Computer Networking

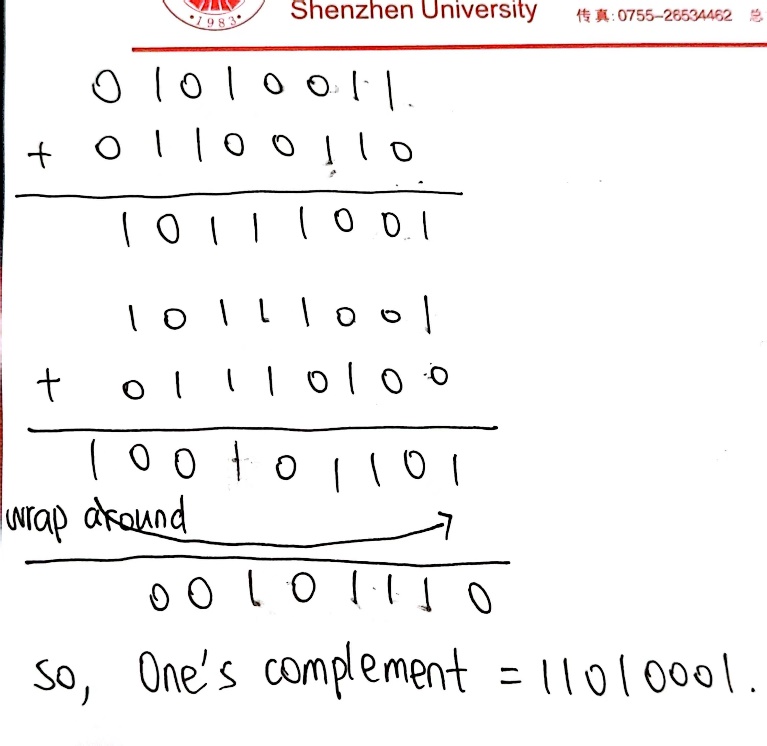
Assignment 2

**Name: 陈应权**

**Student number: 2022280297**

Problem 1: UPD (20 points)

Solution:



To detect errors, the receiver adds the four words including the three original words and the checksum. If the sum contains a zero, the receiver knows there has been an error.

All one-bit errors can be detected, but two-bit errors can be undetected for example when the last digit of the first word is converted to a 0 and the last digit of the second word is converted to a 1.

Problem 2: Reliable Data Transfer (10 points)

Solution:

Suppose the sender is in state “Wait for call 1 from above” and the receiver is in state “Wait for 1 from below.” The sender sends a packet with sequence number 1, and transitions to the state of “Wait for ACK or NAK 1,” then waiting for an ACK or NAK.

Suppose now the receiver receives the packet with sequence number 1 correctly, sends an ACK, and transitions to the state of “Wait for 0 from below,” waiting for a data packet with sequence number 0.

However, the ACK is corrupted. When the rdt2.1 sender gets the corrupted ACK, it resends the packet with sequence number 1. However, the receiver is waiting for a packet with sequence number 0 and always sends a NAK until it get a packet with sequence number 0.

Therefore the sender will always be sending a packet with sequence number 1, and the receiver will always be sending a NAK. The sender will then send packet 1, and the receiver will continue to send NAK, again and again. Then the sender and receiver to enter into a deadlock state, where each is waiting for an event that will never occur.

Problem 3: Pipelining (20 points)

Solution:

(1) Here we have a window size of N=3. Suppose the receiver has received packet k-1,

and all other preceding packets has been ACKed. If all of these ACK's have been received by sender, then sender's window is [k, k+N-1].

Suppose next that none of the ACKs have been received at the sender. In this second case, the sender's window contains k-1 and the N packets up to and including k-1. The sender's window is thus

[k-N,k-1]. By these arguments, the senders window is of size 3 and begins somewhere in the

range [k-N,k].

(2) If the receiver is waiting for packet k, then it has received and ACKed packet k-1 and the N-1 packets before that. If none of those N ACKs have been yet received by the sender, then ACK messages with values of [k-N,k-1] may still be propagating back. Because the sender has sent packets [k-N, k-1], it must be the case that the sender has already received an ACK for k-N-1. Once the receiver has sent an ACK for k-N-1 it will never send an ACK that is less that k-N-1. Thus the range of inflight ACK values can range from k-N-1 to k-1.

Problem 4: TCP (20 points)

Solution:

(1)

GoBackN:

A sends 9 segments in total. They are initially sent segments 1, 2, 3, 4, 5 and later resent segments 2, 3, 4, and 5. So, 5 + 4 = 9

B sends 8 ACKs. They are 4 ACKS with sequence number 1, and 4 ACKS with sequence numbers 2, 3, 4, and 5.

Selective Repeat:

A sends 6 segments in total. They are initially sent segments 1, 2, 3, 4, 5 and later resent segments 2.

B sends 5 ACKs. They are 4 ACKS with sequence number 1, 3, 4, 5. And there is one ACK with sequence number 2.

TCP:

A sends 6 segments in total. They are initially sent segments 1, 2, 3, 4, 5 and later resent segments 2.

B sends 5 ACKs. They are 4 ACKS with sequence number 2. There is one ACK with sequence numbers 6 which means 2-5 segments has been successfully received. So , 4 + 1 = 5

(2)

TCP. This is because TCP uses fast retransmit without waiting until time out.

Problem 5: Congestion Control (30 points)

(1) TCP slowstart is operating in the intervals [1,6] and [23,26]. As shown in the graph, When the congestion window size grows by 2 to the NTH power instead of linearly is TCP slowstart.

(2) TCP congestion avoidance is operating in the intervals [6,16] and [17,22]. As shown in the graph, When the congestion window size grows by linearly is TCP congestion avoidance.

(3) After the 16th transmission round, packet loss is detected by a triple duplicate

ACK. Because if there was a timeout, the congestion window size would have dropped to 1.But in

Graph the congestion window size is about 24,so it is recognized by a triple duplicate ACK

(4) After the 22nd transmission round, segment loss is detected due to timeout, because

the congestion window size have dropped to 1.

(5) The threshold is initially about 32, because it is at this window size that slow start stops and

congestion avoidance begins. It is a turning point.

(6) As teacher said in class , the threshold is set to half the value of the congestion window when packet loss is detected. When loss is detected during transmission round 16, the congestion windows size is 42. Therefore the threshold is 21 during the 18th transmission round.

(7) As teacher said in class , the threshold is set to half the value of the congestion window when packet loss is detected. When loss is detected during transmission round 22, the congestion windows size is 29. 29/2 =14.5.We should take the lower floor than 14.5. Therefore the threshold is 14 during the 24th transmission round.

(8) packet 70 is sent in the 7th transmission round.

We should take the method of summing and adding During the 1st transmission round, packet 1 is sent; packet 2-3 are sent in the 2nd transmission round; packets 4-7 are sent in the 3rd transmission round; packets 8-15 are sent in the 4th transmission round; packets 16-31 are sent in the 5th transmission round; packets 32-63 are sent in the 6th transmission round; packets 64 – 96 are sent in the 7th transmission round. Thus packet 70 is sent in the 7th transmission round.

Next is a visual diagram:

|  |  |  |
| --- | --- | --- |
| Transmission round | segments | Sum of segments |
| 1 | 1 | 1 |
| 2 | 2 | 3 |
| 3 | 4 | 7 |
| 4 | 8 | 15 |
| 5 | 16 | 31 |
| 6 | 32 | 63 |
| 7 | 33 | 96 |

(9) The threshold will be set to half the current value of the congestion window (8) when

the loss occurred and congestion window will be set to the new threshold value + 3 MSS . because the window size is 8, therefore new values of the threshold is 8/2 = 4 and new window size will be 7.

(10) TCP Tahoe: Tahoe implements a basic congestion control algorithm. It uses slow start and congestion avoidance to control the sending rate of packets. Upon packet loss, Tahoe enters slow start, **reducing the congestion window size to 1 and gradually increasing it to recover the sending rate.**

The threshold will be set to half the current value of the congestion window (42) when

the loss occurred.

So threshold is 21, and congestion window size is 1.

(11)

TCP Tahoe: Tahoe implements a basic congestion control algorithm. It uses slow start and congestion avoidance to control the sending rate of packets. Upon packet loss, Tahoe enters slow start, **reducing the congestion window size to 1 and gradually increasing it to recover the sending rate.**

So, round 17, 1 packet; round 18, 2 packets; round 19, 4 packets; round 20, 8 packets;

round 21, 16 packets; round 22, 21 packets.

So, the total number is 52.

|  |  |  |
| --- | --- | --- |
| Transmission round | segments | Sum of segments |
| 17 | 1 | 1 |
| 18 | 2 | 3 |
| 19 | 4 | 7 |
| 20 | 8 | 15 |
| 21 | 16 | 31 |
| 22 | 21 | 52 |