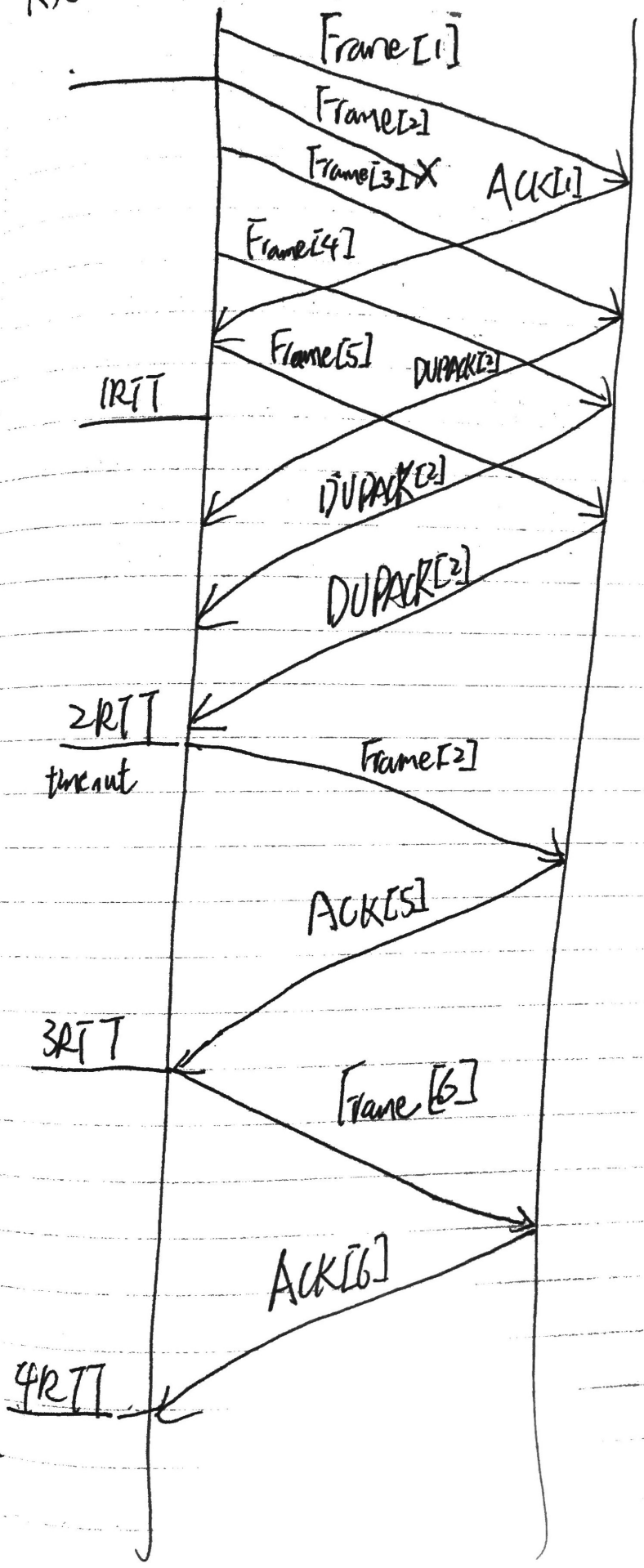
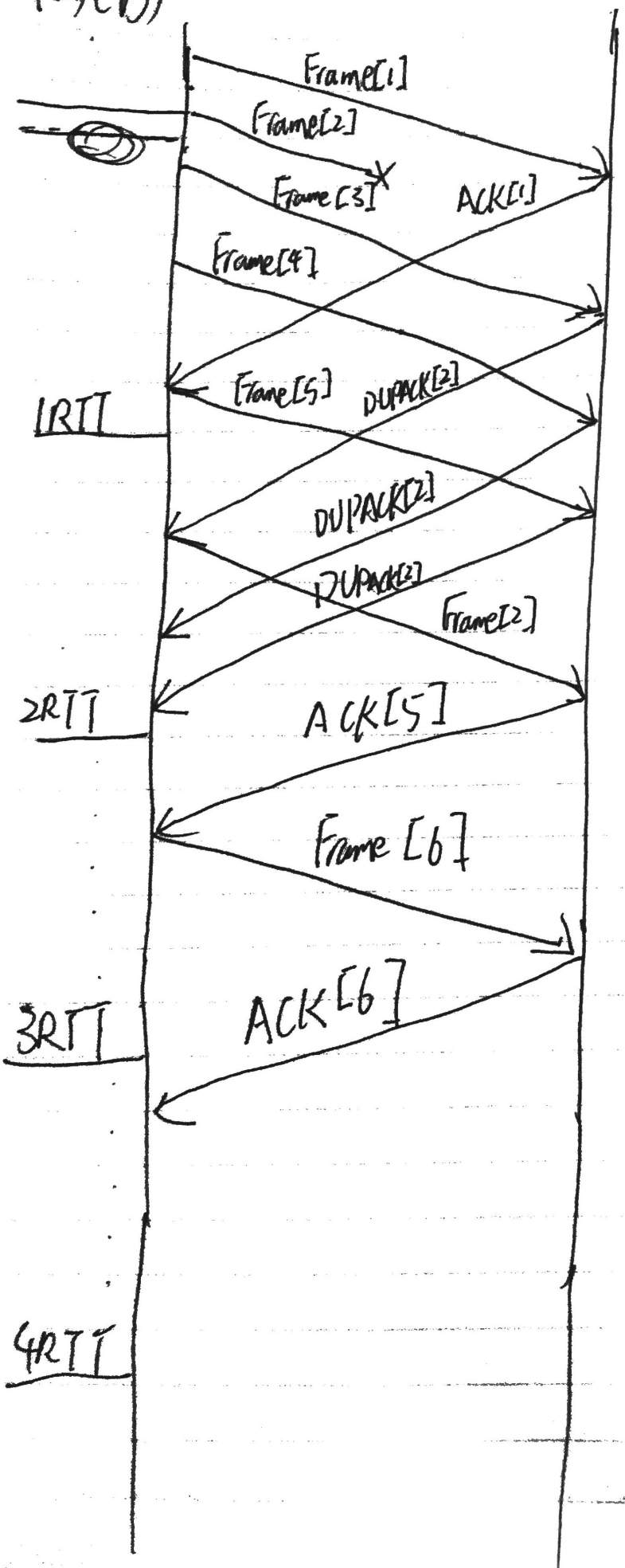
1.(1)(a)


Frame[1]
Frame[2]
Frame[3]
ACK[1]
Frame[4]
Frame[5]
DUPACK[2]
1RTT
DUPACK[2]
DUPACK[2]
2RTT
timeout
Frame[2]
ACK[5]
3RTT
Frame[6]
ACK[6]
4RTT

1.9(b)



This reduce the transaction time about 1 RTT

5 3

1. (2) $N = SWS + RWS - 1 = 8$

~~When~~ sender window sent 0 1 2 3 4 to ~~8~~ receiver window, ~~receiver window~~ will accept all of them but all the ACKs are ~~lost~~ since $SWS > RWS$. RWS ~~expected~~ now expects to receive 5 6 7

If we have DATA[8] in receive window, the earliest possible receive window is DATA[6], DATA[7], DATA[8]. The ACK[6] has been received and DATA[5] was delivered. But because SWS = 5, so all the DATA[0]'s sent were sent before DATA[5]. The DATA[0] can't arrive later ~~for~~ by the no-out-of-order arrival hypothesis.

2, BF?

	Address	Port
B1	MAC(A)	A
	MAC(C)	B2
B2	MAC(A)	B1
	MAC(C)	B3
	MAC(D)	B4
B3	MAC(A)	B2
	MAC(C)	C
	MAC(D)	B4
B4	MAC(A)	B2
	MAC(D)	D

When M sends to L, both B1 and B2 will receive the message and send it to L, but they'll also send a message to each other. ~~Since the message they send to L has been received, they both know the write right path towards L~~. So the packet will circle endlessly in both $M \to B_2 \to L \to B_1$ and $M \to B_1 \to L \to B_2$ ways.