EE122 MIDTERM EXAM: 2003-10-13

Scott Shenker, Ion Stoica

Last name _____ **First name** _____

Student ID _____ **Login: ee122-**____

Please circle the last two letters of your login.

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z

Discussion section meeting time _____ TA's name _____

This is a closed-book exam. This booklet has 10 numbered pages including the cover page, and contains 4 questions. You have 1.5 hours to complete the exam. Good luck.

Question	Max Score	Your Score
1	18	
2	20	
3	20	
4	42	

Question 1 (18 pts)

Node A (IP address: 111.111.111.111) connects to router R (IP address: 222.222.222.222). The store-and-forward router R connects to node B (IP address: 123.123.123.123). All links have 40kbps bandwidth and have propagation delay 0.75ms. A sends packets, each of which is 50 bytes, to B. B sends an ACK to A each time it receives a packet. The size of the ACK packet is 5 bytes. The protocol is stop-and-go. (i.e. The sender sends a new packet only after the previous packet has been acked.)

You can assume that there is no loss during the transmission and the connection A to B is the only connection for the links. You can ignore the processing delay and queuing delay of a packet, but not the transmission time.

A sends a packet to B through R.

(4 pts) (a) What are the source IP address and destination IP address of the header of the packet when node A sends the packet to the router?
(4 pts) (b) What are the source IP address and destination IP address of the header of the packet when the router sends the packet to node B?
(5 pts) (c) What is the throughput of the flow from A to B?
(5 pts) (d) Are the terms bandwidth and throughput the same? If not, why not?

Soln:
(a) source IP address: 111.111.111.111 (2 pts)
destination IP address: 123.123.123.123 (2 pts)
(b) same as (i) (4 pts)
(c) Total time:
delay from A to R + delay from R to B
+ ACK time from B to R + ACK time from R to A
= ((50*8)/40k) + 0.75ms + ((50*8)/40k) + 0.75ms
+ ((5*8)/40k) + 0.75ms + ((5*8)/40k) + 0.75ms
= 25ms (2.5 pts)
Throughput = (50*8)/25ms = 16kbps (2.5 pts)
(d) No. (1 pt)
Bandwidth refers to the number of bits per second that can be transmitted on the link. (2 pts)
Throughput refers to the performance of the flow and can be expressed as ”TransferSize/TransferTime” (2 pts)

Question 2 (20 pts)

Consider two routers A, B that are connected by a 10 Mbps link. Suppose 4 TCP flows and a constant UDP flow of 5 Mbps traverse this link, and that this link is the bottleneck for all these flows. Also assume all the TCP flows have the same RTT and each of them can use up to 5 Mbps alone.

(5 pts) (a) What will be the average throughput of each of the TCP flows? What will be the average throughput of the UDP flow?

(5 pts) (b) Suppose that router A implements the round-robin scheduling discipline, and that the packet size of TCP-1 and TCP-2 is 500 bytes, the packet size of both TCP-3 and TCP-4 is 1500 bytes, and the packet size of the UDP flow is 1000 bytes. What is the average throughput of each flow?

(5 pts) (c) Suppose FQ is implemented at router A (FQ implements max-min fairness). What will be the average throughput of each of the TCP flows? What will be the average throughput of the UDP flow?

(5 pts) (d) Now, suppose that the maximum possible rates for each of the TCP flows are as follows: TCP-1 (1 Mbps), TCP-2 (2 Mbps), TCP-3 (2.5 Mbps), TCP-4 (3 Mbps). As before, the UDP flow has a constant rate of 5 Mbps. If the router A uses FQ, what is the average throughput of each of the TCP flows and the UDP flow?

Soln:

(a) Each TCP flow has a throughput of 1.25 Mbps. (2.5 pts)
The UDP flow has a throughput of 5 Mbps. (2.5 pts)

(b) TCP-1 and TCP-2 (1 Mbps each) (2 pts)
TCP-3 and TCP-4 (3 Mbps each) (2 pts)
UDP (2 Mbps each) (1 pts)

(c) Under FQ, the TCP and UDP flows have a throughput of 2 Mbps. (5 pts)

(d) TCP Flow 1: 1 Mbps (1 pts)
TCP Flow 2: 2 Mbps (1 pts)
TCP Flow 3: 2.33 Mbps (1 pts)
TCP Flow 4: 2.33 Mbps (1 pts)
UDP Flow : 2.33 Mbps (1 pts)

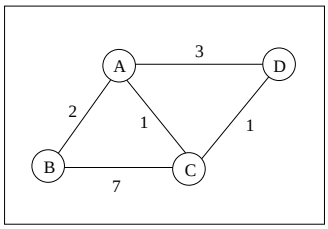
Question 3 (20 pts)

Compute the routing tables for each node in the graph below using the distance vector method. Note that:

- The routing tables reflect the routing state in each router at the start of each time unit.
- Whenever an entry in the routing table of a node changes, that node propagates its routing table to each of its neighbors.
- At the end of each time unit, each node updates its routing table based on the routing tables received from its neighbors. A router updates its routing table instantaneously, i.e., you can neglect the time taken by the update operation.
- Propagation time along all links takes 1 time unit (ie. sending it at the start of the interval causes the information to reach the other end of the link at the end of the time unit, in time for updating), except for the link between nodes A and C, which takes 2 time units.

Give the routing tables at the start of each time unit until the algorithm converges. The first set of tables have been done for you.

Solution:
In general, most people can handle basic distance vector updates. Some people basically ignored the fact that updates going to and from A to C takes 1 more time unit than usual. If this is the case, then lots of alarm bells should have sounded in our minds, because it would end up being EXACTLY the same as in the lecture notes!!
There are two main parts to the question: basic distance vector updating, and taking into account the additional delay between A and C. We took into account propagating errors, ie. we'll only penalize once if the entries at the later times depend on those earlier. We deduct 1 point for each incorrect "normal" distance vector update, and a one-time 5 points if the delayed propagation is not handled correctly. Also, benefit of the doubt is always given.



Time 0

Node A

Destination	Cost	Next Hop
B	2	B
C	1	C
D	3	D

Node B

Destination	Cost	Next Hop
A	2	A
C	7	C
D	inf	–

Node C

Destination	Cost	Next Hop
A	1	A
B	7	B
D	1	D

Node D

Destination	Cost	Next Hop
A	3	A
B	inf	–
C	1	C

Time 1

Node A

Destination	Cost	Next Hop
B	2	B
C	1	C
D	3	D

Node B

Destination	Cost	Next Hop
A	2	A
C	3	A
D	5	A

Node C

Destination	Cost	Next Hop
A	1	A
B	7	B
D	1	D

Node D

Destination	Cost	Next Hop
A	2	C
B	5	A
C	1	C

Time 2

Node A

Destination	Cost	Next Hop
B	2	B
C	1	C
D	2	C

Node B

Destination	Cost	Next Hop
A	2	A
C	3	A
D	5	A

Node C

Destination	Cost	Next Hop
A	1	A
B	3	A
D	1	D

Node D

Destination	Cost	Next Hop
A	2	C
B	5	A
C	1	C

Time 3

Node A

Destination	Cost	Next Hop
B	2	B
C	1	C
D	2	C

Node B

Destination	Cost	Next Hop
A	2	A
C	3	A
D	4	A

Node C

Destination	Cost	Next Hop
A	1	A
B	3	A
D	1	D

Node D

Destination	Cost	Next Hop
A	2	C
B	4	C
C	1	C

Question 4 (42 pts)

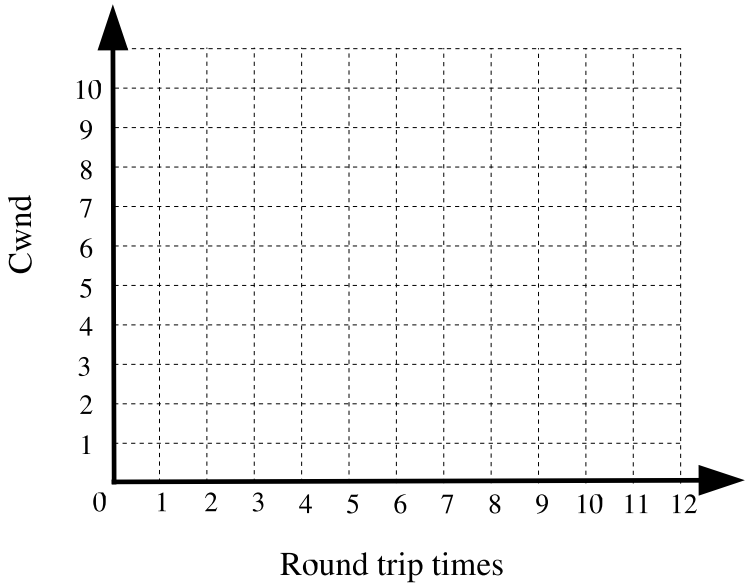
(5 pts) (a) What is the purpose of the *congestion window* (cwnd) in TCP?

(5 pts) (b) What is the *slow-start* phase in TCP? What is its purpose?

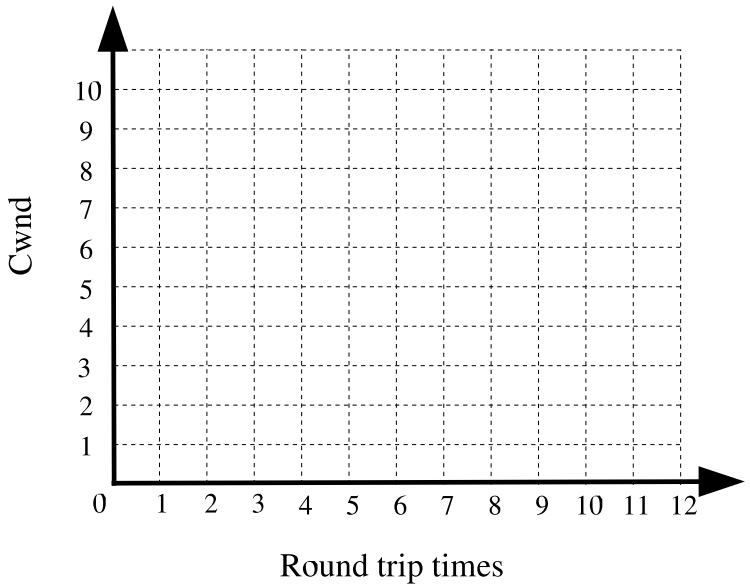
(5 pts) (c) What is the *congestion avoidance* phase in TCP? What is its purpose?

(12 pts) (d) Suppose a source S and a destination D are connected via a single link with round trip time RTT. Suppose S starts sending to D at time $t = 0$ using TCP Tahoe, and suppose the 14th segment sent is lost. Note that TCP's cwnd is initially set to 1. Assume TCP's RTO is equal to 3 RTTs (ie, TCP will detect

loss when the RTO expires). Fill in the diagram below showing how the congestion window grows over time. Label which periods of time are *slow-start* and which are *congestion avoidance*.



(10 pts) (e) Repeat part (d), but this time assume the 12th segment is lost, and assume TCP Reno is used. Note that TCP Reno also detects loss upon receiving 3 duplicate acks.



(5 pts) (f) In part (d), after 10 RTTs, how many segments has *D* received?

Soln:

(a) cwnd limits the sending rate so as to avoid congestion inside the network. The TCP algorithm never sends more than the minimum of cwnd and the flow control window. cwnd is maintained in bytes, and is incremented in slow-start by the segment size every time an ACK is received. cwnd is flow control imposed by the sender, while the advertised window is flow-control imposed by the receiver.

(b) To ramp up quickly to the capacity of the path, without overflowing queues. It works by controlling the rate packets are injected into the network based on the rate at which ACKs are received. By limiting the rate at which we send, we can avoid overflowing queues at routers, which would cause packet losses (losses can greatly decrease throughput, since we need to wait for a timeout to detect them before sending more packets). However, we do increase the window quickly (at an exponential rate, in fact). This allows us to quickly reach the available bandwidth of the path.

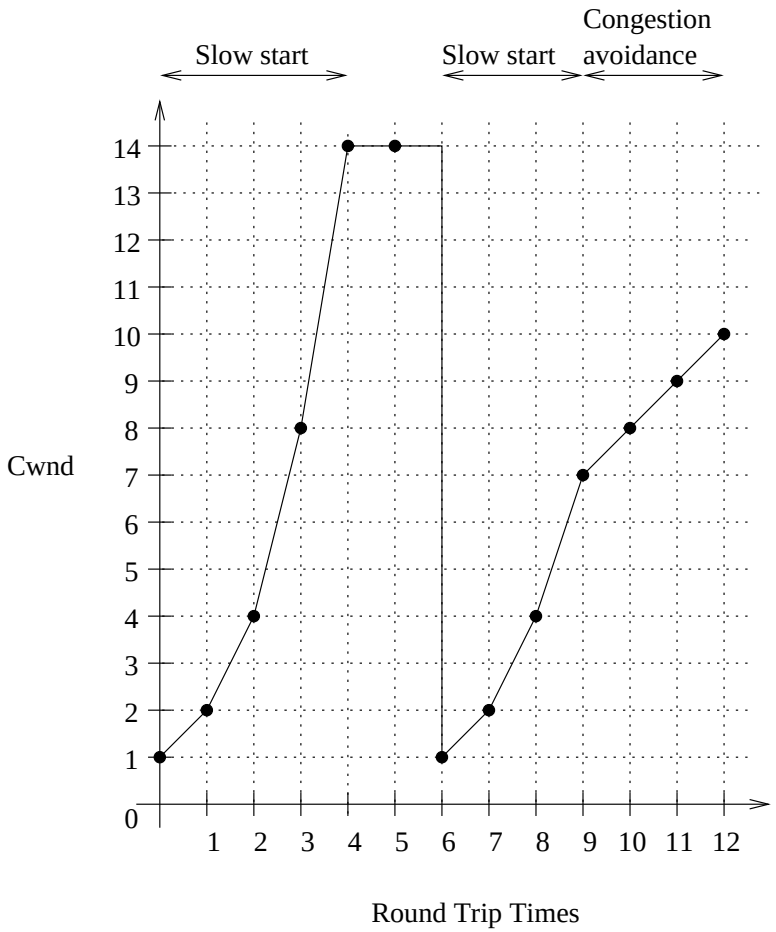
(c) The key observation in congestion avoidance is that, when a packet loss occurs, we are nearing the saturation point of the path. Hence, we keep an estimate of the available capacity of the link (ssthresh), and we slow down the rate at which we increase cwnd when we reach this point. ssthresh represents a estimate of the lower bound on the capacity of the link. Once we exceed ssthresh, it is likely that we are nearing capacity. So, we start growing cwnd linearly instead of exponentially. This allows us to continue exploring available capacity while avoiding congestion.

(d) We accepted answers which assumed that Tahoe has fast retransmit + no fast recovery, and has no fast retransmit + no fast recovery. Note that during slow start and congestion avoidance phase, cwnd will increase only if the ack received is for new data, ie. not dupacks.
Assume Tahoe = no fast retransmission + no fast recovery

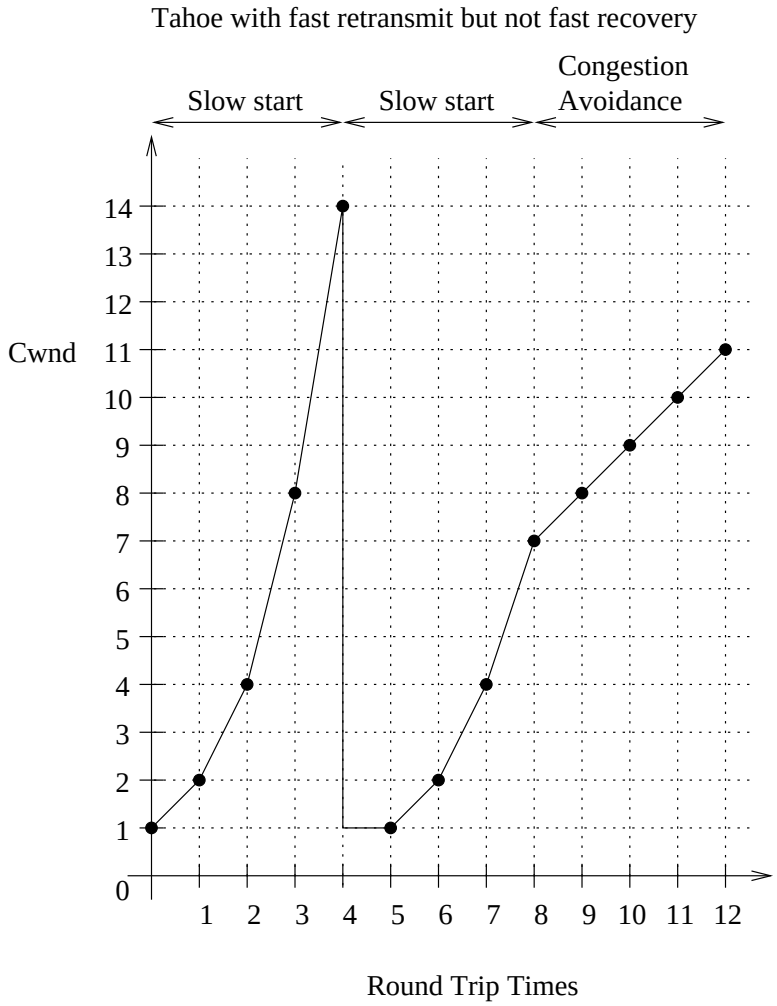
Cwnd	RTT	Segments sent
1	0	1
2	1	2,3
4	2	4,5,6,7
8	3	8,9,10,11,12,13,14,15
14	4	16 to 27
14	5	-
1	6	14
2	7	28,29
4	8	30,31,32,33
7	9	34,35,36,37,38,39,40
8	10	41 to 48
9	11	48 to 56
10	12	57 to 66

Next assume Tahoe = fast retransmission + no fast recovery

Tahoe with no fast retransmit and no fast recovery

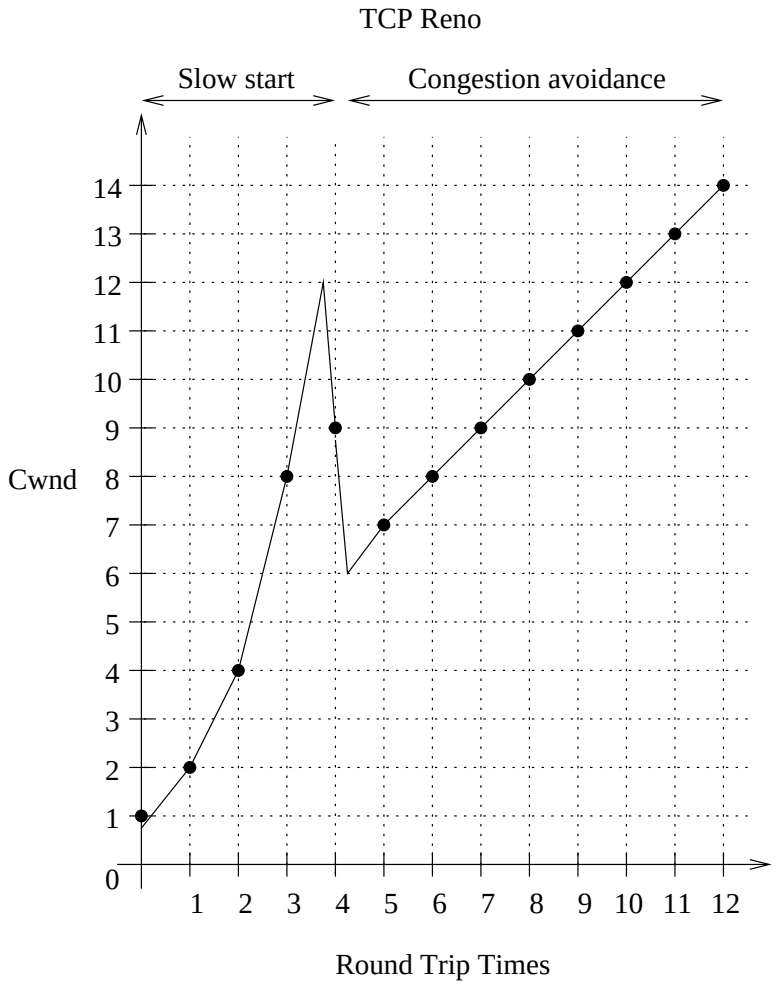


Cwnd	RTT	Segments sent
1	0	1
2	1	2,3
4	2	4,5,6,7
8	3	8,9,10,11,12,13,14,15
14	4	16 to 27
1	5	14
2	6	28,29
4	7	30,31,32,33
7	8	34,35,36,37,38,39,40
8	9	41 to 48
9	10	49 to 57
10	11	58 to 67
11	12	68 to 78



(e) See figure. Note in this case fast-retransmit will be activated since the source will receive 3 duplicate ACKs. cwnd goes up to 12 because we receive the ACKs for segments 8,9,10 and 11, and cwnd increases by 1 for each of these ACKs.

Cwnd	RTT	Segments sent
1	0	1
2	1	2,3
4	2	4,5,6,7
8	3	8,9,10,11,12,13,14,15
9	4	12,16 to 20
7	5	21 to 27
8	6	28 to 35
9	7	36 to 44
10	8	45 to 54
11	9	55 to 65
12	10	66 to 77
13	11	78 to 90
14	12	91 to 105



(f) Tahoe with fast retransmit but not fast recovery: 78 segments
Tahoe with no fast retransmit and no fast recovery: 66 segments