1	2	3	4
10	10	10	9

6.829
Computer Networks
2005/10/14

Dan Ports
drkp@mit.edu
Collaborators: {amdragon}

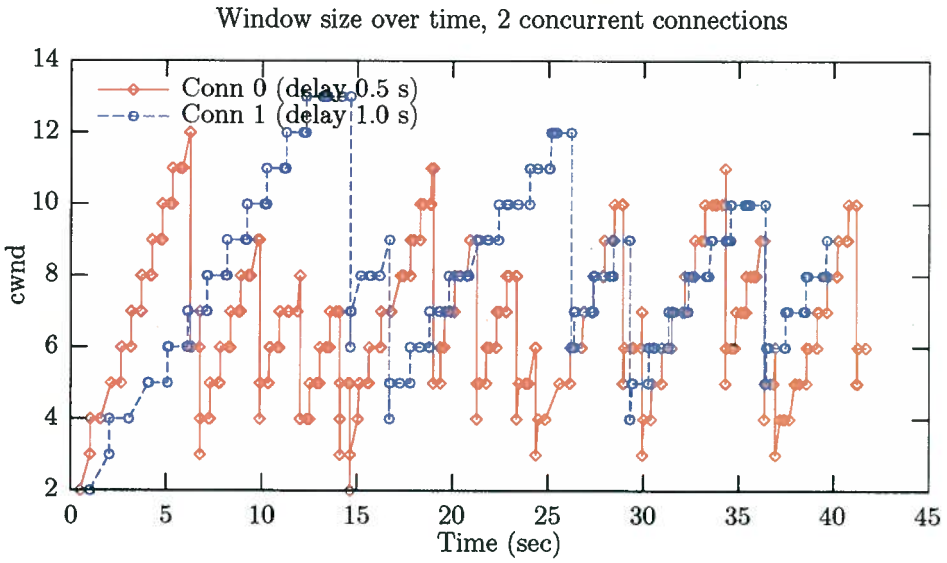
Problem Set 3

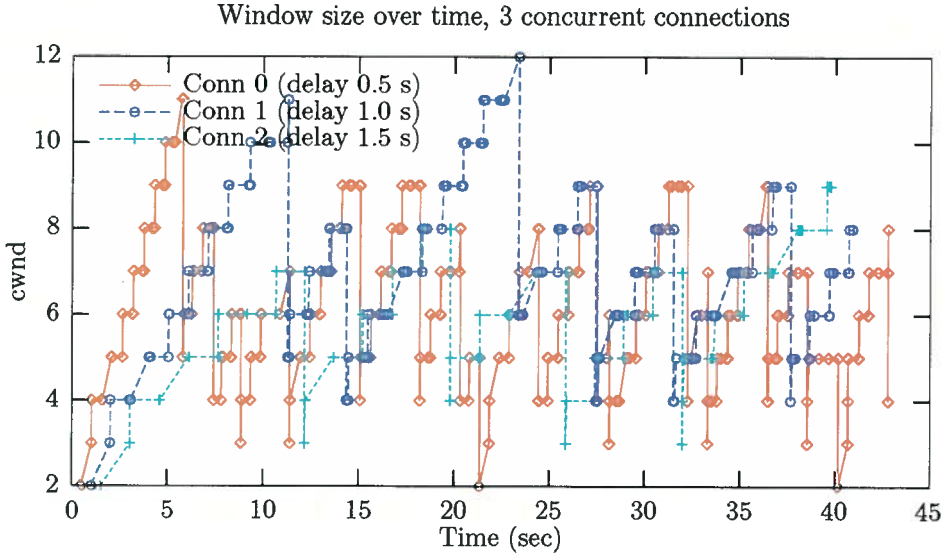
Problem 1:

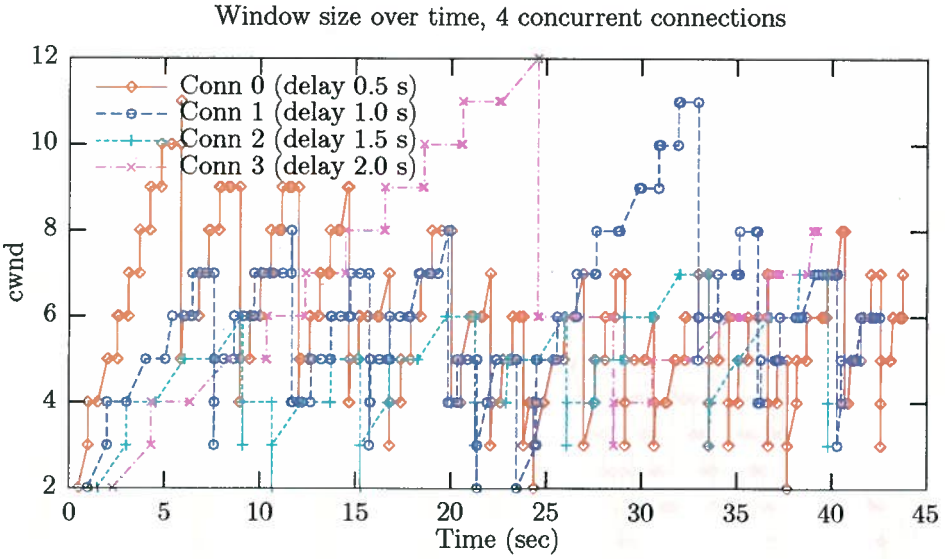
1) *a)* The following table shows the percentage of total bandwidth usage that is used by each connection, with connection i having $0.5i$ delay.

		Connection			
		1	**2**	**3**	**4**
# Connections	**2**	62%	38%		
	3	57%	25%	17%	
	4	54%	26%	12%	8%

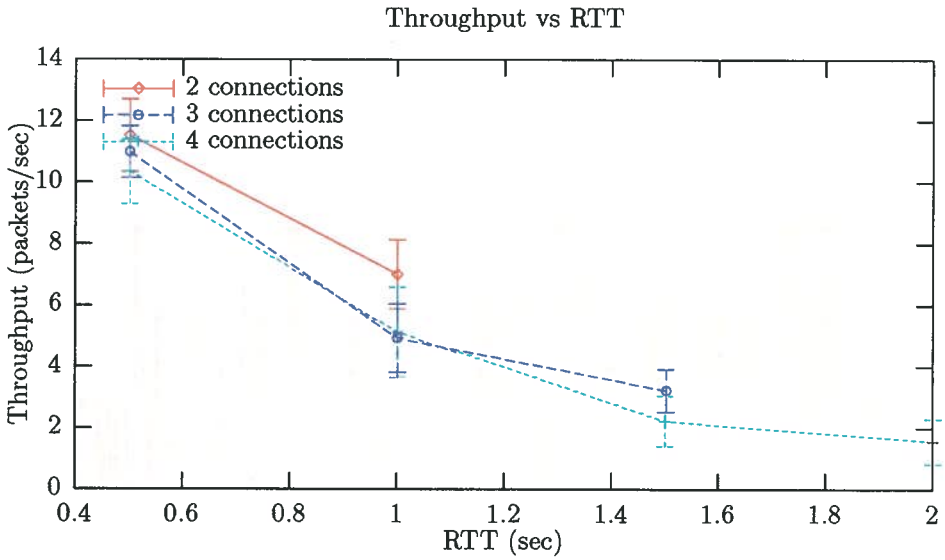
b)




Window size over time, 3 concurrent connections
Conn 0 (delay 0.5 s)
Conn 1 (delay 1.0 s)
Conn 2 (delay 1.5 s)
cwnd
Time (sec)
2
4
6
8
10
12
0
5
10
15
20
25
30
35
40
45


Window size over time, 4 concurrent connections
Conn 0 (delay 0.5 s)
Conn 1 (delay 1.0 s)
Conn 2 (delay 1.5 s)
Conn 3 (delay 2.0 s)
cwnd
Time (sec)
2
4
6
8
10
12
0
5
10
15
20
25
30
35
40
45

c)

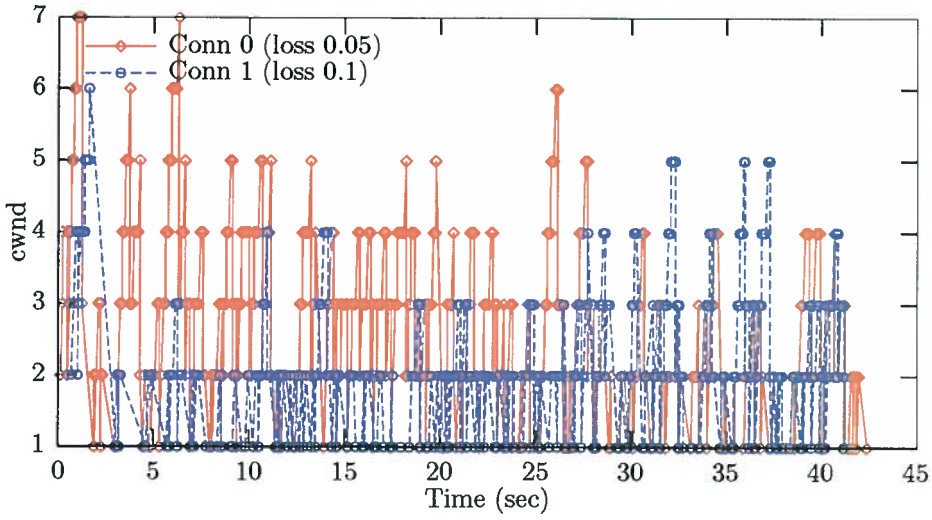


We can see that as the RTT increases, the throughput decreases.

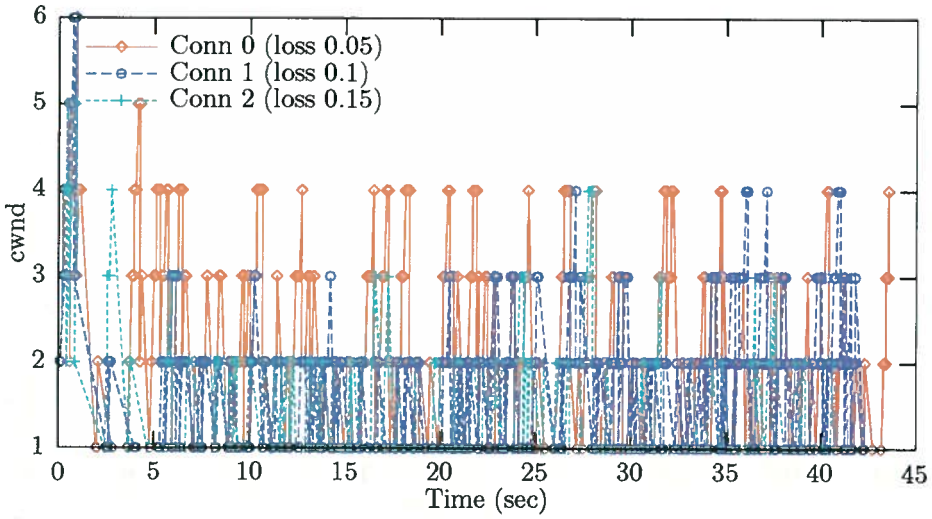
2) *a)* The following table shows the percentage of total bandwidth usage that is used by each connection, with connection i having $0.05i$ loss.

		Connection			
		1	**2**	**3**	**4**
# Connections	**2**	63%	78%		
	3	55%	32%	13%	
	4	52%	23%	16%	9%

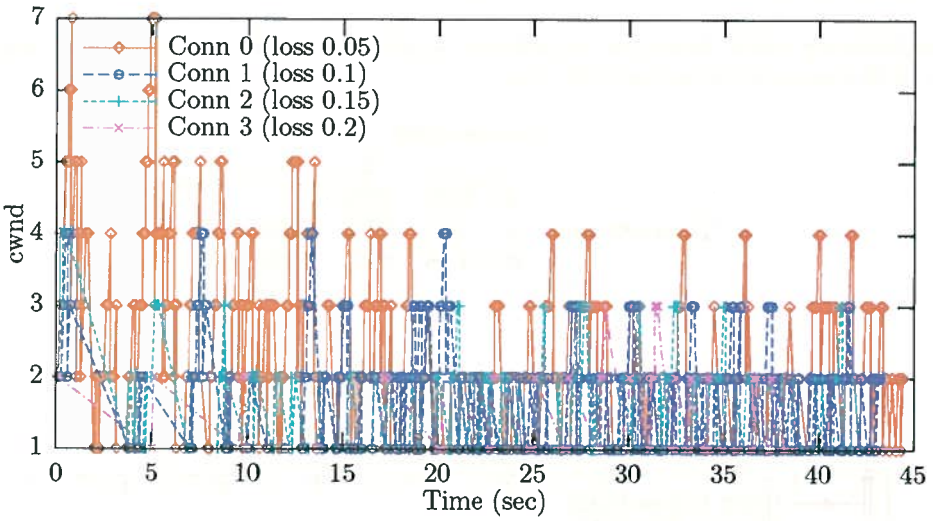
b)



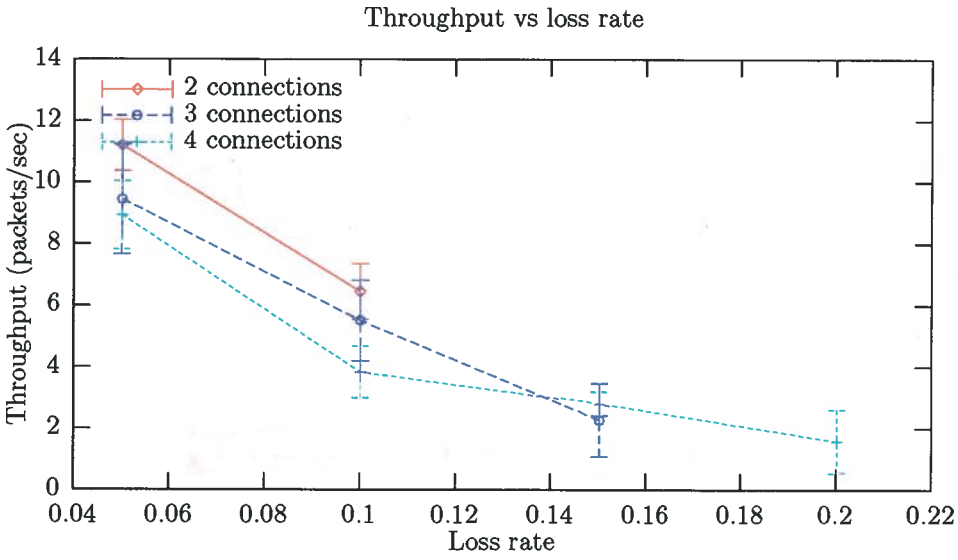
Window size over time, 3 concurrent connections



Window size over time, 4 concurrent connections



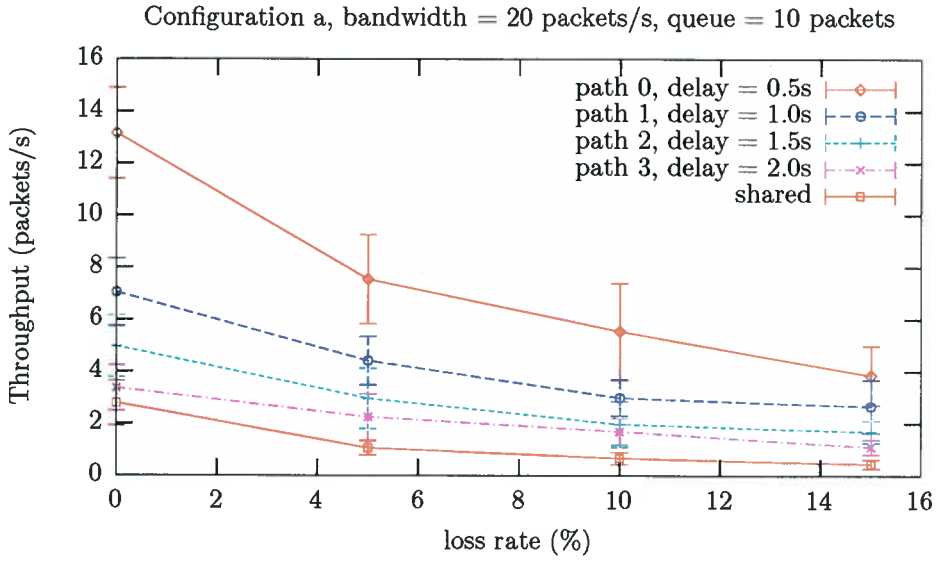
c)

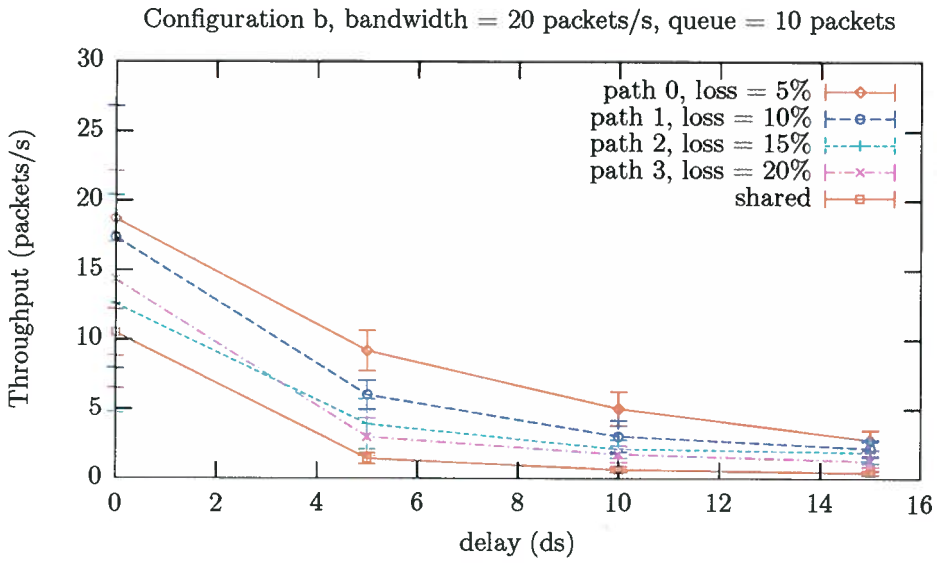


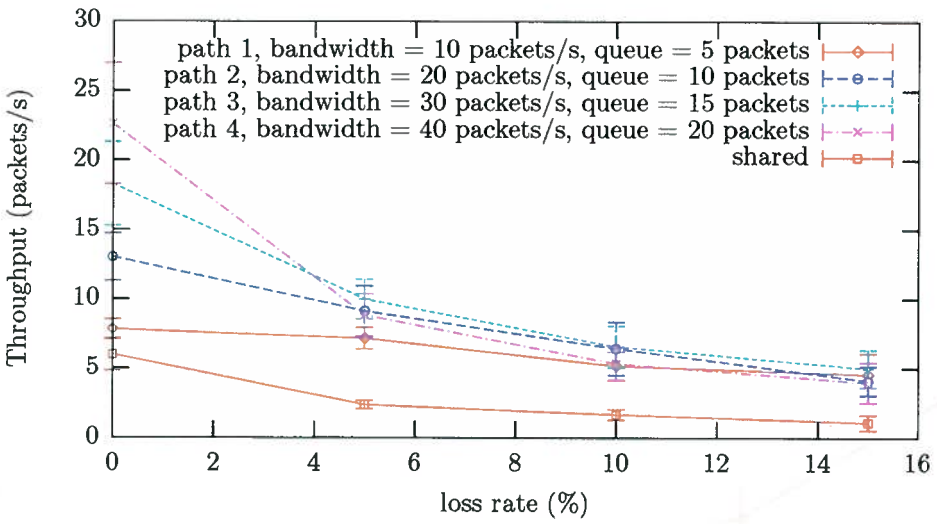
We can see that as the loss rate increases, the throughput decreases.

Problem 2:

1) For each of these three configurations, graphs of the throughput along each path for different parameter values follow. The shared case scales the throughput by 1/4 to show the average throughput on each connection.

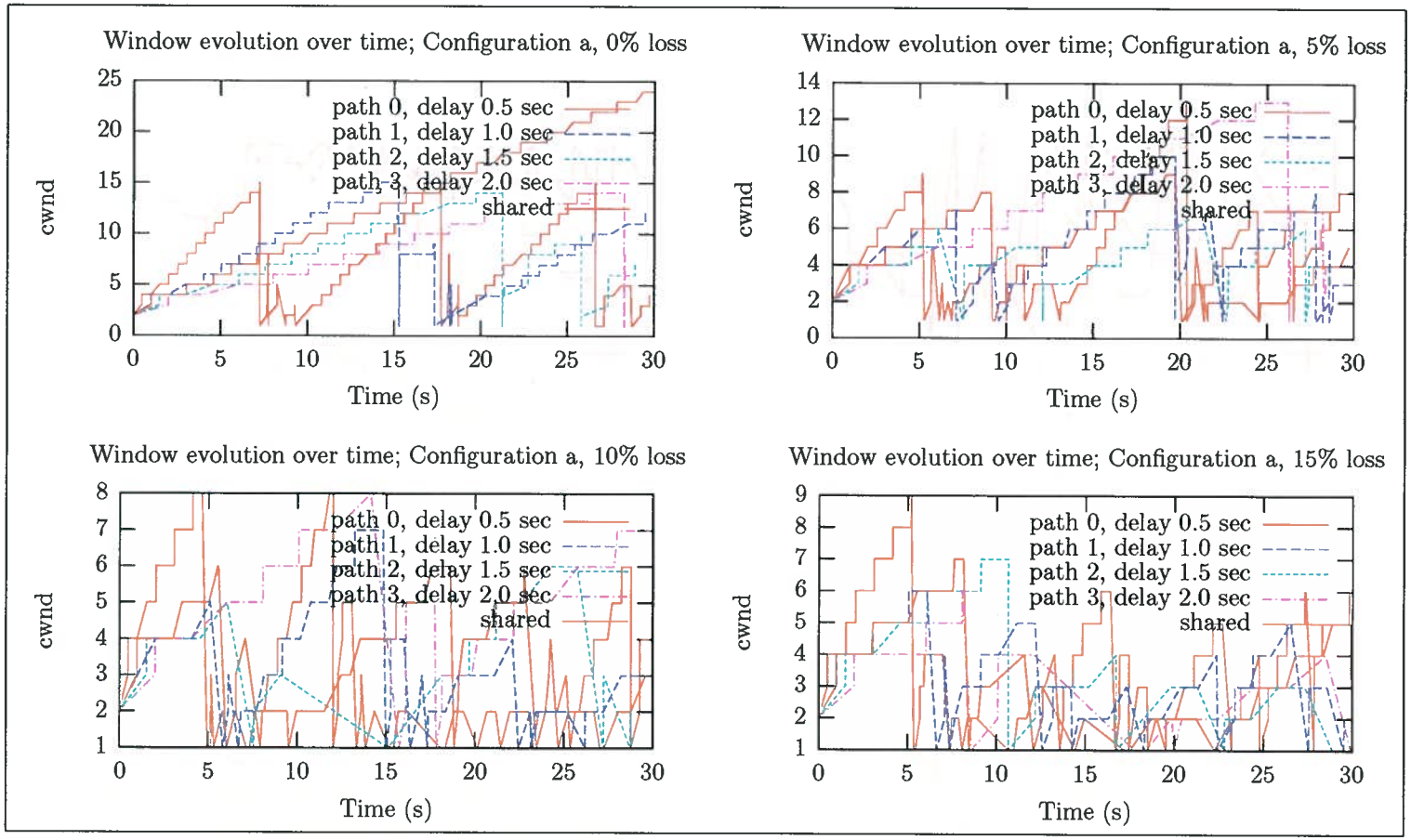



Configuration b, bandwidth = 20 packets/s, queue = 10 packets
Throughput (packets/s)
delay (ds)
path 0, loss = 5%
path 1, loss = 10%
path 2, loss = 15%
path 3, loss = 20%
shared
0
5
10
15
20
25
30
0
2
4
6
8
10
12
14
16

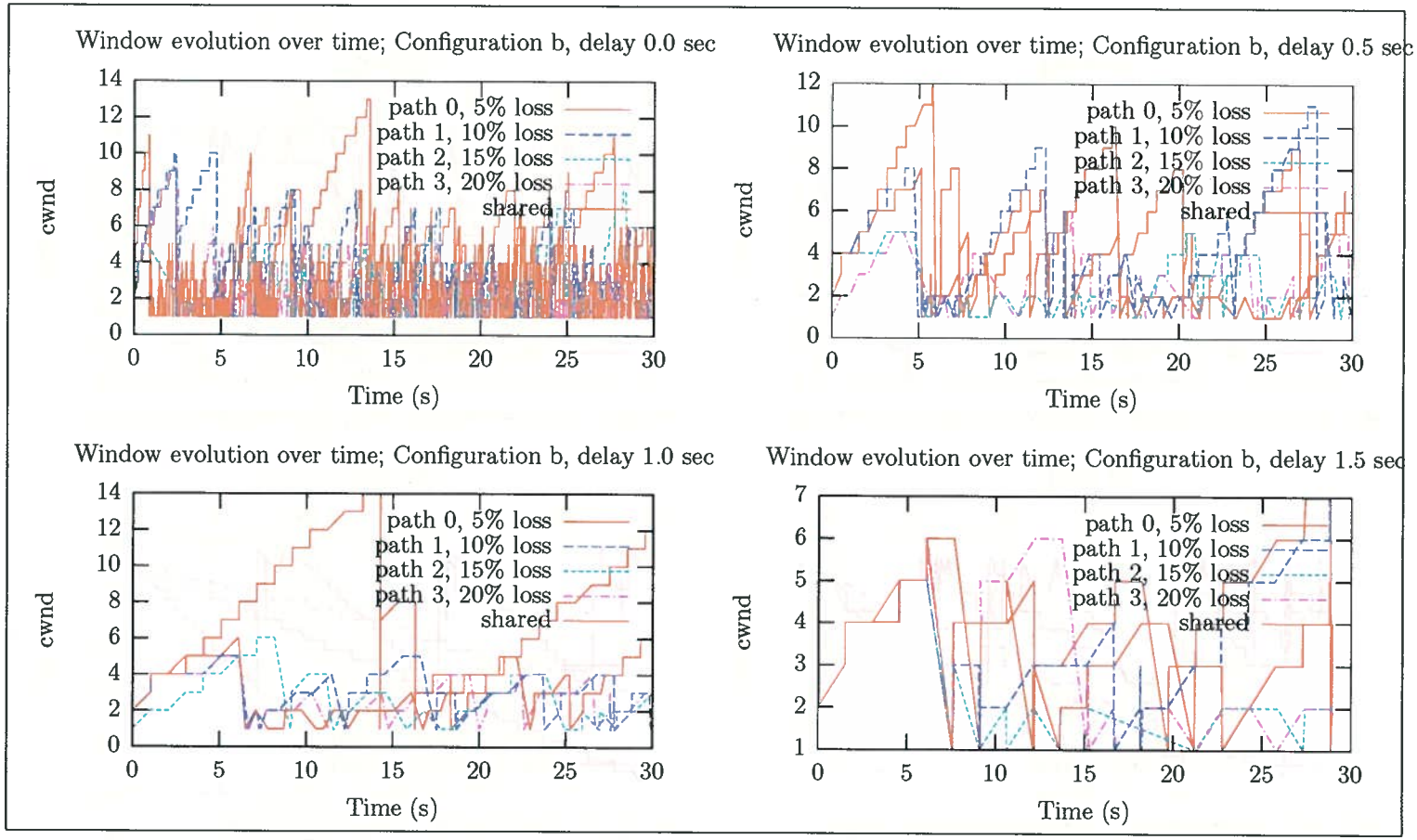

Configuration c, delay 5 ds
Throughput (packets/s)
loss rate (%)
path 1, bandwidth = 10 packets/s, queue = 5 packets
path 2, bandwidth = 20 packets/s, queue = 10 packets
path 3, bandwidth = 30 packets/s, queue = 15 packets
path 4, bandwidth = 40 packets/s, queue = 20 packets
shared
0
5
10
15
20
25
30
0
2
4
6
8
10
12
14
16

2)

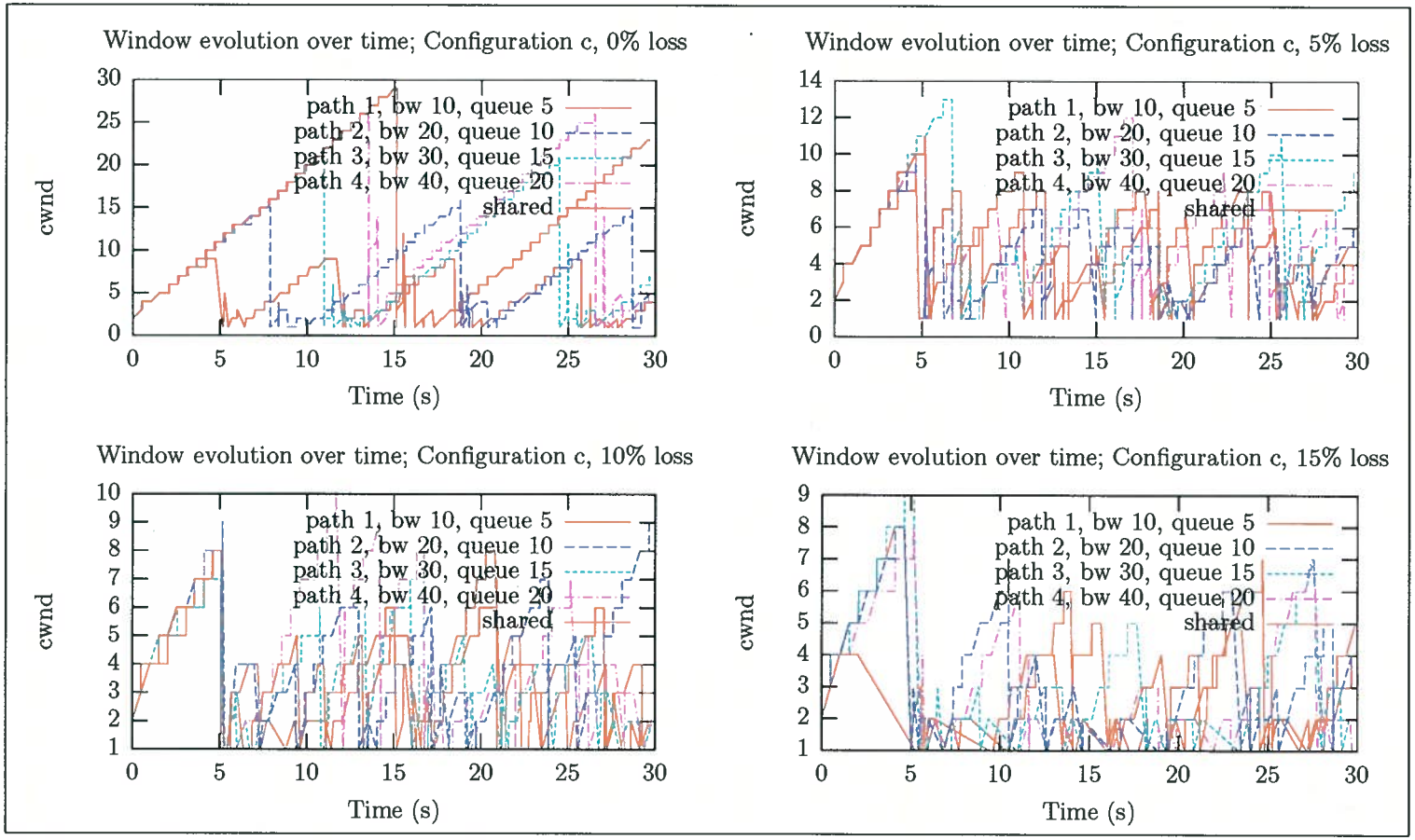
a) The following four graphs show the evolution of the `cwnd` size for configuration a, with varying loss rates.



b) The following four graphs show the evolution of the `cwnd` size for configuration b, with varying delays.



c) The following four graphs show the evolution of the `cwnd` size for configuration c, with varying loss rates.



3) The current design of TCP uses a shared window. In all of these multihoming configurations, we see that the shared window case gives lower average throughput than even the slowest of the paths would with a separate window. This suggests that performance of TCP could be improved if each link could be given a separate window.

4) We see that fast retransmissions improve the performance of `aimd`, and reduce the performance of `shared`. This is because fast retransmit can function normally on the `aimd` case, and improve performance just as it would for a single-homed link. In the `shared` case, fast retransit is not useful: the shared connection window sees many out of order ACKs since packets are reordered by the different delays on the multihomed links, and so packets are retransmitted and cwnd is decreased unnecessarily. This does not occur on the `aimd` case because each link does not adjust cwnd in response to ACKs from other links.

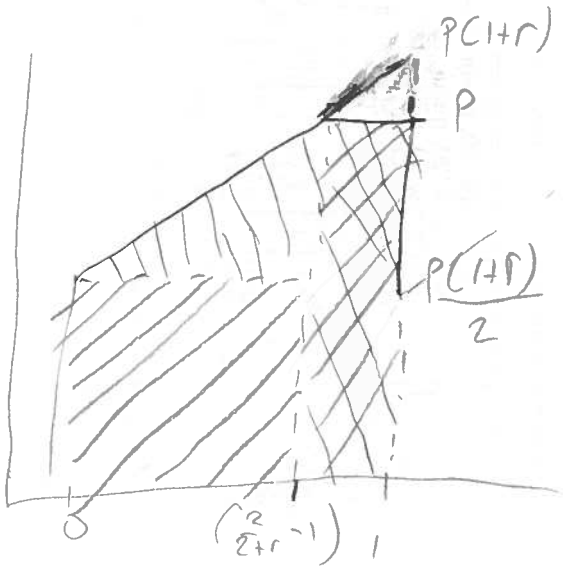
Problem 3:

1) The TCP congestion window over time forms a sawtooth plot in the steady state: it increases additively until it reaches a maximum $W_{\max}$ and a packet is dropped; then it will drop to $\frac{W_{\max}}{2}$ and begin increasing again. Packets are first dropped when both the available bandwidth and buffer space are exhausted. P bytes per RTT can be transmitted over the link, and B bytes can be stored in the buffer, so $W_{\max} = P + B$. This is $2P$ if $B = P$. So the window size is always at least P, and so at least P packets are transmitted each RTT: this is full link utilization by definition.

For 40 Gb/s and 100 ms, this is 4 gigabits, or half a gigabyte of buffer memory.

2) If negligible buffer space is available, packets will be dropped as soon as the window size exceeds P. So the window size will oscillate between $P/2$ and P; thus, the average window size is $3/4P$, and so the average link utilization is $3/4$.

3) Let $B = rP$. Then by the reasoning of problem 3.1, the TCP window size will oscillate between $P + B = P(1 + r)$ and half this amount. The link utilization will be $\frac{1}{P}$ the window size, but not greater than 1. We can consider this geometrically:

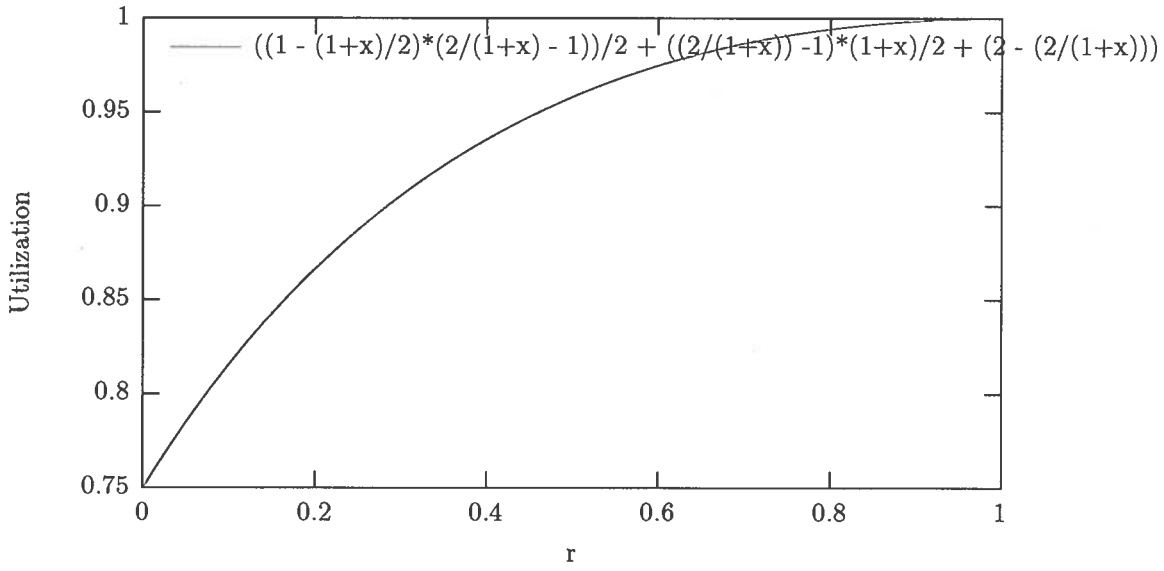


The total area is the area of the triangle from 0 to $\left(\frac{2}{2+r} - 1\right)$, the area of the rectangle below

it, and the area of the rectangle that represents the region truncated at 1:

$$U(r) = \frac{\left(1 - \frac{1+r}{2}\right)\left(\frac{2}{1+r} - 1\right)}{2} + \left(\frac{2}{1+x} - 1\right)\left(\frac{1+r}{2}\right) + \left(2 - \frac{2}{1+r}\right)$$
$$= \frac{5}{4} + \frac{r}{4} - \frac{r-1}{2} - \frac{1}{r+1}$$

4) $U(r)$ is plotted below:



Problem 4:

1) The fairness controller attempts to make changes proportional to the expected packet rate; since we are reporting the throughput directly, this is the throughput reported by the sender. So lying about the RTT will have no effect on the fairness controller because it allocates proportional to feedback.

(In XCP as in the paper, with `cwnd` as a header parameter, lying about the RTT would cause extra positive feedback to be applied, but not affect negative feedback: rather than the congestion window, If the link is underutilized and positive feedback is being applied, sending a high RTT would cause extra positive feedback to be applied to the sender because rtt^2 appears in the numerator, and only one factor of rtt is canceled by the scaling by $\frac{true\_RTT}{declared\_RTT}$. The negative feedback term has only one rtt factor in the numerator, so this is canceled, and thus lying has no effect.)

The efficiency controller sets its aggregate feedback proportional to the average RTT multiplied by the amount of spare bandwidth. So if a sender sends a high RTT, it may increase the average RTT enough to cause the efficiency control to be underdamped and oscillate. If it sends a low RTT, it may decrease the average RTT enough to cause the efficiency controller's adjustments to be too small, and converge too slowly to be useful.

2) The fairness controller makes positive changes equally to all flows, and negative changes inversely proportional to each link's throughput. So if the link is underutilized and positive feedback is being applied, lying about the throughput has no effect; if the link is congested and negative feedback is being applied, giving a low throughput value would cause the link to receive less feedback and thus

gain and advantage.

The utilization controller does not take congestion windows into account, so lying about this would not affect link utilization.