Lab 3: Packet Scheduling

Harmanprit Tatla – tatlahar
Shunta Chimura – chimuras

Exercise 1.3 Observing a FIFO Scheduler at different loads

Note: The waiting time was calculated by taking the difference of the departure time of a **non-dropped** packet at the scheduler sender and the arrival time of this packet at the scheduler receiver. For each trace below we only used the first **10,000** packets of the Poisson 3 trace.

N = 1 (Low Load)

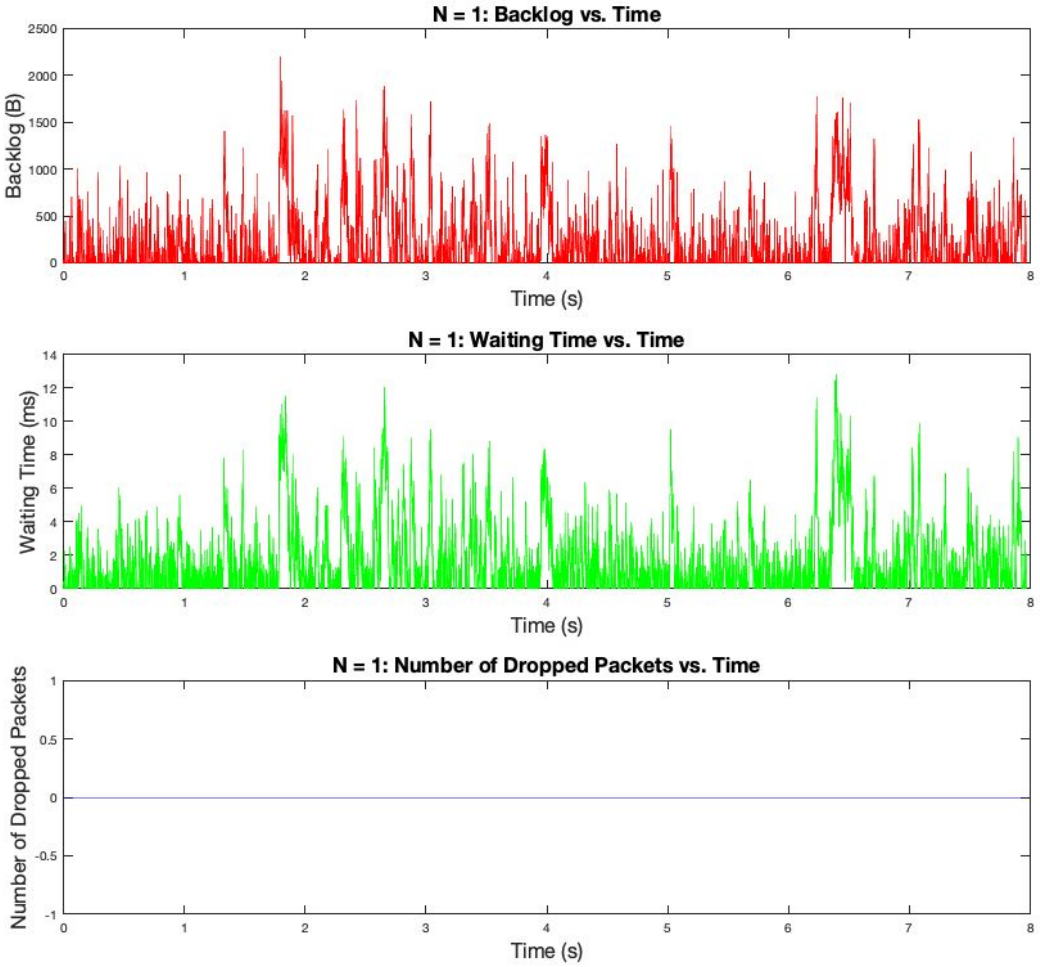

Figure 1: Poisson Trace – N = 1: Backlog, Waiting Time, Number of Dropped packets vs. Time

Observations:

From figure 1, we see that when the Poisson traffic arrival is scaled to 1 Mbps there are no dropped packets and neither the backlog nor waiting-time grows arbitrarily large. The culmination of these two observations suggests the FIFO scheduler is in a regime of low load.

N= 5 (Medium Load)

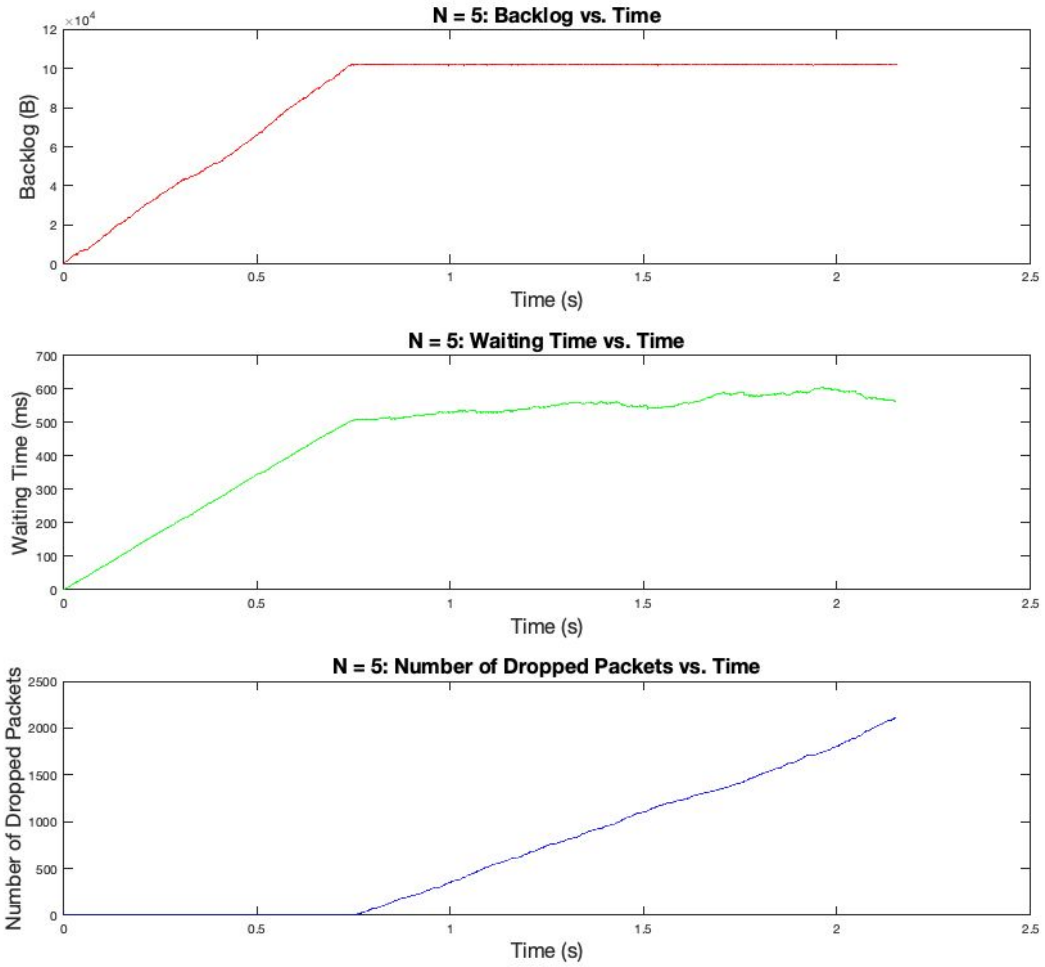

Figure 2: Poisson Trace – N = 5: Backlog, Waiting Time, Number of Dropped packets vs. Time

Observations:

From figure 2, we see that when the arrival rate of the Poisson traffic is scaled to 5 Mbps, the backlog reaches its maximum value of 100 kB after ~0.7 s and then remains approximately constant thereafter. In addition, we also see that in figure 2 that the waiting time and number of dropped packets grows very large after ~0.7 s, which makes sense as the buffer is full at this time. Comparing the N = 5 scenario to the N = 9 case on the next page (i.e. figure 3), we observe it takes longer for the backlog to reach its

maximum value in the N = 5 case than it does for the N = 9 instance. Accordingly, we deem N = 5 to represent a regime of medium load.

N = 9 (High Load)

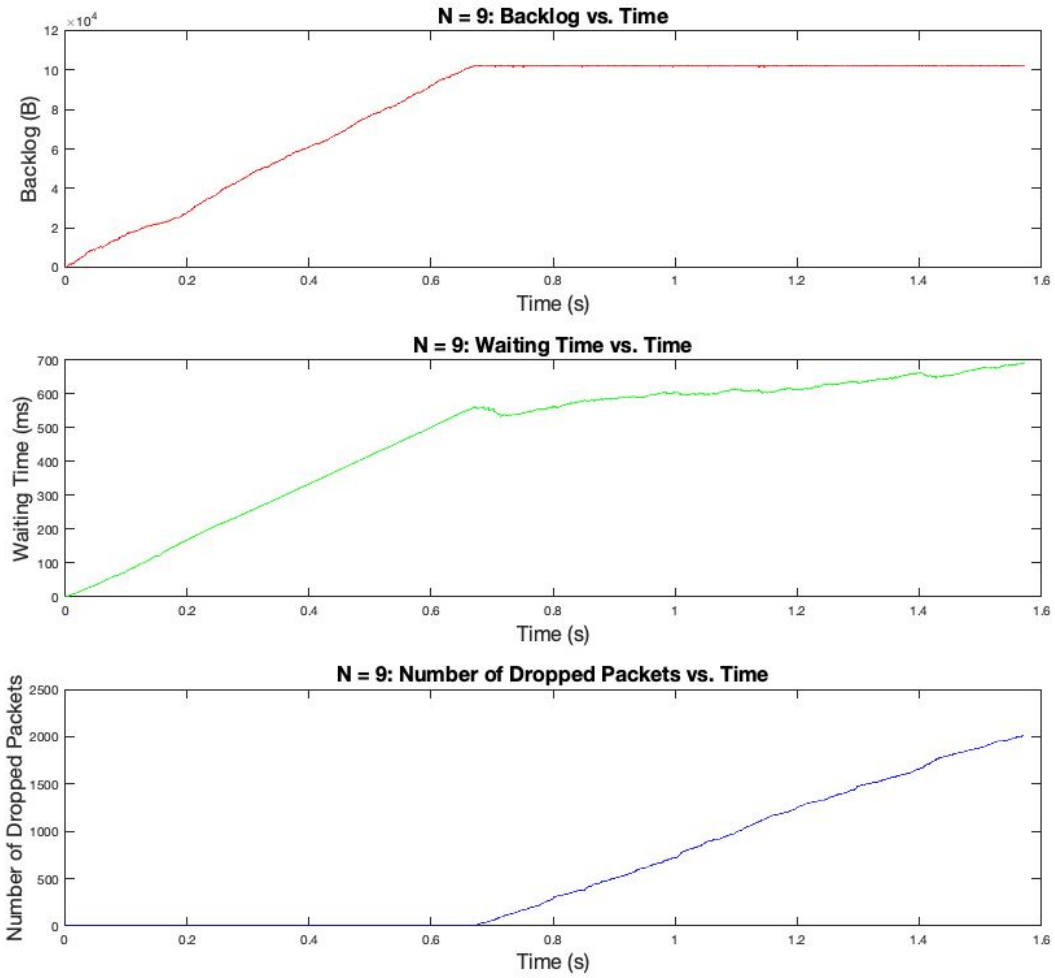

Figure 3: Poisson Trace – N = 9: Backlog, Waiting Time, Number of Dropped packets vs. Time

Observations:

We observe in figure 3 that when the Poisson trace is scaled to 9 Mbps, the buffer overflows after ~0.65 s and both the waiting time and number of dropped packets grow very large thereafter. As it takes less time for the buffer to overflow in the N = 9 case than it did for the N = 5 scenario (i.e. ~0.65 s vs ~0.7 s) we designate the N = 9 case as a regime of high load.

Designate ranges of N, where the FIFO scheduler is in a regime of low load and high load. Justify your choice:

Observing figure 1, we see that for N = 1 the backlog never overflows and thus there are no dropped packets and waiting-time does not grow arbitrarily large. Accordingly (and as mentioned before) we treat N = 1 as a regime of low load. Comparing figures 2 and 3 we see that in both cases the backlog overflows but since it takes longer in the N = 5 case for this to happen than in the N = 9 scenario, we designate N = 5 as a regime of medium load and N = 9 as a regime of high load. Putting these observations together, we treat the following ranges as having the indicated load regimes: (i) $1 \leq N < 5$ - Low Load (ii) $5 \leq N < 9$ - Medium Load (iii) $N \geq 9$ - High Load.

Exercise 2.3 Evaluation of the priority scheduler

Note: For each trace below we only used the first **10,000** packets of the Poisson 3 trace. As well, we only used the first **1000** frames of the video trace with a **33 us** delay between frames.

Furthermore, we apologize for the x and y-axis scaling of the plots below, we were pressed for time and my partner and I are now in completely different time zones.

Video Trace (High Priority) and Poisson Trace (Low Priority)

N = 1 (Low Load)

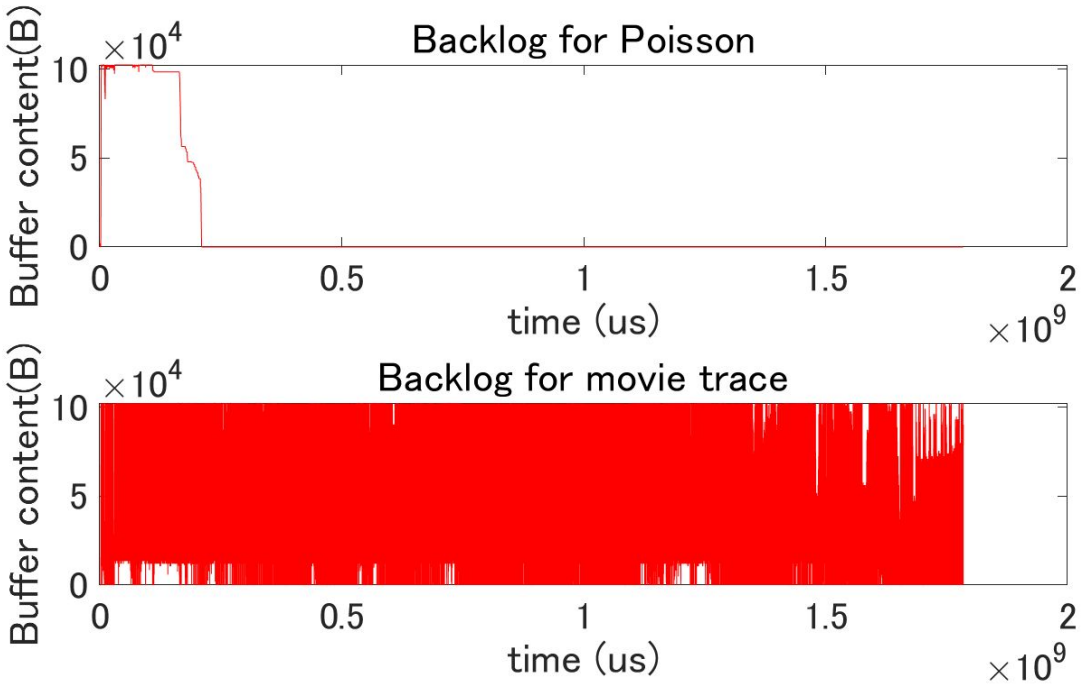

Figure 4: Backlog of Low Priority Poisson Trace at 1 Mbps and High Priority Video Trace when N =1

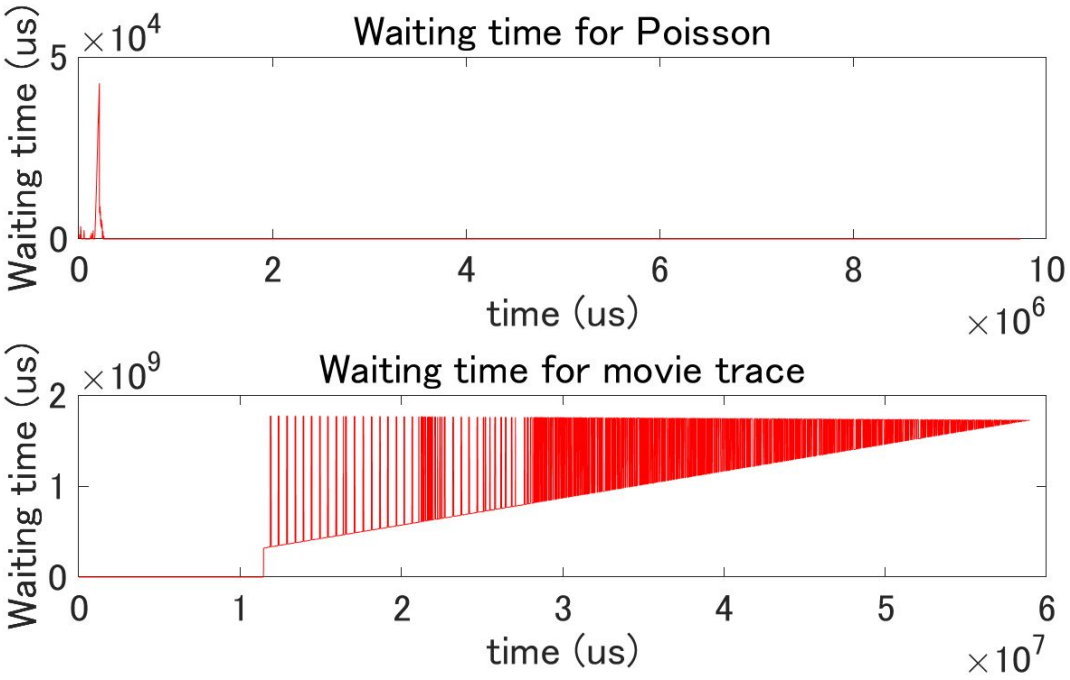

Figure 5: Waiting time of Low Priority Poisson Trace at 1 Mbps and High Priority Video Trace when N = 1

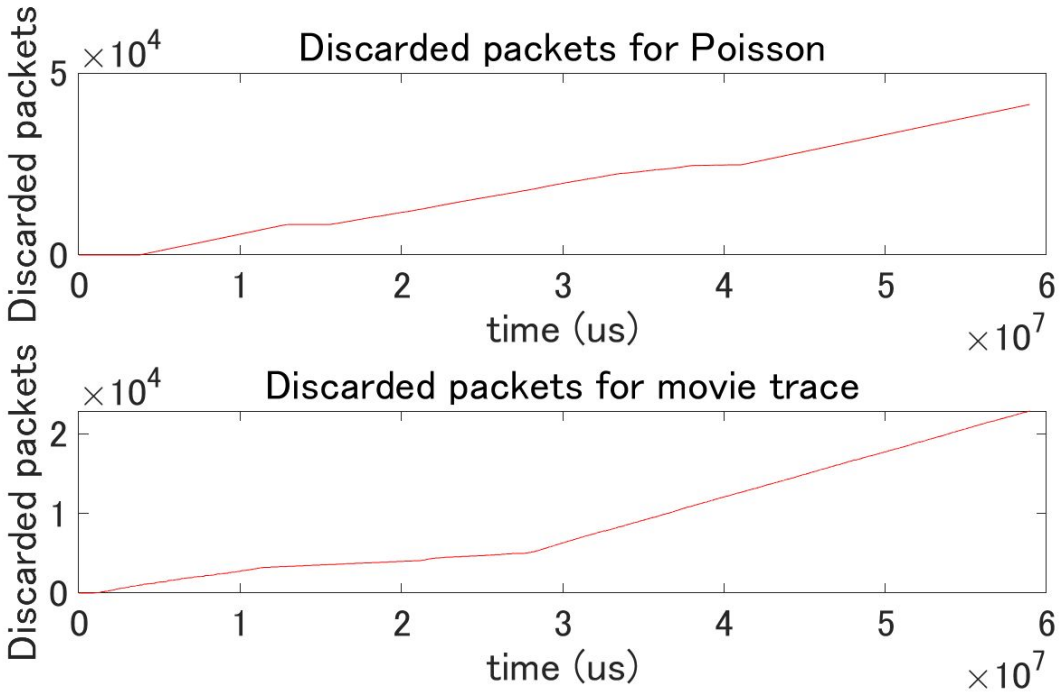

Figure 6: Number of Dropped Packets of Low Priority Poisson Trace at 1 Mbps and High Priority Video Trace when N = 1

Observations:

From figure 4, we see that the backlog for the Poisson 3 trace accumulates initially and is fully serviced around ~0.25x10^9 ms, after this point, the Poisson buffer does not accumulate to a noticeably large value. We believe this could be due to a lot of traffic arriving from the video trace initially causing a large backlog build-up for the Poisson trace, which is then quickly serviced as there may have been a delay before several more video packets arrived. Continuing with figure 4, we see that the backlog for the video trace oscillates between the maximum buffer size and a value of 0, which makes sense as it generates far more traffic than the Poisson trace and is also the higher priority traffic (i.e. gets serviced before the Poisson trace if there is a backlog).

Observing figure 5, we see that for the Poisson trace the waiting time accumulates to very large values initially then tapers off toward much smaller waiting-times (i.e. close to 0 seconds). This might be again due to an initial burst of several video packets arriving then allowing for a small delay where the Poisson trace is serviced, and afterwards, there may be small delays (i.e. between bursts of video packets) where the Poisson packets in the backlog can be quickly serviced. Also in figure 5 we see that the waiting time for the video trace oscillates quickly between large and small values, this makes sense as for a given frame exceeding the max packet size, several many packets are generated and arrive in a burst where some packets have to wait increasingly large wait-times.

Analyzing figure 6, we see that there are more packets dropped for the Poisson trace than for the video trace, this makes sense as the scheduler is likely spending more time servicing the many packets that arrive for each video frame while allowing the backlog for the Poisson trace to build up and overflow.

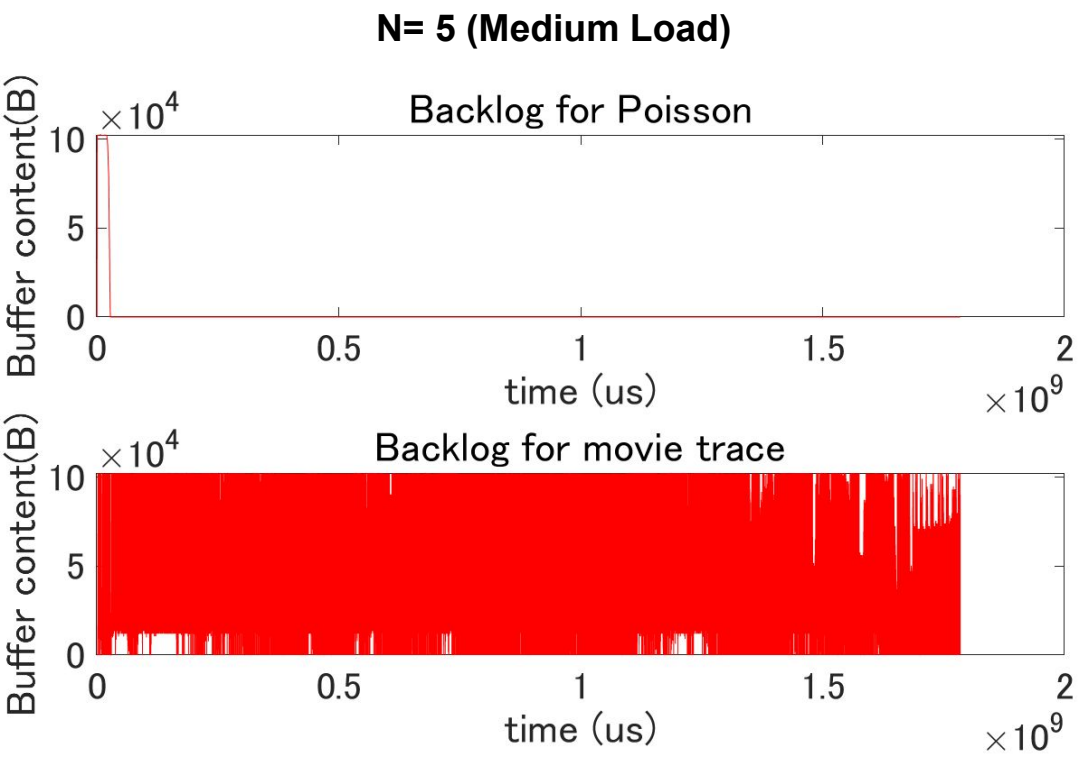

Figure 7: Backlog of Low Priority Poisson Trace at 5 Mbps and High Priority Video Trace when N = 5

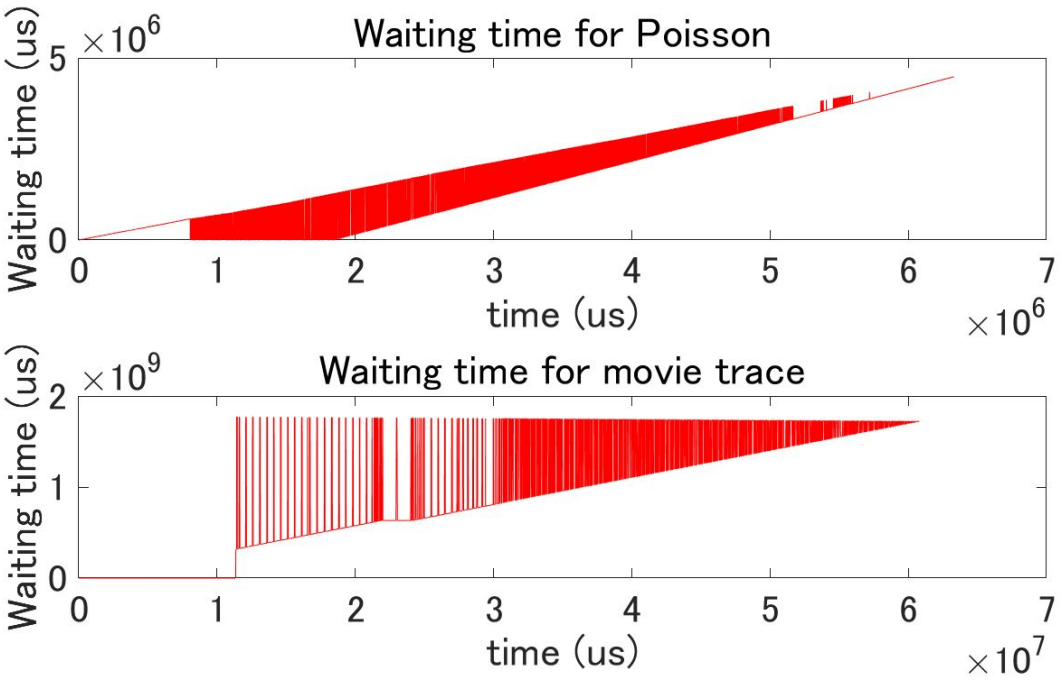

Figure 8: Waiting time of Low Priority Poisson Trace at 5 Mbps and High Priority Video Trace when N = 5

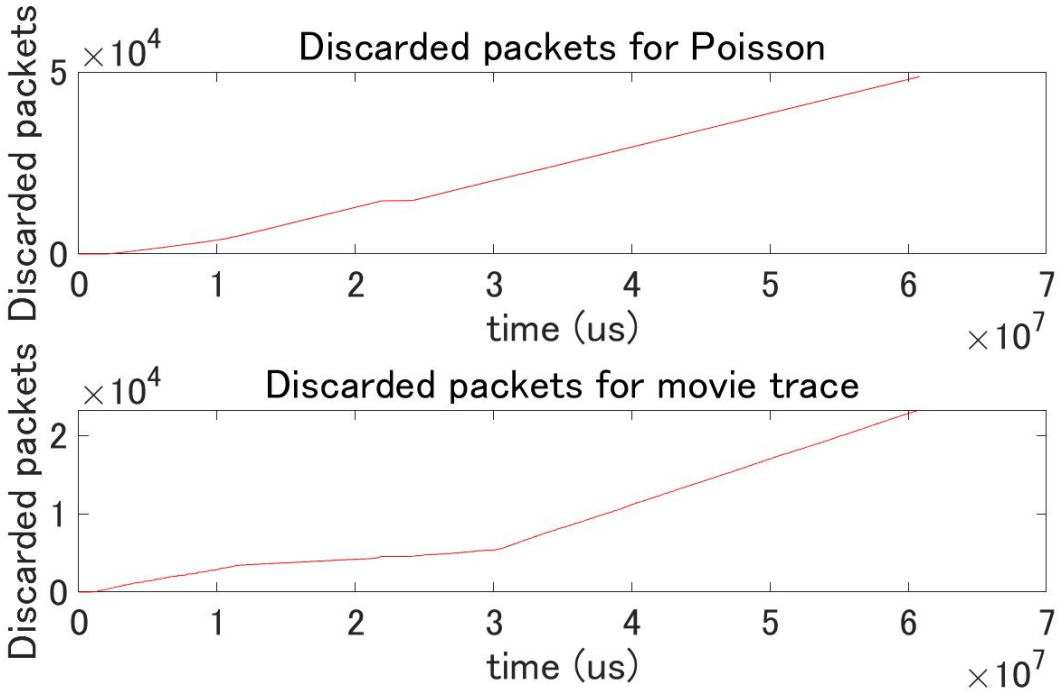

Figure 9: Number of dropped packets of Low Priority Poisson Trace at 5 Mbps and High Priority Video Trace when N = 5

Observations:

Observing figures 5 through 8 for the Poisson 3 trace at 5 Mbps, we see that there is a moment when the backlog overflows and both the waiting times and the number of discarded packets grow to much larger values in this scenario than for the 1 Mbps Poisson trace scenario. These results make sense as the Poisson trace being a lower priority traffic is now arriving faster, thus allowing for the backlog to overflow sooner and for many packets to be waiting longer before transmission.

Analyzing figures 5 through 8 again but for the video trace, we observe that the behaviour is not too different than when the Poisson trace was arriving at the 1 Mbps rate. In general, we still observe an oscillation in the backlog between its max value and 0, along with oscillating but linearly growing waiting times, and lastly, a comparable number of packets dropped as we saw in the experiment with a Poisson trace arriving at 1 Mbps.

N = 9 (High Load)

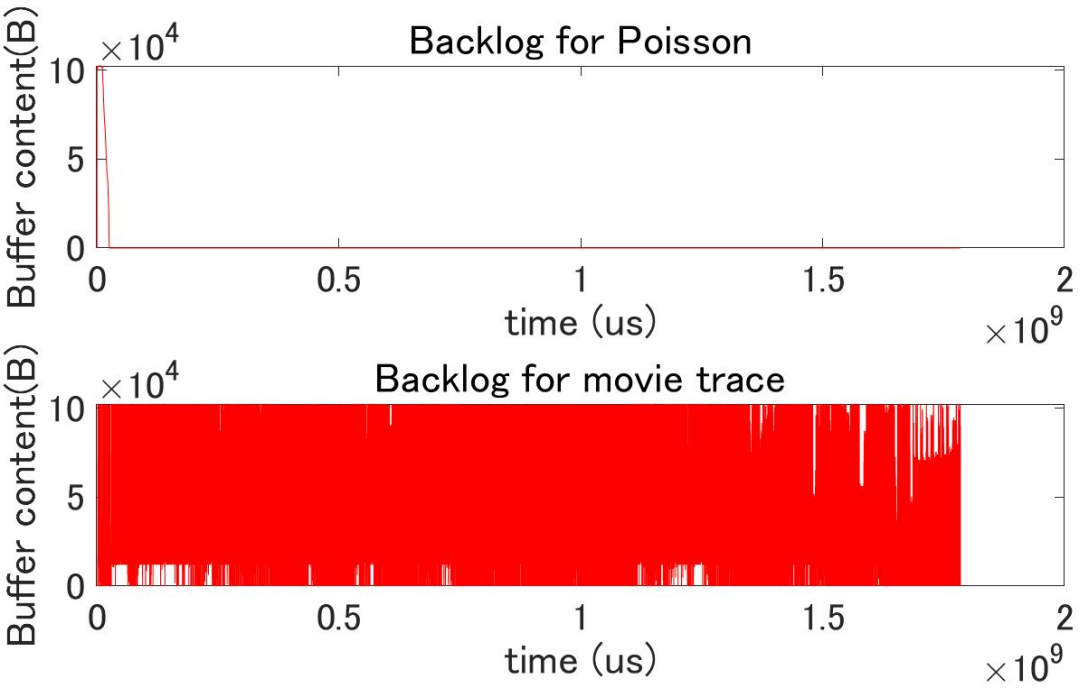

Figure 10: Backlog of Low Priority Poisson Trace at 5 Mbps and High Priority Video Trace when N = 9

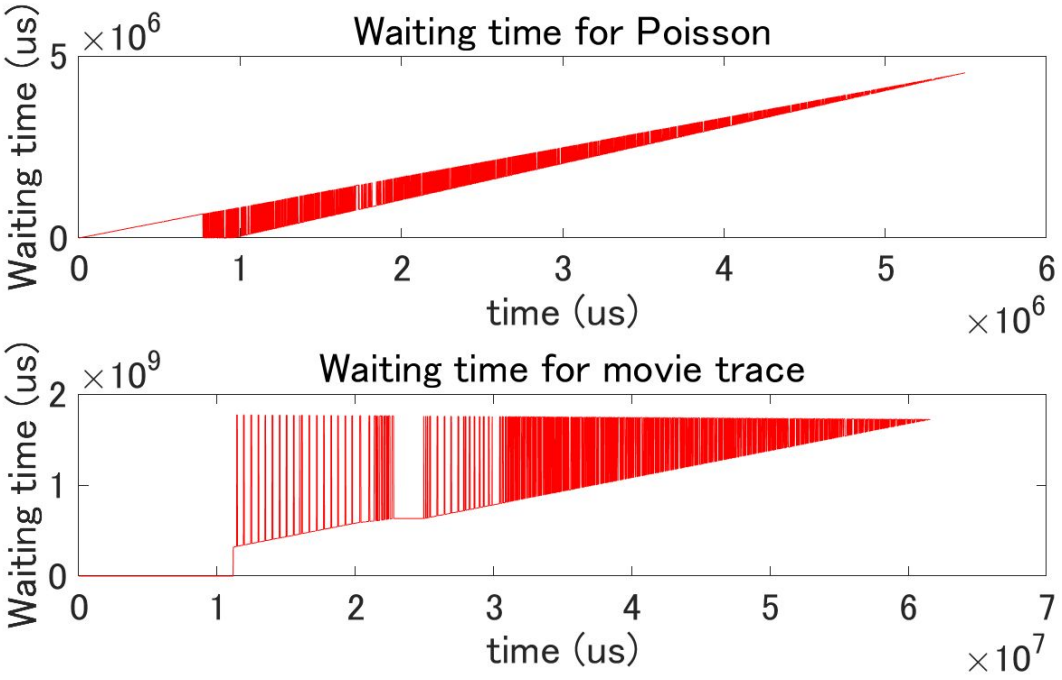

Figure 11: Waiting time of Low Priority Poisson Trace at 5 Mbps and High Priority Video Trace when N = 9

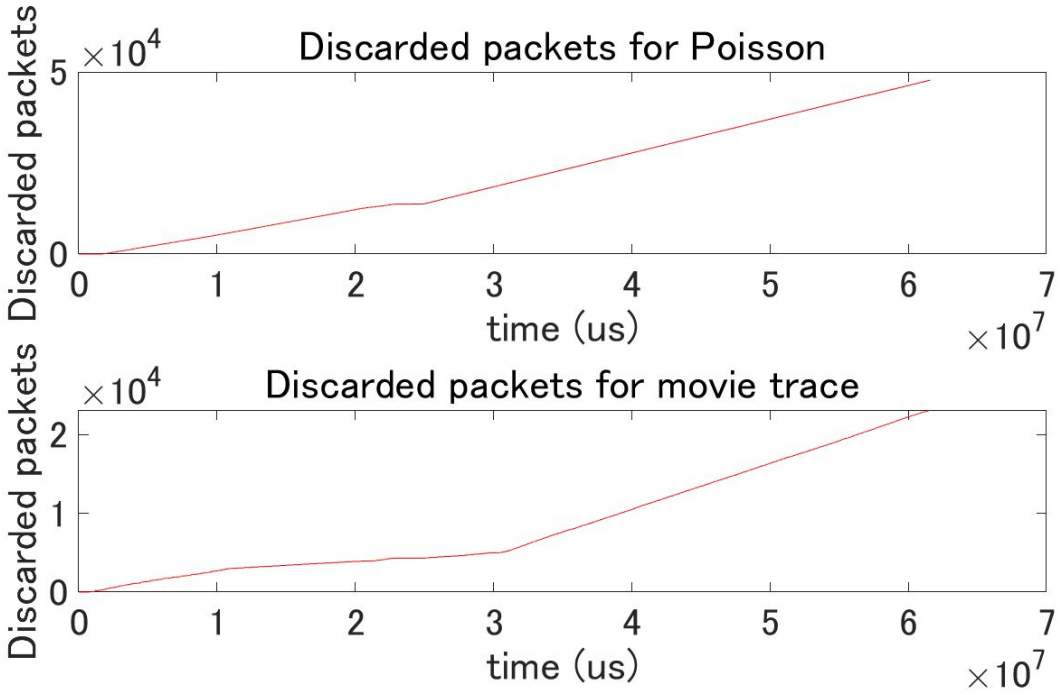

Figure 12: Number of Dropped Packets of Low Priority Poisson Trace at 5 Mbps and High Priority Video Trace when N = 9

Observations:

Analyzing figures 10 through 13 for the Poisson trace at 9 Mbps, we observe behaviour similar to that seen when the arrival rate was 5 Mbps (though more severe). Precisely we see that the backlog overflows, increasingly large waiting times and much more packet losses. These results make sense as the Poisson trace is of low priority and is now arriving at a much faster rate.

Similar to before, we observe that the video trace stays much the same (i.e. it exhibits similar behaviour to what was observed when the Poisson trace was arriving at a rate of 5 Mbps). With the video trace being the higher priority traffic flow that is always serviced (whenever its backlog is non-zero) before the Poisson trace, this similarity in behaviour to prior experiments thus makes sense.

Comparison between Exercise 1.3

The Priority scheduler deals with two incoming flows with 100kB buffers where the first flow is a high volume and high priority video flow and the second, a low priority Poisson flow whose arrival rate is varied between regimes of low to high load. As for the FIFO

scheduler in experiment 1.3, we simply observed the effects on a 100kB buffer contending with a Poisson trace that again varied from regimes of low to high load.

For backlog, Priority Scheduling has less backlogged packets, since the prioritized packets will be classified into separate buffers with different capacities, and as the transmission rate increases, it takes more time for the backlog to be filled up (i.e. compare figures 1-3 and the Poisson portion of figures 4, 7 and 10).

Regarding wait times, it will depend on the priority. For higher priority packets, it will take less time compared to FIFO, since the packets will be sent more frequently. However, lower priority packets will have a longer waiting time, since they will be “skipped” by the scheduler. Priority Scheduling requires some time for traffic classification too. The increase in waiting time also affects the possibility of packets being dropped. High prioritized packets may have less packets to be dropped compared to FIFO, however, low prioritized packets will have more packets to be dropped compared to FIFO. These observations can be seen in figures 1-3 and the Poisson portion in figures 5-6, 8-9 and 11-12.

In general, FIFO can be better when the network is used for multiple purposes, but when it is used particularly for a specific service, Priority Scheduling can increase the transmission rate for that service, by prioritizing its source as high.

NOTE: For sections 3.2 and 3.3, only the first 10,000 packets of the Poisson 3 trace were used for each traffic generator. Each figure indicates the **cumulative** number of transmitted packets and bytes for each source in the time range **0 ≤ t ≤ 100 ms**. Furthermore, for each source the quantum was calculated as $Q_i = \phi_i * min_j\, Q_j$, where Q_j = 1500 B for each of the three flows, this value was chosen so that the requirement $Q_i > L^{max_i}$ is not violated. Ultimately, with the aforementioned choice for Q_j this led to the result $Q_i = 1500 * \phi_i$ B.

As each of the three generators transmit the first 10,000 packets from the Poisson 3 trace to the scheduler receiver, this means a total of 30,000 packets should be received by the scheduler receiver. However, we observed that often **~25,000 – 27,000** packets actually make it to the scheduler receiver. This is true even after applying the T.A's suggestions to both my laptop's OS X and on Ubuntu running on a VM (i.e. we tried setting net.inet.udp.recvspace=1048576 on OS X Catalina and net.core.rmem_default=1048576 on Ubuntu).

Also note we used the stated buffer size in the lab of 100 kB for each flow.

Exercise 3.2 Evaluation of a DRR scheduler: Equal weights

N = 8 with $w_1 = 1$

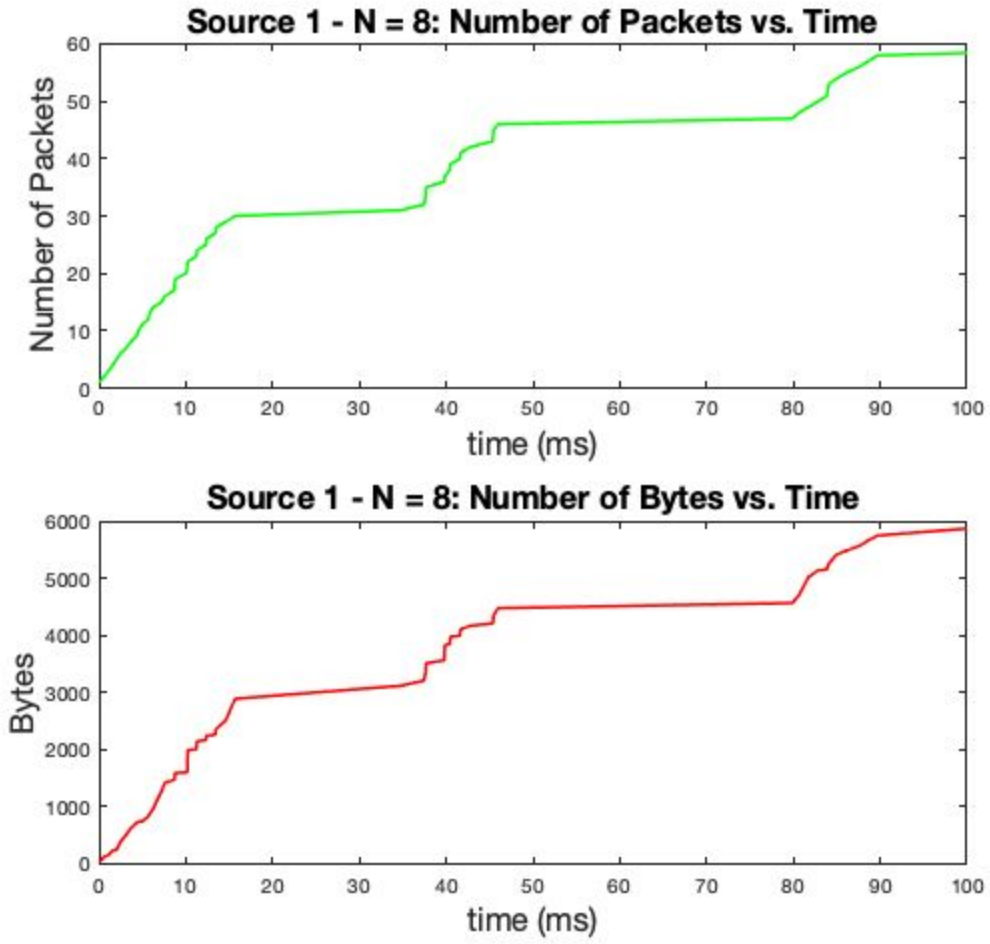

Figure 13: Poisson Trace – N = 8: Number of Packets, Number of Bytes vs. Time

N= 6 with $w_2 = 1$

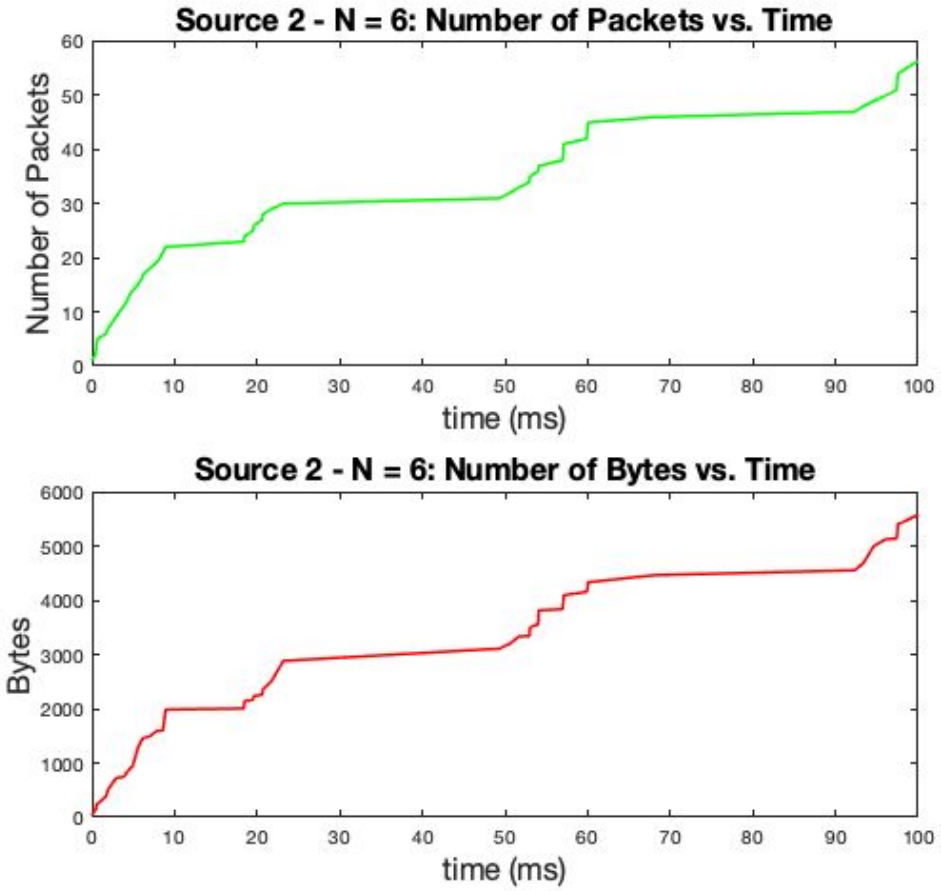

Figure 14: Poisson Trace – N = 6: Number of Packets, Number of Bytes vs. Time

N = 2 with $w_3 = 1$

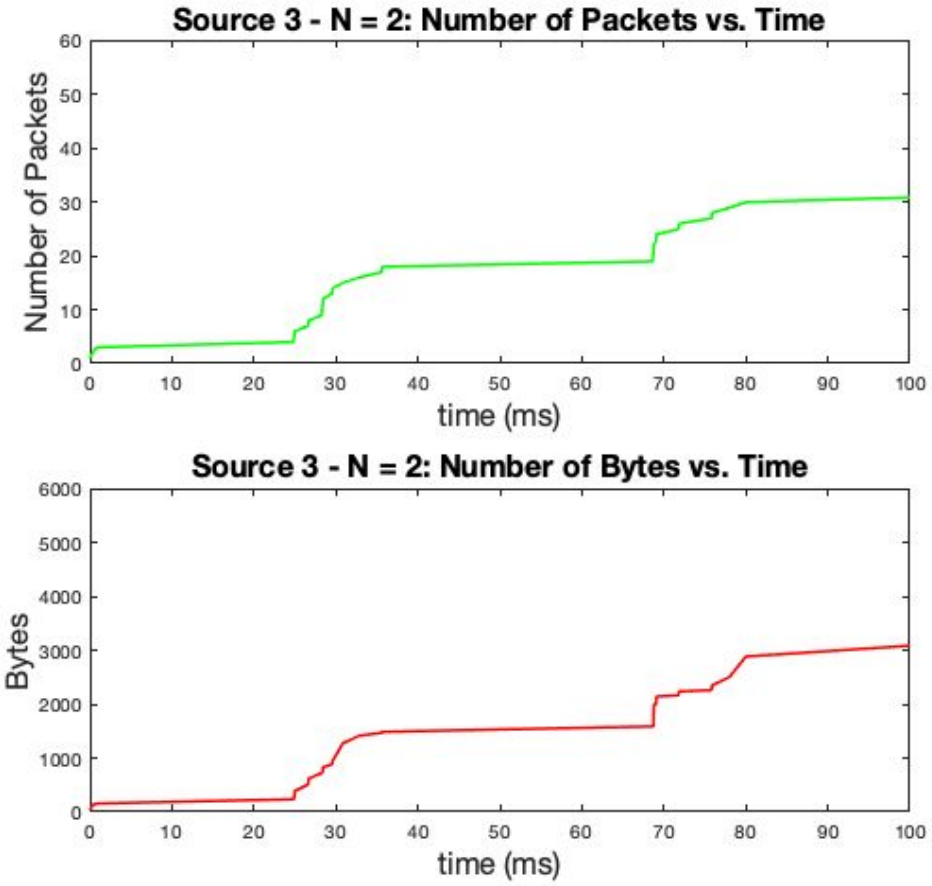

Figure 15: Poisson Trace – N = 2: Number of Packets, Number of Bytes vs. Time

Compare the plots with the theoretically expected values of a PS scheduler

From the lab documentation for this experiment, we observe that for the scenario involving three sources with equal weights having rates of 8 Mbps, 6 Mbps and 2 Mbps respectively, a Process Sharing algorithm (PS) would yield a shared bandwidth allocation ratio of 4:4:2 (i.e. 4 Mbps allocated for the N = 8 Mbps and N = 6 Mbps sources while the N = 2 Mbps source has its requested rate satisfied). Therefore, if one were to plot the number of bytes and packets transmitted in this idealized scenario, the N = 8 and N = 6 source should have identical plots whereas for the N = 2 source, it should have half as many bytes and (possibly) packets transmitted.

For this experiment, we used Deficit Round Robbin (DRR), which is at best an approximation of the idealized PS algorithm when weights of all traffic flows are equal. Observing figures 13 and 14 above, we see that the number of bytes and packets transmitted over time for flows N = 6 and N = 8 is similar. We also see that the aforementioned flows tend to nearly identical values for the total number of bytes and packets transmitted at the 100 ms mark. These observations suggest that the N = 6 and N = 8 flows have nearly identical link bandwidth allocation. As for the N = 2 case, observing figure 15 above, we see that the number of bytes and packets transmitted overtime is close to half of that in the N = 8 case and almost half of that for the N = 6 scenario. Accordingly, this observation suggests that the link bandwidth allocation for the N = 2 flow is approximately half of that allocated to the N = 8 and N = 6 flow.

Altogether, the above results suggest that the DRR in this context is a reasonable approximation of the PS scheduler, as it nearly achieves a shared bandwidth allocation ratio of 4:4:2.

Exercise 3.3 Evaluation of a DRR scheduler: Different weights

N = 8 with $w_1 = 3$

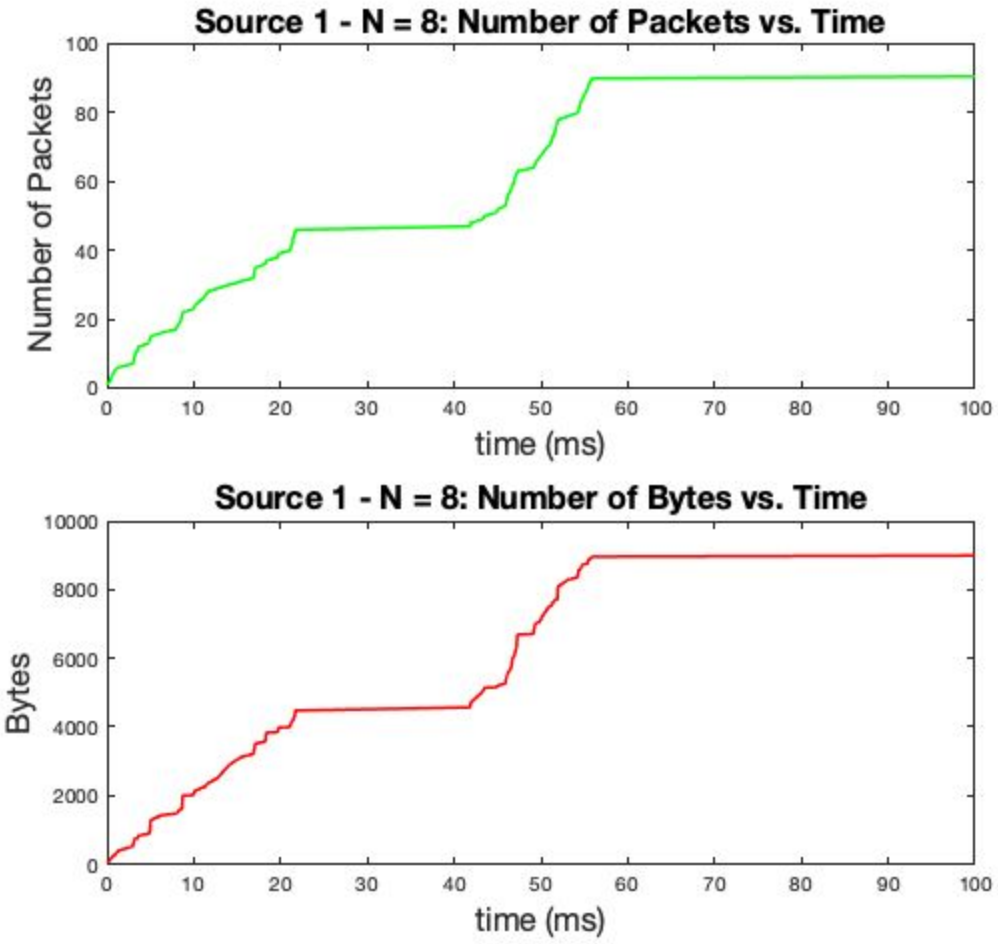

Figure 16: Poisson Trace – N = 8 with w_1 = 3: Number of Packets, Number of Bytes vs. Time

N= 6 with $w_2 = 1$

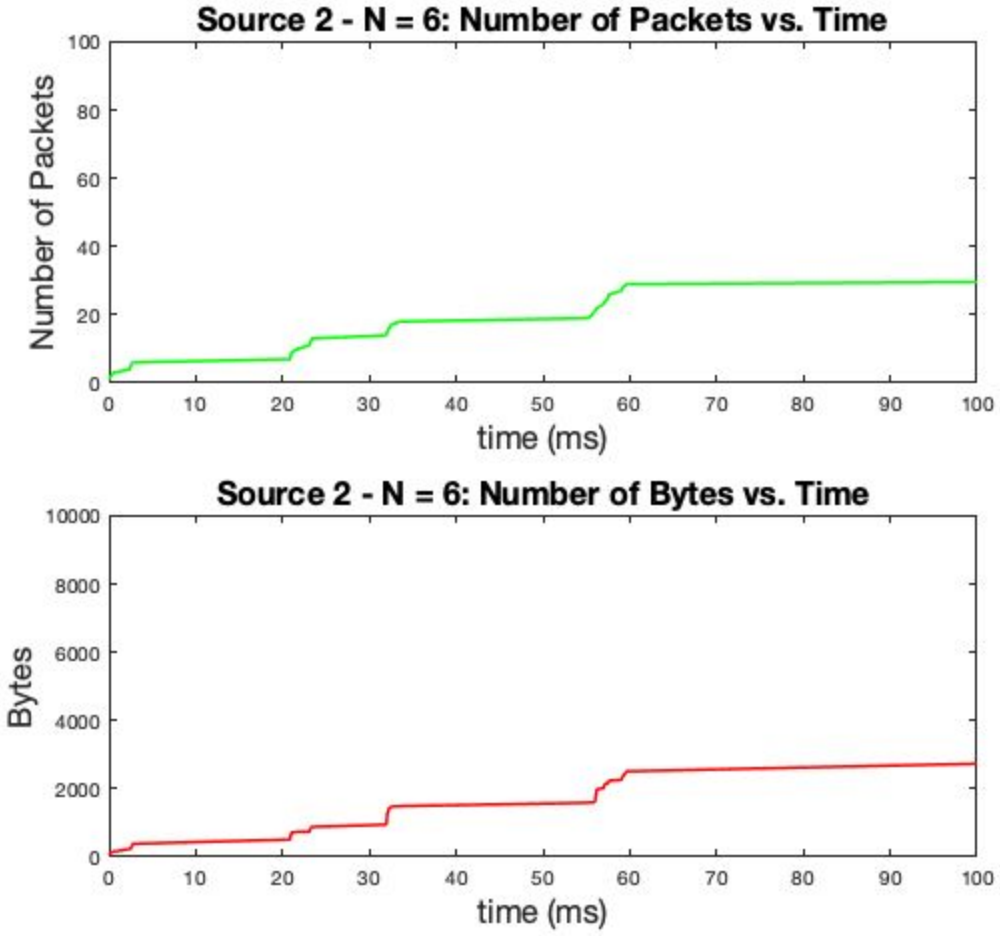

Figure 17: Poisson Trace – N = 6 with w_2 = 1: Number of Packets, Number of Bytes vs. Time

N = 2 with $w_3 = 1$

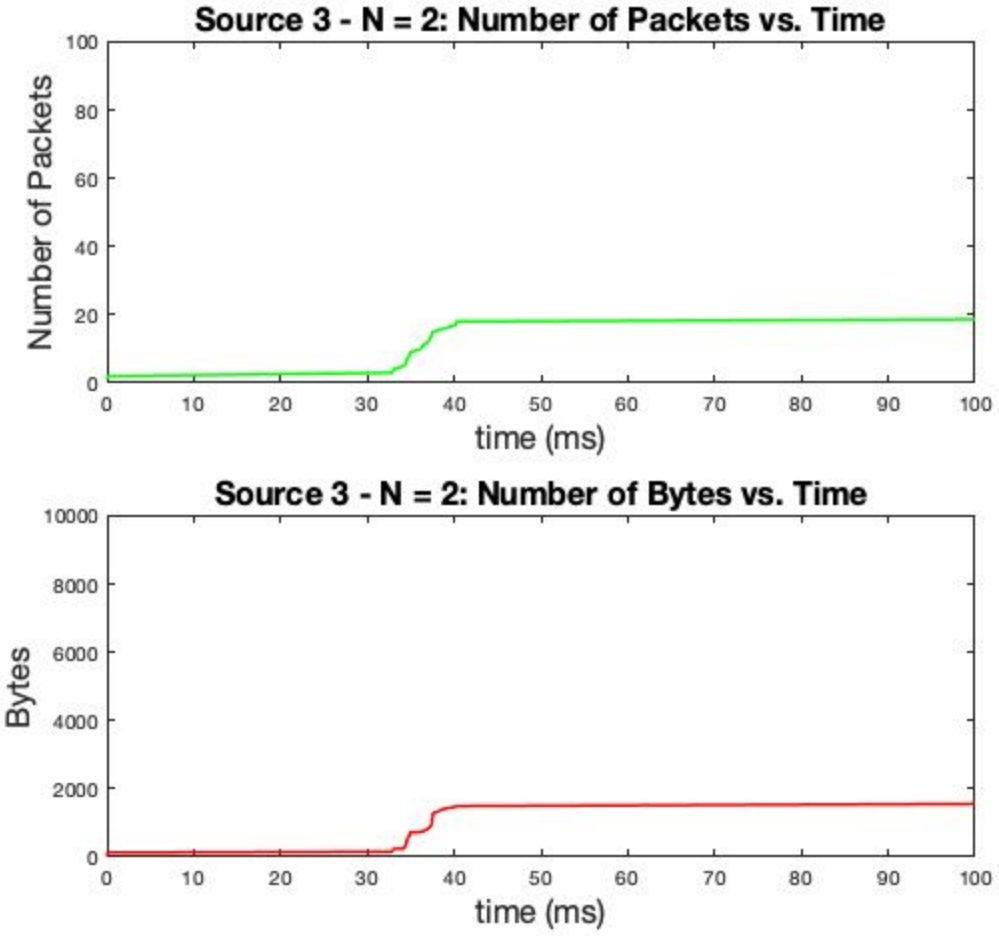

Figure 18: Poisson Trace – N = 2 with w_3 = 1: Number of Packets, Number of Bytes vs. Time

Compare the plots with the theoretically expected values of a GPS scheduler

Following a similar discussion as in section 3.2, we keep all else the same except that now the three sources are no longer equally weighted. Specifically, the N = 8 source has a weight of 3 while the N = 6 and N = 1 sources are equally weighted at a value of 1. From the perspective of the Generalized Processor Sharing algorithm (GPS), the flows would now have a shared bandwidth allocation ratio of 6:2:2 (i.e. 6 Mbps allocated for the N = 8 source while the N = 6 and N = 2 sources are both allocated 2 Mbps). Accordingly, if one were to plot the total number of bytes and packets transmitted for the

three flows, we would see that the plots for N = 6 and N = 1 would be identical while the N = 8 plot would have 3x more transmitted bytes and (potentially) packets.

Similarly, we use DRR to approximate GPS when flows have unequal weights. Comparing figure 16 with figure 17 it appears there is ~4.5x as many packets and bytes transmitted in the N = 8 flow than there is for the N = 2 flow. Similarly, comparing figure 16 and figure 18, there is ~3x as many packets and bytes transmitted in the N = 8 flow than there is for the N = 6 flow. Finally, comparing the N = 6 and N = 2 flows (see figures 17 and 18), we observe that the difference in cumulative number of bytes and packets transmitted over time between these two flows is very small and in some instances they are equal. Altogether, these observations suggest that the shared bandwidth allocation of the N = 8 flow is either ~3x or ~4.5x larger than the other two flows, where each of these flows (i.e. N = 2 and N = 6) have very similar bandwidth allocations.

Hence, the DRR approximation seems to suggest a shared bandwidth allocation that appears to be very similar to that proposed by the idealised GPS algorithm (i.e. 6:2:2).