PA5 Report

Gather timing data on the same set of n, b, w, m arguments on each of PA5 FIFO and PA5 MQ. Note that you need to modify the default capacity of the message queue to test out different m values that are larger than the default.

n = 15000, b = 500, w = 20, p = 1

m	runtime
256	1.917562
512	1.916941
1024	1.916853
2048	1.916755
4096	1.916211
8192	1.915946

Table 1. Runtime with different message buffer capacities on FIFO

n = 15000, b = 500, w = 20, p = 1

m	runtime
256	1.990344
512	1.984391
1024	1.974433
2048	1.969677
4096	1.953864
8192	1.949611

Table 2. Runtime with different message buffer capacities on MQ

Present a performance comparison of the different IPC mechanisms based on this data and attempt to provide explanation for any differences and similarities.

By using two different IPC mechanisms, we collect the timing data on the same set of number of requests per patient, number of worker threads and capacity of the request buffer while varying the capacity of the message queue as above tables. From the tables, we can tell that both mechanisms show a general decrease in the runtime as the message buffer capacity increases from 256 to 8192; however, the trend of decreasing is more dramatic in MQ mechanism. Also, the runtime by using MQ method takes a little bit longer than that by FIFO method. In other words, it seems like FIFO mechanism is a more efficient way to run this program.

Present the results in separate graphs using PA5 FIFO (i.e., PA4) as the baseline for comparison.

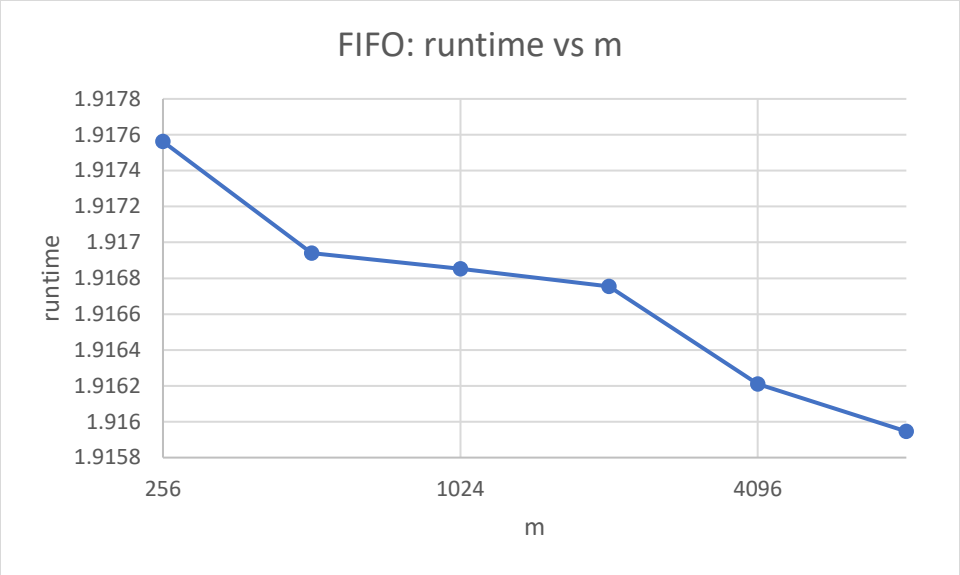

Diagram 1. FIFO: runtime vs. m

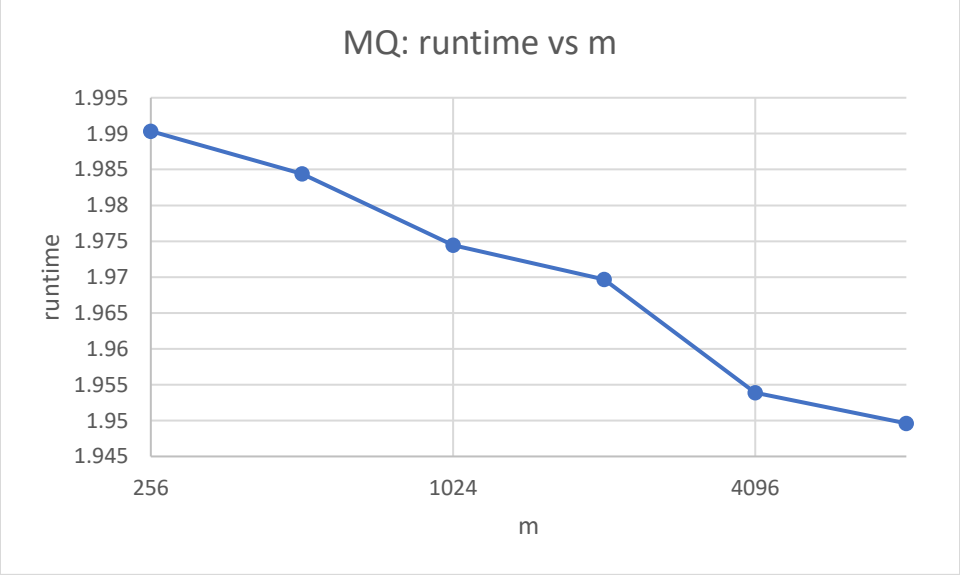

Diagram 2. MQ: runtime vs. m

What is the maximum w and thus the max number of RequestChannels that you can use for each IPC? How much more can you go beyond the limit in PA4?

In my case, the maximum w is 127, which means I can use 127 worker threads for the request channels in each IPC. In PA4, the maximum w is 2007, which is 1880 greater than IPC methods used in PA5.

What are some of the limits encountered by each class, either due to the specific implementation or to operating system limitations, and how does the program behave when it encounters them?

As for my OS, the maximum of the number of worker threads is 127. When the input number of worker threads exceeds the maximum, the program will break and output the error message "/mq_data128_1: No space left on device" in the terminal. Similarly, there is a limit for the message buffer capacity. When the massage buffer capacity exceeds its maximum, it will output "/mq_control: Invalid argument" in the terminal. In some cases when the input is inappropriate, it might also lead to segmentation fault and thus breaking the program.

Describe the clean-up activity you have done for each IPC

To remove the shared memory segment, you could type in the command line "ipcrm shm <shmid>". For instance, "ipcrm shm 5" can remove the shared memory segments allocated in shmid 5.