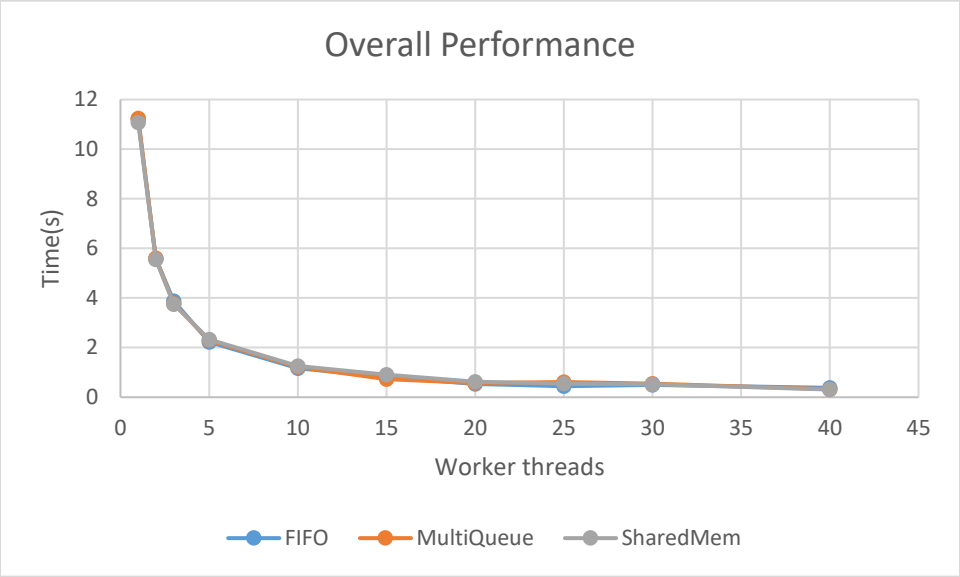
David Qin
PA6: IPC Mechanisms
Dr. Ahmed
11/29/18

Graph:

Using a buffer size of 6, and 1000 requests:



The time needed was about the same for all three types of IPCs. I believe a difference would be more noticeable if I used more threads and more requests, but I was doing this on c9 and I wasn't sure if it could handle it.

FIFOs could only have up to 509 worker threads. MQ and SHM should be able to have more than that but c9 cannot handle it.

Limitations:

FIFO: Easy implementation. Uses files for reading and writing. However it creates tons of files, and does have a limit as to how much it can create. Also relies on using the disk, which means it'll be slower than the other two.

MultiQueue: Similar to FIFO, except we just use ftok(), letting us not have to use multiple files. This means we can create as many queues as the OS can handle. Also stores in memory so it's faster than FIFO.

SHM: Uses bounded buffers to do things. Needs synchronization but is faster than MQ.

Cleanup:

The channels will be deleted after all threads are done. All cleanups will be performed in their respective deconstructors and the function call msgctl(). In order for SHMs to clean up, we must clear both the reading and writing buffers by deleting their ids. We also need delete semaphore objects. A similar thing must be done for the MQs where we delete the ids for the writing queue and reading queue. In FIFO we close the writer and reader file descriptors and remove the files.

Bonus:

I was able to use only one file with ftok(). I used the same file but different project ids in order to generate unique keys. The id was got from the channel name. For example, both the server and client use 11 as their project id for data1_1 and 12 for data_1_2. Control channel uses 9991 and 9992, they will create the same key. Then when they create a MQ/SHM object they will be using the same mqid/shared memory id.