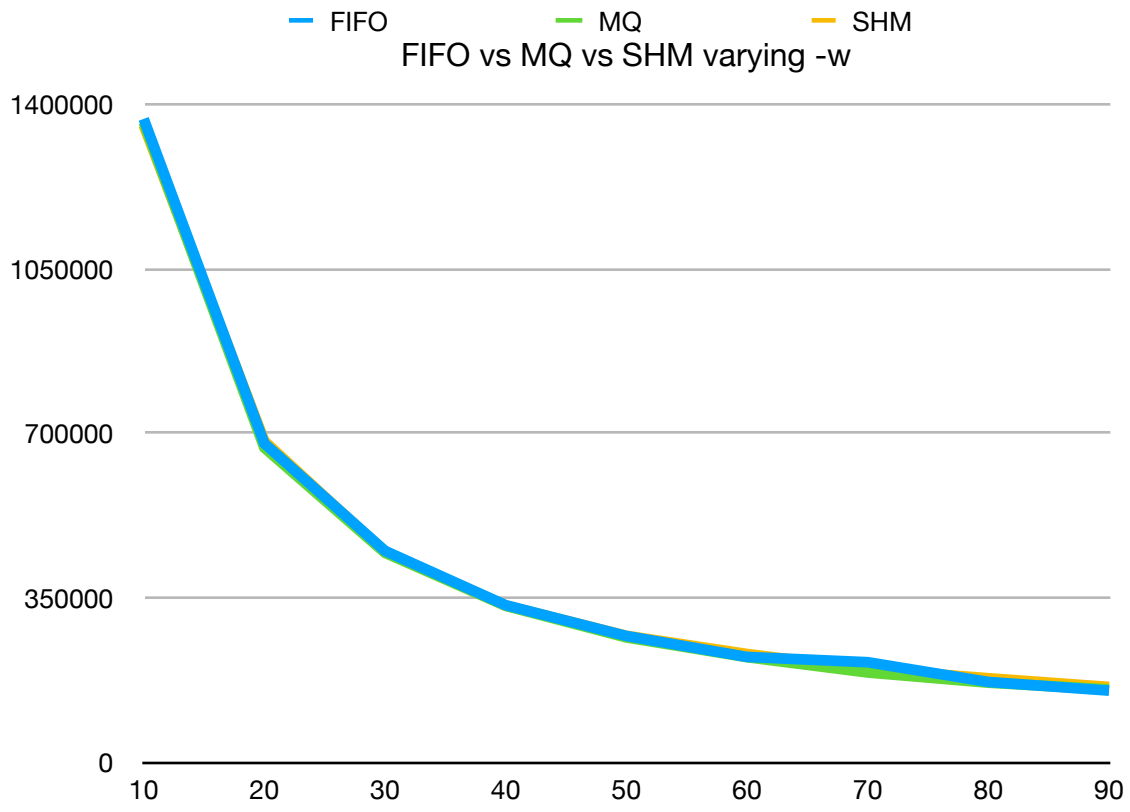
Rahul Shah PA5 Report

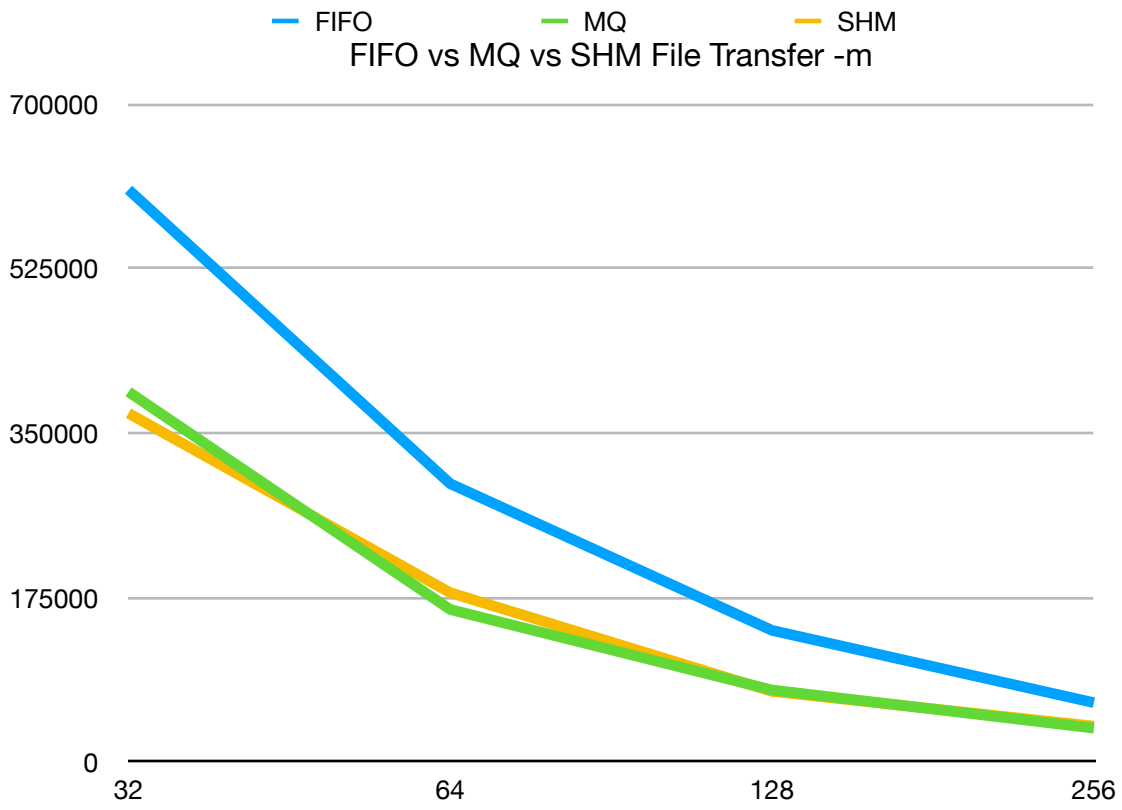
Introduction:

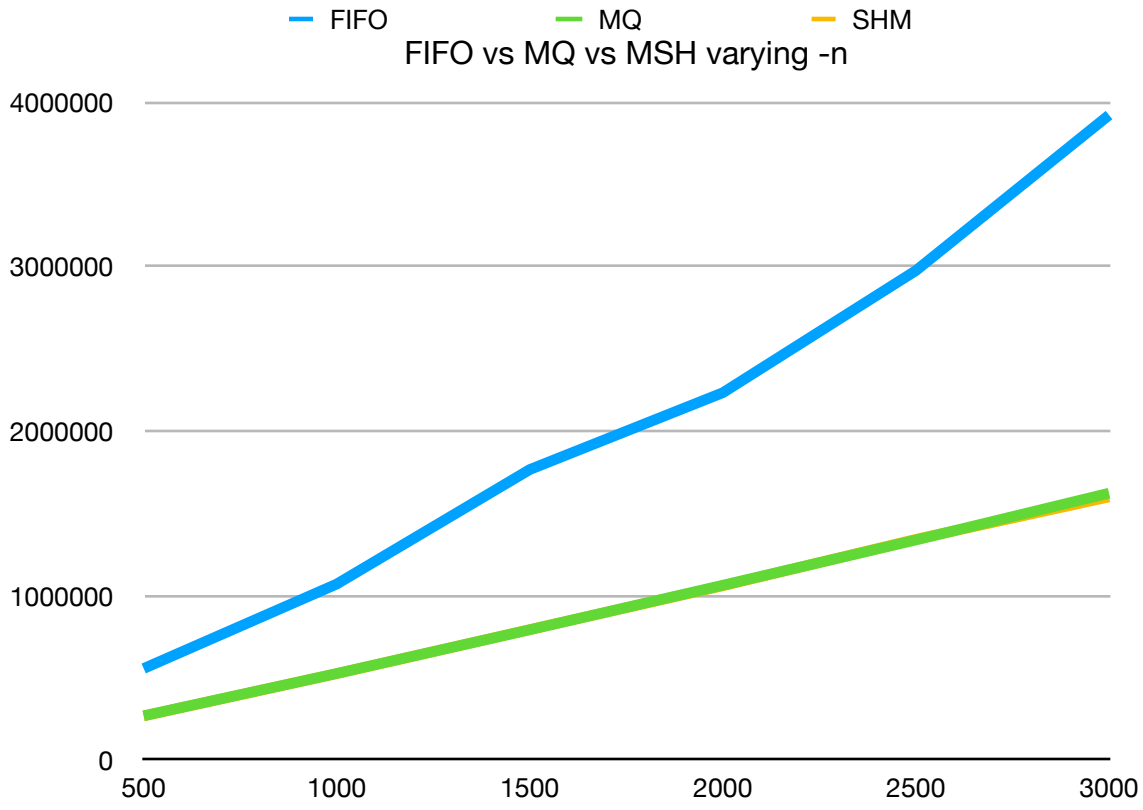
In this PA, we implemented properly structured data requests to a server via a pipe-based IPC (Inter-Process Communication) protocol. There are 3 kinds of messages , a “data message”, “file message”, and “new channel” message. These 3 kinds of messages allow the client to request a specific data point from a file, an entire file transmitted via fixed size buffer, or request the creation of a whole new communication channel to communicate across. We utilized the power of multithreading to parallelize our system. We have implemented multiple IPC protocols- FIFO and Message Queues.

Data Request Timing Data:



Rahul Shah PA5 Report





Analysis:

1) According to my analysis, message queues and shared memory provide increased performance in multithreaded scenarios when compared to FIFO pipes. A performance comparison of message queues against PA4's FIFO pipes shows that the message queue performance graph is vertically shifted downwards, showing that the speedup does not vary with the number of worker threads. The power of message queues is more apparent during file transfers. FIFO requires data to pass through the kernel, which requires expensive context switches, unlike Shared Memory and Message Queues. Shared memory is faster than Message Queues, as messages need to be allocated and written once in the shared memory space.

2) For message queues, I can have a maximum of 127 worker threads before receiving a "not enough memory" error. However, the number of worker threads is limited by the OS's limit of maximum open file descriptors, which can be manually raised such that thousands of worker thread are able to run simultaneously.

3) The main limitations of FIFO is that the size of the buffer is limited by the kernel, as all transmitted data is stored in kernel memory. Thus, there can be portability issues across different machines. Also, FIFO pipes send raw bytes, while message queues are able to send more complex data structures, allowing for more advanced consumption. FIFO's are limited by the OS's open file descriptor limit. Message queues allow for asynchronous messaging between processes. Shared memory have the issue of synchronization, thus named semaphores must be used to protect the shared memory space.

4) To clean up MQ, first you must close the file descriptors, then unlink the thread from the MQ. For FIFO, you must first close the file descriptors, then close the pipe.