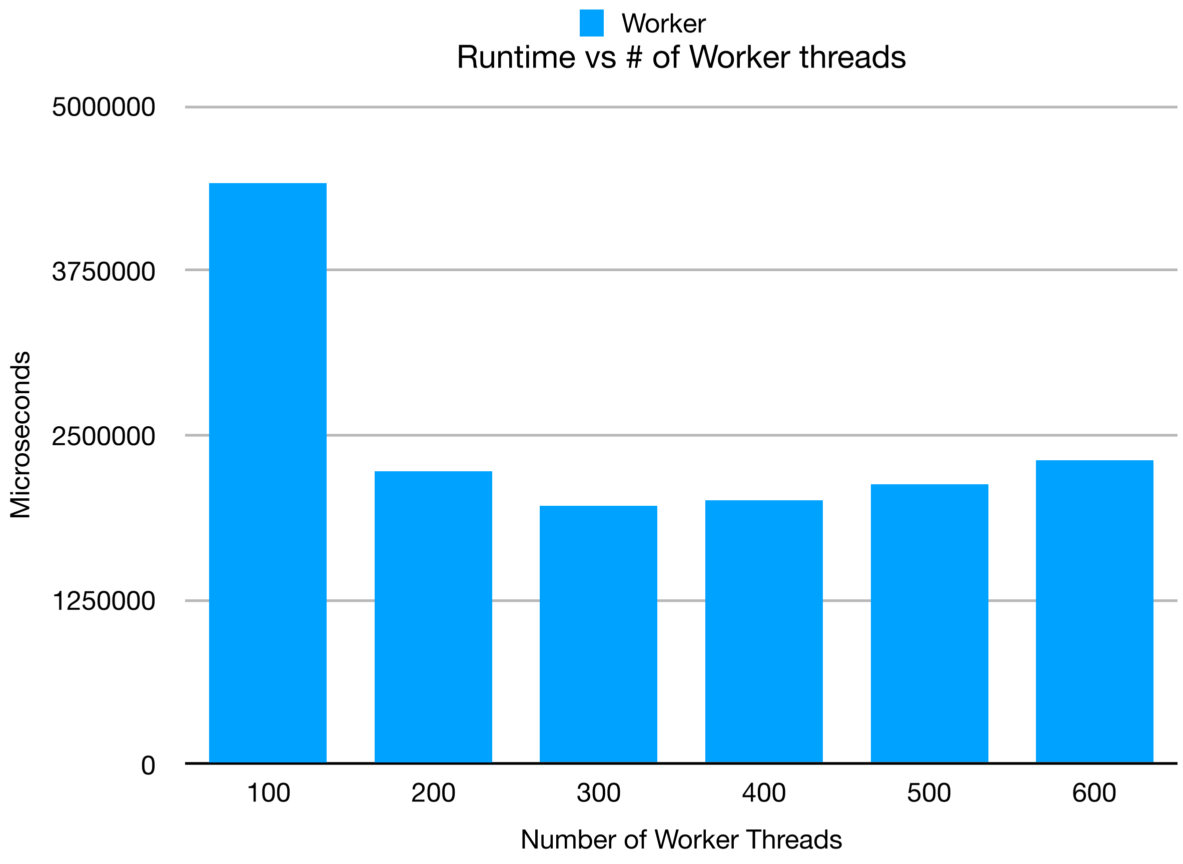
Rahul Shah PA4 Report

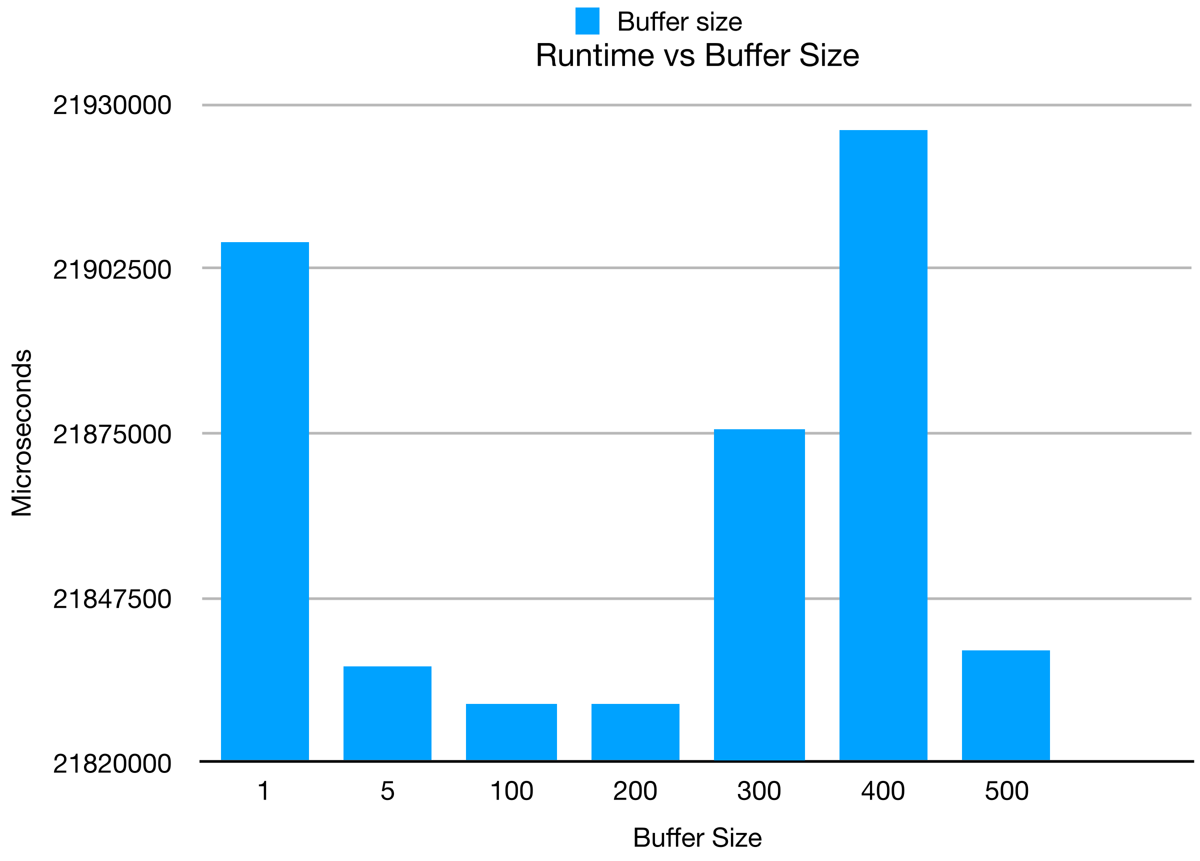
Introduction:

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.

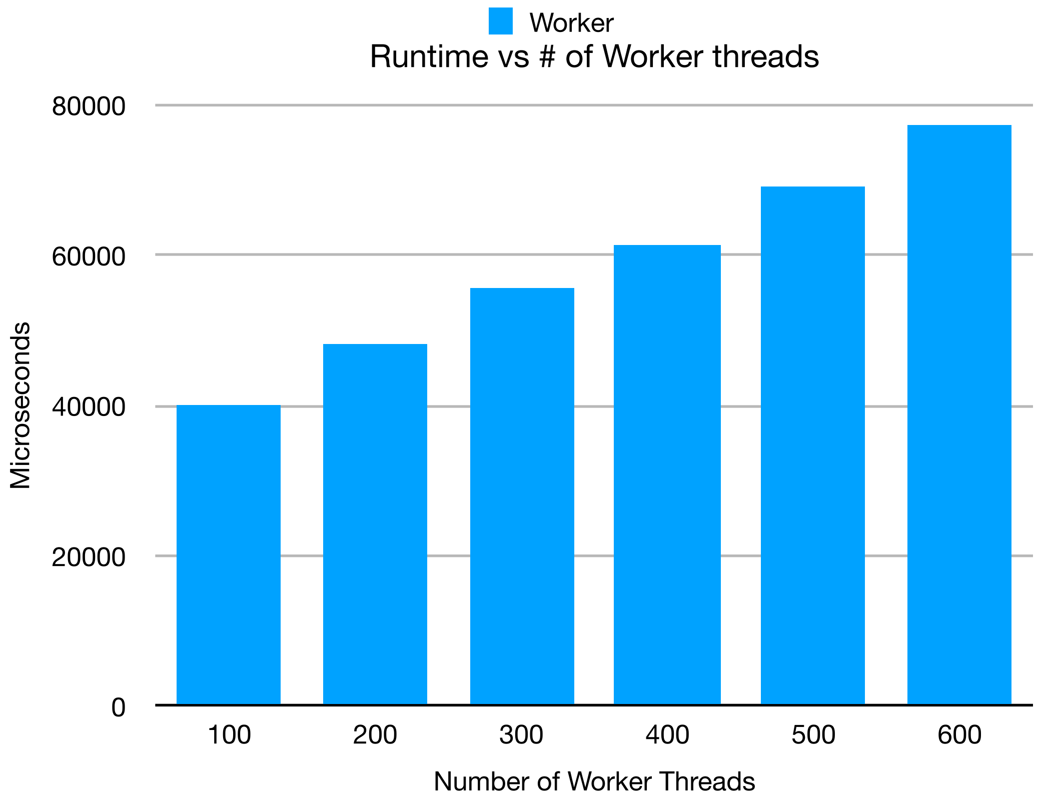
Data Request Timing Data:

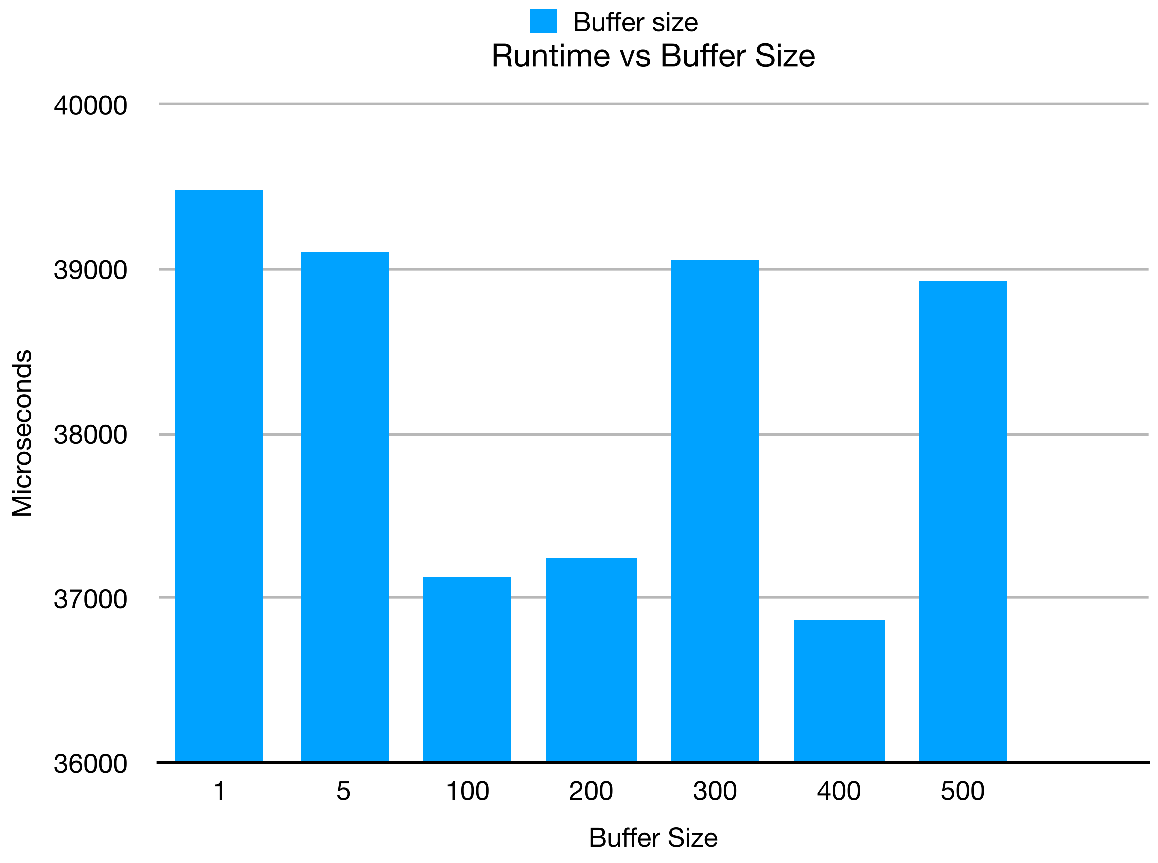


Rahul Shah PA4 Report



<u>File Request Timing Data:</u>





Analysis:

1) Increasing the number of worker threads seemed to increase runtime in the long run. This is because the worker threads are not all being executed in true parallel. Parallelization is limited by the number of CPU cores, thus an increased core count

beyond that point is actually detrimental, as the number of thread switches increases linearly with thread count. Therefore, a high number of threads will decrease performance.

2) There seems to be no influence on buffer size and performance, as runtime seems to fluctuate randomly. This is because beyond a certain point, the size of the buffer does not become a bottleneck in the system as the requests are being ingested by the workers at a sufficiently fast rate. We can see that beyond a buffer size of 200, we get seemingly random fluctuations in runtime. However, for buffer sizes between 1 and 200, there seems to be an exponential decrease in runtime.

3) For file messages, increasing the number of worker threads is actually a detriment to performance. Our results show that runtime increases linearly with the number of threads. This is because the number of thread switches increases as we create more and more workers, thus slowing down the system.

4) Buffer size does not speed up the system significantly after a certain point for file messages. This is because the worker threads are ingesting requests quickly enough to not hit the buffer capacity maximum. We can see by the graph that a buffer size of 100 is ideal for the system, beyond that the performance varies wildly.