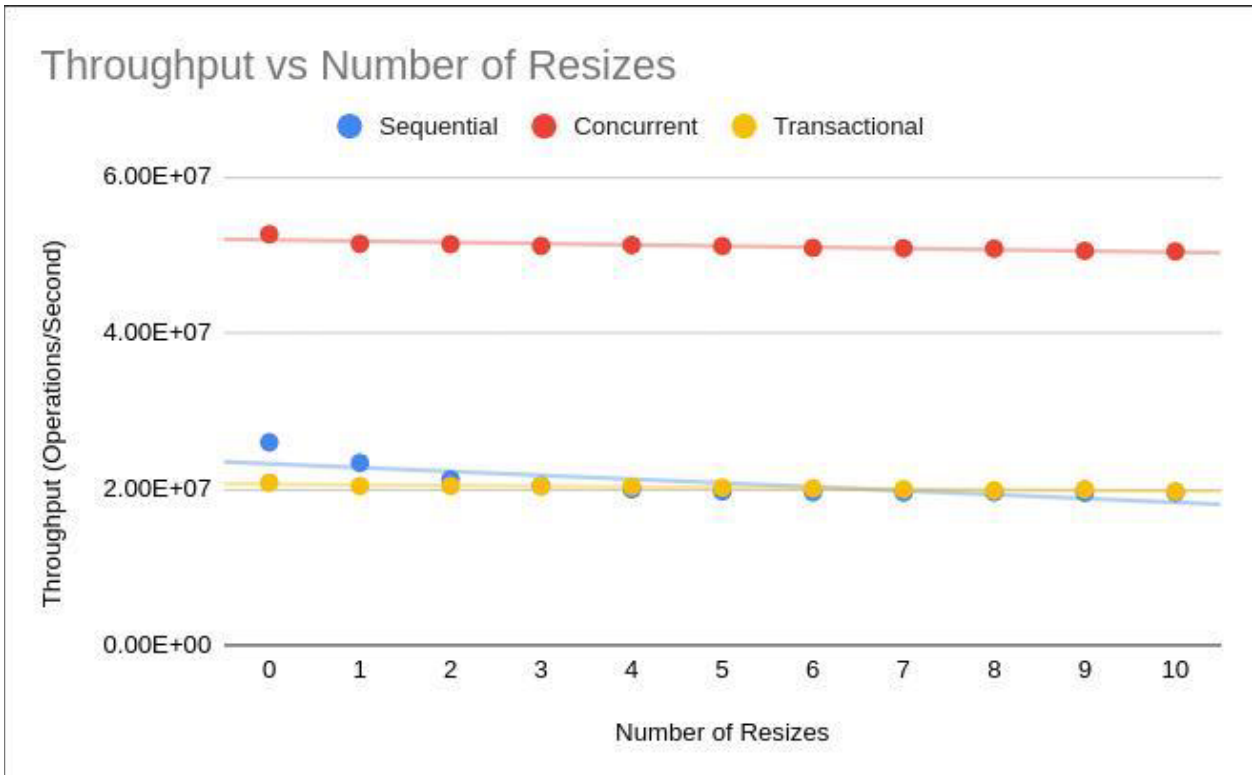
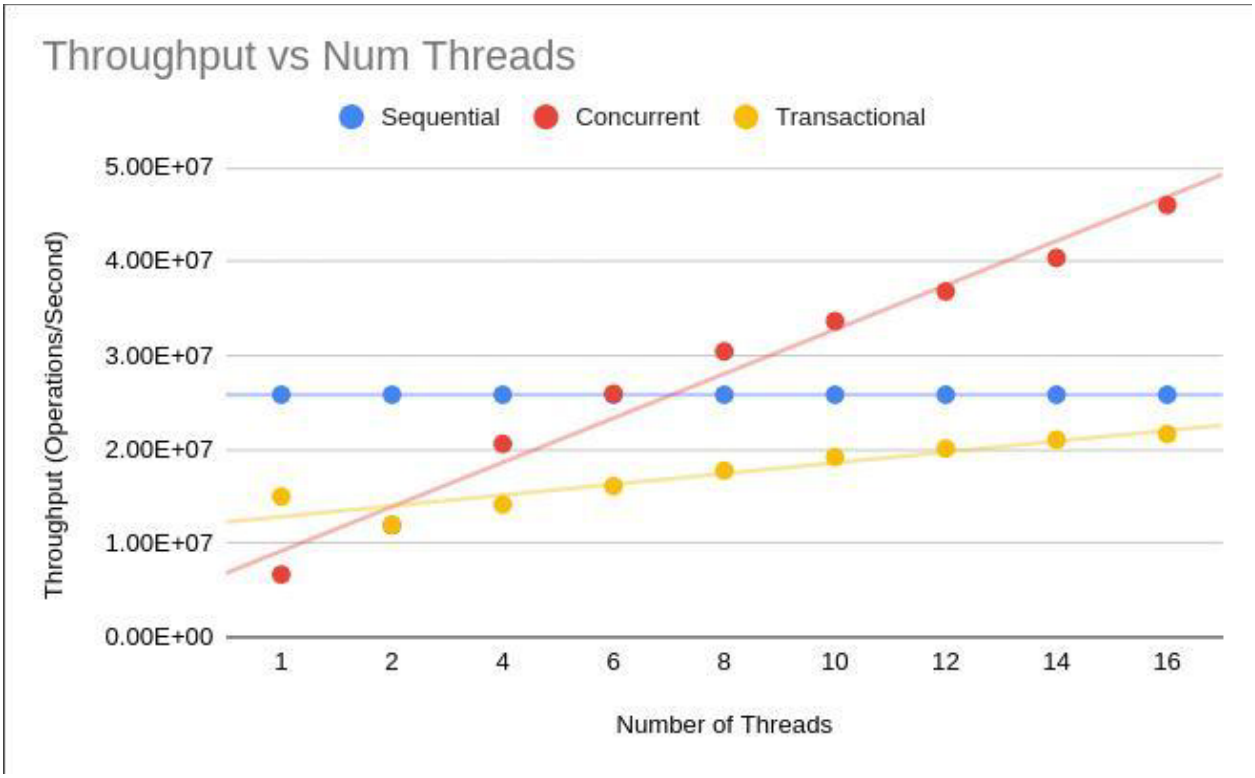


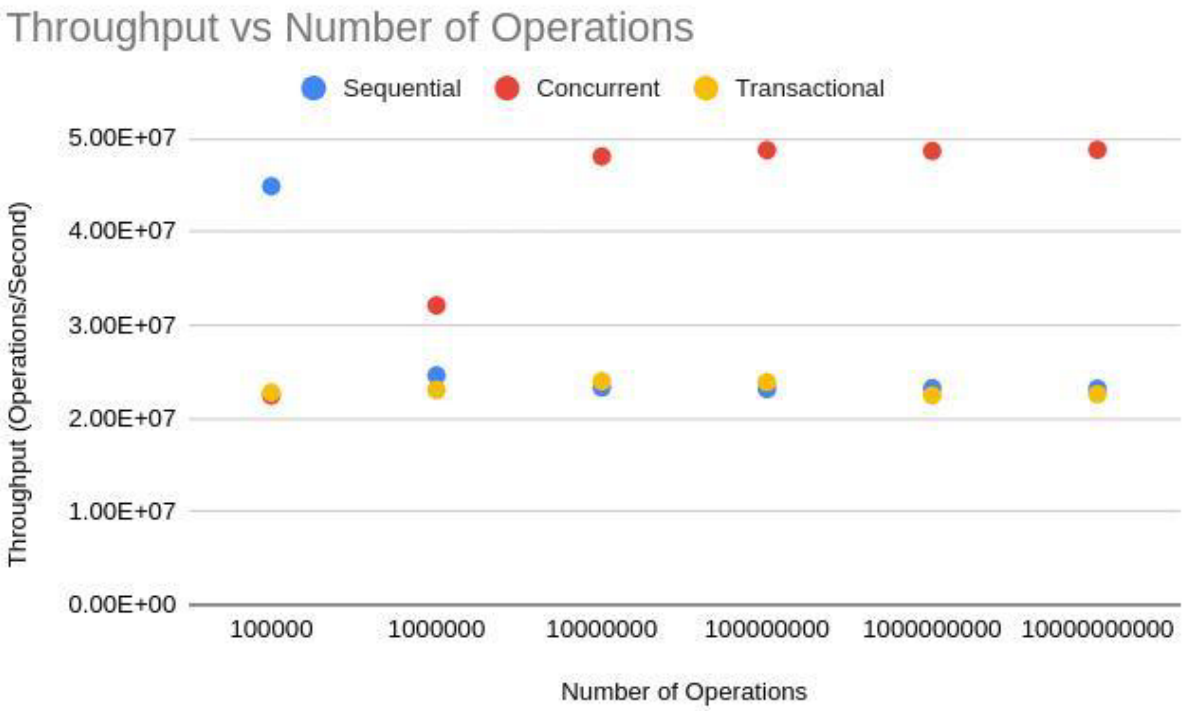
num_ops = 500000000, max_val = 1000000, capacity = 16, num_threads = 16;

Note that these resizes happen after populating each set. Each curve has a slight but certainly downward trend. I expected the performance degradation from increasing the number of resizes to be much more significant, especially in the concurrent and transactional implementations. It stands to reason that many more resizes would need to kick in to degrade performance at the expected exponential level. I would have liked to collect better data (more resizes), but it would have taken me a very long time as the only way I was able to benchmark this was by manipulating the size of the initial population. I'm sure I could have come up with a slightly less consistent but faster way to get this data.



int num_ops = 50000000, num_elms = 1000000, max_val = 100000000, capacity = 100000000

The behavior described by this plot is expected and not very interesting. I am very pleased that the transactional version was able to get so close in performance to the sequential version. I would have liked to make the concurrent version faster, but I did not have the time to enjoy this assignment as much as I would have liked. Overall, I am happy with my results.



num_elms = 1000000, max_val = 1000000, capacity = 16000000, num_threads = 16;

I thought it would be interesting how throughput changed as the number of operations increased. It stands to reason that the concurrent version would increase in throughput as operations are added until it hits an asymtote. I would have thought the sequential version to be more consistent. I also am interested to see that the transactional version is so consistent. I would have thought that it would behave more like the concurrent version.