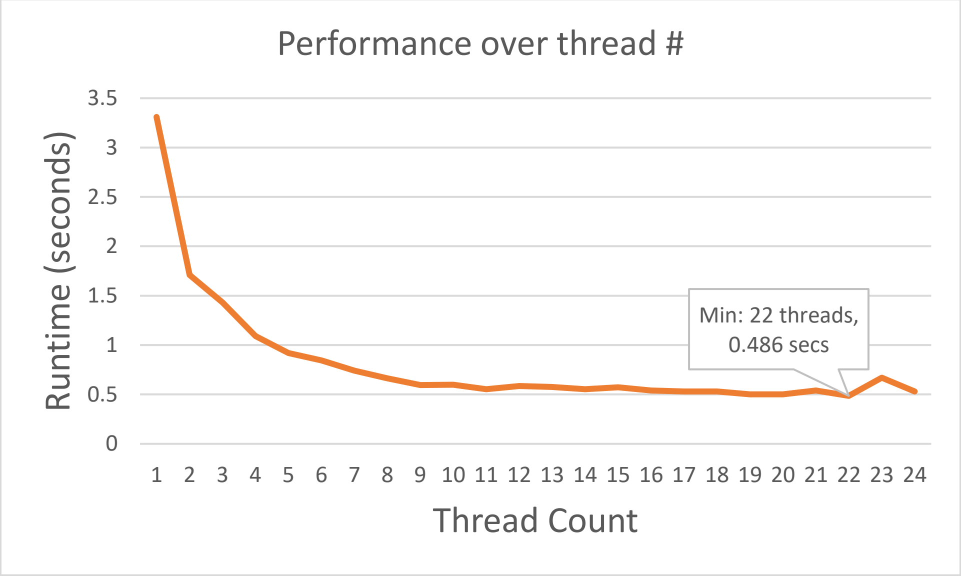
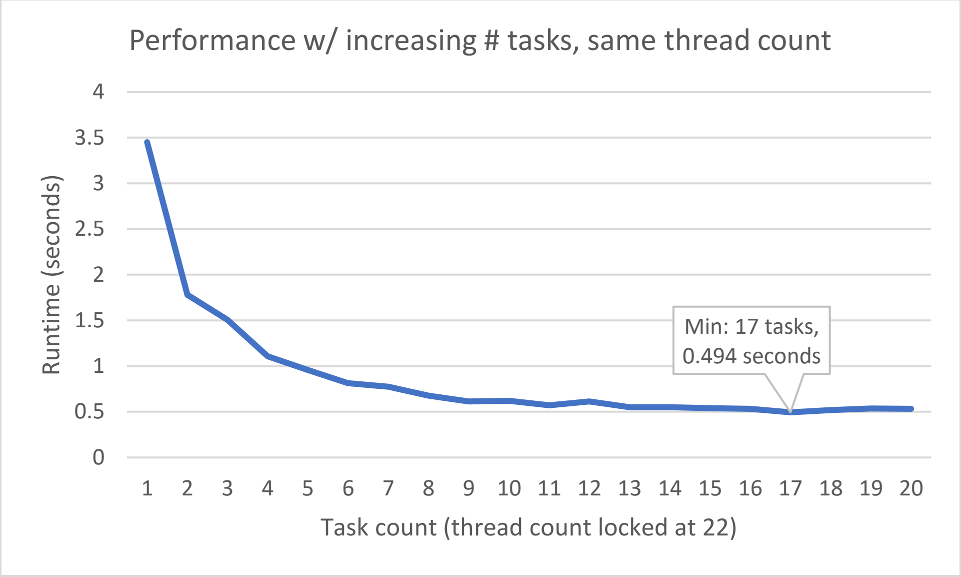
Project 3 Evaluation and Report

Operating Systems Principles
Matthew Phelps

1. Plot of fastest runtimes for fractalthread depending on number of threads utilized:



2. The optimal number of threads for fractalthread was 22 threads, however, the optimization potential as thread number increased plateaued quickly. Any runtime reductions from thread counts of 9 to 24 were indistinguishable.
3. As thread count increases initially, runtime is significantly reduced. But with larger thread counts, the overhead required to manage 22 threads and smaller and smaller marginal reductions to runtime balance each other out. So there is no need to try and maximize thread count. Further, though the graph does not show this, a higher number of threads might actually reduce runtime because at that point the overhead outweighs the reductions in runtime by splitting up work.
4. Plot of fastest runtimes for fractaltask with 22 threads depending on number of tasks assigned:



5. Likewise to fractalthread, the above graph highlights the benefits related to dividing up work beginning to plateau as task count increases. Thus, there is no need to try and maximize the number of tasks that the threads are assigned to. In fact, with a greater number of tasks (not shown by this image), splicing up the work would actually slow down runtime! A balance must be struck.