Jess Conway

February 12, 2018

COMP 460; Sekharan

Homework 2

I have performed both of these tasks using static scheduling of threads. To my understanding, static scheduling is best used in situations where all threads essentially perform the same task, while dynamic scheduling is best suited for situations where the workload may vary depending on which thread one is analyzing. For that reason, static scheduling seemed best for these tasks because I used a very basic algorithm for each, and simply partitioned these algorithms into smaller parts when using multiple threads.

I have attached code and graphs for each problem and associated graphs which compare various runs of the programs using different sized data sets and different numbers of threads. These are labelled as appropriate.

One will notice from the matrix graphs that the speedup from multithreading is most obvious when multiplying very large matrices. This is appropriate, as larger matrices, even if sped up by the same factor as smaller matrices, would allow for a more notable effect on time-decrease.

Develop and implement multi-threaded Java (or Python) code for factoring large integers. You can use the straightforward algorithm with some simple efficiency improvements as discussed in class. You must first make a conscious choice as to which type of scheduling you will use, static or dynamic and write the reasoning for it before you implement it. Your reasoning is important in both self-assessment as well as for me to gauge your level of understanding of the material. Also plot some graphs involving run-times, threads used and size of the integer in some suitable form that portray your work in the best light. That is quite a bit of freedom, flexibility, and yes, responsibility. Reason wisely! (Yes, You will use cs460 via Guacamole to benchmark the code.)