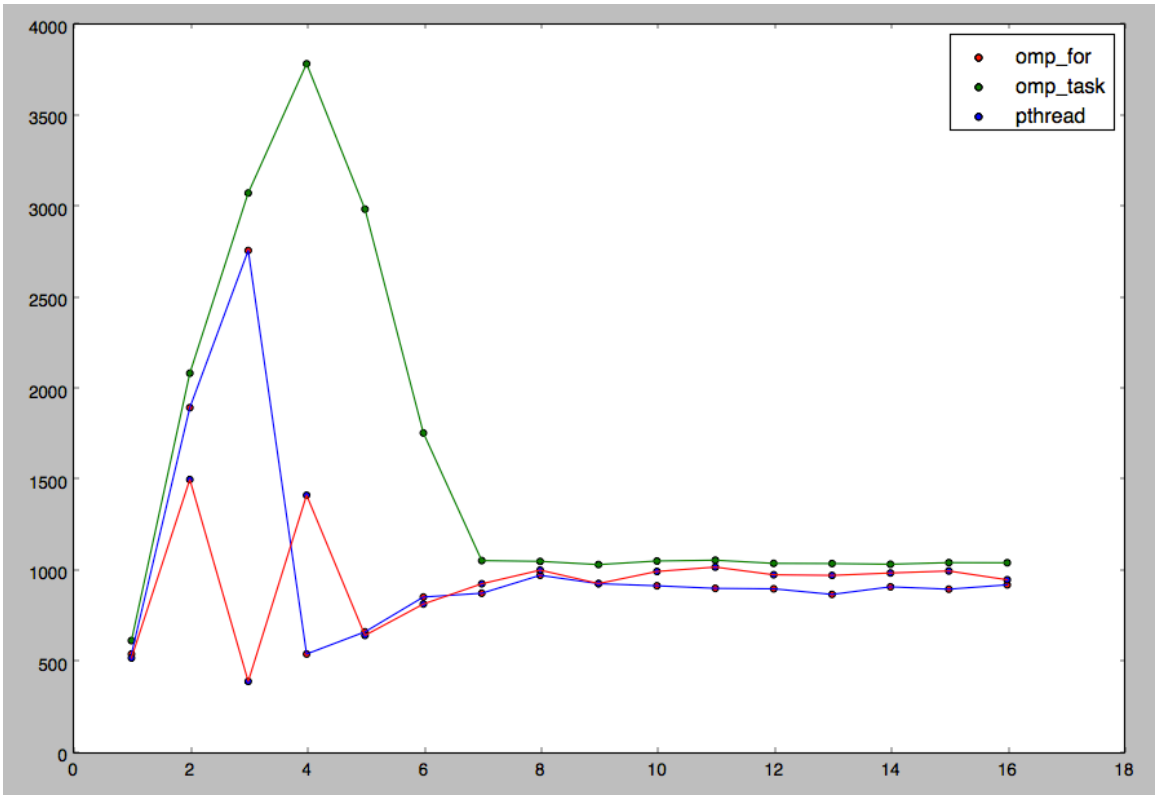
Nikhil Unni
UC Berkeley

Assignment 2

3.2.1



3.2.2

My fastest was with 4 threads with omp_task, which was about 3x faster than the scalar baseline. Obviously there are fluctuations, but I think it's reasonable to say that the omp_task implementation achieved an almost linear speedup. I'd say any big hindrances are the startup time to spin up threads, or an asymmetrically faster performance on some tasks than other tasks. My guess is that there could be better cache performance.

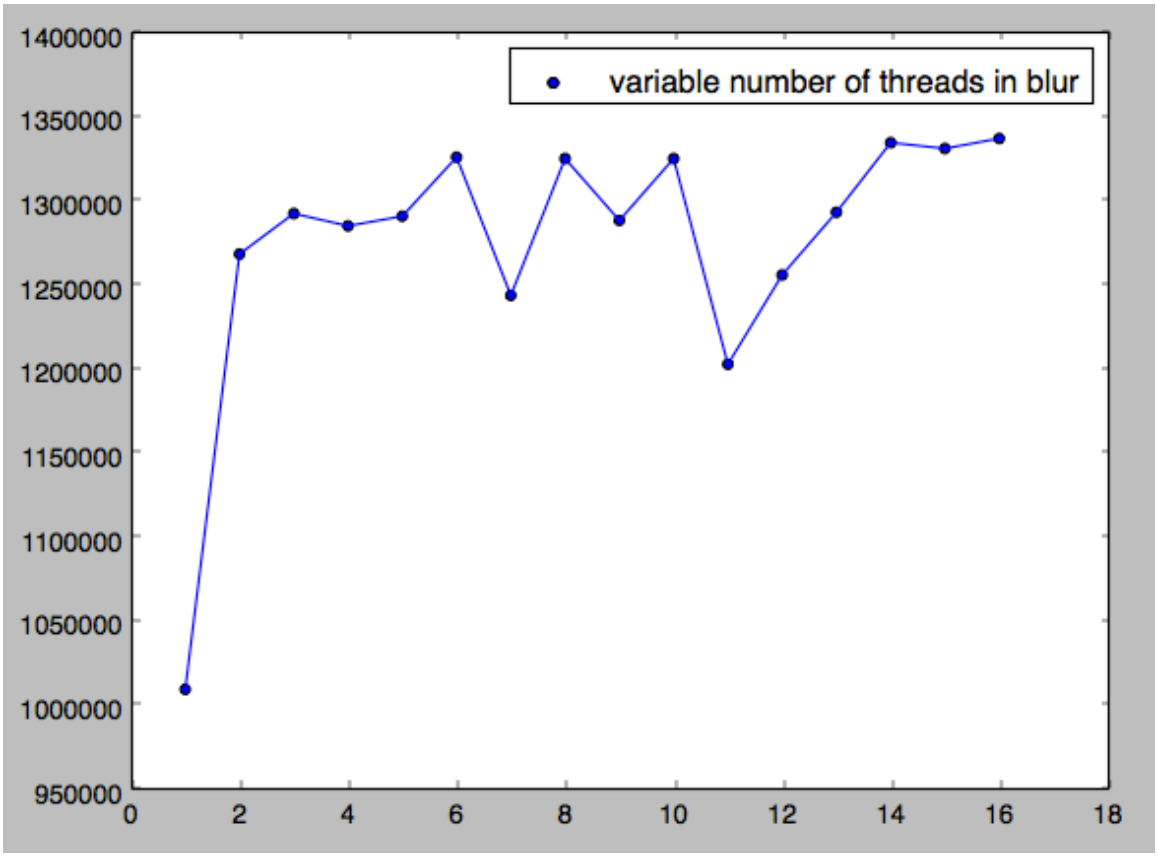
3.2.3

My peak was not even close. This could be because I was ssh'ing into the machine, so results are not going to measure up to the peak performance of the machine in general. And matrix multiplication as an algorithm doesn't achieve nearly as good cache performance as just a linear scan through an array, for example. So I'd say it's partly algorithmic, partly the environment.

3.2.4
I prefer pthreads just because I feel like I know exactly what's happening and what I'm doing. It gives me some peace of mind for profiling or trying to chase race conditions.

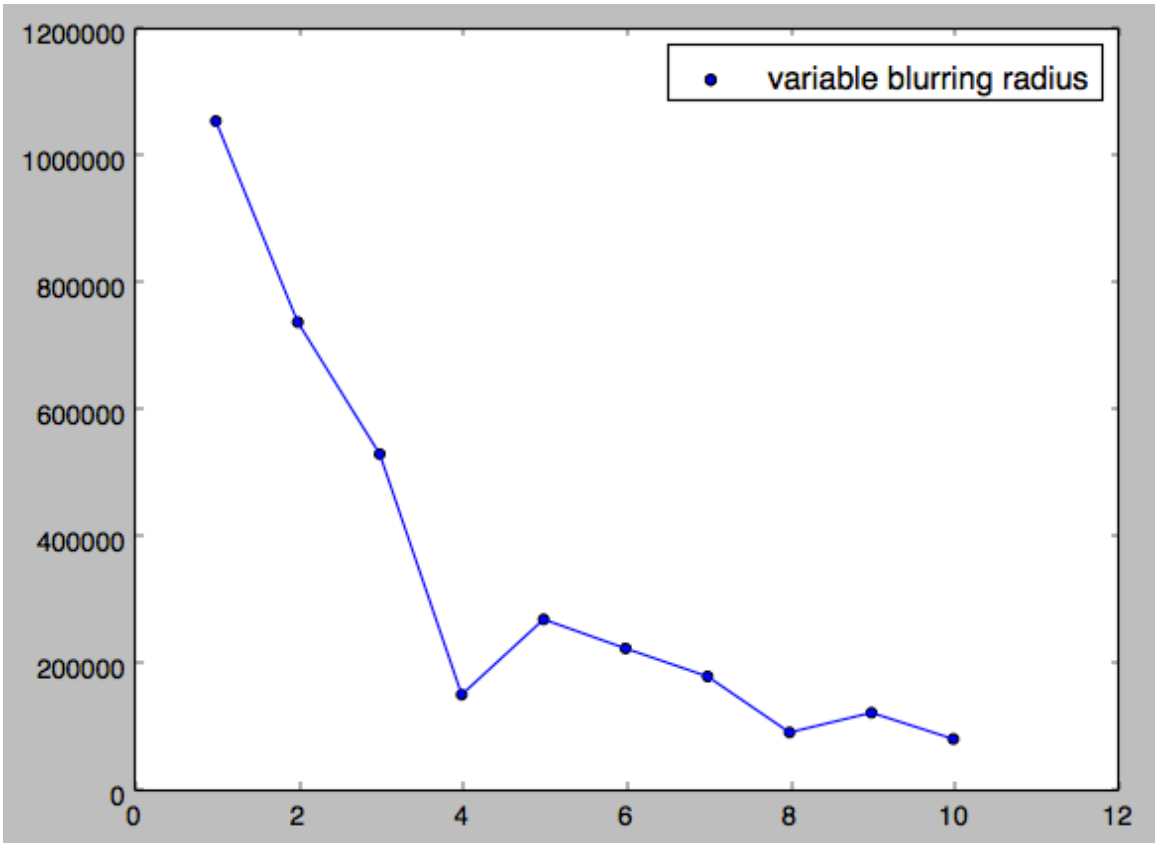
Nikhil Unni
UC Berkeley

4.1.1



4.1.2

Nikhil Unni
UC Berkeley



My guess for why performance drops so quickly with blur radius is twofold: first, the radius of 10 is an order of magnitude more operations than a radius of one, so a 1/10th performance wouldn't be unreasonable. But also, now since the radius is bigger, cache performance drops as memory locality on average decreases for the average lookup, since we're doing much wider sweeps through memory.

4.2
I spent a couple of hours on the assignment.