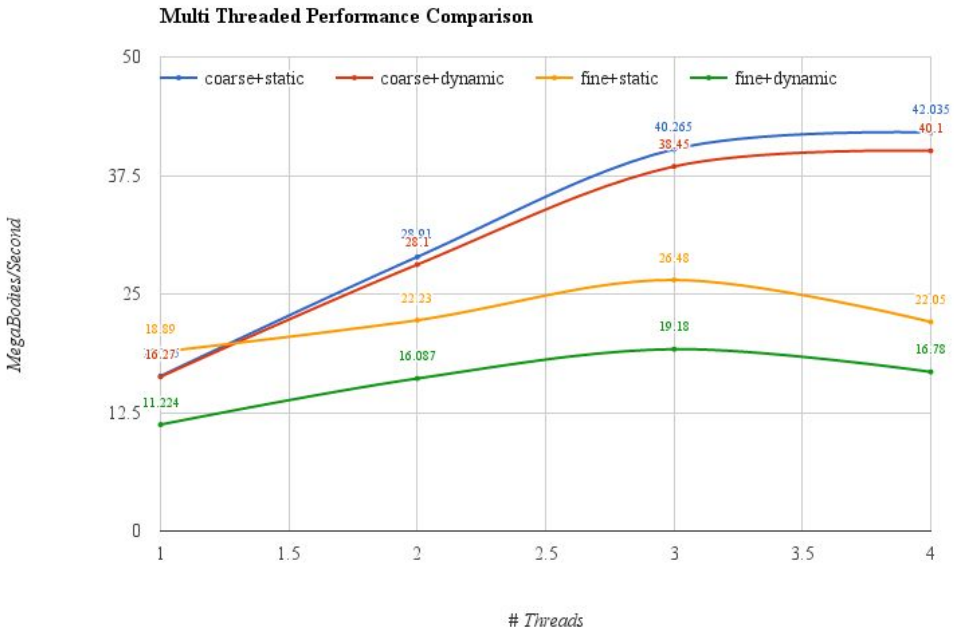
Cody Malick
Project 1
CS 475

1. I ran this on my home desktop, which is running an Intel Sandy Bridge processor with four cores, no hyper threading. It has 8 Gigabytes of RAM.
2. Here is the table of my results:

#Threads	coarse+static	coarse+dynamic	fine+static	fine+dynamic
1	16.345	16.27	18.89	11.224
2	28.91	28.1	22.23	16.087
3	40.265	38.45	26.48	19.18
4	42.035	40.1	22.05	16.78

3. Here is a screen capture of the performance of the different runs. The graph's x axis → number of threads and the y axis → performance in megabodies/second.



4. It seems that coarse parallelism is performing better in this particular use case. Another interesting observation is that static and dynamic scheduling don't seem to have a large impact on coarse grain parallelism, while it has a greater effect on fine grain parallelism.

5. I suspect the reason for this being the amount of reduction that needs to occur when using fine grain parallelism.Each variable fx, fy, fz each need to be reduced later. Another reason is that the problem may not be large enough to warrant splitting the problem down into threads at that for loop. It may be too inefficient.

   I believe the reason we don't see any large differences in performance with different scheduling methods is that the problem isn't big enough. We aren't see a large difference in performance because the problem executes so quickly. If we had a problem with a larger computation time, we might see a larger gap between methods.