Hualiang Li
ICS632
Assignment #4

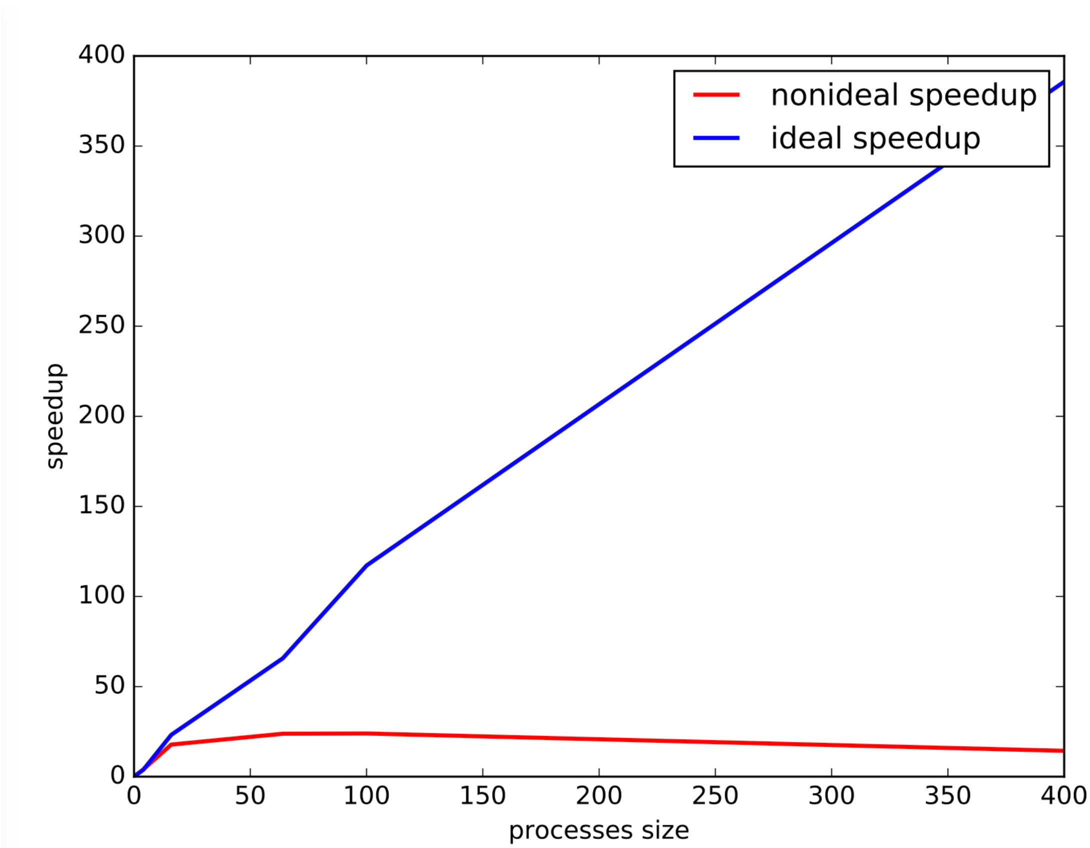
Topic #3

What is challenging for me is how to convert the index between local space and global space. I draw out a sample matrix to ease the pain. Additionally, I learned malloc does not initialize the memory for me, which is a hidden bug for my program. I am happy with my final implementation.

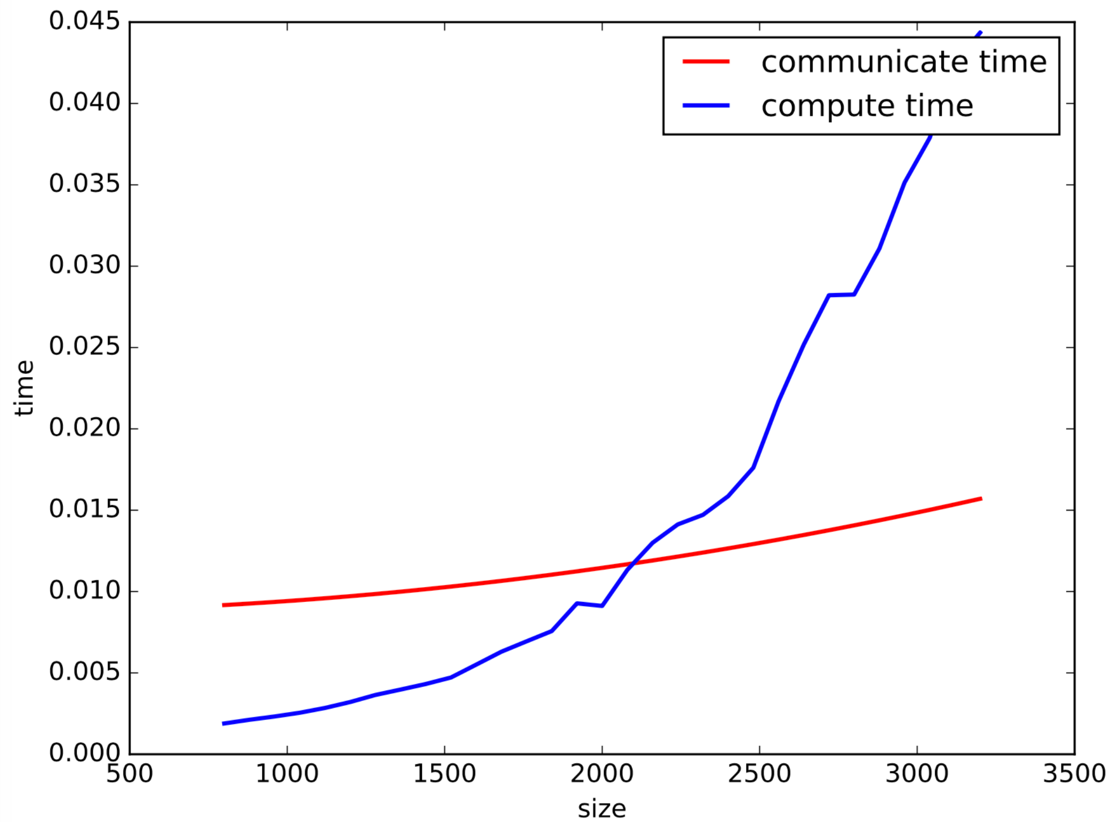
Topic #4

Activity #1

Below is my plot for speed-up curves (speedup vs. number of processors) of ideal and non-ideal platform. For p=400, the parallel efficiency for ideal platform is 385/400 = 96.3%. While for non-ideal platform, the parallel efficiency only 21/400 = 5.25%. Therefore, on a real machine, if we compute this large size with large amount of processes, the parallel efficiency would be very low. That means, it is not worth to have so many processes computing the matrix multiplication.



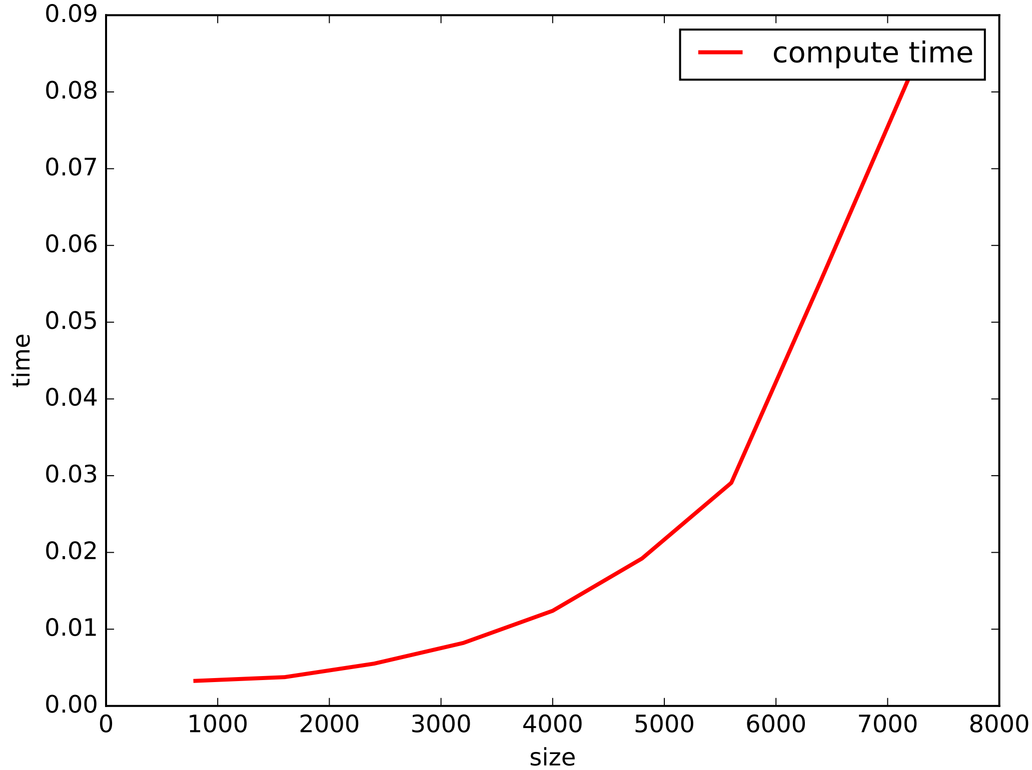
Below is my plot showing the communication time vs. the matrix size.



From the graph, we can see that the computation time and communication time is growing faster and faster as the size of matrix increases, especially for compute time. However, as we can see from the graph, the compute time grows much faster than the communication time. When the matrix size is sufficient large, the compute time would become the bottleneck instead of communication time. This is a good news for multiplying large matrices because the communication overhead is insignificant compared to the compute time.

Activity #2

Below is my plot for average efficiency vs. matrix size curve.



From the plot above, we can see that the compute time is growing faster as the matrix size increases. It is like a n^3 plot, which is matching the time complexity of matrix multiplication. So, we need more work on it if the matrix size is large.

For this topic, the hardest thing is to gather data and plot it. One biggest mistake I made is that I did not consider the matrix multiplication is 3 nested for loop, so my flops calculation just use size*Factor instead of $size^3$*Factor.