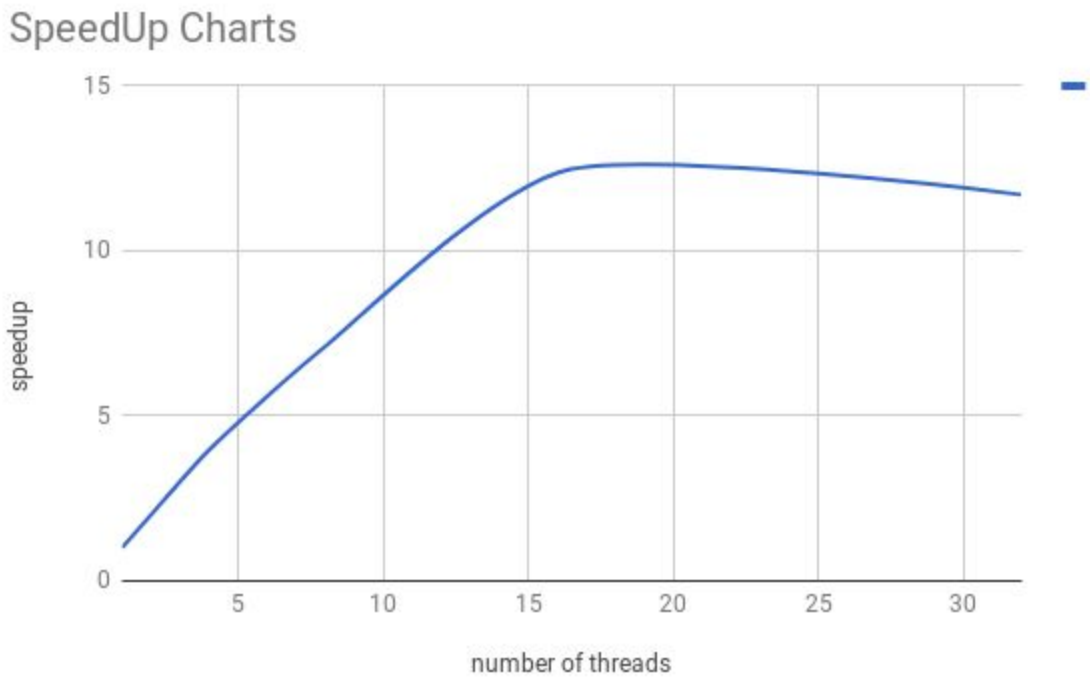
Project 2 WriteUp

Part I

(1a). I think the type of speedup should be sublinear. I think the running time will become larger as the number of threads grows, but it will become stuck at some point when the number of threads continue to grow. The reasons are the following: Firstly, when number of thread = 1, we don't have some race conditions to happen. But when we have 2 threads, since random object is synchronized, so it will have race conditions to happen. When the number of threads continue to grow, the race conditions will happen more frequently, and when one thread has the random object, other threads must wait, so it will take additional time compared to the case where we only have 1 thread. But when the number of threads grows, due to the number of cores and other kinds of resources are limited, so the race conditions' frequency will not be able to continue to grow. At that point, the running time will be stuck at some point.

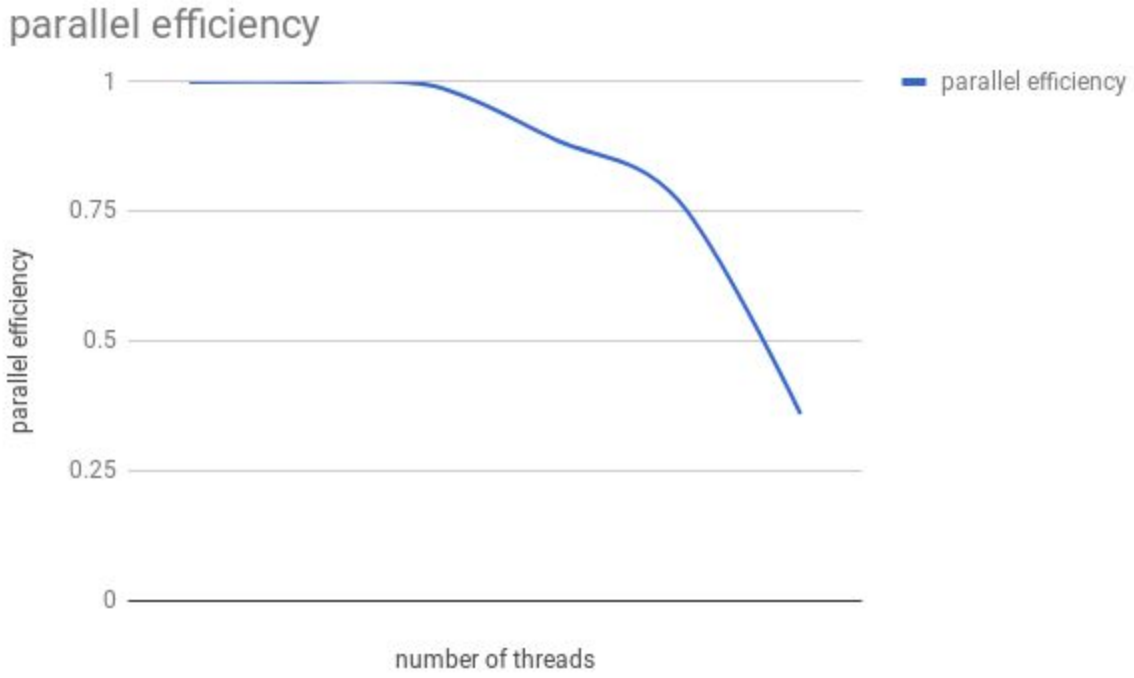
(1b). In this case, since the random object is synchronized, so it can only be accessed by one thread at the same time. Thus, it will have race conditions, which is not helpful for our parallelism. Thus, it doesn't help to parallelize our program. Also, due to the fact that the single random object is shared by all the threads, so switching between threads to get access the random also takes a lot of time, which is not helpful in parallel program. Waiting for another thread to finish using random object, which makes the other threads waiting for getting access to random object become stuck, which is really not efficient. Also, the startup and skew can be against parallelism in this case. When the number of threads continue to grow, the startup cost become large, which needs more time. Also, when we distribute the task into more threads, it is not guaranteed that every thread will finish their own tasks at the same time, maybe some thread will finish its task very late which is also against parallelism.

(2a)



Characterize the plot: If we see the picture above, we can know that it is linear when number of threads <= 16, but becomes sublinear when number of threads >=16.

(2b)



Characterize this plot: If we see the picture, we can know that it is sublinear. At the beginning, the efficiency didn't dramatically decrease. When the number of threads>=4, the efficiency dramatically decrease.

Part II: