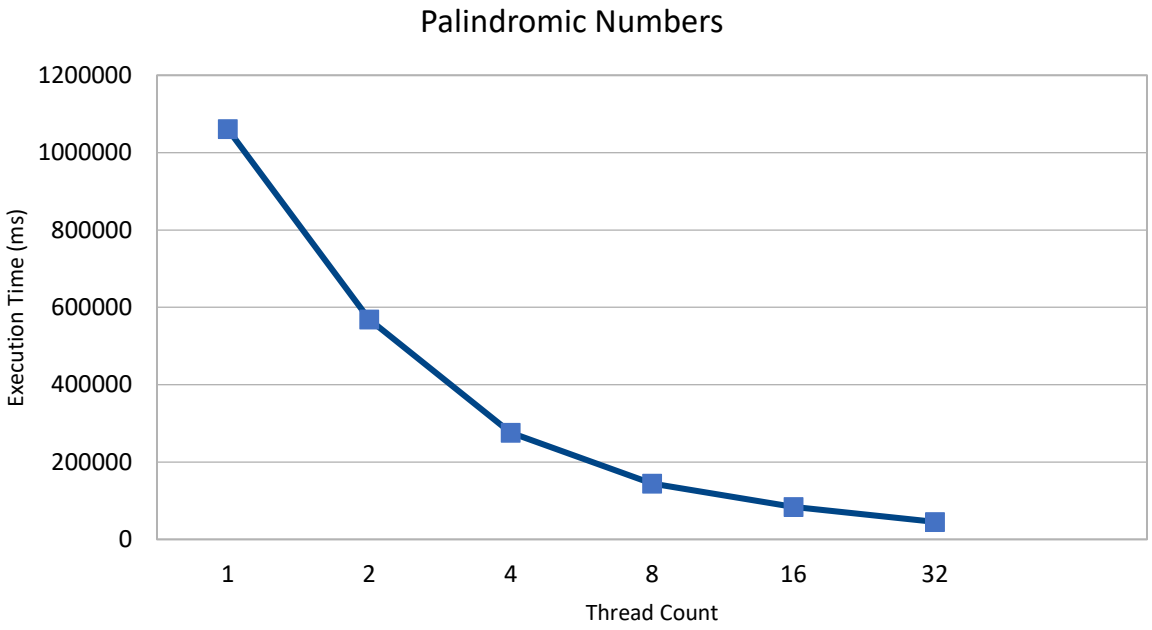
Bong Lee
Asst06
CS302

Multi-Thread Palindrome Calculation



Above line graph is a visual representation of how multiple threads can significantly reduce the execution time of calculating palindromes. The number of palindromes between 1 and 10 billion was calculated with varying number of threads to see how the execution times would differ. As the graph suggests increasing the number of threads from 1 to 2 almost reduced the execution time by half. Increasing from 2 threads to 4 threads had similar results. At certain point however increasing the number of threads did not significantly reduced the execution time. In this experiment having more than 8 threads for such task seems like an overkill.

Speed-Up Percentage Chart

Thread Count	1	2	4	8	16	32
Execution Time (ms)	1060941	568792	276059	144114	83902	45341
Speed-Up	N/A	187%	384%	736%	1,265%	2,340%

Bong Lee
Asst06
CS302

The program was set up so that each thread function would receive a block of numbers to calculate the palindromes. A mutex lock was placed inside the function that provided the numbers to the treads. The same function also incremented each time it provided the numbers to the threads. Removing the mutex lock caused the results to become inconsistent. The inconsistency was caused be race condition when the threads would call the function to receive their numbers. The threads were able to call the function to receive their block of numbers and increment the counter without waiting for other threads.