Star Catalog Report

Overview:

This assignment requires the programmer to multi-thread the application and find the optimal number of threads between 2, 4, 10, 25, 100, and 1000. The program must be able to execute 1 thread and the program must not exhibit race conditions or deadlock. The code provided before modifications serially calculates the angular distance between 30,000 stars in the Tycho Star Catalogue. Unmodified, the code takes a significant amount of time to run. Multithreading the code provided should reduce the run time.

Code implementation:

The library added to the program is <pthread.h>. This library is necessary to create threads using pthread_create, pthread_join, pthread_exit, mutex_lock, and mutex_unlock. Pthread_create creates the threads responsible for dividing the work. The ID's are passed in the arguments and used in the function created to execute the desired work by the threads. Pthread_join makes a thread wait for another thread to terminate. The function mutex_lock is used to keep multiple threads from accessing the same variable and altering the data in the variable. The function mutex_unlock releases the thread and gives access for another thread to alter the variable. Global variables were used in this assignment and were locked and unlocked to prevent race conditions. The timing method for this assignment is the Linux based command "time" on the command line followed by a space and execution of the file such as ./findAngular. I chose to use this command for my timing method because it is easy to implement and gives you the real, user, and system time. The real time is what will be taken into consideration since the assignment states we should focus on the run time of the program.

Anomalies:

The anomaly I encountered was that the run time continued to decrease as the threads increased. I expected the output to have an optimal number of threads within 2-10 range. I anticipated that the more threads I created the slower the run time would be due to locking and context switching. This was not the case for my program.

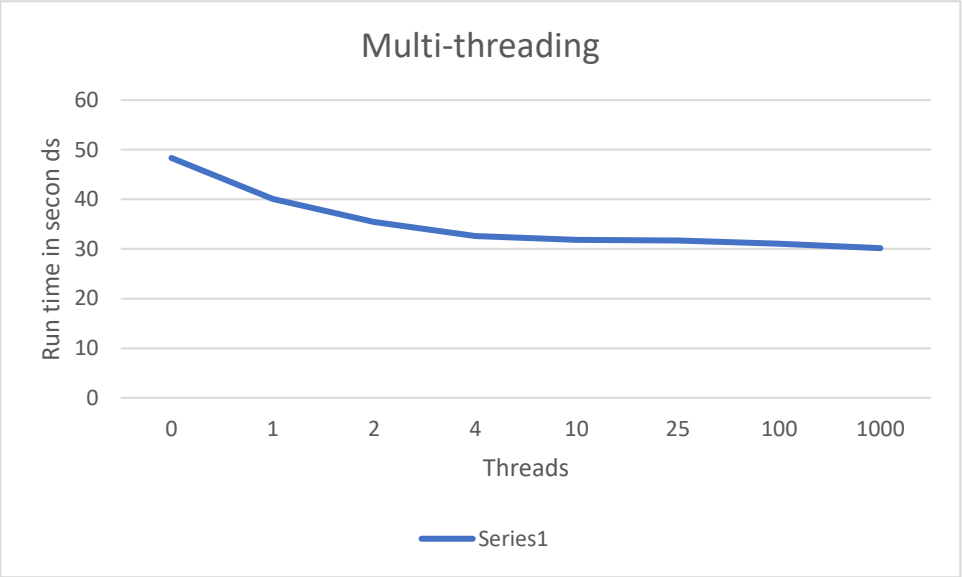
Conclusion:

I found that 1,000 threads are the most optimal when I gathered my runtimes for 1, 2, 4, 10, 100, and 1,000 threads. The optimal number of threads is 1,000 because timing method shows it has the least real time for the execution of the program. The 1,000 threads, however, has the highest system time showing that it spends more time in the kernel the more threads one has implemented.

Notes: The average found in the modified code varies slightly than the unmodified code, but this is due to the formula used to calculate mean and the variable's data type float. The average, min, and max calculated from the modified code stays consistent from thread 1 to 1000 showing there is no race condition in the code within this range of threads.

Graphs and Tables

Threads	Runtime
0	48.347
1	40.079
2	35.467
4	32.601
10	31.852
25	31.74
100	31.067
1000	30.164



Screenshots

Baseline

```
@johnnyG93-cyber ➜/workspaces/Star-Catalog-Assignment (master) $ time ./findAngular
30000 records read
Average distance found is 31.904232
Minimum distance found is 0.000225
Maximum distance found is 179.569720

real    0m48.352s
user    0m47.261s
sys     0m0.377s
```

Threads

```
@johnnyG93-cyber →/workspaces/star-catalog-multithreading-johnnyG93-cyber (master) $ time ./findAngular -t 1
30000 records read
1 threads created
Average distance found is 31.902105
Minimum distance found is 0.000225
Maximum distance found is 179.569720

real    0m39.622s
user    0m39.315s
sys     0m0.024s
@johnnyG93-cyber →/workspaces/star-catalog-multithreading-johnnyG93-cyber (master) $ time ./findAngular -t 2
30000 records read
2 threads created
Average distance found is 31.902105
Minimum distance found is 0.000225
Maximum distance found is 179.569720

real    0m35.315s
user    0m48.967s
sys     0m0.036s
@johnnyG93-cyber →/workspaces/star-catalog-multithreading-johnnyG93-cyber (master) $ time ./findAngular -t 4
30000 records read
4 threads created
Average distance found is 31.902105
Minimum distance found is 0.000225
Maximum distance found is 179.569720

real    0m32.692s
user    0m56.524s
sys     0m0.043s
@johnnyG93-cyber →/workspaces/star-catalog-multithreading-johnnyG93-cyber (master) $ time ./findAngular -t 10
30000 records read
10 threads created
Average distance found is 31.902105
Minimum distance found is 0.000225
Maximum distance found is 179.569720

real    0m31.387s
user    0m57.272s
sys     0m0.067s
```

```
@johnnyG93-cyber →/workspaces/star-catalog-multithreading-johnnyG93-cyber (master) $ time ./findAngular -t 25
30000 records read
25 threads created
Average distance found is 31.902105
Minimum distance found is 0.000225
Maximum distance found is 179.569720

real    0m31.281s
user    0m57.676s
sys     0m0.055s
@johnnyG93-cyber →/workspaces/star-catalog-multithreading-johnnyG93-cyber (master) $ time ./findAngular -t 100
30000 records read
100 threads created
Average distance found is 31.902105
Minimum distance found is 0.000225
Maximum distance found is 179.569720

real    0m31.063s
user    0m57.850s
sys     0m0.086s
@johnnyG93-cyber →/workspaces/star-catalog-multithreading-johnnyG93-cyber (master) $ time ./findAngular -t 1000
30000 records read
1000 threads created
Average distance found is 31.902105
Minimum distance found is 0.000225
Maximum distance found is 179.569720

real    0m30.358s
user    0m57.901s
sys     0m0.094s
```