Multithreading Report (Github for code and images)

Approach

My first priority in this project was determining the correct order in which the data and functions needed to execute. In multithreading, it is essential to properly sequence tasks to avoid race conditions, deadlocks, and incorrect results. Ensuring that each statistical function ran at the appropriate time was a key part of the design.

Findings

After reviewing the assignment requirements, I determined that only one statistic—standard deviation—depends on another value being computed ahead of time (the mean). Because of this dependency, all other statistical operations can run in parallel except the standard deviation.

Therefore, the execution order is:

mean_thread() → mode_thread() → median_thread() → max_thread()

Only after mean_thread() completes can std_thread() begin.

Thread Implementations

max_thread(): Iterates through the array and compares each element to track the highest value. The largest value found is stored in the global variable.

median_thread(): Creates a copy of the array and sorts the copy. If the count is even, it averages the two middle values; if odd, it selects the middle element and stores to global variable.

mode_thread(): Selects each value in the array using an outer loop and counts its occurrences with a nested loop. This $O(n^2)$ algorithm determines the number with the highest frequency.

mean_thread(): Sums all values in the array and divides by the total count to compute the mean stores result in global variable.

std_thread(): Uses the mean to compute variance by summing the squared differences and then taking the square root for the standard deviation then stores result in global variable.

Difficulties

The most difficult part of the project was calculating the mode. My initial attempts were unsuccessful, so I referenced an external source to understand the proper logic. Once I saw a working version, the algorithm became clear.

Conclusion

The multithreaded implementation showed slightly faster results on average. However, due to the relatively small datasize, the difference was not significant. For small programs, a sequential approach can be cleaner and easier to manage. For larger datasets or computation-heavy workloads, multithreading offers meaningful performance benefits.

Sources

How to Find the Mode of Numbers in an Array in C? - GeeksforGeeks