# Report

You can run the program with ./“program name” “number of integers” “number of threads” command after executing the make file. I named the program as myprogram.out in the makefile. Example execution is “./myprogram.out 10000 10”. If there is no number of threads given, the default number of threads is 1.

I implemented the program such that it can run with 1 to 10 threads. If the number of given threads are greater than 10, the program runs with 10 threads. If you give the number of threads parameter as 1 there will be no extra thread creation. All functions run in the main thread. After executing functions, main thread writes the results to the output file. Thread numbers between 2 and 10 creates extra threads in addition to main thread. If the number of threads parameter is given as 2 there will be 2 more thread creation in addition to the main thread. Main thread waits the other threads while other threads are running. After other threads are done main thread continues its execution and prints the results. The same execution is made for other thread numbers except 1.

I shared the functions among threads for the thread numbers between 2 and 10. For example if the thread number is given as 2, minimum value, arithmetic mean, range, mode, median calculations are made in one thread, sum, maximum value, harmonic mean, standard deviation, interquartile range calculations are made in the other thread. I shared the functions among threads unequally if the number of functions, which is 10, cannot be divided to number of threads.

I created 2 types of function for each operation. One type is for calculations and the other type is for threads. For the sake of simplicity, let’s call them thread type functions and calculator functions respectively. Threads call the thread type functions, which takes void* parameter and returns void* type value. The function that is called, calls the functions that needed for calculation. After the function calculates the result, thread type function assigns the result to the proper variable.

I created given number of additional threads to the main thread and each calls one thread type function. I checked the number of threads in the called function. If the number of threads are 10, the function calls only its calculator function. If the number of threads are different than 10, the function calls proper calculator functions according to the sharing I made. After the calculations, return values are assigned to the proper variables. I have to denote that there is no additional thread creation inside of the thread type functions.

I observed that 5 and 10 threads are faster than 1 thread but there is not much execution time difference between 5 and 10 threads. For example, 1-thread execution finishes in 72.66 ms for 100000 inputs whereas 5-thread execution finishes in 32.73 ms and 10-thread execution finishes 23.32 ms. The reason why the 5 and 10 threads are faster than 1 thread is that 1 thread executes the functions one by one, so it takes more time to finish execution. The functions do not wait each other to finish and they execute at the same time in 10 threads. In 5 threads, each function waits only 1 function so it is a lot faster than 1 thread but a little slower than 10 threads.

I put the outputs to the output files ended with number of threads. For example, if the program is run with 4 threads, the output file will be “output4.txt”. There is a possibility that median and interquartile range are floating point numbers, so program prints them as floats even if they are integers. For instance, if the median is 5, the output will be 5.00000.

Ali Başaran
2020400357