Operating Systems 2

Tanmay Garg CS20BTECH11063

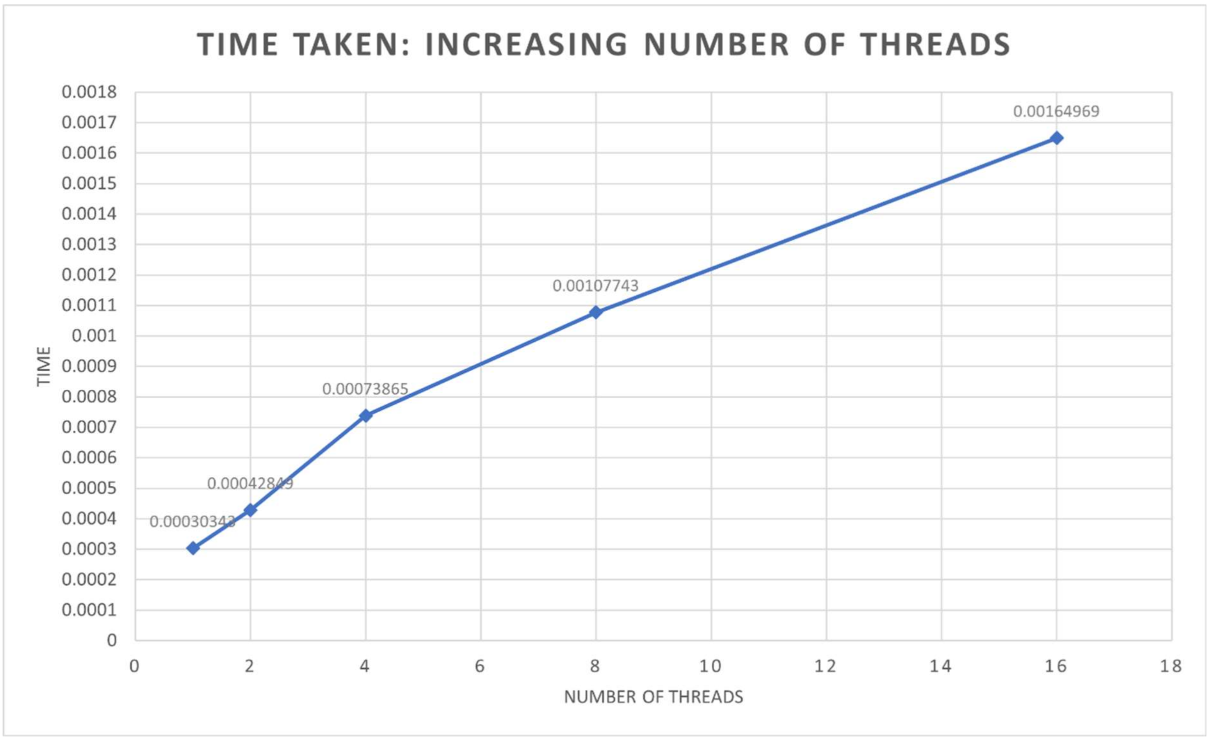
Programming Assignment 1

Program Design:

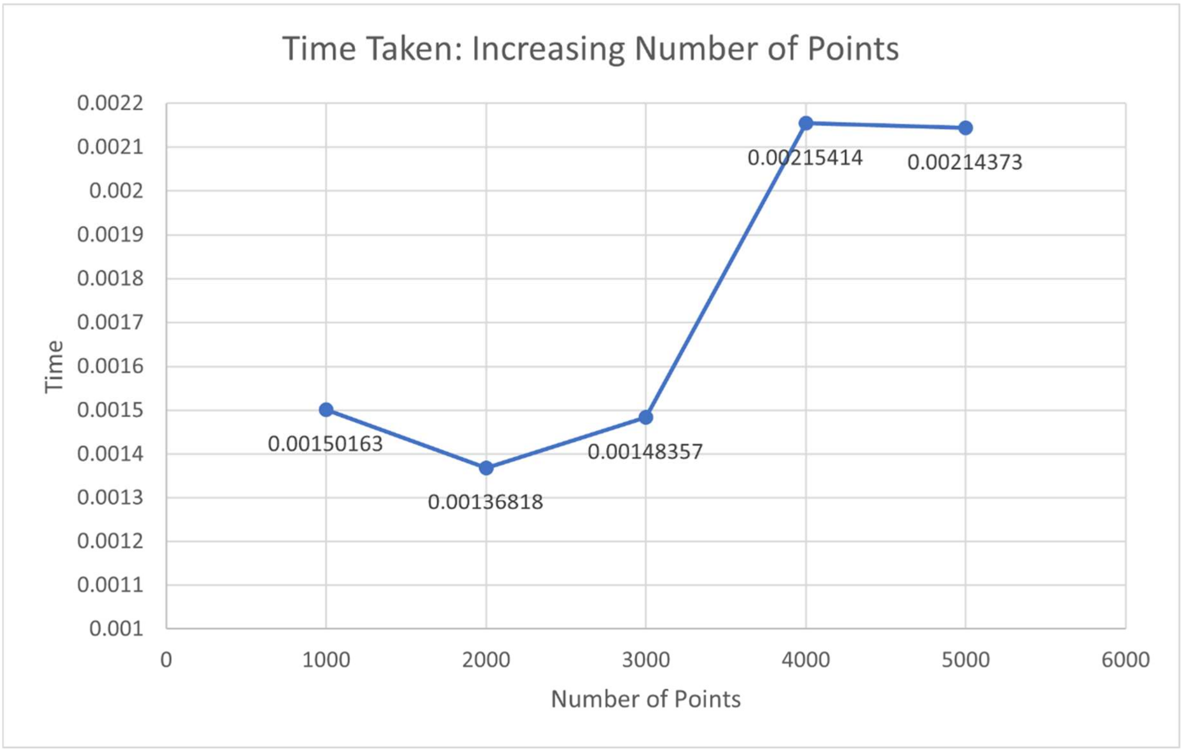
- First the program opens the input.txt file which has the required input for number of threads, number of points, source coordinate and all the other coordinates
- It then scans from the file and places in their respective variables
- An array of **pthreads** is created and an array of struct **arguements_findNearest** is also created which will have the parameters that need to be passed in the thread for **find_nearest_point** function
- The entire array is divided into N sub arrays where N is the number of threads, and each subarray has a size of $\left\lfloor \frac{size\ of\ array}{number\ of\ threads} \right\rfloor$
- Each thread calls the **find_nearest_point** function
- The **argument_multithreading_t** is provided as argument which contains starting index, ending index and the nearest point within the sub array
- The nearest point is initialized at the end of the **find_nearest_point** function
- After all the threads are joined then the main thread finds the nearest point amongst the shortlisted nearest points of the subarrays
- The function is **find_nearest_point** which works simply by looping through the subarray
    - It first initializes the startIndex and endIndex from the arguments passed into the function
    - It then loops through the entire subarray to find the minimum distance between the 2 points
    - It then initializes the point_t inside **argument_multithreading_t** to the nearest point
- The time has been calculated between the following 2 points
    - Just after the points have been initialized
    - Just after finding the nearest point
- For the Comparison Part
    - The program needs to be run multiple times and number of points and number of threads should be changed in the source code at appropriate lines (Mentioned in the comments in the source code and readme file)

Sequential v/s Parallel:

- The first comparison is when the number of threads is increasing, and the number of points is fixed
    - The program has been run 100 times for each number of thread and then the averages for all have been plotted. Number of Points is 5000



- The second comparison is when the number of points is increasing, and the number of threads is constant
    - The program has been run 100 times for each size and then the averages for all have been plotted. Number of threads is 16



Note:

- The following comparison has been done only for 100 iterations for each type and the average has been taken for the values
- The values may differ based on the on the user's system and specifications and compiler optimizations and will also depend if the program is run on different Operating Systems, its architectures and whether it is a Virtual Machine or not
- To run the first part of the assignment, which is taking input from file and giving output to file, comment the second main function and uncomment the first main function
- To run the report part of the assignment, comment the first main function and uncomment the second main function and follow the instructions given in the readme file
- The report part may give the same point as output as when it was run before if the program is run multiple times (see readme file) due to *rand()* function and the OS it runs on, it is preferable to have a small time gap between sequential runs to give different outputs