Name: Aravinda Kumar Reddy Thippareddy

Roll No: CS20BTECH11053

Course: Operating Systems - II

Assignment: Finding nearest point using multiple threads

Low Level Design of My Program:

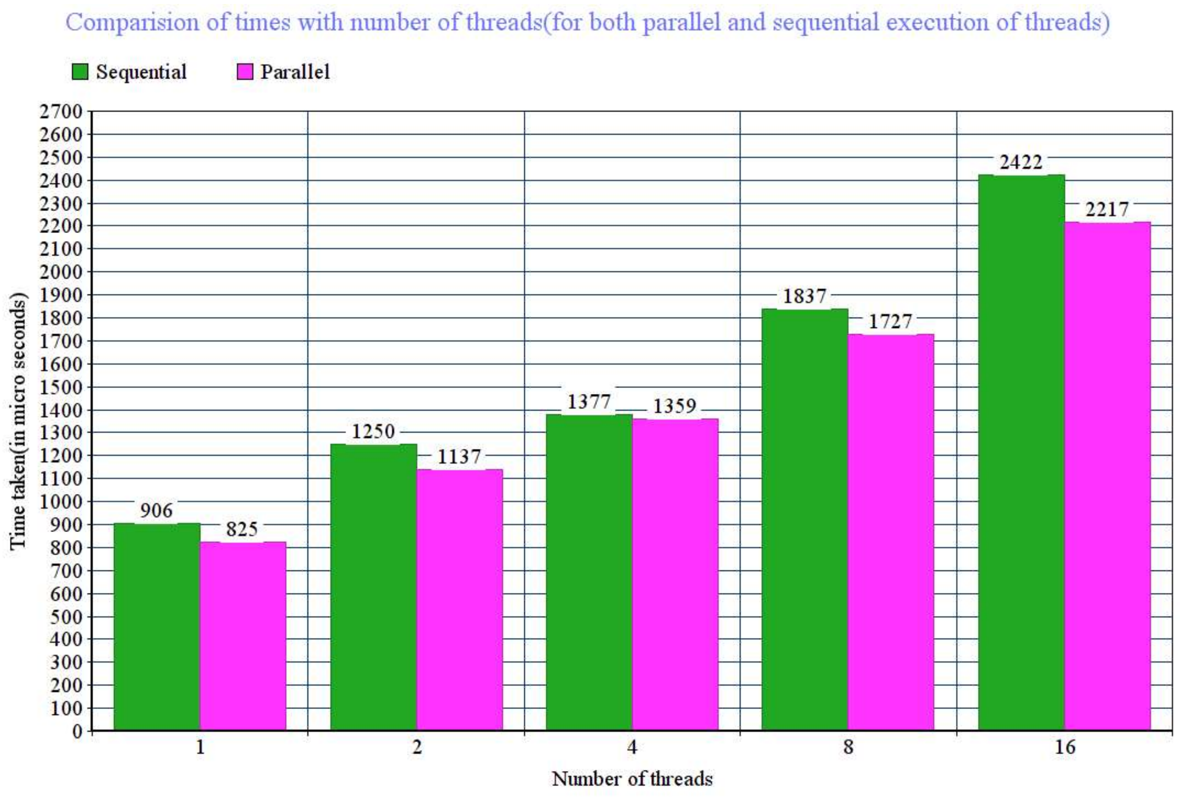
1. I have distributed all the destination points to the threads.
2. We have to input the name of the input text file.
3. I have calculated the nearest point of each thread and stored each of them in an array.
4. And then I had found the nearest point among all those nearest points stored in the array stated in the above point.
5. Thread implementation:
    1) My current program is the implementation of parallel execution of threads.
    2) The thread starts when the function pthread_create(…) is called, and this we have to pass 4 arguments to this function.
    3) Of those 4 arguments, 1st argument represents thread, 2nd argument represents attributes, 3rd argument represents the function to be executed under this thread, 4th argument represents the argument to be passed to the function represented by the 3rd argument.
    4) If pthread_create(…) function of all threads is called before calling the pthread_join(…) function of all threads, then the threads execute in parallel.
    5) If pthread_create(…) function of one thread is called only after calling the pthread_join(…) function of the executing thread, then the threads will execute sequentially.
6. Finding Nearest Point:
    1) Assume the first point of the list of points as the nearest point.

2) Check all the points one by one if they are near than the current nearest point, and update the nearest point accordingly.
3) Finally, you will get the nearest point.

Analysis of output:

1. The program will output nearest point among all the destination points.
2. The program also outputs the time taken for finding the nearest point in micro seconds.
3. The time keeps changing on running the program again and again.

Graphs:

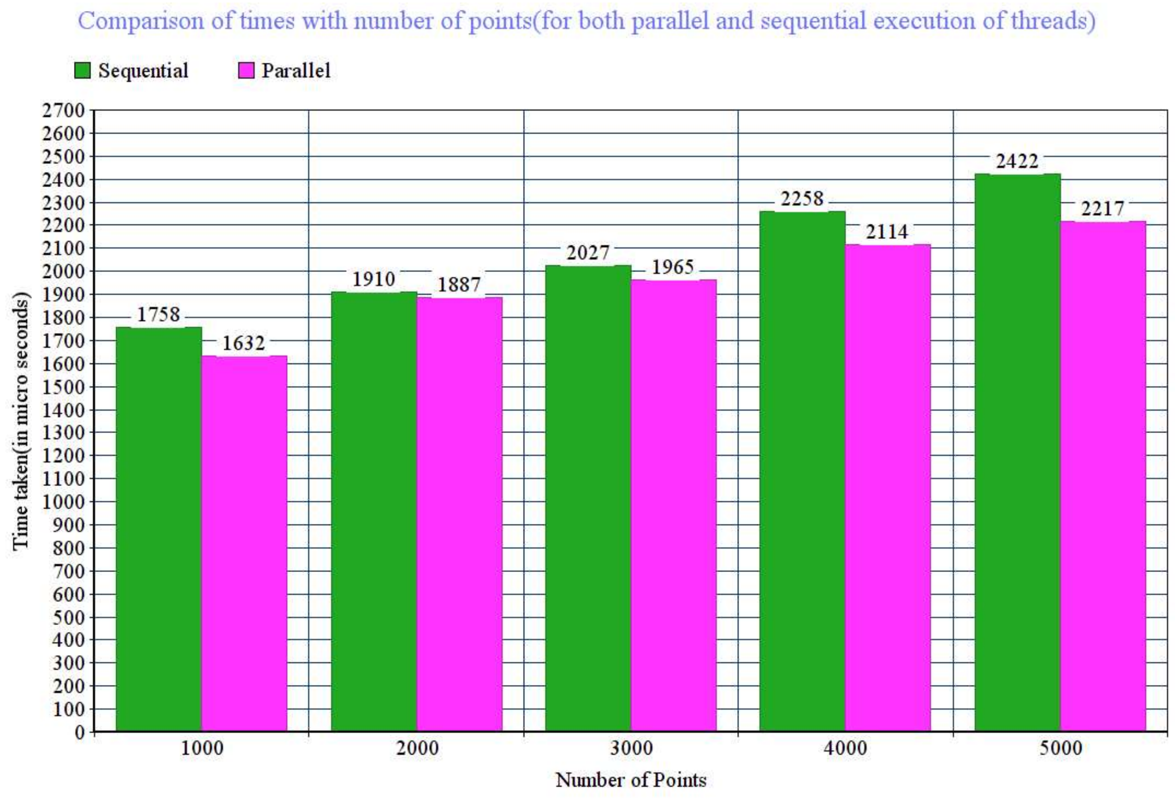


➔ Above is the bar graph of (time taken) versus (number of threads) for both parallel and sequential execution of threads.

➔ Sequential execution of threads takes a little more time than parallel execution of threads.

➔ This is because in parallel execution all threads execute at the same time, where as in sequential execution only after the completion of execution of the current thread, the next thread starts execution.

➔ As the number of threads increases time taken also increases. This is because thread creation also takes some time.



➔ In the above graph time increases as the number of points increases, because we have to calculate nearest point among more number of points (as we move to right).