Operating Systems CS3523
Programming Assignment 1
ES21BTECH11022

High Level Design of Code

1. Taking Input: The program uses command line arguments to take the input file containing values of n and k. Here n is the number of random points to be generated to calculate the value of pi, and k is the number of threads used. Here we also check for two errors being improper number of command line arguments and the case of fopen returning NULL pointer. In both cases the error is printed onto the output file.

2. CreatingThreads: The program creates k threads using a for loop. As each thread executes one function with one void* argument, we pass a pointer to a struct type named arg in the code. We create an array of k such arg structs to pass to each thread. This struct contains one integer j representing the thread number and a pointer to another struct type named point (an array). The point struct represents a 2-D point in the Cartesian Coordinate System and has an additional attribute bool in, which indicates whether the point lies within the circle described by the equation,

$$x^2 + y^2 = 1$$

Then the for loop is run creating the k threads. Then each thread runs the function func and after completion we join all threads using pthread_join.

3. Function func: Each thread runs this function. The initialization of the argument of type arg occurs in the for loop which creates the thread itself. The task of the function is to just generate approximately n/k points that lie inside the region $-1 \leq x \leq 1$ and $-1 \leq y \leq 1$. Then the function also updates the attribute 'in' for the point generated. The log of all the points generated is kept in the points array which is inside the arg struct. After having checked all the generated points the thread updates the global variable that keeps count of total points inside the square and circle.

4. Final Output File: The final output file named output.txt contains the time taken to calculate the value of pi, the obtained value of pi and finally log files for each thread. The log file for a particular thread contains the thread number, and the number of points it generated within the square and circle.

Low Level Design of Code

Considering low level design, we first create k thread ids using an array of tid_t. Then we simply create k threads using,

pthread_create(&threads[i], NULL, perfect, (void*) p);

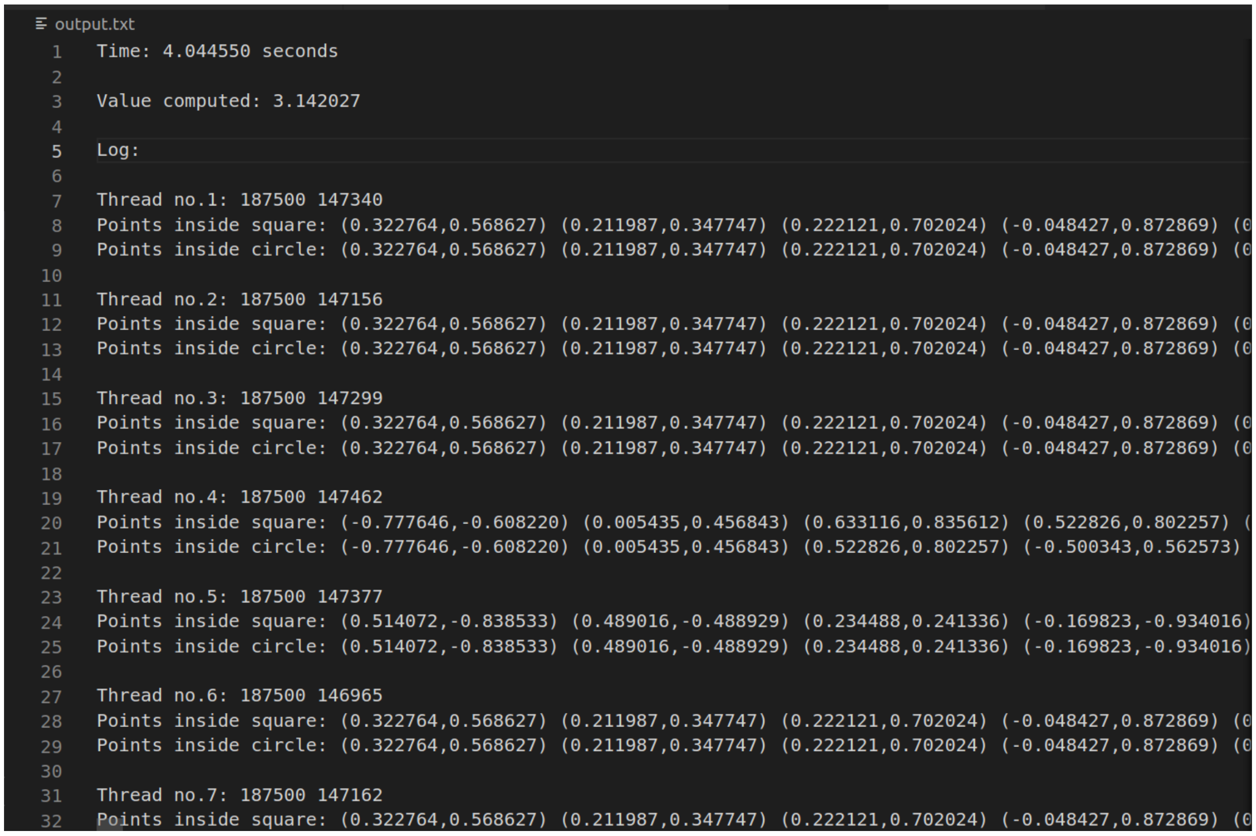
Thus one thread is created for each tid with NULL attributes and each of them executes the function func with argument args[i]. The variable args is basically of type struct arg* and to pass it as an argument we perform type casting. Then in the function random points are generated and checked if they are within the circle and keep track of all this information. This marks the end of tasks for one thread. In the program k such threads are created. Finally we join all these k threads using,

pthread_join(pthread[i], NULL);

In the context of task splitting as each thread generates random points the task is split equally among the threads.

Analysis of Output:

Output file using n = 6000000 and k = 32:

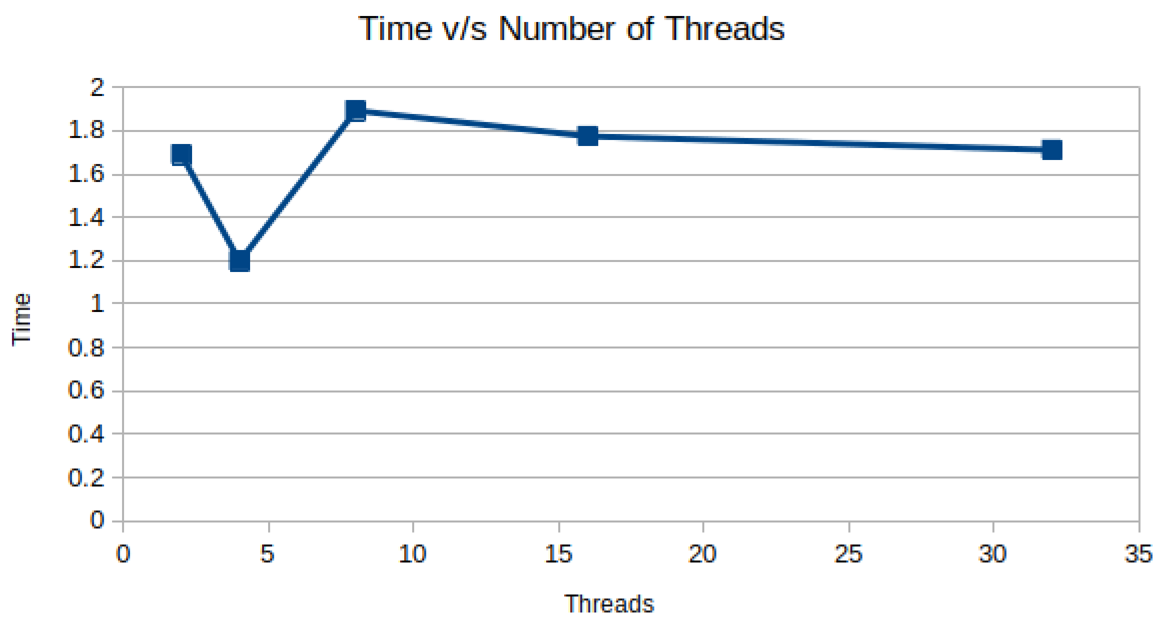


```
output.txt
Time: 4.044550 seconds

Value computed: 3.142027

Log:

Thread no.1: 187500 147340
Points inside square: (0.322764,0.568627) (0.211987,0.347747) (0.222121,0.702024) (-0.048427,0.872869) (0
Points inside circle: (0.322764,0.568627) (0.211987,0.347747) (0.222121,0.702024) (-0.048427,0.872869) (0

Thread no.2: 187500 147156
Points inside square: (0.322764,0.568627) (0.211987,0.347747) (0.222121,0.702024) (-0.048427,0.872869) (0
Points inside circle: (0.322764,0.568627) (0.211987,0.347747) (0.222121,0.702024) (-0.048427,0.872869) (0

Thread no.3: 187500 147299
Points inside square: (0.322764,0.568627) (0.211987,0.347747) (0.222121,0.702024) (-0.048427,0.872869) (0
Points inside circle: (0.322764,0.568627) (0.211987,0.347747) (0.222121,0.702024) (-0.048427,0.872869) (0

Thread no.4: 187500 147462
Points inside square: (-0.777646,-0.608220) (0.005435,0.456843) (0.633116,0.835612) (0.522826,0.802257) (
Points inside circle: (-0.777646,-0.608220) (0.005435,0.456843) (0.522826,0.802257) (-0.500343,0.562573)

Thread no.5: 187500 147377
Points inside square: (0.514072,-0.838533) (0.489016,-0.488929) (0.234488,0.241336) (-0.169823,-0.934016)
Points inside circle: (0.514072,-0.838533) (0.489016,-0.488929) (0.234488,0.241336) (-0.169823,-0.934016)

Thread no.6: 187500 146965
Points inside square: (0.322764,0.568627) (0.211987,0.347747) (0.222121,0.702024) (-0.048427,0.872869) (0
Points inside circle: (0.322764,0.568627) (0.211987,0.347747) (0.222121,0.702024) (-0.048427,0.872869) (0

Thread no.7: 187500 147162
Points inside square: (0.322764,0.568627) (0.211987,0.347747) (0.222121,0.702024) (-0.048427,0.872869) (0
```

Graphs:

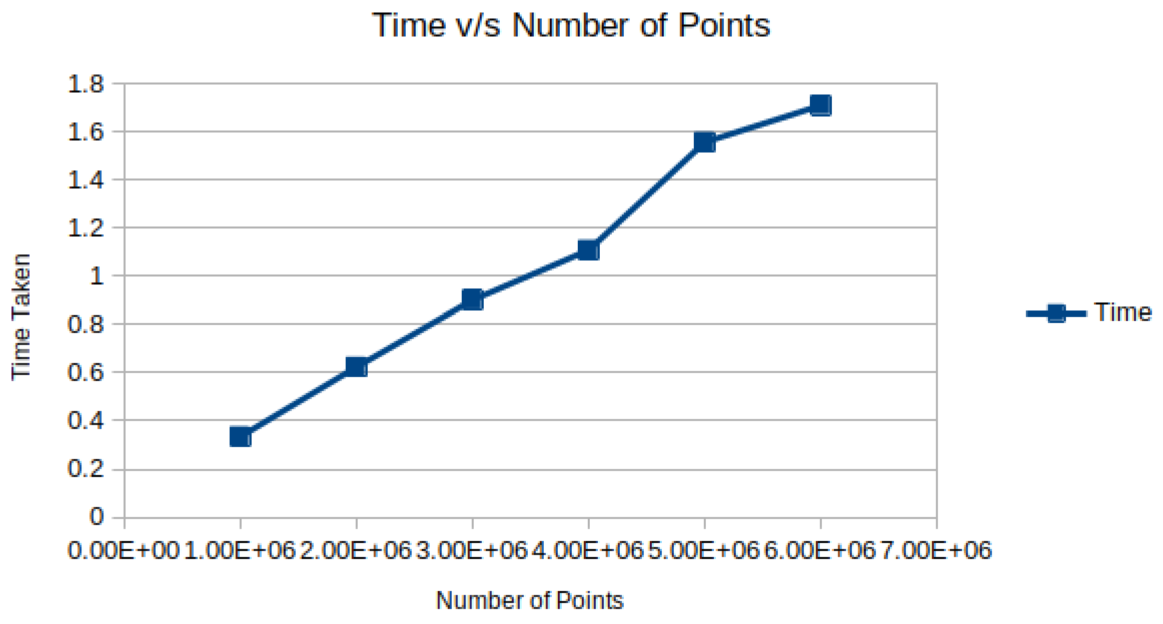
Graph of Time taken v/s Number of threads:

Threads	Time
2	1.688359
4	1.196056
8	1.892381
16	1.77436
32	1.710721



Graph of Time taken v/s Number of threads:

Points	Time
1.00E+06	0.334864
2.00E+06	0.624216
3.00E+06	0.901861
4.00E+06	1.10872
5.00E+06	1.557634
6.00E+06	1.710721


Time v/s Number of Points
Time Taken
0.00E+00
1.00E+06
2.00E+06
3.00E+06
4.00E+06
5.00E+06
6.00E+06
7.00E+06
Number of Points
Time