Programming Assignment 1: Parallel Monte Carlo technique for calculating π using Multi-Threading in C++

CO21BTECH11004

Global arrays are declared : -

- double *x:- Stores the x-coordinate of the randomly generated point.
- double *y:- Stores the y-coordinate of the randomly generated point.
- int *arr: - For i_th point have value 1 is it is inside the circle otherwise 0.
- int *c_points:- Stores no. of points each thread found inside the circle.
- Int *sq_points:- Stores no. of points each thread found inside the square.

Void *TheradFunc(void* k_i) :

- k_i is type casted to long, two integers in_points and all_points are initialized to 0.
- For all i in [0,n) when (i%k)==k_i , a random point is generated i.e., it's X and Y coordinates are generated and stored at i_th index in arrays x and y respectively.
    - Each thread has to update the global array where index%k equals ki to avoid simultaneous access of an index in the array.
    - If the randomly generated point is inside the circle in_points is incremented by 1 and the i_th index of arr changed to 1, if not in circle the i_th index of arr changed to 0.
    - All_points is increment by 1
- After generating and checking desired points, k_i index of c_points is changed to in_points and sq_points is changed to all_points.

main():

- n and k are read from file "inp.txt".
- k sized array k_i is created to send the mod value to each thread contains i+1 at i_th index.
- Global arrays are dynamically allocated:

        - x :- n*sizeof(double)
        - y :- n*sizeof(double)
        - arr :- n*sizeof(int)
        - c_points :- k*sizeof(int)
        - Sq_points :- k*sizeof(int)
    - Clock is started now.
    - pthread_t tid[k], thread identifiers are created.
    - K threads are created using for loop and pthread_create(&tid[i], NULL, ThreadFunc, (void*)k_i[i]);
    - K threads are joined using for loop and pthread_join(tid[i],NULL);
    - Points inside circle and total points are calculated by summing up the data in the c_points and sq_points which each thread had updated.
    - Pi value is calculated as 4*(Points in circle)/(Total Points).
    - Clock is stopped.
    - Time elapsed in calculating pi is calculated by subtracting end_time and start_time.

Results i.e., the time taken to compute the value of pi and the value computed, with logs of each threads is printed in “Output.txt”.

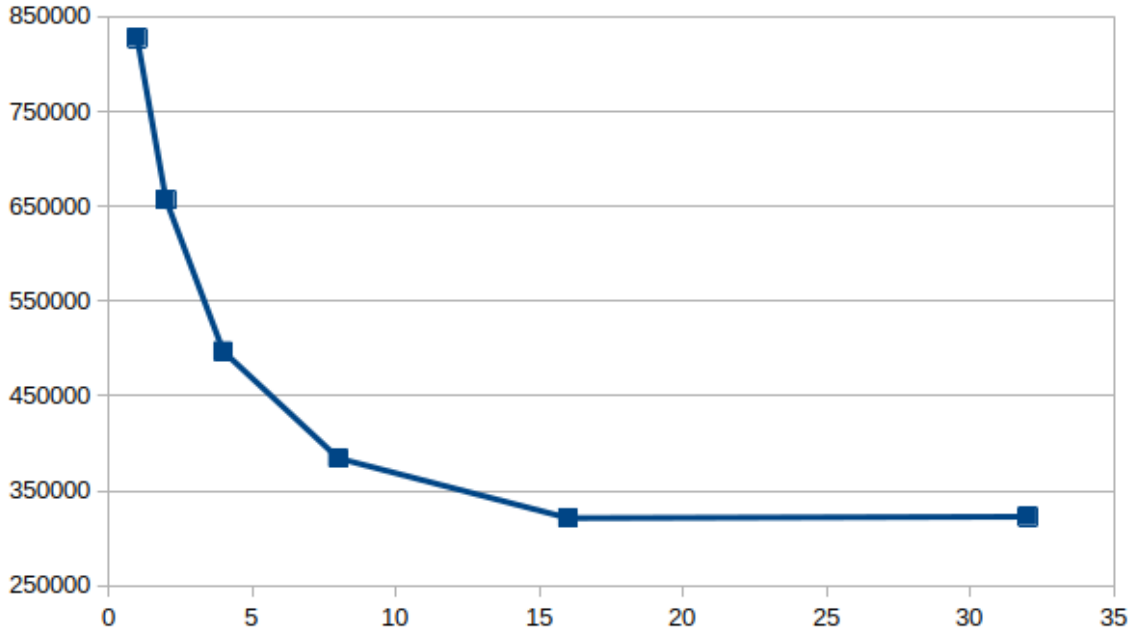
Generation of Random Number:-

- random library is used to generate random numbers numbers in range -1 to 1.
- Links:
    - https://cplusplus.com/reference/random/
    - https://cplusplus.com/reference/random/uniform_real_distribution/
    - https://cplusplus.com/reference/random/mt19937/\
    - https://cplusplus.com/reference/random/random_device/

Calculation of Time :-

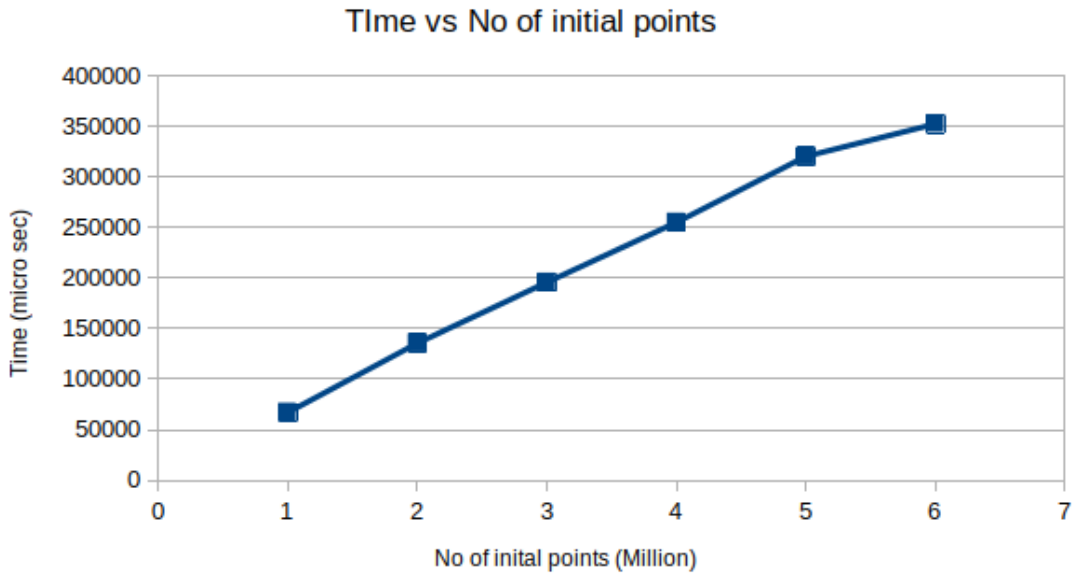
- chrono library, high_resolution_clock is used to measure time.
- Links:
    - https://en.cppreference.com/w/cpp/header/chrono
    - https://en.cppreference.com/w/cpp/chrono/high_resolution_clock
    - https://en.cppreference.com/w/cpp/chrono/duration

Plot1 :- Time Taken vs Number of threads.



Time decreases as expected from 2 to 16, but decrease from 16 to 32 is not as much as from 2-4. This is because my system have 8 cores and 2 threads per core so in total 16 threads, time decreases

Plot2 :- Time Taken vs Number of initial points.



Here time increas as number of points increases keeping no. of threads constant.