CSE 6010 / CX 4010 Assignment 7
Monte Carlo Integration using OpenMP

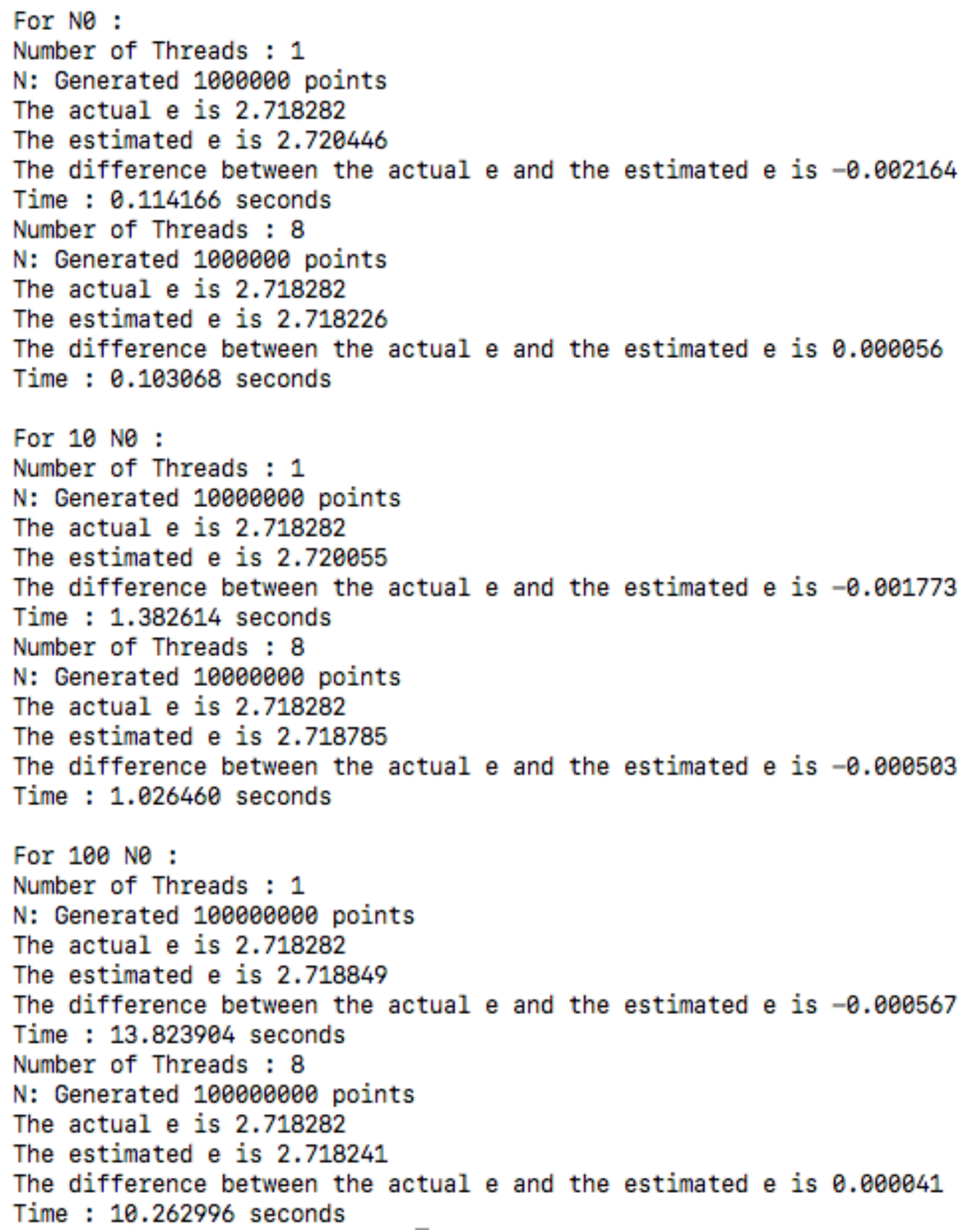
Ting Liao, GTID: 903278773

11/13/2020

In this assignment, I developed two functions, serial() and reduction(), to do the serial computing and parallel computing seperately. For the serial part, it is quite simple. I simply used a for-loop to calculate the points, which are generated randomly, under the exp curve. For the reduction (parallel) part, the concept is similar to the serial part, but I added two #pragma omp parallel function to make the parallel computing. Also, I adjusted the method of generating random numbers in the parallel part by using the code from Canvas (rand_r function) to reduce the time consuming. Finally, I print out all the results on the screen.

Testing procedure and Evidence

- I first implemented the serial function as the standard of my program. Then, I added the parallel function and compared with the serial function.
- From the result, I first know that the value of the estimated e should be correct. Then, I check the number of threads, and it showed that it is equal to what I have defined in the beginning of the code.
- At last, from the time consuming, it showed that using the parallel function can truly accelerate the computing process.



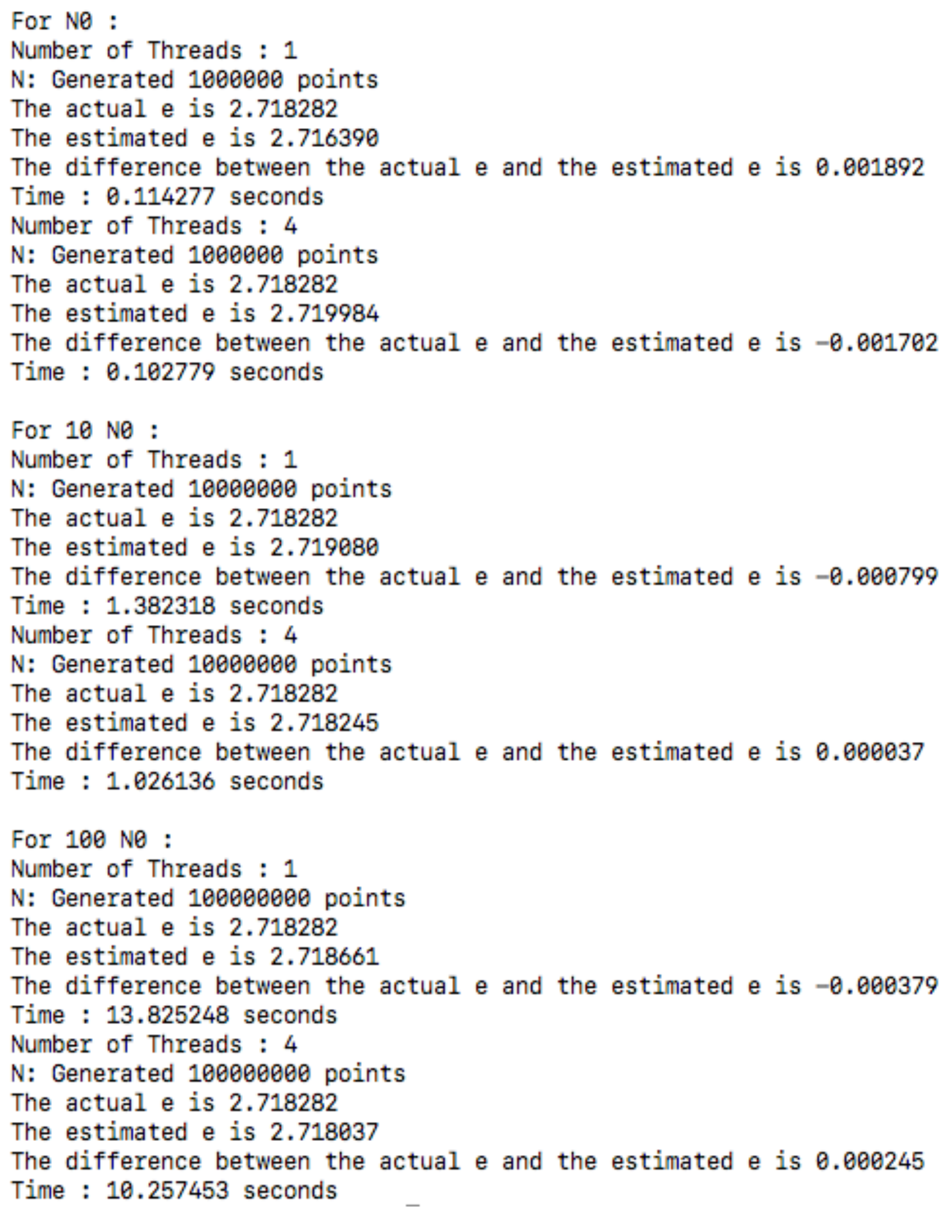
```
For N0 :
Number of Threads : 1
N: Generated 1000000 points
The actual e is 2.718282
The estimated e is 2.720446
The difference between the actual e and the estimated e is -0.002164
Time : 0.114166 seconds
Number of Threads : 8
N: Generated 1000000 points
The actual e is 2.718282
The estimated e is 2.718226
The difference between the actual e and the estimated e is 0.000056
Time : 0.103068 seconds

For 10 N0 :
Number of Threads : 1
N: Generated 10000000 points
The actual e is 2.718282
The estimated e is 2.720055
The difference between the actual e and the estimated e is -0.001773
Time : 1.382614 seconds
Number of Threads : 8
N: Generated 10000000 points
The actual e is 2.718282
The estimated e is 2.718785
The difference between the actual e and the estimated e is -0.000503
Time : 1.026460 seconds

For 100 N0 :
Number of Threads : 1
N: Generated 100000000 points
The actual e is 2.718282
The estimated e is 2.718849
The difference between the actual e and the estimated e is -0.000567
Time : 13.823904 seconds
Number of Threads : 8
N: Generated 100000000 points
The actual e is 2.718282
The estimated e is 2.718241
The difference between the actual e and the estimated e is 0.000041
Time : 10.262996 seconds
```

Summary & Results

- From the result, I found that my reduction (parallel) function is not work as its concept, but similar to a critical function. I might have missed something or typed the wrong parameter because the time consuming didn't decrease when I increased the number of Threads. Then, I realized that the time scale didn't increase when I used more threads because the #pragma reduction is inside another #pragma parallel. That is, my program actually should be considered as using a critical method based on the outer #pragma and thus the runtime scale was similar to the critical method. However, the value of N truly impacted the run time. When N increased, it cost more time to complete the program. All in all, the Monte Carlo Integration is useful because it provides me an accurate value of the estimated e.



```
For N0 :
Number of Threads : 1
N: Generated 1000000 points
The actual e is 2.718282
The estimated e is 2.716390
The difference between the actual e and the estimated e is 0.001892
Time : 0.114277 seconds
Number of Threads : 4
N: Generated 1000000 points
The actual e is 2.718282
The estimated e is 2.719984
The difference between the actual e and the estimated e is -0.001702
Time : 0.102779 seconds

For 10 N0 :
Number of Threads : 1
N: Generated 10000000 points
The actual e is 2.718282
The estimated e is 2.719080
The difference between the actual e and the estimated e is -0.000799
Time : 1.382318 seconds
Number of Threads : 4
N: Generated 10000000 points
The actual e is 2.718282
The estimated e is 2.718245
The difference between the actual e and the estimated e is 0.000037
Time : 1.026136 seconds

For 100 N0 :
Number of Threads : 1
N: Generated 100000000 points
The actual e is 2.718282
The estimated e is 2.718661
The difference between the actual e and the estimated e is -0.000379
Time : 13.825248 seconds
Number of Threads : 4
N: Generated 100000000 points
The actual e is 2.718282
The estimated e is 2.718037
The difference between the actual e and the estimated e is 0.000245
Time : 10.257453 seconds
```