CS575:Parallel Programming
Numeric Integration (project#2)

Student Name: Ping-Jui Liao
Student Email: liaop@oregonstate.edu
Project Number: 2
Project Name: Numeric integration

I do monte-carlo simulation with
NUMT (Number Of Threads) : 1 2 4 8 threads
NUMNODES (Number Of NODES) : 10 20 50 100 200 500 1000 2000 nodes

I implemented this by editing .cpp file and using bash script to run multiple times.

1. **Tell what machine you ran this on**

   I ran it on Ubuntu 18.04 on my laptop.

```
/numeric-integration
$ uname -a
Linux pingjuicybersecuritygram-15Z980-U-AAS5U1 4.18.0-17-generic #18~18.04.1-Ubuntu
 SMP Fri Mar 15 15:27:12 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux
```

2. **What do you think the actual volume is?**

   It should be around 28. For 100 tries I get a value around 28.6883

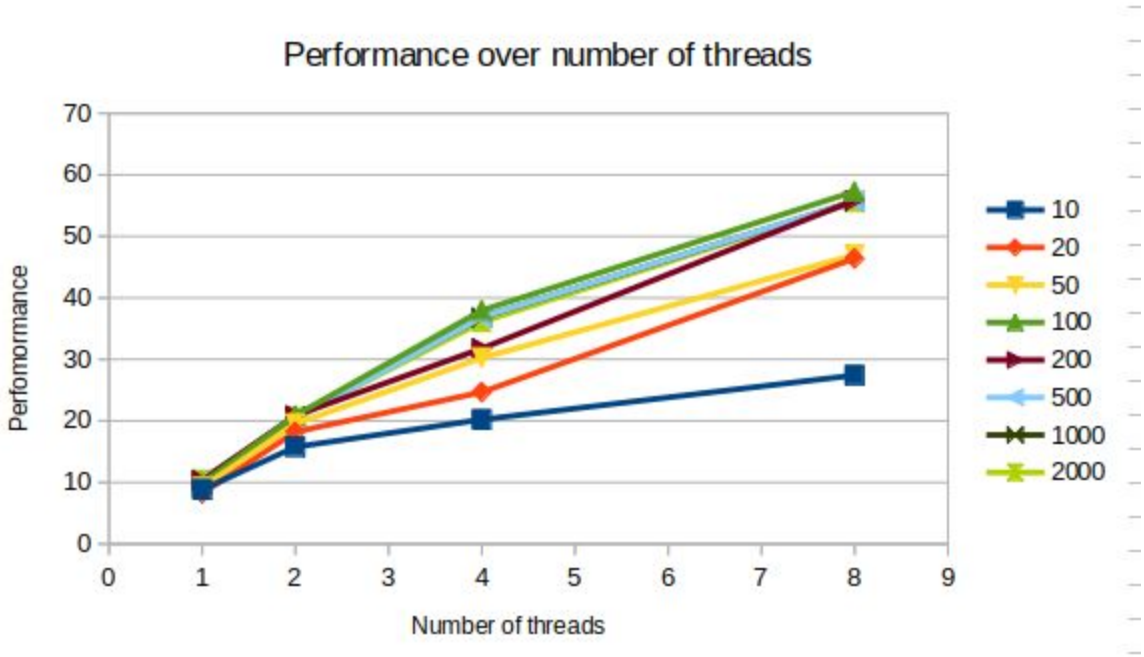
3 . Show the performances you achieved in tables and graphs as a function of NUMNODES and NUMT

1.

Table :

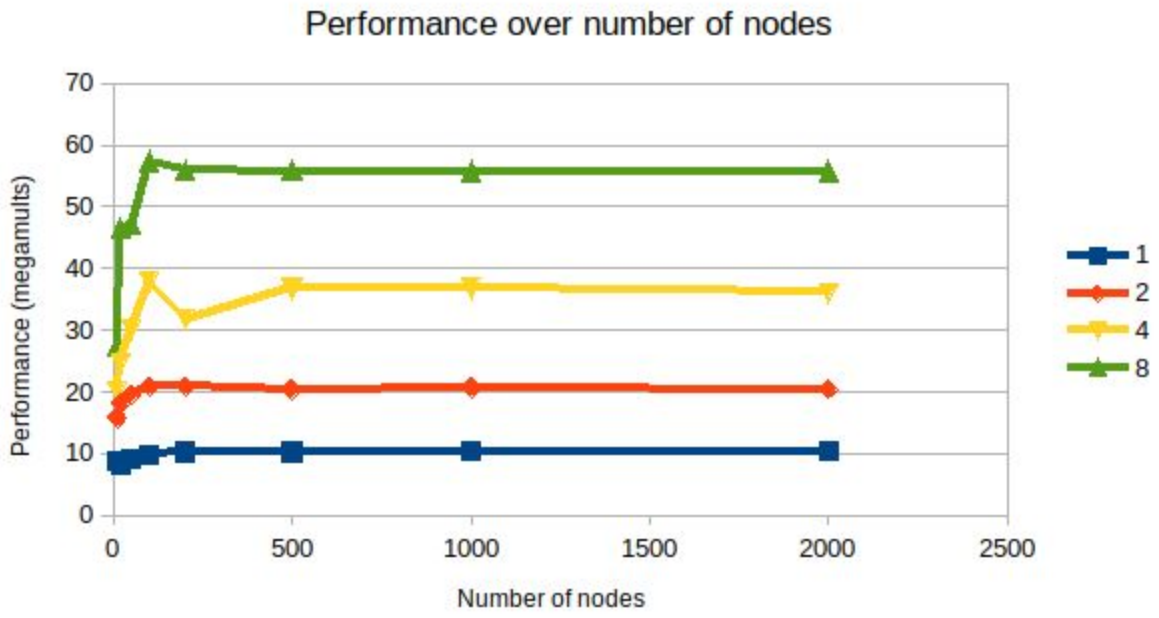
	10	20	50	100	200	500	1000	2000
1	8.82846	8.20159	9.19882	9.82623	10.2696	10.2895	10.4078	10.3724
2	15.7604	18.2083	19.5149	20.8732	20.877	20.2907	20.6855	20.4751
4	20.2224	24.6639	30.2404	37.9649	31.7708	36.8573	36.8185	36.0926
8	27.4124	46.4037	46.9889	57.3033	55.8786	55.7161	55.7048	55.6

Comparison Between Different NUMNODES
Color : Number of Nodes
Y axis: Performance
X axis: Number of Threads



The Number of thread do not have much influences on performances

Comparison Between Different NUMNODES
Color : Number of Threads
Y axis: Performance
X axis: Number of Nodes



As the thread number increases, the performance are significantly better.

4. What patterns are you seeing in the speeds?

As the threads increase, despite some anomaly data, the performance gets better.

The nodes have no effect on performance.

5. Why do you think it is behaving this way?

I believe my numbers are weird due to cache misses. My numnodes size are not set to 2 to the N so that the cache line cannot encapsulate all the data it needs for the calculation.

6. What is the Parallel Fraction for this application, using the Inverse Amdahl equation?

Calculating the Fp as follow:

Speedup(8 thread vs. 1 thread, NUMNODES=2,000) = P8/ P1
= 55.6000 / 10.3724
= 5.360379

Fp = 8/7 * (1. - (1. / 5.360379)
= 0.92965262076053

```
>>> speedup = 55.6 / 10.3724
>>> speedup
5.360379468589719
>>> numt = 8.
>>> fp = (numt/(numt-1.)) * (1.-(1./speedup))
>>> fp
0.9296526207605345
>>>
```

7. Given that Parallel Fraction, what is the maximum speed-up you could *ever* get?

Lim Speedup = 1 / (1 - 0.92965262076) = 14.21517