Project #1
OpenMP: Numeric Integration with OpenMP

By Michael Rose

Due: April 19th 2016

Written for: CS475 Spring 2016

Oregon State University

Table of Contents

Project Background Information and Calculated Volume

This project was run on my local machine which contains an intel i5 core. Based on the results that I got from running this using a variety of threads and divisions, I believe that the volume is around 14.1. The implementation of this project was simple. To actually calculate the area, I split the for loop using #pragma omp parallel for reduction(+:volume). Within the loop, I checked to see if iu was either 0 or NUMS-1 to find out if it was on an edge. If it was, I decreased my area multiplier by half. I did the same thing with iv, allowing this code to handle both edges and corners. To try and stabilize my millions of volume calculations per second, I did 10 runs with each NUMS and NUMT.

Results

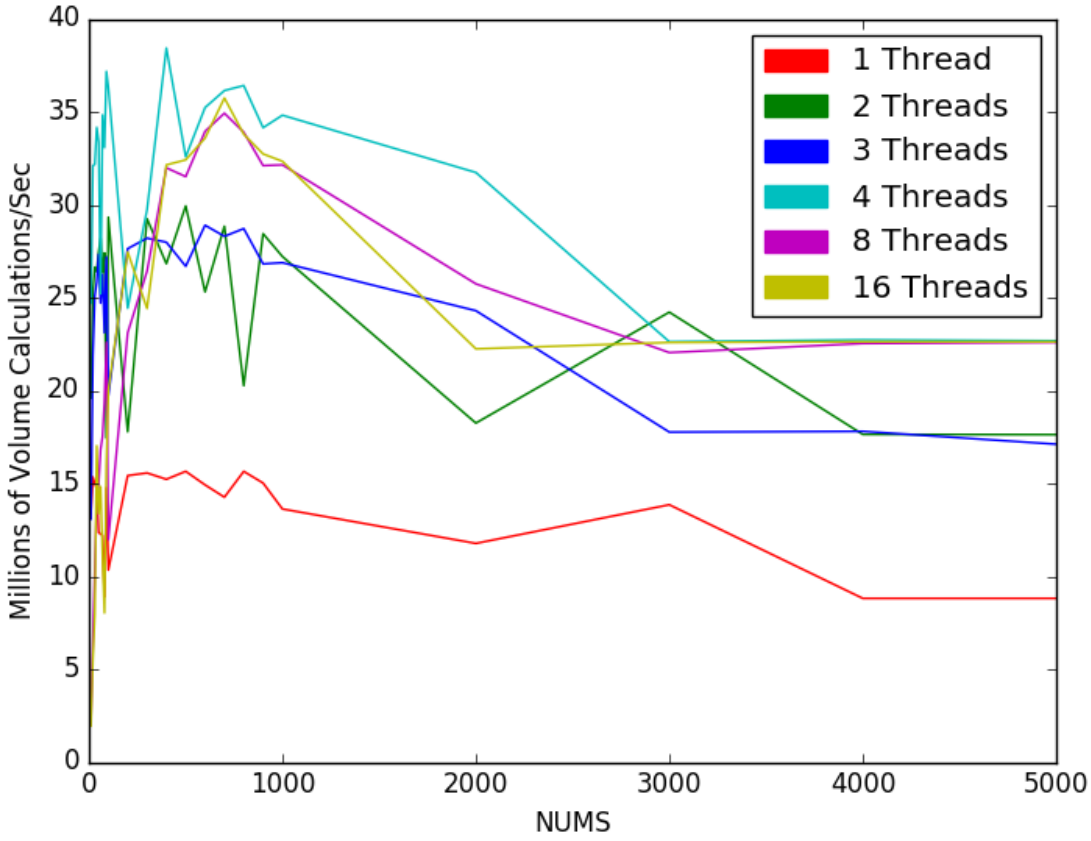

Figure 1-Overview of all the datapoints.

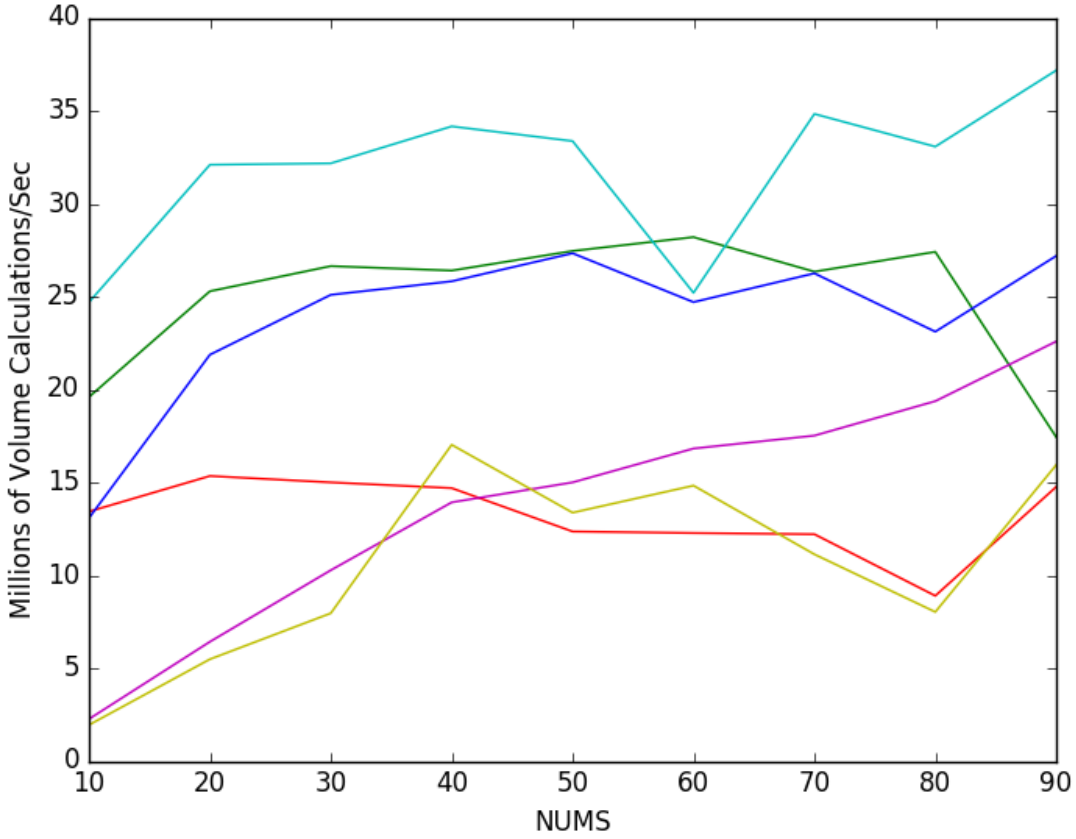

Figure 2- A zoomed in view using NUMS from 10 to 90. The legend is the same as above.

Based on the graphs, going from really small numbers of NUMS to between 500 and 1000 has a great increase. At that point the capacity of my processor is reached and the performance starts to decrease. 4 threads, which is also the number of cores that my computer has is the line that has the least decreased performance over time, but eventually reaches the same performance as more than 4 threads. I thought it was interesting that 2 and 3 threads ended up around the same performance. I believe that this behavior is caused by the increase of number of computations passed the point of maximum CPU usage.

SpeedUp and Parallel Fraction

Included below is a table that lists my results for volume, millions of volume calculations per second and speedup for NUMS = 5000 for a variety of numbers of threads. I decided to run 32, 64 and 128 threads just for fun and included those values. These values are not shown in the graphs above because they basically show exactly the same thing and just end up cluttering the graph.

NUMT	NUMS	Volume	Millions of Volume Calculations/Sec	Speedup
1	5000	14.34207	8.812985	
2	5000	14.039	17.624985	1.999888
3	5000	14.19093	17.110653	1.941528
4	5000	14.04985	22.676784	2.57311
8	5000	14.03398	22.589754	2.563235
16	5000	14.08005	22.623029	2.567011
32	5000	14.06054	22.672312	2.572603
64	5000	14.0632	22.615123	2.566114
128	5000	14.06261	22.654763	2.570612

For the parallel fraction, I am going to use the speedup from 1 to 4 threads because that is the number of cores that my machine has.

$$Speedup = \frac{22.676784}{8.812985} = 2.57311 = \frac{1}{\frac{F_{Parallel}}{5000} + (1 - F_{Parallel})}$$

$$F_{Parallel} = .61149$$

Max Speedup = $\frac{1}{1 - F_{Parallel}} = 2.57392$