Trevor Stahl
Project 2
CS475 X400 S2019
Professor Bailey

openMP: Numeric Integration
Volume between two Bezier Surfaces

Tell what machine you ran this on:
I ran my code on my 2017 MacBook Air
MacBook Air (13-inch, 2017)
2.2 Ghz intel core i7
8 GB Ram

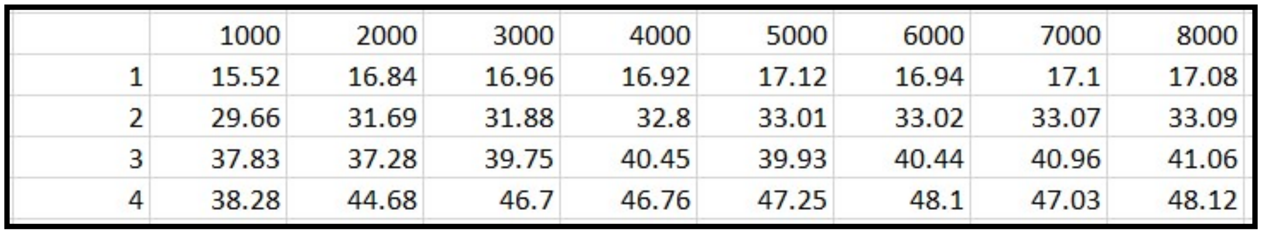
Compiled with g++-8 -o proj2 project2.cpp -fopenmp -lm
g++-8 is my prefix for Homebrew GCC 8.3.0

What do you think the actual volume is?
~28.69

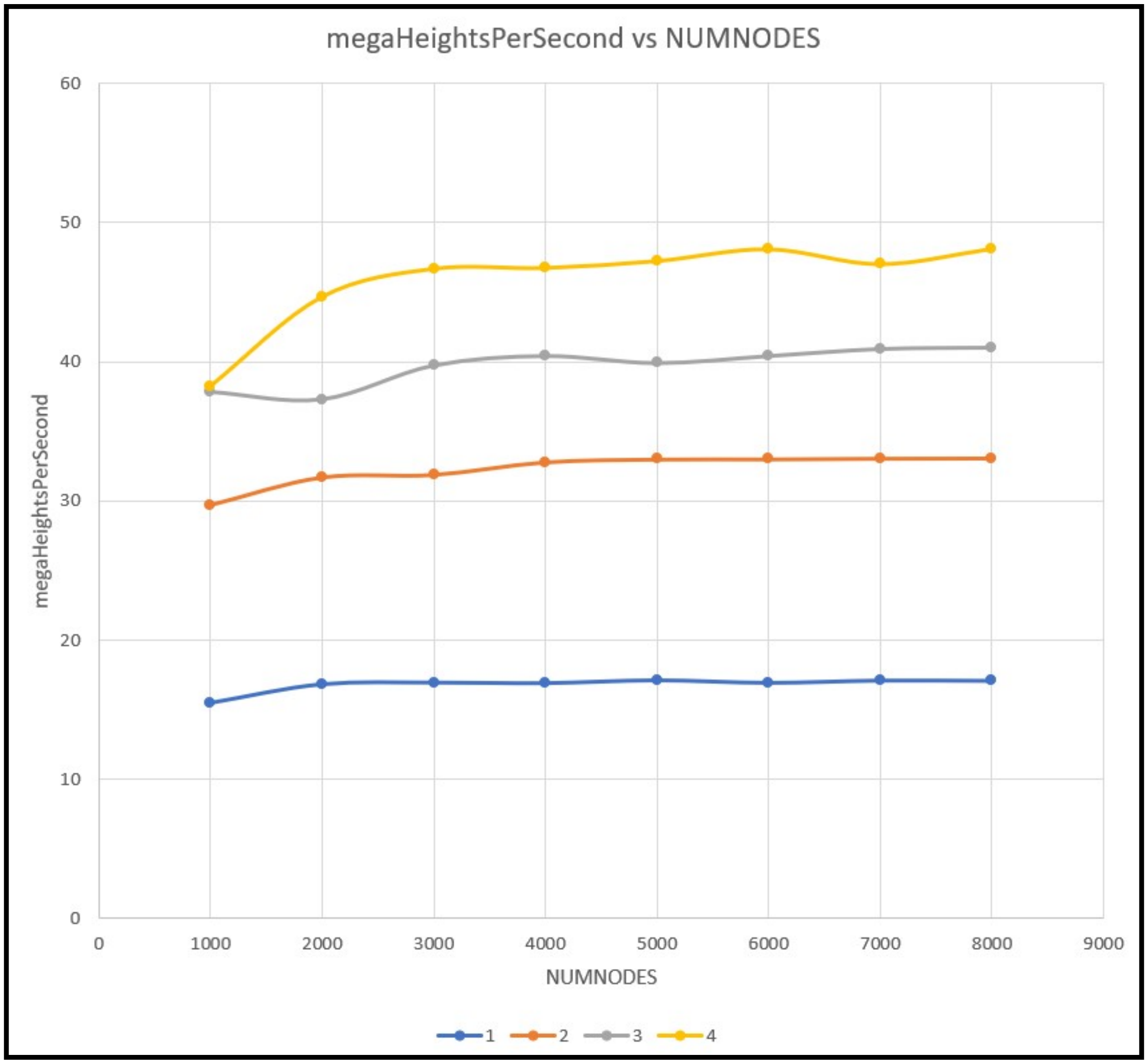
All volumes found were:
(1000 to 8000 by 1000 increments for NUMNODES horizontally)
(1, 2, 3, 4 NUMT vertically)

28.68	28.67	28.85	28.54	29.13	30.96	28.28	20.46
28.68	28.67	28.67	28.47	28.30	29.94	29.68	27.90
28.68	28.68	28.64	28.61	29.01	28.70	28.55	30.51
28.68	28.69	28.66	28.56	28.82	28.84	28.33	29.72

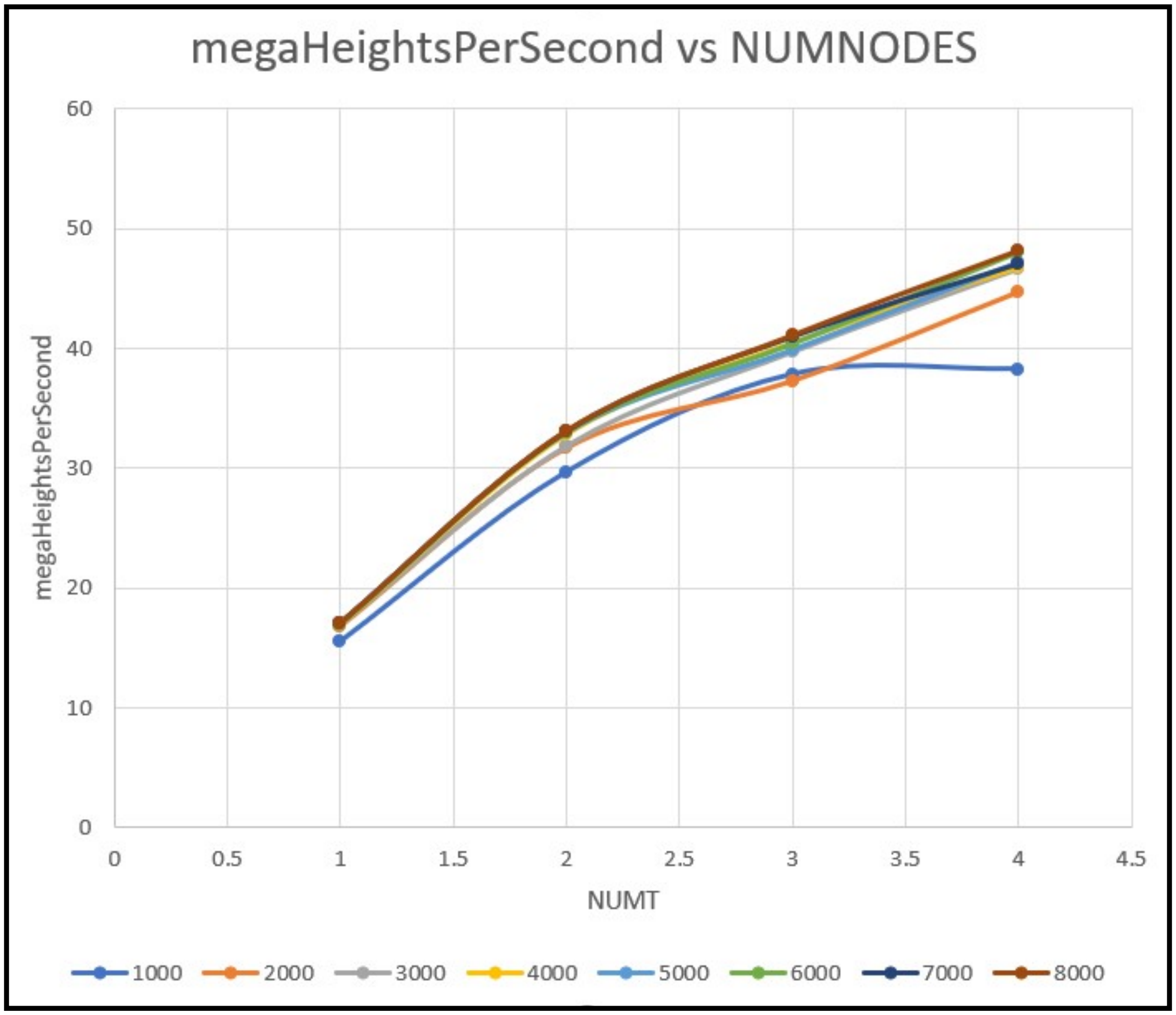
Show the performances you achieved in tables and graphs as a function of NUMNODES and NUMT:
Using NUMNODES values of: 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000
Using NUMT values of: 1, 2, 3, 4
Performance recorded in megaHeightsPerSecond



	1000	2000	3000	4000	5000	6000	7000	8000
1	15.52	16.84	16.96	16.92	17.12	16.94	17.1	17.08
2	29.66	31.69	31.88	32.8	33.01	33.02	33.07	33.09
3	37.83	37.28	39.75	40.45	39.93	40.44	40.96	41.06
4	38.28	44.68	46.7	46.76	47.25	48.1	47.03	48.12



megaHeightsPerSecond vs NUMNODES
megaHeightsPerSecond
NUMNODES
0
10
20
30
40
50
60
0
1000
2000
3000
4000
5000
6000
7000
8000
9000
1
2
3
4



What patterns are you seeing in the speeds?
As NUMNODES increases performance increases.
As NUMT increases performance almost increases by the factor the NUMT increased.

Why do you think it is behaving this way?
Having larger NUMNODES value means that the parallel fraction becomes greater.
Performance increases with NUMT in this way because there is twice as much compute power available per second however there is the overhead of spinning up the extra thread. This is also why lines in graph two directly above criss cross each other.

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

Speed up for 1 thread to 4 threads NUMNODES = 8000:
Performance of 4 threads/performance of 1 thread =
2.8173

Inverse Amdahl is:
1 / Speedup = (Fp/n) + Fs = (Fp/n) + (1-Fp)

Fp = (4.0/3.0)*(1.0 - (1.0/2.8173)) = 0.86

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

MaxS = 1 / (1-0.86) = 1 / 0.14 = 7.143