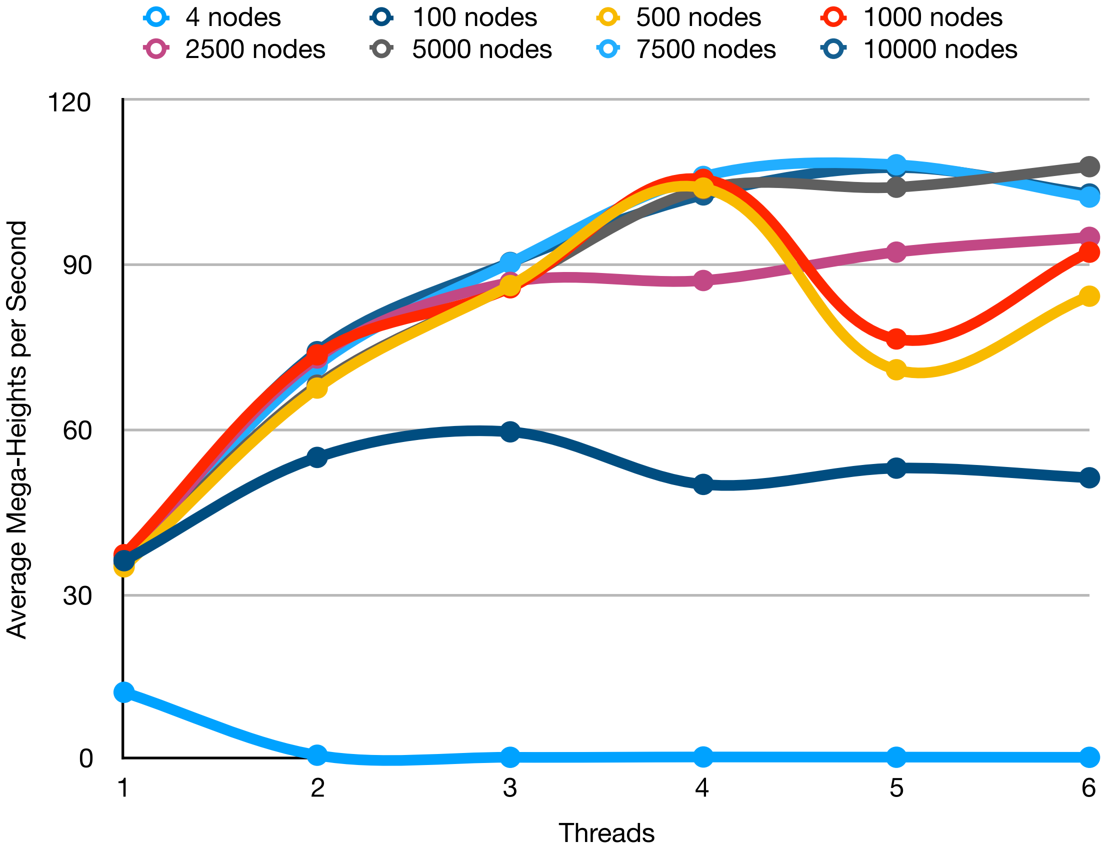
Aaron Berns
Project 1: OpenMP: Numeric Integration with OpenMP

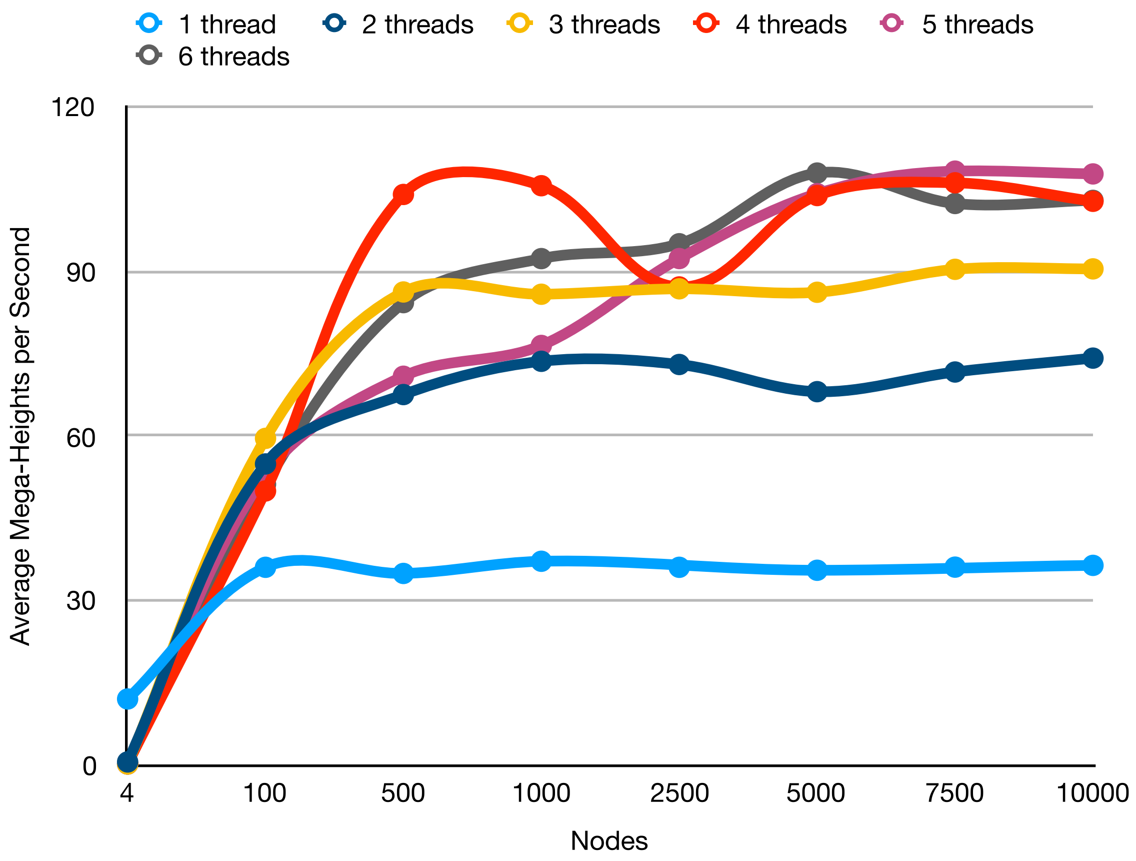
1. Machine: MacBook Pro, i5, 2 cores

2. The calculated volume leveled off at 25.3125 after 5000 nodes so I'll go with that.

Mega-Heights per Second

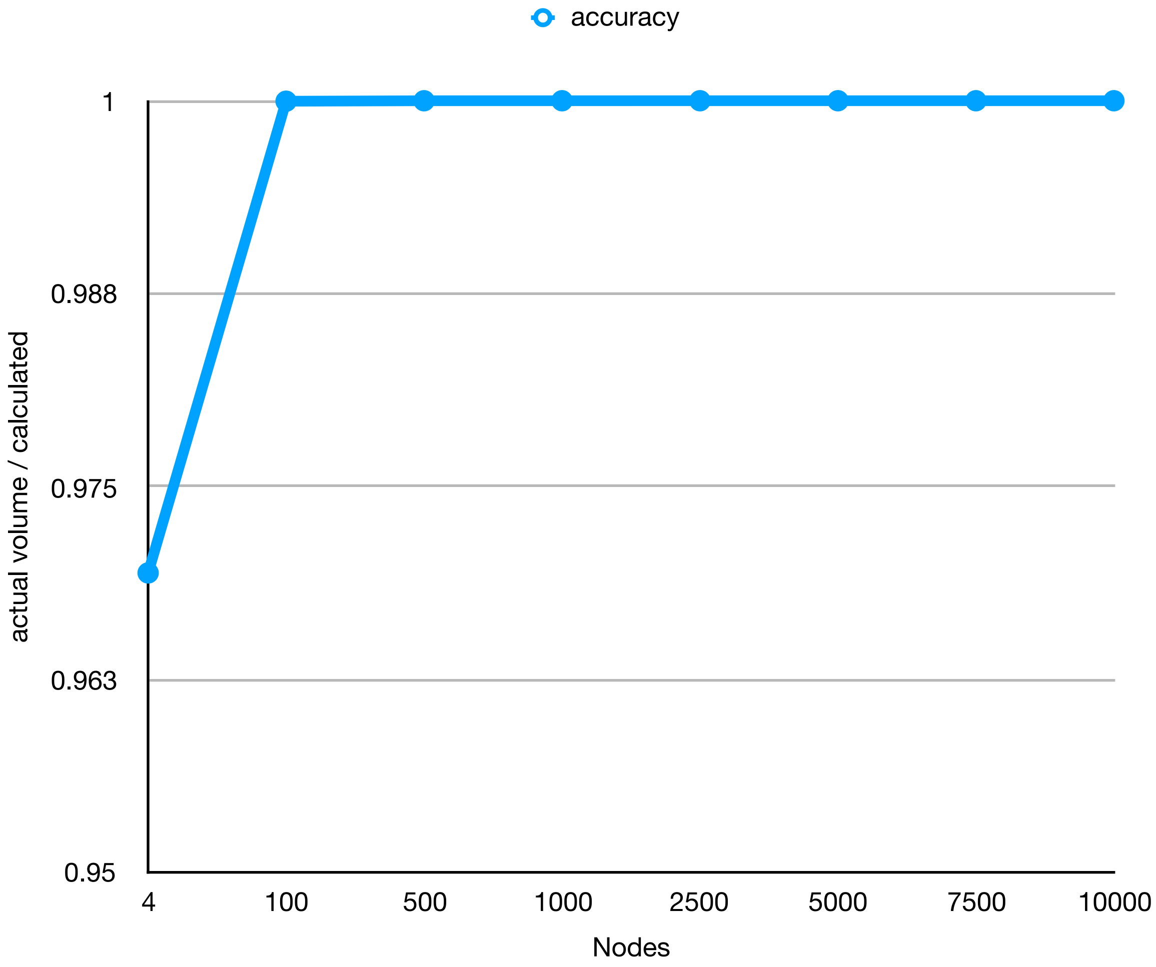
	4 nodes	100 nodes	500 nodes	1000 nodes	2500 nodes	5000 nodes	7500 nodes	10000 nodes
1 thread	12.10	36.09	34.97	37.19	36.09	35.55	36.12	36.44
2 threads	0.63	54.92	67.56	73.62	73.04	68.09	71.67	74.20
3 threads	0.25	59.58	86.25	85.84	86.83	86.20	90.38	90.46
4 threads	0.31	50.00	104.02	105.61	87.17	103.82	106.14	102.77
5 threads	0.26	52.96	70.90	76.51	92.34	104.17	108.27	107.76
6 threads	0.25	51.21	84.31	92.33	95.04	107.92	102.37	102.94





Calculated Volume Accuracy

	4 nodes	100 nodes	500 nodes	1000 nodes	2500 nodes	5000 nodes	7500 nodes	10000 nodes
volume	26.111111	25.313303	25.312532	25.312508	25.312501	25.312500	25.312500	25.312500
percentage of actual	0.969414934868721	0.999968277549556	0.999998735804067	0.999999683950717	0.999999960493829	1	1	1



4,5. Using more than one thread leads to worse performance for small numbers of nodes, less than 75 or so. This most likely happens because the overhead of creating, running and maintaining multiple threads uses more resources than simply completing the calculations. At 100 nodes, there is improved performance with 2 or more threads, but the number of threads above 1 doesn't really make a difference. This most likely occurs because the amount of work that can be parallelized isn't large enough to divvy up yet. Around 500 nodes, having 4 threads leads to the the best performance, with more or less falling short. This might be because having two threads running per core on this machine is optimal. The core i5 processor doesn't have hyper-threading so I'm not sure why 4 threads is so much better than 2. It does suggest that the two cores are not as busy when they are only dealing with one thread each. I'm not sure about the subsequent dip in performance for 4 threads between 1000 and 5000 nodes or the lack of performance of 5 or 6 threads from 100 to 2500 but by the time the number of nodes becomes really large, around 10000, it's clear that 4,5 or 6 threads give the best performance over 1,2 or 3. The difference between 4, 5 and 6 threads is minimal. Using more than 10000 nodes causes the area to be divided into fractional pieces that are too small to store with double precision, leading to increasing incorrect answers.

6. Speedup with 2 threads (one thread per core on this machine) = 2.73 / 1.376 = 1.98
Parallel Fraction with 2 threads = (2) * (1 - (1/1.98)) = 0.9898

Speedup with 6 threads = 2.73 / 0.92 = 2.97
Parallel Fraction with 6 threads = (6/5) * (1 - (1/2.97)) = 0.796

7. The maximum Speedup based on 2 threads = 1 / (1 - Fp) = 1 / (1 - 0.9898) = 98.99
   The maximum Speedup based on 6 threads = 1 / (1 - Fp) = 1 / (1 - 0.796) = 4.9