HW 3
Numerical Integration with Posix threads

Carlos Vazquez Gomez

March 18, 2019

1 Code

The code for this assignment can be found on GitHub.

2 Strong Scaling Analysis

In order to see the effects of different physical core numbers on the strong scaling, I ran my program on both the H2P Cluster login node which has 12 cores (2 CPU sockets with 6 cores per socket, as printed by the 'lscpu' linux command), and on my VirtualBox Ubuntu Linux VM with 2 cores (1 socket). Figure 1 shows the strong scaling efficiency curves for both machines. Strong scaling efficiency is defined as:

$$\text{Efficiency} = \frac{T_{\text{serial}}}{T_{\text{parallel}} \cdot \text{NumThreads}}$$

and requires that the number of total points to integrate remains constant as thread count increases. We can observe that in the H2P login node case, once the number of threads surpasses the number of physical cores (12), the efficiencies begin to approximately half as the thread count doubles. In the Linux VM with 2 cores, however, the efficiency drops all the way to 0.67 with only two threads, leading us to believe that the one of the cores is busy with another task. Interestingly, in the H2P login node, even with more than 12 threads the performance improves (albeit slightly). But in the Linux VM, running with more than 2 threads causes the total time to grow, most likely because of mutex contention and thread scheduling overhead.

# Threads	Time (s)	Efficiency
1	8.528	1 (benchmark)
2	4.873	0.875
4	2.847	0.749
8	1.717	0.621
16	1.198	0.445
32	1.027	0.260
64	0.968	0.138

Table 1: Running with 4 billion on H2P Cluster Login Node with 12 cores

# Threads	Time (s)	Efficiency
1	1.544	1 (benchmark)
2	1.146	0.673
4	1.144	0.337
8	1.148	0.168
16	1.871	0.052
32	1.246	0.039
64	1.775	0.014

Table 2: Running with 400 million points on my Ubuntu Linux VM with 2 cores

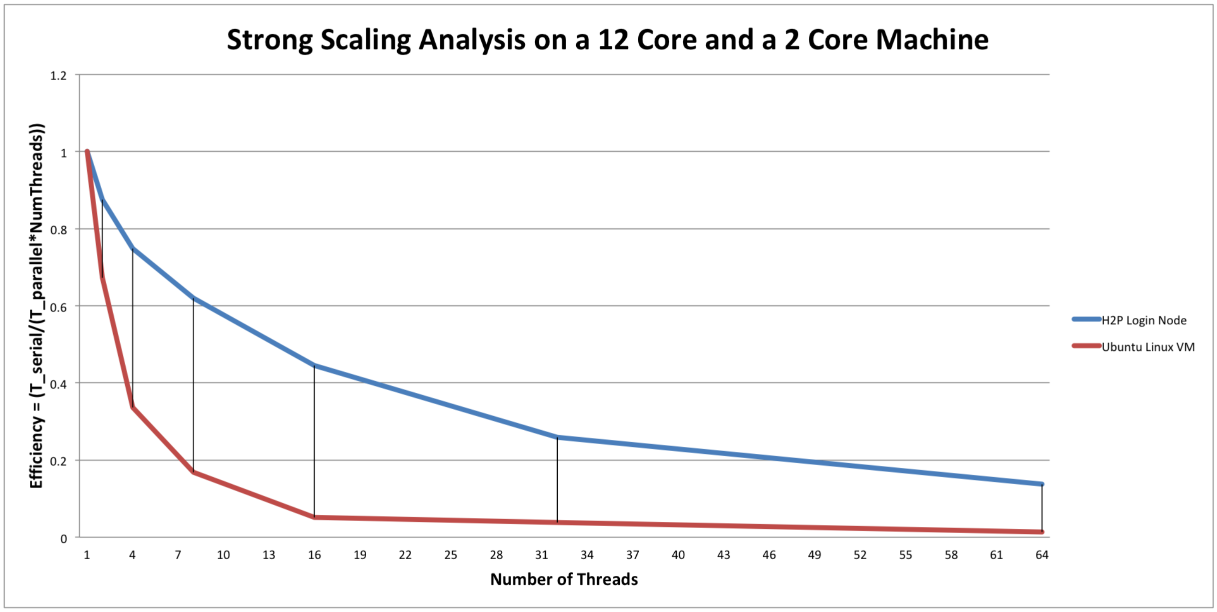

Figure 1: Strong scaling analysis