Assignment: Numerical integration and Shared-memory Parallelization

5. Scaling Analysis

CSCD92

Harri Pahirathan

2020-09-09

Introduction:

"In this section, we perform a scaling analysis to compare the strong and weak scaling samples and determine whether or not the parallel code is functioning normally or strangely, and determine the optimal parameters for optimal efficiency. To make the specifics of our parallel code more understandable, we will use a table and a graphical depiction. Because efficiency is key in parallel computing, scalability is a crucial consideration."

Table 1: Weak Scaling

Threads	Num of Partition	Time
1	100	0.0001454 sec
2	200	0.0002399 sec
4	400	0.0002736 sec
8	800	0.0004299 sec
16	1600	0.004381 sec

- We can see the difference in time has increased over the increase of partition and threads if we take a look at Graph 1, we can see an exponential trend. As the workload and threads increase linearly, the time increases slightly.

Graph 1: Weak Scaling Graph

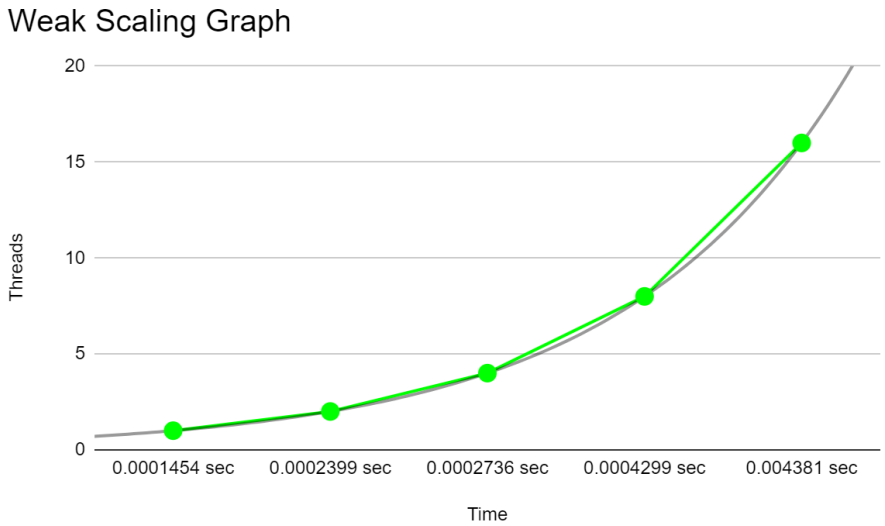
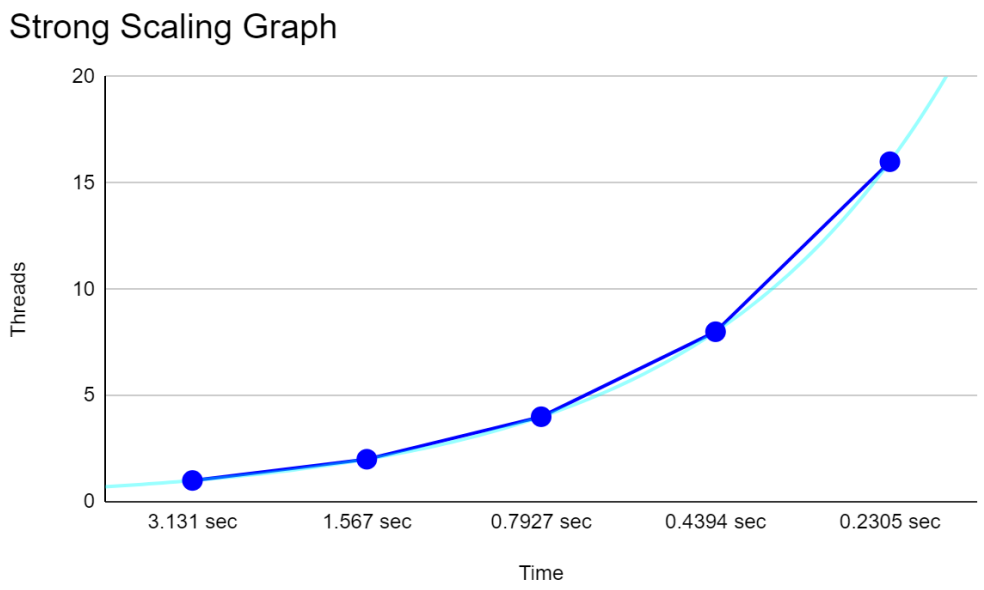


Table 2: Strong Scaling

Threads	Num of Partition	Time
1	100000000	3.131 sec
2	100000000	1.567 sec
4	100000000	0.7927 sec
8	100000000	0.4394 sec
16	100000000	0.2305 sec

- We can see has the partitions stay fixed and the threads increase the time it takes to compute parallelly is less as seen on the Graph 2. As the workload stays the same the increase of threads does compute the code faster but will be marginal as it keeps increasing threads.

Graph 2: Strong Scaling Graph



Conclusion:

"After observing the weak scaling, which shows that the partition and thread will lengthen the time required, and the strong scaling, which shows that the fixed partition and thread increase will shorten the time required to a marginal level, we can draw the conclusion that we can determine the ideal number of threads and workload required to run our code at the ideal level of efficiency. Finding the most effective settings for any code that can be parallelized is a key component of scalability."