Parallel Computing Lab2

Chuanyang Jin

April 26, 2024

Times (seconds):

N	1	2	4	8
# unknowns 8	0.000077	0.000918	0.001900	0.003133
16	0.000128	0.001306	0.001872	0.003086
32	0.000414	0.000626	0.002217	0.003371
64	0.001278	0.001872	0.002533	0.003064
128	0.004242	0.003045	0.002896	0.004305
256	0.016130	0.008985	0.006552	0.005744
512	0.064784	0.034976	0.020864	0.015886
1,024	0.271344	0.136500	0.070721	0.041414

Speedup:

N	1	2	4	8
# unknowns 8	1	0.083878	0.040526	0.024577
16	1	0.098009	0.068376	0.041478
32	1	0.661342	0.186739	0.122812
64	1	0.682692	0.504540	0.417102
128	1	1.393103	1.464779	0.985366
256	1	1.795214	2.461844	2.808148
512	1	1.852242	3.105061	4.078056
1,024	1	1.987868	3.836824	6.551987

Efficiency:

N	1	2	4	8
# unknowns 8	1	0.041939	0.010132	0.003072
16	1	0.049005	0.017094	0.005185
32	1	0.330671	0.046685	0.015352
64	1	0.341346	0.126135	0.052138
128	1	0.696552	0.366195	0.123171
256	1	0.897607	0.615461	0.351019
512	1	0.926121	0.776265	0.509757
1,024	1	0.993934	0.959206	0.818998

a) What is the trend you see in Tables 1 and 2 (expected to be the same trend)?

With a fixed number of unknowns, when the number of unknowns is large, the time decreases and the speedup increases as the number of threads increases; when the number of unknowns is small, the time increases and the speedup decreases as the number of threads increases.

With a fixed number of threads, the time increases and the speedup increases as the number of unknowns increases.

b) What is your interpretation of that trend?

With a fixed number of unknowns, when the number of unknowns is large, the time decreases and the speedup increases as the number of threads increases because the computational work can be effectively distributed among the threads, leading to faster execution times, while the overhead is relatively minor.

With a fixed number of unknowns, when the number of unknowns is small, the time increases and the speedup decreases as the number of threads increases because the overhead of managing multiple threads can outweigh the benefits gained from parallel computing.

With a fixed number of threads, increasing the problem size leads to longer execution times, and also results in greater speedup due to enhanced parallel computing benefits.

c) What is the trend you see in Table 3 (expected to be the same trend)?

With a fixed number of unknowns, the efficiency decreases as the number of threads increases.

With a fixed number of threads, the efficiency increases as the number of unknowns increases.

d) What is your interpretation of that trend?

With a fixed number of unknowns, as the number of threads increases, each thread has less work to do, so the efficiency decreases due to the increased proportion of overhead of managing multiple threads.

With a fixed number of threads, as the number of unknowns increases, the efficiency increases because each thread can handle more work without being idle.