Merge Sort in OpenMP

JIALI CLAUDIO, HUANG Personal code: 11032111
Peng, Rao Personal code: 11022931

Experimental setup

We have implemented the merge sort algorithm in C++ using OpenMP. The code is compiled using the GCC compiler. The code is run on a machine with the following specifications:

Table 1: Experimental setup

Machine	MacOS with Apple M1 pro chip(8 cores)
Compiler Option	g++14 -fopenmp -O3
OMP_NUM_THREADS	10
OMP_SCHEDULE	static

Performance measurements

We have measured the performance of the merge sort algorithm by varying the size of the input array and the deep of the Parallelism. The performance is measured in terms of the execution time of the algorithm. The results are shown in the following tables and figures.

Table 2: Execution time of merge sort algorithm with different input sizes and depth

Input size	Depth	Execution time (s)
10^3	5	0.005392
10^3	8	0.000148
10^3	10	0.000151
10^4	5	0.005111
10^4	8	0.000254
10^4	10	0.000414
10^5	5	0.000947
10^5	8	0.003079
10^5	10	0.001315

Array Size	Depth	Time (seconds)
10^6	5	0.012315
10^6	8	0.009194
10^6	10	0.009971
10^7	5	0.135189
10^7	8	0.103922
10^7	10	0.10051
10^8	5	1.47982
10^8	8	1.29516
10^8	10	1.13882

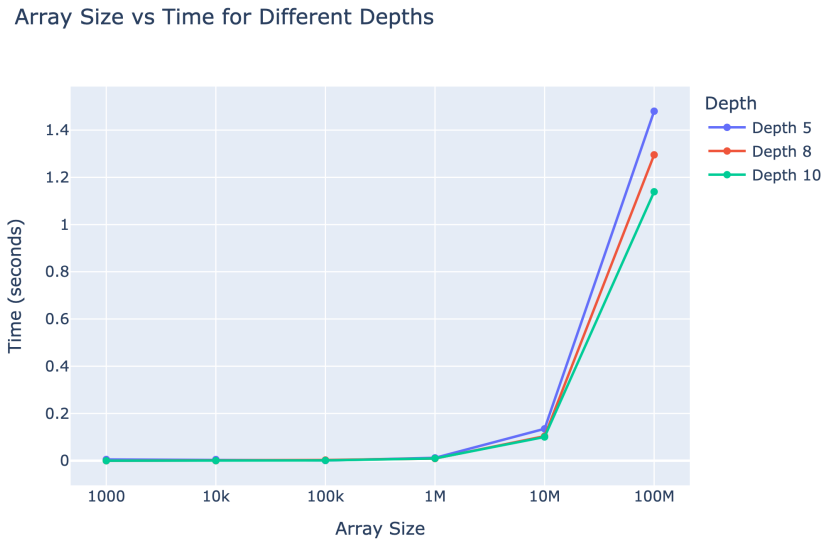

Figure 1: Execution time of merge sort algorithm with different input5sizes and depth

Explanation of design choices

The merge sort algorithm is a divide-and-conquer algorithm that including two main steps: divide and merge. So we can parallelize the divide step and merge step separately. This is the method of *Introduce to Algorithm*[1]

Design of OpenMP parallelization:

- We used the omp task method for parallelization. Because of the iteration of recursion is unknown, so we used the omp task method for parallelization.
- Use depth limited mechanism to prevent too much task generated to block threads.
- When the array size <= 800, it is more efficient sort by serial, so we used the cutoff mechanism to get better performance.
- Use omp_get_max_threads() function to get threads dynamically so that it can perform well in different computer.
- In the merge sort algorithm, the scale of each recursion is similar, so we set schedule to static to avoid unnecessary operate.

Design of the division:

- We divided the array into two parts and then recursively divided the two parts into two parts until the size of the array is less than or equal to 800.
- We used the omp task method to parallelize the division step.
- We used the depth limited mechanism to prevent too much task generated to block threads.

Design of the merge:

- We used the omp task method to parallelize the merge step.
- We used the depth limited mechanism to prevent too much task generated to block threads.
- We used binary search to find the position of the element in the right part of the array.

Bibliography

[1] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. *Introduction To Algorithms*. Mit Electrical Engineering and Computer Science. MIT Press, 2001.