High Performance Computing
Homework 10

Sara Restrepo Velasquez and Dai Nam Nguyen
April 23rd, 2024

For this assignment, the **Linear Search** and **Binary Search** algorithms were implemented in the Fortran module file, **searchutils.mod**. Additionally, the do loop of **Linear Search** is parallelize with OpenMP. This module is used in the **main_program.f90** script with a do loop is used in the script to set OMP threads number (1,2,4,8 and 16), which was compiled with the **make** command and executed through the **./main_program.exe** command. The results of the execution were saved into the **results.txt** file.

In the case of **Linear Search** using OpenMP to parallelize, the speedup goes up along with the increased of OMP threads number, with 16 threads the speedup reaches 7.46. The efficiency peaks at around 1.39 with 2 threads but declines afterwards, reaching 0.46 at 16 threads as shown in Table 1.

Table 1: Linear Search execution time, speed up and efficiency with sorted array.

OMP threads number	CPU time (s)	Speedup	Efficiency
1	3.47E-02	1	1
2	1.24E-02	2.79876185	1.39938092
4	1.15E-02	3.02751492	0.75687873
8	7.44E-03	4.66307899	0.58288487
16	4.65E-03	7.46867785	0.46679237

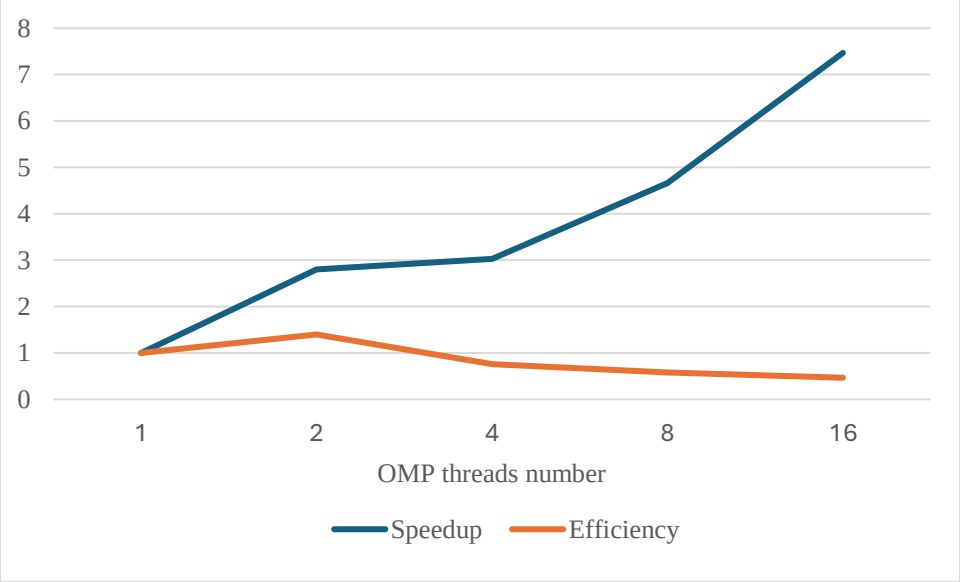

Figure 1: Linear Search speedup and efficiency.

The CPU time for using **Binary Search** is 1.91E-006 which is about 2436 times faster than **Linear Search** parallelize using 16 threads. Even though the **Linear Search** speed is improved by parallelizing, **Binary Search** algorithm is superior in case of sorted array.