MERGESORT MPI

1. Cerintele

Proiectul pe care l-am dezvoltat se axeaza pe algoritmul mergesort. Acesta reprezinta unul dintre cele mai eficiente metode de sortare a unui array, bazandu-se pe tehnica Divide and Conquer, astfel se imparte array-ul initial in doua parti egale, se sorteaza fiecare sub-array in parte, dupa care cele doua sunt reunite si sortate in final.

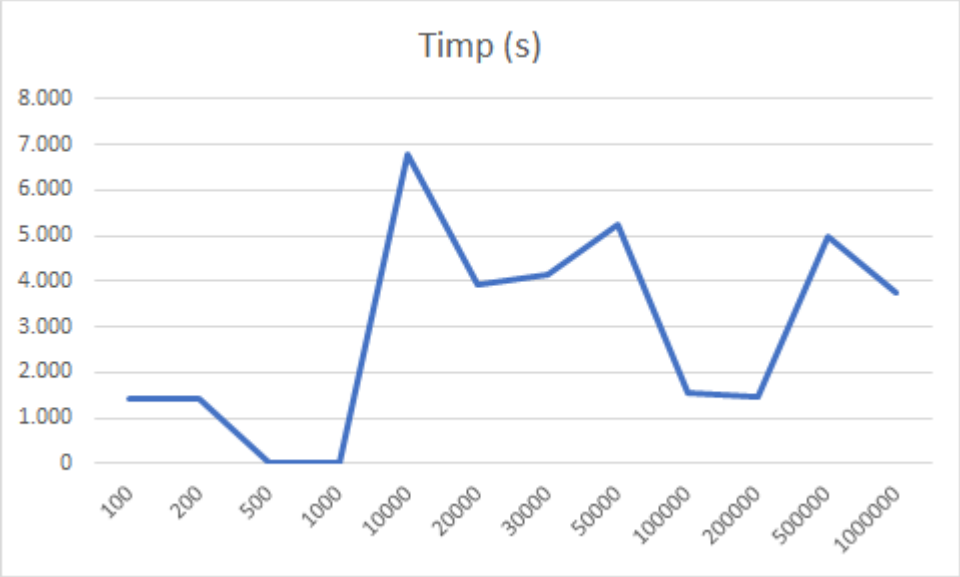
Pentru implementarea acestui algoritm am folosit MPI.

2. Informatii in ceea ce priveste rularea codului

Codul a fost rulat pe urmatorul sistem:

- **Operating System**: Windows 10 Pro 64-bit
- **System Manufacturer**: HP
- **System Model**: OMEN 25L Desktop GT12-0xxx
- **BIOS**: F.12
- **Processor**: AMD Ryzen 7 3700X 8-Core Processor (16CPUS), ~3,6GHZ
- **GPU**: NVIDIA GeForce RTX 2060 SUPER
- **Memory**: 16384MB RAM
- **DirectX Version**: DirectX 12

3. Rezultate experimentale



Teste	Dimensiune	Timp de executie (s)
1	100	1.399972
2	200	1.412402
3	500	0.594642
4	1000	0.870897
5	10000	6.772418
6	20000	3.918946
7	30000	4.144114
8	50000	5.247893
9	100000	1.537559
10	200000	1.468945
11	500000	4.986279
12	1000000	3.755118

4. Eficienta

Determinarea algoritmului mai eficient intre mergesort secvential si mergesort paralel depinde de dimensiunea vectorului de intrare si de numarul de procesoare disponibile pentru algoritmul paralel.

Pentru vectori de dimensiuni mici, mergesort secvential este, de obicei, mai eficient, deoarece costul suplimentar de comunicare intre procese in mergesort paralel poate duce la o crestere a timpului de executie.

Pe de alta parte, pentru vectori de dimensiuni mari si numarul mare de procesoare disponibile, mergesort paralel poate fi mai eficient decat mergesort secvential. Acest lucru se datoreaza faptului ca impartirea datelor intre procesoare poate reduce timpul total de sortare si poate imbunatati performanta algoritmului.

In plus, mergesort paralel poate fi scalabil, ceea ce inseamna ca poate fi utilizat pentru a sorta vectori de intrare mult mai mari decat capacitatea unui singur nod de procesare.

In concluzie, eficienta algoritmului mergesort depinde de dimensiunea vectorului de intrare si de numarul de procesoare disponibile. Pentru vectori de dimensiuni mari si numarul mare de procesoare disponibile,algoritmul paralel poate fi mai eficient decat cel secvential.

5. Referinte:

https://www.geeksforgeeks.org/

https://chat.openai.com/