Algoritmi Paraleli si distribuiti

Bubble sort

Rezultate si concluzii

23.05.2023

CHINTESCU BOGDAN COSTINEL

CALCULATOARE ROMANA ANUL III

1. Introducere

Fiind un algoritm simplu de sortare care trece repetat prin lista de intrare element cu element, comparand elemental current cu cel de dupa el si interchimbandu-le valorile daca este necesar. Aceste treceri prin listă se repetă până când nu este necesară nicio interschimbare în timpul unei treceri, ceea ce înseamnă că lista a fost complet sortată. Algoritmul, care este o sortare prin comparație, este numit astfel datorită modului în care elementele mai mari "bulează" către partea de sus a listei.

Ca si performanta este una slaba in utilizarea reala, dar este unul dintre primii algoritmi de obicei predati. Timpul de rulare este un lucru important de luat in considerare, iar Bubble sort are un timp mediu si in cel mai rau caz de $O(N^2)$.

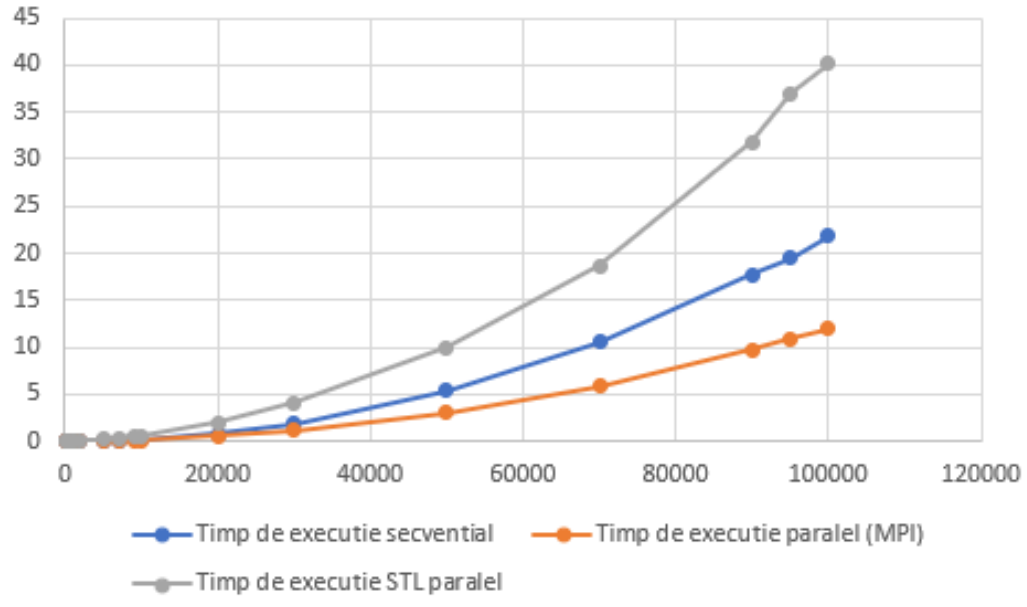
Testul algoritmului a fost executat pe urmatorul system:

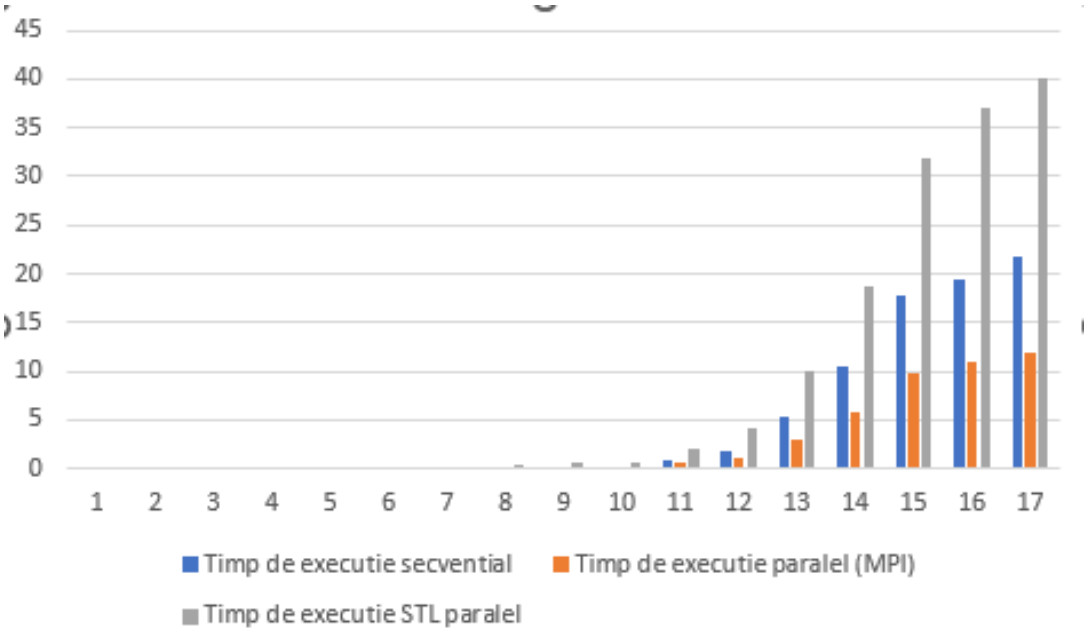
- SO: Windows 10 Pro, C++
- Procesor: Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz (12M up to 5.00 GHz)
- RAM: 16 GB DDR 4
- System type: 64-bit operating system, x64-based processor
- GPU: GeForce RTX 2060 6GB

2. Rezultate

Observatii

Rezultatele urmatoare au fost colectate in urma a 17 teste folosind un vector de o dimensiune intre 100 si 200000, cu o generare aleatoare de numere pana la 2^{32}. In urma a mai multor experimente se poate observa o dublare de fie a timpului de executie la sortarea unui vector de dimensiuni mai mici decat 10000, totusi timpul de executie fiind mai mici de 1 secunda, dar dupa un set de date mai mari de 2000 putem observa ca timpul de executie trece de 1 secunda putem observa cat de rapid creste timpul de executie fata de dimensiunea vectorului.





Se poate observa diferentele de timp si cum creste acest timp de executie in functie de dimensiunea vectorului si dovedind cat de inefficient este in cazul unei dimensiuni prea mari a vectorului.

Index	Dimensiune	Timp de executie secvential	Timp de executie Parallel(MPI)	Timp de executie STL paralel
1	100	0.000168	0.001031	0.00094
2	200	0.000263	0.001262	0.002734
3	500	0.000561	0.001441	0.014129
4	900	0.001281	0.002018	0.016089
5	1000	0.001682	0.00254	0.025065
6	2000	0.006387	0.00592	0.056837
7	5000	0.040877	0.031178	0.208773
8	7000	0.088311	0.059587	0.34768
9	9000	0.14825	0.099363	0.515545
10	10000	0.183584	0.120275	0.599324
11	20000	0.824611	0.48644	1.9271
12	30000	1.8725	1.0794	4.1252
13	50000	5.3449	3.0303	10.0051
14	70000	10.5718	5.8035	18.6137
15	90000	17.6698	9.6563	31.7693
16	95000	19.5255	10.8392	36.9384
17	100000	21.8133	11.9117	40.1478

In tabel se poate observa dimensiunea vectorului si cu timpul corespunzator de executie, acest timp de executie putand sa aiba mici variatii in functie de cum sunt generate numerele.

Cea mai eficienta varianta este varianta paralele folosind MPI, apoi varianta secventiala, fara vreo paralelizare, iar varianta cu STL parallel fiind cea mai ineficienta.

3. Concluzii

Astfel se poate observa cat de inefficient este acest algoritm avand timpul de rulare de O(N^2), mai ales pentru dimensiuni foarte mari, fata de cat de usoare poate fi de inteles.

4. Referinte

https://www.geeksforgeeks.org/bubble-sort/

http://users.atw.hu/parallelcomp/ch09lev1sec3.html

https://www.vik-20.com/java/3-7-sequential-sorting-algorithms/