Laborator 3 PPD – Salsigan Razvan-Dan 236/2

Data predarii: 09.11.2021

# Cerinta

Scrieti un program bazat pe MPI care face suma a 2 numere mari.‘numar mare’ = numar cu mai mult de 10 cifre Reprezentare = tablou de cifre (numere intregi fara semn -byte) in care cifra cea mai nesemnificativa estepe prima pozitie.Cele 2 numere mari se citesc din fisierele “Numar1.txt” (un numar cu N\_1 cifre) si “Numar2.txt”(un numar cu N\_2 cifre).Fiecare din aceste fisiere contine la inceput un numar (N) care reprezinta numarul de cifre si apoi cifrele numarului respectiv.

## Varianta 1

–considera rezolvarea problemei prin executia urmatoarelor etape:

1)id\_proces\_curent=1

2)procesul 0 repeta urmatoarele actiuni pana cand se citesc toate cifrele numerelor

a.citeste cate N/p cifre din cele 2 fisiere

b.le trimite procesului “id\_proces\_curent”

c.incrementeaza “id\_proces\_curent”

3)procesele fac suma cifrelor primite si calculeaza “report” (carry) corespunzator;

4)fiecare process (cu exceptia ultimului) trimit “reportul” la procesul urmator care il foloseste pentru actualizarea rezultatului (procesul id=1 nu primeste carry -il considera egal 0)

5)rezultatul final se obtine in procesul 0. care scrie rezultatul in fisierul “output.txt”

Procesele primesc cifrele pe care trebuie sa le adune si apoi carry de la precedent

Am folosit MPI\_Send si MPI\_Recv

## Varianta 2

– considera rezolvarea problemei prin executia urmatoarelor etape:

1)procesul 0 citeste cele 2 numere si le stocheaza in 2 tablouri:

2)cifrele celor 2 numere se distribuire proceselor folosind MPI\_Scatter (daca nu este valabila conditia p|N, unde N=max{N\_1,N\_2}, N\_1 nr de cifre ale primului numar, N\_2 nr de cifre ale celui de-al doilea, atunci se mareste N corespunzator si se completeaza cu 0-uri)

3)procesele fac suma cifrelor primite si calculeaza “report” (carry) corespunzator

4)fiecare process (cu exceptia ultimului) trimit “reportul” la procesul urmator care il foloseste pentru actualizarea rezultatului

5)rezultatul final se obtine in procesul 0 (MPI\_Gather)

6)procesul0 scrie rezultatul in fisierul “output.txt”

Am folosit MPI\_Scatter si MPI\_Gather

# Analiza timpilor de executie

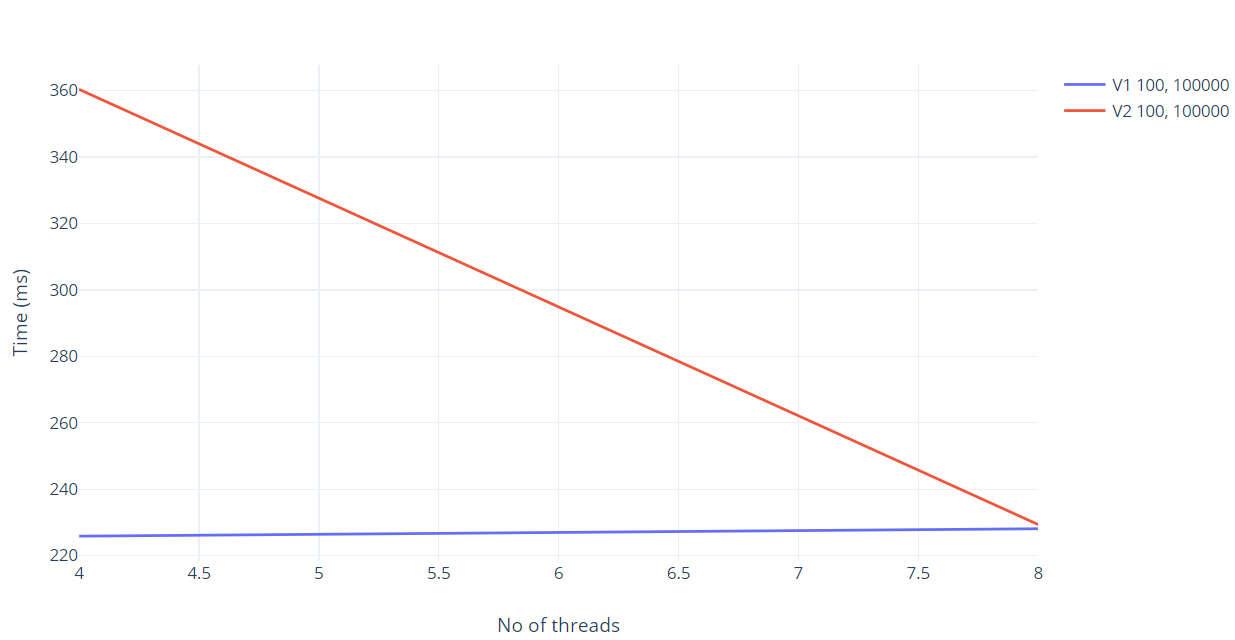
Testele au fost efectuate pe un procesor i5-9300H (4C, 8T) , in Windows ; implementare C++

Timpul afisat este in ms

|  |  |  |  |
| --- | --- | --- | --- |
| Dimensiune numere | Varianta | Nr threads | Timp executie |
| 18, 18 | V1 | 4 | 14.12476 |
| 8 | 20.75976 |
| V2 | 4 | 15.35827 |
| 8 | 21.06437 |
| 1000, 1000 | V1 | 4 | 16.62451 |
| 8 | 24.93434 |
| V2 | 4 | 27.93288 |
| 8 | 33.55829 |
| 100, 100000 | V1 | 4 | 225.8858 |
| 8 | 228.115 |
| V2 | 4 | 360.3381 |
| 8 | 229.3912 |

#### Grafice





## Concluzie

Prima varianta (cea in care se trimit cu send si recv elementele) a dat rezultate putin mai optime in toate cazurile. Varianta de 4 threaduri a rulat mai bine in toate cazurile, mai putin in V2 100x100000, unde a avut un rezultat mult mai slab.