Haja Florin-Gabriel

Programare Paralelă și Distribuită – Laborator 1

Se consideră o imagine reprezentată printr-o matrice de pixeli, F, de dimensiune (NxM). Se cere transformarea ei aplicand o filtrare cu o fereastra definita de multimea de indici W cu coeficientii wkl (reprezentati prin matricea W[k,l], unde 0<=k<n, 0<=l<m; si n<N, m<M). Transformarea unui pixel:

De exemplu:

Se cere asigurea următoarei postcondiții: matricea rezultat V contine imaginea filtrata a imaginii initiale F (V<>F)

A) Program secvential

B) Program paralel: folositi p threaduri pentru calcul..

Obiectiv: Împartire cât mai echilibrată a calculului pe threaduri!

Datele de intrare se citesc dintr-un fisier de intrare “date.txt”. (Fisierul trebuie creat anterior prin adaugare de numere generate aleator.)

Implementare:

a) Java

b) C++ ( cel putin C++11 ) i. matricile sunt alocate static ii. matricile sunt alocate dinamic

Testare: măsurați timpul de execuție pentru

1) N=M=10 și n=m=3; p=4;

2) N=M=1000 si n=m=5; p=2,4,8,16

3) N=10 M=10000 si n=m=5; p=2,4,8,16

4) N=10000 M=10 si n=m=5; p=2,4,8,16

Observații: Fiecare test trebuie repetat de 5 ori, iar pentru evaluarea timpului de execuție se consideră media aritmetică a celor 5 rulări.

- Pentru fiecare varianta a cazului de testare 1) folositi acelasi fisier “date.txt”; similar pentru cazurile 2) , 3) si 4) - Pentru programul C++ testati si comparati:

Analiză: comparati performanta pentru fiecare caz – secvential versus paralel si variantele paralele intre ele. Comparati timpii obtinuti cu implementarea Java versus implementarea C++. Comparati cele doua variante pentru implemntarea C++

Rulând un script care automatizează probarea tuturor cazurilor de testare și care creează automat un raport în Excel, observăm următoarele rezultate pe o configurație cu procesor Intel Core i5-8250U, 8 GB RAM DDR4 la 2400 MHz și Windows 10 (build 19041.572):

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Run no. | Language | Execution time | p | Allocation type | N | M | n=m |
| 1 | C++ | 0.0028 | 4 | static | 10 | 10 | 3 |
| 2 | C++ | 0.0382 | 2 | static | 1000 | 1000 | 5 |
| 3 | C++ | 0.0326 | 4 | static | 1000 | 1000 | 5 |
| 4 | C++ | 0.0296 | 8 | static | 1000 | 1000 | 5 |
| 5 | C++ | 0.0284 | 16 | static | 1000 | 1000 | 5 |
| 6 | C++ | 0.0074 | 2 | static | 10 | 10000 | 5 |
| 7 | C++ | 0.0066 | 4 | static | 10 | 10000 | 5 |
| 8 | C++ | 0.0066 | 8 | static | 10 | 10000 | 5 |
| 9 | C++ | 0.009 | 16 | static | 10 | 10000 | 5 |
| 10 | C++ | 0.0136 | 2 | static | 10000 | 10 | 5 |
| 11 | C++ | 0.0122 | 4 | static | 10000 | 10 | 5 |
| 12 | C++ | 0.0098 | 8 | static | 10000 | 10 | 5 |
| 13 | C++ | 0.0118 | 16 | static | 10000 | 10 | 5 |
| 14 | C++ | 0.0024 | 4 | dynamic | 10 | 10 | 3 |
| 15 | C++ | 0.0458 | 2 | dynamic | 1000 | 1000 | 5 |
| 16 | C++ | 0.038 | 4 | dynamic | 1000 | 1000 | 5 |
| 17 | C++ | 0.0262 | 8 | dynamic | 1000 | 1000 | 5 |
| 18 | C++ | 0.027 | 16 | dynamic | 1000 | 1000 | 5 |
| 19 | C++ | 0.0076 | 2 | dynamic | 10 | 10000 | 5 |
| 20 | C++ | 0.0062 | 4 | dynamic | 10 | 10000 | 5 |
| 21 | C++ | 0.0052 | 8 | dynamic | 10 | 10000 | 5 |
| 22 | C++ | 0.0094 | 16 | dynamic | 10 | 10000 | 5 |
| 23 | C++ | 0.0072 | 2 | dynamic | 10000 | 10 | 5 |
| 24 | C++ | 0.0062 | 4 | dynamic | 10000 | 10 | 5 |
| 25 | C++ | 0.0056 | 8 | dynamic | 10000 | 10 | 5 |
| 26 | C++ | 0.0086 | 16 | dynamic | 10000 | 10 | 5 |
| 27 | Java | 0 | 4 | dynamic | 10 | 10 | 3 |
| 28 | Java | 0.1144 | 2 | dynamic | 1000 | 1000 | 5 |
| 29 | Java | 0.1204 | 4 | dynamic | 1000 | 1000 | 5 |
| 30 | Java | 0.1094 | 8 | dynamic | 1000 | 1000 | 5 |
| 31 | Java | 0.113 | 16 | dynamic | 1000 | 1000 | 5 |
| 32 | Java | 0.072 | 2 | dynamic | 10 | 10000 | 5 |
| 33 | Java | 0.0686 | 4 | dynamic | 10 | 10000 | 5 |
| 34 | Java | 0.0752 | 8 | dynamic | 10 | 10000 | 5 |
| 35 | Java | 0.0946 | 16 | dynamic | 10 | 10000 | 5 |
| 36 | Java | 0.0354 | 2 | dynamic | 10000 | 10 | 5 |
| 37 | Java | 0.051 | 4 | dynamic | 10000 | 10 | 5 |
| 38 | Java | 0.0474 | 8 | dynamic | 10000 | 10 | 5 |
| 39 | Java | 0.0758 | 16 | dynamic | 10000 | 10 | 5 |

Graficul rulării programelor arată astfel:

Fig. N=M=1000, n=5, program C++ cu alocare dinamică

Fig. N=10000, M=10, n=5, program C++ cu alocare dinamică

Fig. N=M=1000, n=5, program Java