Documentatie Tema 1

# Cerinte

Considerand ca se da o matrice F(n,m) si o matrice de convolutie C(k,k) se cere sa se calculeze matricea V(n,m) rezultata in urma aplicarii convolutiei cu matricea de convolutie C pe matricea F.

1. Program secvential
2. Program paralel: folositi **p** threaduri pentru calcul.
   1. Descompunere geometrica pe orizontala (mai multe linii alocate unui thread)
   2. Descompunere geometrica pe verticala (mai multe coloane alocate unui thread)

**Datele de intrare** se citesc dintr-un fisier de intrare “date.txt“.

Fisierul trebuie creat anterior prin adaugarea de numere generate aleator.

Toate rularile trebuie executate cu acelasi fisier.

**Date de iesire :** output.txt fisier care contine matricea rezultat

# Proiectare

## Structuri de date folosite

1. Matricea principala

* Tip : int[][]
* Matricea principala pe care se aplica convolutia.
* Matrice bidimensionala cu dimensiuni variabile cu elemente intregi

1. Matricea de convolutie

* Tip : int[][]
* Matrice de convolutie care se aplica elementelor matricei principale
* Dimensiunea e specificata prin k

1. Matricea bordata

* Tip : int[][]
* Matricea principala careia i-au fost adaugate margini pentru a putea aplica convolutia si pe elementele de pe marginea matricei

1. Matricea rezultat

* Tip : int[][]
* Matricea finala care contine rezultatele dupa aplicarea convolutiei

1. Thread

* Tip : PersonalThread[]
* Un sir de threaduri retine fiecare thread care e responsabil de parcurgerea unei parti din matrice si aplicarea convolutiei asupra acestei parti

## Functii utilizate

### aplicareConvolutie

Aplica o matrice de convolutie asupra unui element specific din matricea principala

Parametrii :

* matrice - matricea originala pe care se aplica convolutia
* matriceConvolutie - matricea de convolutie (filtru) utilizata pt transformare
* k - dimensiunea matricii de convolutie
* linieElement - linia elementului central asupra caruia se aplica convolutia
* coloanaElement - coloana elementului central asupra caruia se aplica convolutia

Returneaza valoarea noua a elementului rezultat in urma aplicarii convolutiei.

### bordareMactrice

Creeaza o matrice bordata prin adaugarea de randuri si coloane suplimentare la margini. Elementele de pe bordura sunt replicate din marginea matricii originale

Parametrii :

* matrice - matricea originala
* n - numarul de linii al matricii originale
* m - numarul de coloane al matrciii originale
* k - dimensiunea matricii de convolutie

Returneaza matricea bordata.

### parcurgereSecventiala

Parcurge matricea originala secvential si aplica matricea de convolutie asupra fiecarui element

Parametrii :

* matrice - matricea bordata
* n - numarul de linii al matricii originale
* m - numarul de coloane al matricii originale
* matriceConvolutie - matricea de convolutie
* k - dimensiunea matricii de convolutie

Returneaza matricea rezultata prin aplicarea convolutiei.

### genereazaMatriceRandom

Genereaza o matrice si o matrice de convolutie cu valori random si le salveaza in fisier

Parametrii :

* n – numar de linii ale matricii principale
* m – numar de coloane ale matricii principale
* k – dimensiunea matricii de convolutie
* numeFisier – numele fisierului in care se afiseaza]

### parcurgereParalela

Aplica convolutia asupra unei matrice in mod paralel utilizand mai multe threaduri.

Parametrii:

* matrice – matricea bordata
* n – numar de linii al matricii originale
* m – numar de coloane al matricii originale
* matriceConvolutie – matricea de convolutie
* k – dimneisunea matricii de convolutie
* nrThreaduri – numar de threaduri care se executa in parallel

Returneaza matricea rezultata dupa aplicarea convolutiei in mod paralel.

### scrieInFisier

Scrie continutul unei matrici in fisier.

Parametrii :

* matrice – matricea care se scrie in fisier
* n – numar linii
* m – numar coloane
* numeFisier – numele fisierului in care se scrie

### main

Citeste datele din fisier, aplica convolutia in mod secvential/paralel si scrie rezultatul in fisier.

Args[0] = nr threaduri

Args[1] = numele fisierului de iesire

Args[2] = tipul de parcurgere

## Clasa PersonalThread

PersonalThread este o clasa care extinde Thread si e responsabila pentru aplicarea convolutiei pe o parte din matrice intr-un fir de executie separat.

Atribute :

* start – indica linia/coloana de la care acest thread incepe procesarea
* step – determina intervalul dintre liniile/coloanele pe care le proceseaza un thread
* n, m, k – dimensiunile matricilor
* matrice – matricea bordata pe care se aplica convolutia
* matriceConvolutie – matricea de convolutie
* matriceRezultat – matricea rezultata prin aplicarea convolutiei

Metoda run() :

* implementarea care se executa atunci cand un thread e pornit
* se parcurg liniile/coloanele din matrice incepand de la start+k/2 (pentru a tine cont de marginea bordata) pana la ultima liniei cu un pas de step (se sare peste liniile procesate de alte threaduri)
* pentru fiecare linie si fiecare coloana se aplica functia aplicareConvolutie, care calculeaza noul element rezultat din aplicarea matricii de convolutie pe elementele din jur

## Fluxul de executie in cazul parcurgerii paralele

1. Se creeaza un array de threaduri PersonalThread[], fiecare thread fiind initializat cu parametri diferiti pentru a acoperi parti diferite din matrice.
2. Fiecare thread e pronit folosind metoda start() care declanseaza executia metodei run().
3. Dupa pornirea fiecarui thread, functia principala asteapta ca toate threadurile sat ermine executia folosind metoda join() pentru a se asigura ca toate calculele sunt complete inainte de a continua.

# Java

|  |  |  |
| --- | --- | --- |
| Tip Matrice | Numar thread | Timp mediu de executie |
| N=M=10  K=3 | Secvential | 48 |
| 4 coloana | 49.3 |
| 4 linie | 49.1 |
| N=M=1000  K=5 | Secvential | 51 |
| 2 coloana | 53.8 |
| 2 linie | 51.3 |
| 4 coloana | 52 |
| 4 linie | 50 |
| 8 coloana | 52.6 |
| 8 linie | 50.2 |
| 16 coloana | 49.9 |
| 16 linie | 52.7 |
| N=10  M=10000  K=5 | Secvential | 57 |
| 2 coloana | 49.6 |
| 2 linie | 49.2 |
| 4 coloana | 50.8 |
| 4 linie | 50.5 |
| 8 coloana | 50.5 |
| 8 linie | 50.7 |
| 16 coloana | 50.1 |
| 16 linie | 51 |
| N=10000  M=10  K=5 | Secvential | 57 |
| 2 coloana | 49.6 |
| 2 linie | 52.2 |
| 4 coloana | 55 |
| 4 linie | 53.1 |
| 8 coloana | 53 |
| 8 linie | 53.1 |
| 16 coloana | 53.4 |
| 16 linie | 53 |
| N=M=10000  K=5 | Secvential | 52 |
| 2 coloana | 52.2 |
| 2 linie | 50.7 |
| 4 coloana | 52.5 |
| 4 linie | 51.5 |
| 8 coloana | 52.9 |
| 8 linie | 53.9 |
| 16 coloana | 52.3 |
| 16 linie | 52.9 |

# C++

|  |  |  |
| --- | --- | --- |
| Tip Matrice | Numar thread | Timp mediu de executie |
| N=M=10  K=3 | Secvential | 50.1(static) 51.6(dinamic) |
| 4 coloana | 52.5(static) 52.5(dinamic) |
| 4 linie | 53.7(static) 52.8(dinamic) |
| N=M=1000  K=5 | Secvential | 52.6(static) 51.7(dinamic) |
| 2 coloana | 53.8(static) 52.7(dinamic) |
| 2 linie | 50.4(static) 51.8(dinamic) |
| 4 coloana | 52.6(static) 52.4(dinamic) |
| 4 linie | 53.6(static) 51.6(dinamic) |
| 8 coloana | 52.7(static) 52.2(dinamic) |
| 8 linie | 53.5(static) 53.8(dinamic) |
| 16 coloana | 53.1(static) 51.8(dinamic) |
| 16 linie | 53.2 (static) 52.2(dinamic) |
| N=10  M=10000  K=5 | Secvential | 52.2 (static) 52.8(dinamic) |
| 2 coloana | 52.5(static) 53.3(dinamic) |
| 2 linie | 52.6(static) 52.9(dinamic) |
| 4 coloana | 51.8(static) 54.9(dinamic) |
| 4 linie | 54(static) 51.2(dinamic) |
| 8 coloana | 53.9(static) 52.2(dinamic) |
| 8 linie | 52.9(static) 51.8(dinamic) |
| 16 coloana | 53.4(static) 52.9(dinamic) |
| 16 linie | 52.7(static) 52.4(dinamic) |
| N=10000  M=10  K=5 | Secvential | 52.5 (static) 52.6(dinamic) |
| 2 coloana | 50.7(static) 51.3(dinamic) |
| 2 linie | 53.4(static) 51.7(dinamic) |
| 4 coloana | 53.5(static) 51.3(dinamic) |
| 4 linie | 53(static) 51.7(dinamic) |
| 8 coloana | 51.6 (static) 51.6(dinamic) |
| 8 linie | 52 (static) 51.6(dinamic) |
| 16 coloana | 52.3 (static) 52.8(dinamic) |
| 16 linie | 52.6 (static) 52.2(dinamic) |
| N=M=10000  K=5 | Secvential | 54.1 (static) 53.2(dinamic) |
| 2 coloana | 51.7 (static) 51.8(dinamic) |
| 2 linie | 52.5 (static) 51.8(dinamic) |
| 4 coloana | 52.7(static) 52(dinamic) |
| 4 linie | 52.4(static) 53.3(dinamic) |
| 8 coloana | 52.2(static) 52.7(dinamic) |
| 8 linie | 52.5 (static) 53.6(dinamic) |
| 16 coloana | 54 (static) 54.3(dinamic) |
| 16 linie | 53 (static) 51.7(dinamic) |