Laborator 1 – PDD

Nașca Razvan-Alexandru

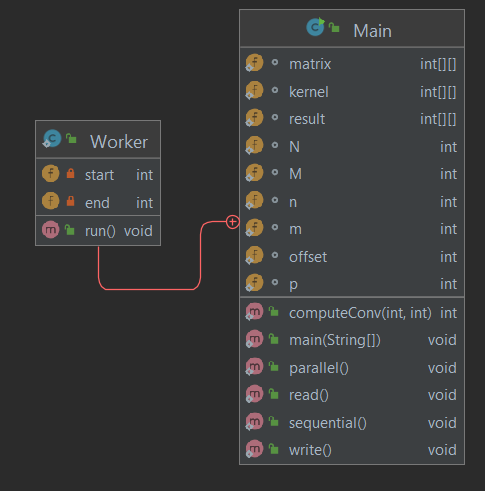
Grupa 235/2

1. Cerinta:

Se consideră o imagine reprezentată printr-o matrice de pixeli, F, de dimensiune (M,N). Se cere transformarea ei aplicând o filtrare cu o fereastră definită de mulțimea de indici W cu coeficienții *wk,l* (reprezentați prin matricea W[k,l]).

Frontiere: f[-1,j] = f[0,j]; f[i,-1] = f[i,0]; f[M,j] = f[M-1,j]; f[i,N] = f[i,N-1].

2. Modul de proiectare



Java

|  |  |  |
| --- | --- | --- |
| Tip Matrice | Nr Threads | Timp executie(ms) |
| N = M = 10  n = m = 3 | Secvential | 0.09632 |
| 4 | 0.97362 |
| N = M = 1000  n = m = 5 | Secvential | 57.44242 |
| 2 | 48.02157 |
| 4 | 39.69983 |
| 8 | 45.3282 |
| 16 | 69.26575 |
| N = 10  M = 10000  n = m = 5 | Secvential | 25.10697 |
| 2 | 31.05911 |
| 4 | 36.75333 |
| 8 | 46.96798 |
| 16 | 46.04184 |
| N = 10000  M = 10  n = m = 5 | Secvential | 21.31592 |
| 2 | 29.35502 |
| 4 | 30.23418 |
| 8 | 37.95382 |
| 16 | 37.51259 |

|  |  |  |  |
| --- | --- | --- | --- |
| Tip matrice | Tip Alocare | Nr. Threads | Timp executie(ms) |
| N = M = 10  n = m = 3 | Static | 4 | 1.17708 |
| Dinamic | 4 | 1.26311 |
| N = M = 1000  n = m = 5 | Static | 1 | 97.76439 |
| 2 | 99.18411 |
| 4 | 98.66483 |
| 8 | 99.30098 |
| 16 | 104.16382 |
| Dinamic | 1 | 105.761 |
| 2 | 106.523 |
| 4 | 107.0639 |
| 8 | 108.116 |
| 16 | 107.1226 |
| N = 10  M = 10000  n = m = 5 | Static | 1 | 10.01933 |
| 2 | 11.2485 |
| 4 | 10.95097 |
| 8 | 11.10748 |
| 16 | 10.89866 |
| Dinamic | 1 | 0.00644 |
| 2 | 1.23843 |
| 4 | 1.42284 |
| 8 | 1.49959 |
| 16 | 1.46379 |
| N = 10000  M = 10  n = m = 5 | Static | 1 | 10.03439 |
| 2 | 11.58576 |
| 4 | 11.7852 |
| 8 | 13.24205 |
| 16 | 12.58414 |
| Dinamic | 1 | 0.00631 |
| 2 | 1.44342 |
| 4 | 1.1545 |
| 8 | 1.34656 |
| 16 | 1.22371 |

C++

4. Analiza:

C++:

* Pentru matricea de 10x10 timpii intre static si dinamic sunt apropiati
* Pentru matricea de 1000x1000 se obtin timpi mai buni pentru implementarea statica, cel mai bun timp fiind obtinut pentru implementarea secventiala
  + - * + Timpii intre cele 2 implementati, statice si dinamice, sunt apropiati
* Pentru matricea de 10x1000 timpii au diferente mari in favoarea implementarii dinamice ( se obtine un timp de 10 ori mai bun), asemanator cu matricea de 1000x10

Java:

* Pentru matricea de 10x10 timpul pentru secvential este de 10 ori mai mic decat la folosirea de 4 threaduri
* Pentru matricea de 1000x1000 cel mai bun timp s-a obtinut pentru implementarea cu 4 threaduri; implementarea secventiala fiind destul de lenta
* Pentru matriciile de 10000x10 si 10x10000, implementarea secevntiala este cea mai buna, fiind usor mai rapida.

Java vs C++

* Se observa ca variant de Java este mai rapid decat C++, cel putin pentru primele seturi de date
* Pentru setul de date cu 10000x10 si 10x10000 C++ dimanic este cu 20 ori