# Procesarea imaginilor: Studiu comparativ al abordarii secventiale, paralelizar in C# si MPI in C++

## Topicul proiectului

Proiectul se concentreaza pe implementarea si compararea unor metode de procesare a imaginilor, cu accent pe filtrare, folosind atat un algoritm secvential, cat si metode de paralelizare pentru a imbunatati performanta. Scopul proiectului este de a observa cum paralelizarea poate accelera procesarea pixelilor in operatiunea de filtrare a imaginilor.

## Cerinte

### **Implementarea Algoritmului Secvential:**

Sa se implementeze un algoritm secvential pentru filtrarea imaginilor in C#.

Algoritmul trebuie sa suporte cel putin doua tipuri de filtru (de ex., filtru Grayscale, filtru Sepia, filtru Negative, filtru Outline).

Sa se masoare timpul de executie pentru procesarea imaginilor de diferite dimensiuni.

### **Implementarea Paralelizarii:**

**Metoda 1:**

Algoritm Paralelizat cu Parallel.For, Task, sau async/await:

Sa se implementeze o versiune paralelizata a algoritmului de filtrare, folosind biblioteci sau tehnici de paralelizare in C# .

**Metoda 2:**

Algoritm Paralelizat cu MPI:

Sa se implementeze o versiune paralelizata a algoritmului de filtrare folosind MS-MPI in C++.

Sa se compare timpii de executie al algoritmilor paralelizati cu cel secvential pentru aceeasi imagine si acelasi filtru.

## Mediu de dezvoltare si testare

**Visual Studio:**

Folosit pentru implementarea si depanarea tuturor celor trei algoritmi (secvential, paralelizat in C# si paralelizat in C++ cu MPI).

**Extensii necesare:**

* Pentru C#: .NET SDK instalat
* Pentru C++ cu MPI: MS-MPI instalat

**Configuratie Proiect**:

* Pentru C#: Proiect de tip Console Application
* Pentru C++: Proiect de tip Console Application cu suport pentru MPI.

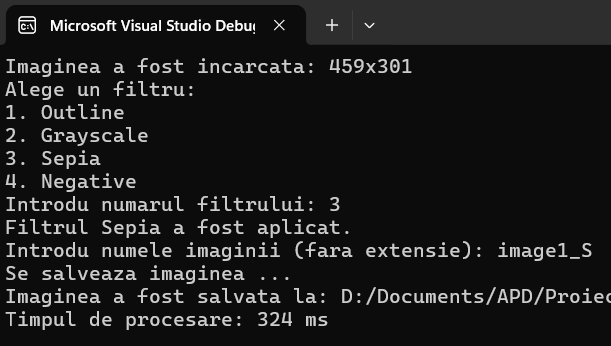
**Testare algoritmi:**

Am folosit imagini de diferite dimensiuni (de la 459x301 pana la 11648x8736) pentru testarea algoritmilor.

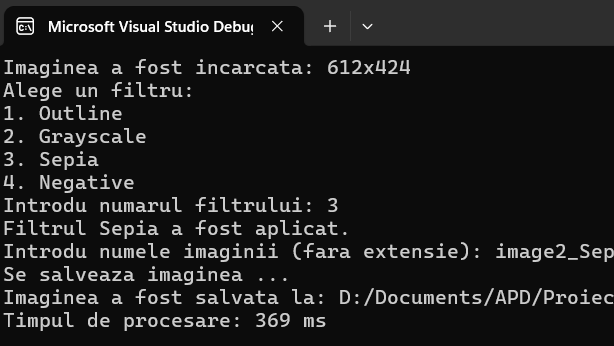
Formate de fisiere: JPEG, PNG.

## Rezultate experimentale

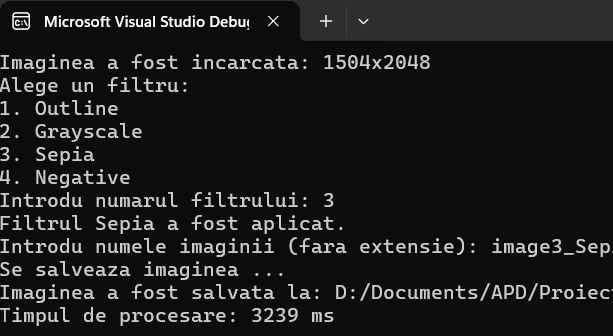
Pentru vizualizarea timpilor de rulare ai algoritmului secvential am aplicat acelasi filtru pentru imagini de diferite dimensiuni.



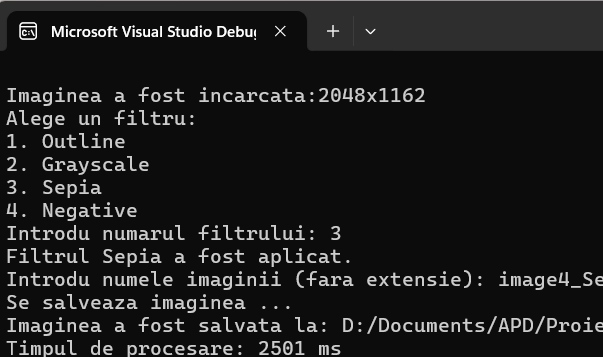
Timp de procesare pentru imaginea 1: **324ms**.



Timp de procesare pentru imaginea 2: **369 ms**.

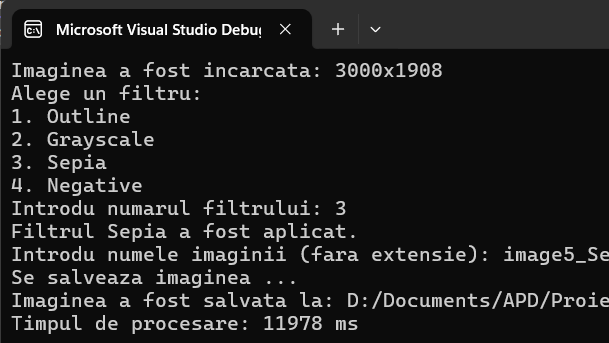


Timp de procesare pentru imaginea 3: **3239 ms**.

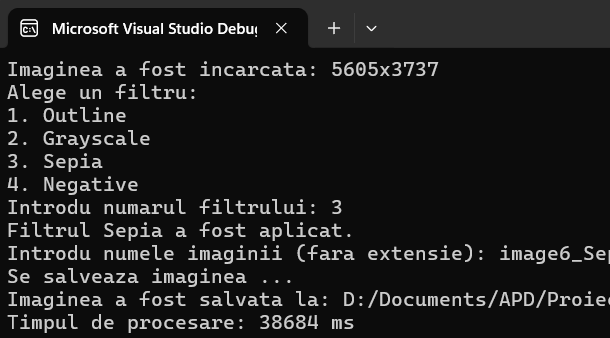


Timp de procesare pentru imaginea 4: **2501 ms**.

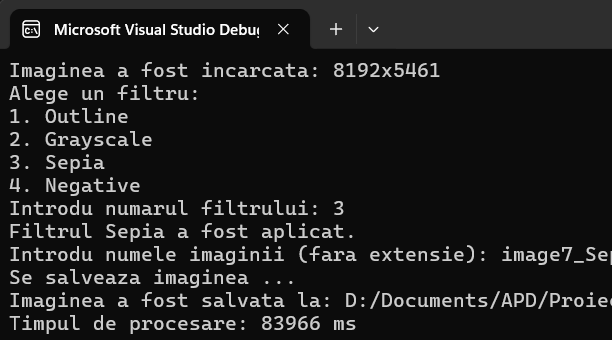
**Observatie**: Timpul de procesare pentru imaginea 4 este mai mic decat cel pentru imaginea 3, desi dimensiunea fisierului imaginii 4 este mai mare decat cea a imaginii 3. Rezolutia totala a imaginii 4 este de 2048 × 1162 = 2.377.216 pixeli, iar rezolutia totala a imaginii 3 este de 1504 × 2048 = 3.080.192 pixeli. Deci, Daca o imagine are mai multi pixeli, aceasta este procesata mai greu, deoarece sunt necesare mai multe calcule si operatii pe pixeli.



Timp de procesare pentru imaginea 5: **11978 ms** **≃ 12 s**

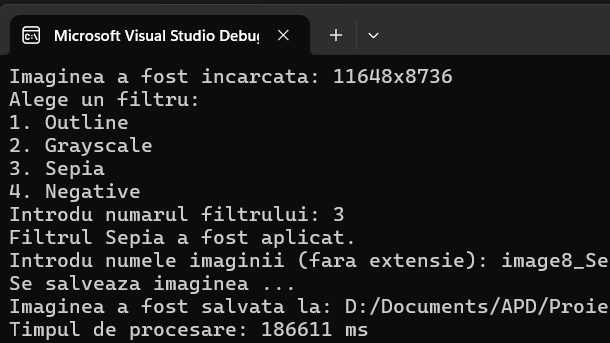


Timp de procesare pentru imaginea 6: **38648 ms** **≃ 39 s**



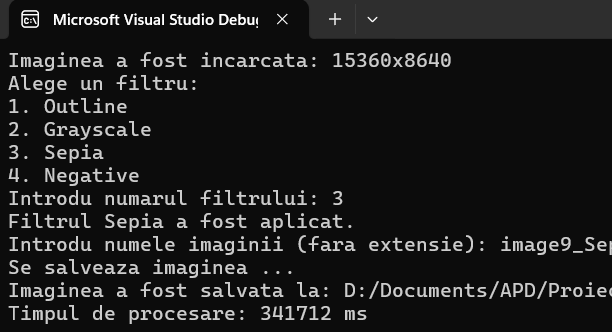
Timp de procesare pentru imaginea 7:

**83966 ms** **≃** **1,4 min**



Timp de procesare pentru imaginea 8:

**186611 ms** **≃** **3 min**



Timp de procesare pentru imaginea 8:

**341712 ms** **≃** **6 min**

**Concluzie:**

Timpul de procesare al imaginilor este influentat de dimensiunea acestora, mai precis de numarul de pixeli. Cu cat o imagine are mai multi pixeli, cu atat este necesara o capacitate de calcul mai mare si un timp de procesare mai lung. Desi, sunt cazuri in care si alte aspecte influenteaza timpul de procesare cum ar fi complexitatea continutului, dar numarul de pixeli ramane un factor determinant.