Tema 2 – Compresia imaginilor utilizând arbori

Mihai Nan, Andrei Sorin Cîrpici, Vicențiu-Octavian Palaga

Data postării: 06.04.2023
Deadline: 2.05.2023 ora 23:59

1 Obiective

În urma realizării acestei teme, studentul va fi capabil:

- să implementeze și să utilizeze arbori în rezolvarea unor probleme;
- să înțeleagă modul de reprezentare a unei imagini;
- să implementeze operații standard de prelucrare a imaginilor;
- să transpună o problemă din viața reală într-o problemă care uzitează arbori cuaternari.

2 Descriere

Trăim într-o lume a imaginii; suntem, fără îndoială, consumatori de imagini. Paradoxal însa, nu suntem nici pregătiți, nici invățați să lecturăm imaginile. Atunci când suntem puși în situația de a le interpreta, o facem mai mult intuitiv, pentru că nu avem prea multe repere în acest domeniu, nu știm ce trebuie să vedem și cum. Fotografia dezvoltă moduri distincte de a vedea lumea. Ea poate evoca sau poate impune reprezentarea care determină recunoașterea lucrului sau ființei care fie că a disparut, fie că s-a transformat după fixarea sa în alb-negru sau color. Prin urmare, ea reprezintă o frântură fractalică a realității.

Această temă dorește să vă introducă în vasta lumea a imaginilor și vă propune spre analiză și implementare o metodă de compresie a imaginilor. Termenul de compresie a imaginilor se referă la o clasă largă de tehnici și metode al căror scop este reprezentarea unei imagini date folosind un număr cât mai mic de biți. Odată cu evoluția tehnologiei și, implicit, cu creșterea calității imaginilor, necesitatea reducerii cantității de informație necesară reprezentării unei imagini a devenit evidentă.

Procesul de recompunere a imaginii inițiale, din reprezentarea restrânsă, se numește decompresie. Este evident că, prin decodare, trebuie să se obțină o imagine cât mai apropiată de imaginea originală.

3 Compresia imaginilor

3.1 Arbore cuaternar

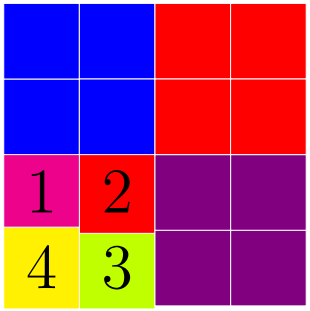
Arborele cuaternar este o structură de date care organizează informații pentru date multidimensionale, fiind folosită în cartografie, procesarea imaginilor, grafică pe calculator etc. Structura este un arbore ce reprezintă o zonă din spațiul N-dimensional (noi vom analiza, în cadrul acestei teme, cazul N = 2), fiecare nod al arborelui păstrează informație pentru o zonă din spațiu, iar nodul are 2^N fii, care reprezintă fiecare o zonă de 2^N ori mai mică decât zona părintelui. Zonele fiilor sunt disjuncte, iar reuniunea lor formează zona părintelui. Cu alte cuvinte, structura pe care o vom utiliza, în cadrul acestei teme, este un arbore în care fiecare nod neterminal va avea fix 4 descendenți.

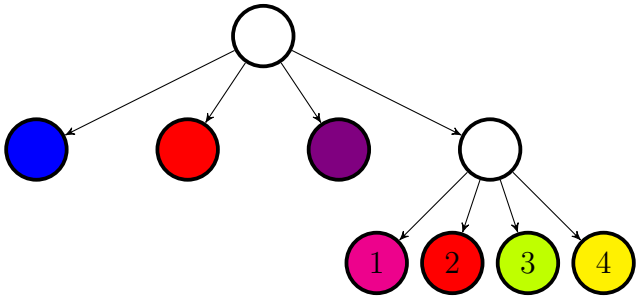
Pentru un arbore cuaternar, rădăcina reprezintă un pătrat, iar descendenții reprezintă cele patru pătrate disjuncte și congruente, în care se poate împărți pătratul inițial. Din câte am prezentat până acum, un arbore cuaternar este inifinit, însă noi vom uzita această structură de date pentru reprezentarea unei imagini cu o dimensiune redusă din plan. Din acest motiv, vom folosi numai câteva niveluri din arborele cuaternar.

3.2 Algoritmul de compresie

Orice imagine pătrată, de dimensiune putere a lui 2, poate fi reprezentată printr-un arbore cuaternar. Nodurile de pe fiecare nivel al arborelui corespund unei împărțiri a unei zone pătrate din imagine în patru sferturi. Rădăcina arborelui este asociată întregii imagini, nodurile de pe primul nivel al arborelui corespund celor patru sferturi ale imaginii (ordinea este: stânga sus, dreapta sus, dreapta jos, stânga jos), nodurile de pe nivelul doi corespund sferturilor fiecărui sfert anterior și așa mai departe. Împărțirea imaginii poate continua până când nodurile nivelului curent al arborelui corespund unor zone pătrate uniforme: dacă regiunea pătrată considerată nu este uniformă, atunci aceasta va fi descompusă, prin tăiere, în patru părți egale și nodul corespunzător va deveni neterminal, având patru descendenți. Dacă regiunea considerată este uniformă (compusă din pixeli de același fel), nodul respectiv devine un nod frunză (terminal) al arborelui. Fiecare nod terminal conține informație, corespunzătoare valorii zonei de imagine la care este asociat.

Pentru o mai bună înțelegere a algoritmului descris în paragraful anterior, se propune analizarea exemplului de mai jos.





4 Decompresia imaginilor

Pentru refacerea imaginii inițiale, din reprezentarea arborescentă, este suficientă alegerea

nodurilor terminale a căror valoare corespunde pixelilor de obiect. Adâncimea la care este plasată în arbore o frunză conține informația de dimensiune a zonei pătrate corespunzătoare din imagine. Poziția frunzei față de nodurile de pe același nivel ce au același predecesor este direct determinată de regula de alocare a sferturilor unei zone la nodurile descendente ale arborelui (regula de alocare trebuie să se păstreze pentru întregul arbore).

5 Formatul fișierelor

În cadrul acestei teme vom utiliza imagini color. Pentru reprezentarea imaginilor color exista mai multe formate disponibile. Indiferent de format, pentru o imagine color vom avea stocată o matrice de pixeli în care fiecare pixel al imaginii este format dintr-un triplet de valori numerice, numite canale de culoare: roșu, verde și albastru (**RGB**). Aceste valori sunt utilizate pentru a descrie cantitatea de culoare specifică fiecărui pixel. Fiecare canal poate avea o valoare cuprinsă între 0 și 255, unde 0 reprezintă absența culorii, iar 255 reprezintă cantitatea maximă de culoare posibilă într-un anumit canal. Prin combinarea valorilor din cele trei canale, se obține o gamă largă de culori, ceea ce permite o varietate mare de imagini color.

Pentru simplitate, am ales ca în cadrul acestei teme să utilizăm formatul **PPM**.

5.1 Fișierul PPM

Un fișier în format **PPM** conține un antet, în format text, care cuprinde: pe prima linie tipul fișierului (în cazul imaginilor folosite în test, o să fie tipul `P6`), pe a doua linie două numere naturale (`width` și `height`), separate prin spațiu, care descriu dimensiunile imaginii, iar a treia linie va conține un număr natural reprezentând valoarea maximă pe care o poate lua o culoare (în cazul testelor folosite, valoarea este `255`); și imaginea propriu-zisă, în format binar.

Important

Este garantat că imaginile primite sunt pătratice și au dimensiunea putere a lui 2.

5.1.1 Imaginea propriu-zisă

Această secțiune este reprezentată printr-o matrice de pixeli care ocupă cea mai mare parte a fișierului. Numărul de elemente din tabou este egal cu produsul dintre numărul de pixeli pe linie (**width**) și numărul de pixeli pe coloană (**height**), tabloul fiind organizat pe linii și coloane, începând cu linia de sus a imaginii, pixelul din stânga.

Observație

Fiecare pixel este descris prin canalele aflate în ordine **RGB** (**R**ed **G**reen **B**lue).

Important

Imaginea propriu-zisă este în format binar și va trebui să fie citită utilizând `fread`.

Informații suplimentare despre acest format și câteva exemple puteți găsi la adresa: http://paulbourke.net/dataformats/ppm/.

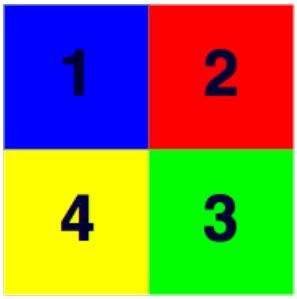
5.2 Fișierul comprimat

Fișierul comprimat va conține informațiile rezultate în urma procesului de compresie:

- ***dimensiune_imagine*** **- de tip** **`unsigned int`** – care specifică dimensiunea imaginii (reamintim că avem imagini pătrate);
- Pentru fiecare nod din parcurgerea pe nivel aplicată arborelui de compresie vom scrie în fișierul comprimat următoarele informații:

1. Dacă nodul este un nod intern:
   - ***tipul_nodului*** **- de tip** **`unsigned char`** – care va avea valoarea 0 în acest caz.
2. Dacă nodul este un nod frunză:
   - ***tipul_nodului*** **- de tip** **`unsigned char`** – care va avea valoarea 1 în acest caz.
   - ***valoarea_pentru_roșu*** **- de tip** **`unsigned char`** – care va indica valoarea componentei responsabilă de culoarea Roșu pentru pixelii din zona descrisă de respectivul nod.
   - ***valoarea_pentru_verde*** **- de tip** **`unsigned char`** – care va indica valoarea componentei responsabilă de culoarea Verde pentru pixelii din zona descrisă de respectivul nod.
   - ***valoarea_pentru_albastru*** **- de tip** **`unsigned char`** – care va indica valoarea componentei responsabilă de culoarea Albastru pentru pixelii din zona descrisă de respectivul nod.

Pentru a înțelege mai bine modul în care o să apară descendenții unui nod, propunem următoarea reprezentare grafică a unei astfel de împărțirii în care sunt evidențiați cele 4 noduri copil.



Fișierul decomprimat este un fișier standard în format **PPM** și conține informațiile extrase din fișierul comprimat dat pentru decompresie.

6 Cerințe

6.1 Cerința 1

Pentru început, vom construi arborele de compresie și vom determina câteva statistici pornind de la acesta. Astfel, inițial vom citi imaginea din fișierul PPM și apoi vom construi arborele de compresie pentru aceasta.

Important

Implementarea arborelui se va face folosind pointeri.

Pentru a determina când s-a ajuns la un bloc care poate fi reprezentat în arborele cuaternar de compresie ca nod frunză, cu alte cuvinte, nu mai este nevoie să fie divizat în alte 4 zone de dimensiune egală, se va calcula culoarea medie a blocului, determinând pentru fiecare canal (**RED**, **GREEN** și **BLUE**) media aritmetică a valorilor din submatricea de pixeli care corespunde blocului. Vom considera că această submatrice va începe cu elementul aflat la coordonatele (x, y), unde x reprezintă indicele liniei și y reprezintă indicele coloanei. De asemenea, știm că orice submatrice analizată trebuie să fie pătratică și vom considera că are dimensiunea $size \times size$.

Pentru a calcula media aritmetică a valorilor din submatricea de pixeli care corespunde unui anumit bloc vom putea utiliza următoarele formule:

$$red = \frac{1}{size \cdot size} \left(\sum_{i=x}^{x+size} \sum_{j=y}^{y+size} grid[i][j].red \right)$$

$$green = \frac{1}{size \cdot size} \left(\sum_{i=x}^{x+size} \sum_{j=y}^{y+size} grid[i][j].green \right)$$

$$blue = \frac{1}{size \cdot size} \left(\sum_{i=x}^{x+size} \sum_{j=y}^{y+size} grid[i][j].blue \right)$$

După ce a fost determinată culoarea medie, se calculează un scor al similarității pentru blocul respectiv, folosind următoarea formulă:

$$mean = \frac{1}{3 \cdot size^2} \sum_{i=x}^{x+size} \sum_{j=y}^{y+size} (red - grid[i][j].red)^2 + (green - grid[i][j].green)^2 + (blue - grid[i][j].blue)^2$$

unde red, $green$, $blue$ reprezintă componentele pentru culoarea medie.

Dacă valoarea obținută pentru scor este mai mică sau egală decât pragul impus, atunci nu o să mai fie nevoie de divizare.

După ce am construit arborele de compresie, dorim să determinăm următoarele informații:

- numărul de niveluri din arborele cuaternar;
- numărul de blocuri din imagine pentru care scorul similarității pixelilor este mai mic sau egal decât factorul furnizat;
- dimensiunea laturii pătratului pentru cea mai mare zonă din imagine care a rămas nedivizată.

Important

Aceste informații se vor scrie în fișier text fiecare pe câte o linie. Numele fișierului este furnizat ca argument în linia de comandă.

6.2 Cerința 2

Pentru a doua cerință a temei, trebuie să se realizeze compresia unei imagini în format **PPM**, folosind algoritmul de compresie detaliat în secțiunea anterioară a enunțului. În rezolvarea acestei cerințe, este **obligatorie** implementarea arborelui cuaternar de compresie.

Pentru construcția arborelui, veți folosi metoda descrisa la cerință anterioară.

După ce este construit acest arbore de compresie, el este parcurs pe nivel și este generat fișierul de compresie conform formatului descris în Secțiunea 5.2.

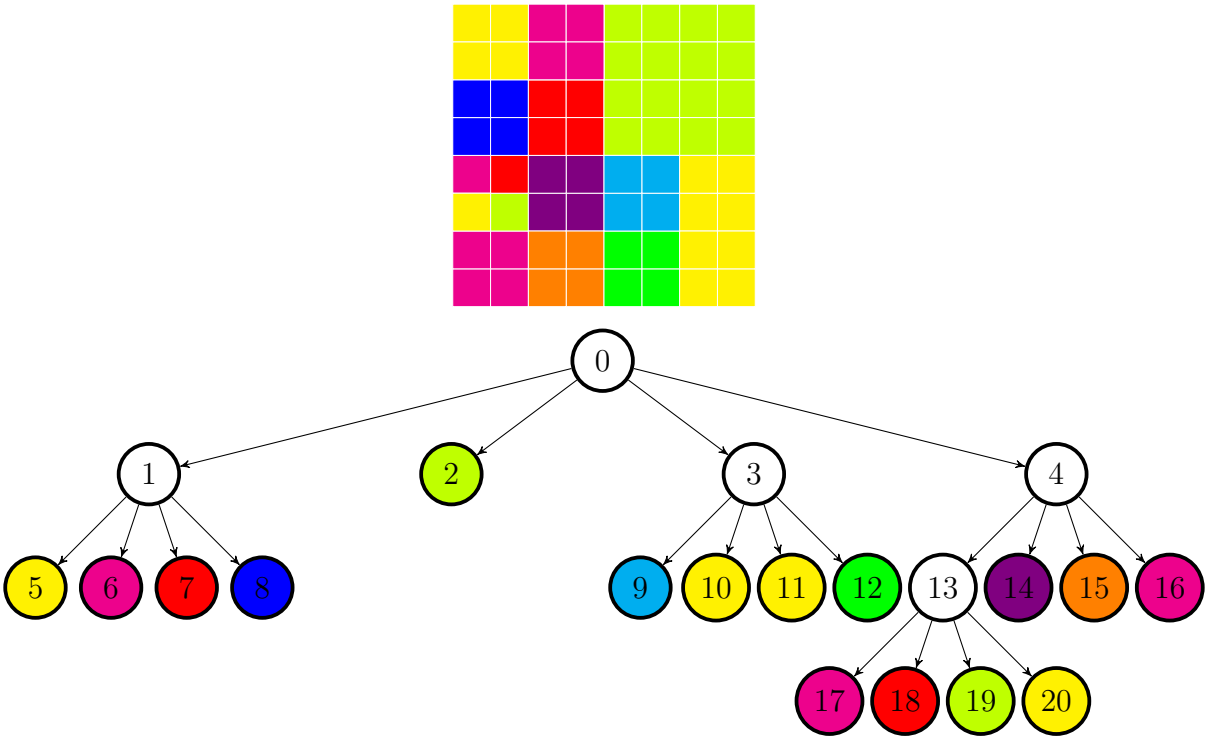
6.3 Cerința 3

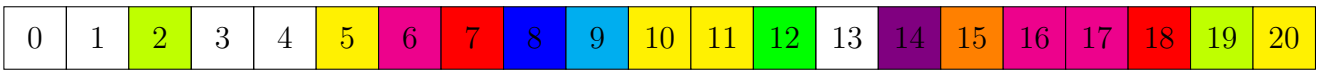
În cadrul acestei cerințe, trebuie să se reconstituie imaginea inițială, pornind de la un fișier rezultat în urma compresiei și utilizând algoritmul de decompresie descris. De asemenea, este **obligatorie** uzitarea structurii de arbore cuaternar în rezolvarea cerinței.

În acest caz, va trebui să reconstruiți arborele de compresie, pornind de la parcurgerea disponibilă în fișierul comprimat, și să generați imaginea pe baza arborelui. Fișierul binar de la care porniți va avea formatul descris în Secțiunea 5.2.

7 Exemplu

Pentru o înțelegere mai bună a cerințelor acestei teme, analizați următorul exemplu, realizat pentru imaginea de mai jos. Imaginile cu care veți lucra, în cadrul acestei teme, nu vor avea zonele despărțite (acele contururi albe care apar în imaginea de mai jos pentru fiecare pătrățel), precum imaginea de mai jos. Inițial, va trebui să citiți imaginea și sa construiți arborele cuaternar de compresie care o va modela, structura cu care veți lucra în cadrul cerințelor următoare.





7.1 Cerința 1

Vom presupune că fiecare pătrățel din imaginea anterioară are dimensiunea 4×4. Numărul de niveluri din arborele de compresie este 4, numărul de blocuri din imagine pentru care scorul similarității pixelilor este mai mic sau egal decât factorul furnizat este 16, iar pătratul care descrie cea mai mare zonă din imagine care a rămas nedivizată are latura de dimensiune 16.

Conținutul fișierului `quadtree.out`:

```
4
16
16
```

7.2 Cerința 2

După ce am citit imaginea și am format arborele cuaternar, putem realiza compresia imaginii. Pentru acest lucru, vom aplica algoritmul de parcurgere pe nivel pentru arborele cuaternar.

Ordinea în care apar nodurile pentru respectivul arbore în rezultatul parcurgerii pe nivel este următoarea:

Pentru arborele de mai sus, prezentăm ce valori vom scrie în fișierul binar pentru fiecare nivel:

```
0: 0
1: 0 {1 191 255 0} 0 0
2: {1 255 235 61} {1 251 49 153} {1 255 0 0} {1 0 0 255} {1 0 185 242}
   {1 255 235 61} {1 255 235 61} {1 0 255 0} 0 {1 128 0 128}
   {1 255 128 0} {1 251 49 153}
3: {1 251 49 153} {1 255 0 0} {1 191 255 0} {1 255 235 61}
```

În această reprezentare, am precizat indicele nivelului la început și apoi valorile. Pentru a fi mai ușor de urmărit, am delimitat prin `{` și `}` cele 4 valori pentru nodurile frunză (`tipul_nodului`, `red`, `green`, `blue`).

Important

În enunț am oferit un exemplu de valori în format text formatat, pentru a fi ușor de citit. În implementare, trebuie să respectați precizările din cadrul Secțiunii **5.2** și să aveți în vedere faptul că lucrați cu un fișier binar. Astfel, veți scrie doar valorile de interes, ținând cont de tipurile specificate, fără spațiu sau alte procesări suplimentare.

Important

Fișierul rezultat în urma compresiei trebuie să fie un fișier binar. Astfel, informațiile vor fi scrise în fisier utilizând funcția `fwrite` și ținând cont de tipurile de date indicate în Secțiunea 5.2.

Pornind de la următoarea poză, o să vedem cum afectează pragul oferit arborele cuaternar și, implicit, operația de compresie.

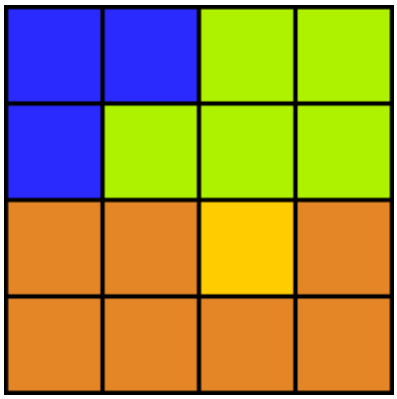
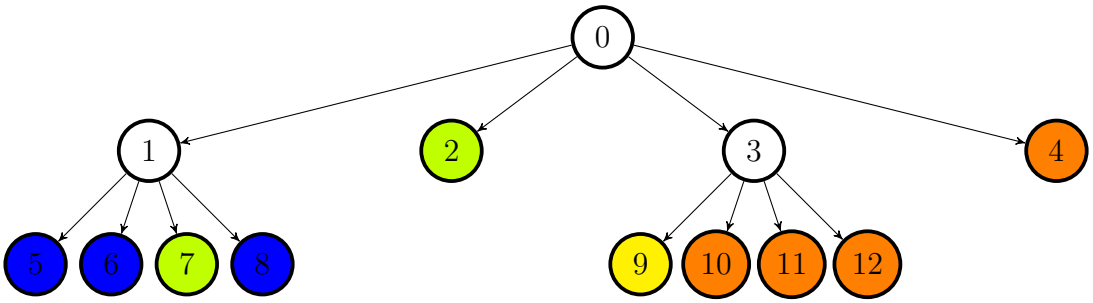


Figure 1: Imaginea inițială

1. Dacă pragul oferit este 0, atunci arborele de compresie o să fie următorul.



Pentru acest arbore, imaginea după decompresie va arăta în felul următor:

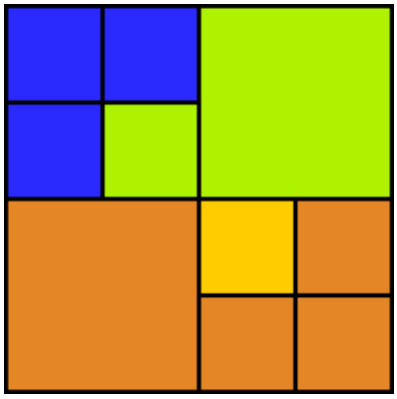
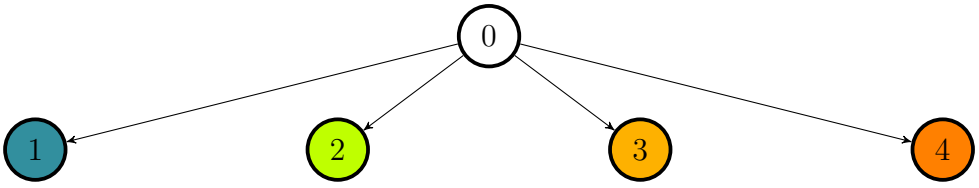


Figure 2: Imagine cu prag 0

2. Dacă pragul oferit este mai mare, atunci arborele de compresie ar putea arăta precum cel mai jos. Fiecare nod frunză va conține culoarea medie a blocului.



Pentru acest arbore, imaginea după decompresie va arăta în felul următor:

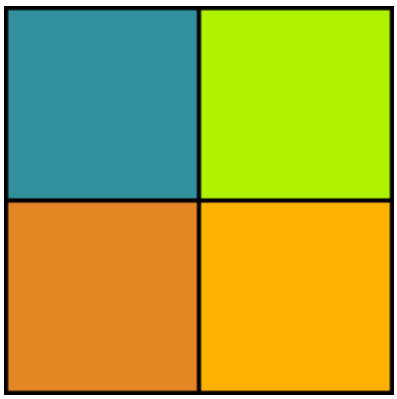


Figure 3: Imagine cu prag mai mare

7.3 Cerința 3

Pentru realizarea acestei cerințe, va trebui să citiți fișierul binar ce reprezintă compresia unei imagini. În acest fișier găsiți informațiile necesare pentru construcția arborelui cuaternar asociat imaginii. Astfel, pe măsură ce citiți conținutul fișierului puteți crea un arbore cuaternar. Având această structură ce modelează o imagine, trebuie doar să formăm imaginea pe baza arborelui.

Important

Trebuie să respectați structura unui fișier **PPM**, descrisă în secțiunea 5.1.

8 Restricții și precizări

Temele trebuie să fie încărcate pe vmchecker. **NU** se acceptă teme trimise pe e-mail sau altfel decât prin intermediul vmchecker-ului.

O rezolvare constă într-o arhivă de tip **zip** care conține toate fișierele sursă alaturi de un **Makefile**, ce va fi folosit pentru compilare, și un fisier README, în care se vor preciza detaliile implementării.

Makefile-ul trebuie să aibă obligatoriu regulile pentru **build** și **clean**. Regula **build** trebuie să aibă ca efect compilarea surselor și crearea binarului **quadtree**.

Programul vostru va primi, ca argumente în linia de comandă, numele fișierului de intrare și a celui de ieșire, dar și o opțiune în felul următor:

./quadtree [-c1 factor | -c2 factor | -d] [fișier_intrare] [fișier_ieșire] unde:

- **-c1 factor** indică faptul că programul va rezolva cerința 1 (**factor** = pragul impus pentru arborele de compresie);
- **-c2 factor** indică faptul că programul va rezolva cerința 2 (**factor** = pragul impus pentru arborele de compresie);
- **-d** indică faptul că programul va rezolva cerința 3;
- **fișier_intrare** reprezintă numele primului fișier de intrare (cel care conține imaginea);

- **fișier_ieșire** reprezintă numele fișierului de ieșire, în care se va scrie, în funcție de comanda primită, rezultatul execuției programului.

9 Referințe

1. PPM Format Specification

10 Punctaj

Cerința	Punctaj
Cerința 1	20 puncte
Cerința 2	30 puncte
Cerința 3	30 puncte
Coding style & warning-uri	15 puncte
README	5 puncte
BONUS (testat cu valgrind)	20 puncte

Atenție!

Orice rezolvare care nu conține structurile de date specificate **NU** este punctată.
Temele vor fi punctate doar pentru testele care sunt trecute pe vmchecker.
Nu lăsați warning-urile nerezolvate, deoarece veți fi depunctați.
Tema este individuală! Toate soluțiile trimise vor fi verificate, folosind o unealtă pentru detectarea plagiatului.