

# Generare Arbori pentru Evaluarea de Binarizări Optime (GAEBO)

@Managementul Proiectelor Software

## Raport dezvoltare Milestone 2 12.12.2022

---

### Autor raport

Șerban Alexandru-George 344C2

Universitatea Politehnica București

Facultatea de Automatică și Calculatoare

Email: alexandru.serban00@stud.acs.upb.ro

ID Teams: Șerban Alexandru-George (110133)

## Componența echipei

- + Project Manager / Manager de proiect: Șerban Alexandru-George 344C2
- + Team Leader / Liderul echipei: Marii Hristofor 344C4
  - + Software Developer / Dezvoltator software: Andrei Alin-Ionuț 344C3
  - + Software Developer / Dezvoltator software: Timpuriu Mircea 342C5
  - + Tester: Popa Marian-Elvis 342C5

## Legături cu documentația anterioară

Detaliile privind descrierea proiectului, planificare, metodologie folosită de dezvoltare a soluțiilor software (Spiral) și alte detalii tehnice este inclusă în: [MILESTONE1/ RAPORT INITIAL MILESTONE 1 PROIECT GAEBO.pdf](#)

Evoluția actualizată a lucrului efectuat este monitorizată în [TRACKING PROIECT GAEBO.pdf](#), dar și [DIAGRAMA GANTT PROIECT GAEBO.pdf](#).


Evidența tuturor documentelor scrise de Project Manager (Șerban Alexandru-George) se găsește în [README](#)

## Stadiu implementare și descriere sumară dezvoltare software

S-a finalizat proiectarea și implementarea procedurilor de binarizare globală.

Etapile de dezvoltare programatică sunt de antrenare (train), validare (validation) și testare (test), dependente de prelucrarea de metadata asociate unor imagini în setul de fișiere CSV. Datele arborelui performant se obțin după stagiile de antrenare și validare, fiind salvate în fișiere de output, pentru a extrage informații de performanță și a compara cu efectele aplicării pragurilor clasice pe imagini.

Modul de construire a arborelui este generativ și progresiv (poate fi asemuit unui pipeline). Astfel, introducerea de noduri tip rădăcină (root-uri) este etapizată și



dependentă aplicării rezultatului de funcții compuse. În urma etapei de antrenare, se preia ca ieșire a stadiului un abore parțial de train/sau testare ( $A_t$ ), fiind prelucrat în stagiul de validare, în urma căruia se obțin în final metadatele structurii arborescente definitive finale ( $A_v$ ).

Diferența dintre antrenare și validare constă în comparațiile de performanță pentru actualizarea arborelui. Pentru prima, inserarea de noduri rădăcină este condiționată de obținerea unei metrice de performanță mai bună decât structura  $A_t$  în care nu s-a adăugat respectivul root, pe când cea din urmă (care preia  $A_t$ ), nu mai apar în esență noi noduri,  $A_v$  obținându-se prin probarea lui  $A_t$  cu alți arbori aleatori.

## Evoluție dezvoltare proiect milestone 2

Întâlnirile în care s-au elaborat arhitectura și s-au dezbătut aspecte tehnice de implementare, precum obținerea efectului determinist al funcțiilor compuse utilizate pentru adăugarea root-urilor în arborele performant, au avut loc în cadrul laboratoarelor de MPS.

S-a păstrat o evidență de tip jurnal, reținându-se punctele succinte atinse ce relevă progresul software.

### 14 Noiembrie 2022


- Parsarea și analiza fișierelor CSV
- Împărțirea setului de date în 3 mulțimi: train, validation, test  
(s-a folosit funcția `train_test_split` din biblioteca `sklearn.model_selection`)

### 21 Noiembrie 2022

- Generare funcții pentru determinare noduri
- Analiză algoritm generare globală arbore

Conceperea arhitecturală a algoritmului din care rezultă arborii globali

Generarea unui arbore:



1. Se alege un număr random de thresholduri folosite într-o listă, număr fix de noduri arbore (8)

2. Se mapează funcția folosită pentru determinarea nodurilor cu parametrii folosiți și valoarea pe care o returnează (pentru un număr fix inițial de noduri)

3. Se generează un număr de noduri initiale, folosind o funcție aleatoare

4. Pentru restul nodurilor, generăm aleator 10 funcții pretendente, din care aplicăm funcția potrivită de determinare, cu valoarea cea mai apropiată de threshold

#### Alegerea arborelui cel mai bun:

1. Generăm un număr de arbori cu threshold-uri diferite folosite, și un număr de noduri inițiale pe care se aplică o funcție aleatoare

2. La introducerea unui nou nod, se determină funcția potrivită pentru fiecare root al fiecărui arbore; Arborele cu nodul care are cea mai slabă performanță nu va mai fi luat în considerare pe viitor.


3. Arborele final este cel care rămâne după toate eliminările de la pasul 2.

### **5 Decembrie 2022**

- Scriere funcției pentru crearea arborilor
- Algoritm (full) aleator de generare și (pseudo) de adăugare nod în arbore
- Creat și aplicat metrica de evaluare a performanței
- Combinarea etapelor anterioare de train, validation, test
- Formatarea outputului în fișier

## **Raport tehnic milestone 2**

Articolul științific în care se ilustrează introducerea în domeniu, descrierea soluției, rezultate intermediare soluție software și concluzii intermediare, se regăsește în



fișierul: [ARTICOL MILESTONE 2 SOLUȚIE DE GENERARE ARBORI PENTRU EVALUAREA DE BINARIZĂRI OPTIME GLOBALE.pdf](#)

## Raport de riscuri milestone 2

Planul de identificare a pericolelor în dezvoltarea proiectului GAEBO este intitulat [PLAN DE RISC PROIECT GAEBO](#). Se enumeră frecvența de apariție a unui eveniment (dată ca probabilitate risc) recunoscut după nume și ID care afectează componenta de cost (privită ca resurse umane), respectiv dimensiunea de posibilă întârziere în planificare.

În matricea de riscuri, rezultatul impactului de întârziere este produsul dintre probabilitatea riscului și întârziere, iar impactul costului este produsul dintre probabilitate risc și metrica de cost. La final, se computează suma impactului de întârziere cu cel de cost, obținându-se rezultatul final numit "*Rezultat risc*".


Se explicitează și măsurile de acțiune la apariția evenimentelor identificate.

## Impact metodologie software aleasă

Tehnica de dezvoltare Spiral a mitigat majoritar riscurile de scriere a surselor cod (source code), precum potențialele blocaje de programare logică ale dezvoltatorilor software.

Acest fapt s-a datorat prin lucrul modularizat, funcționalitățile de bază (precum parsarea setului CSV, generarea mulțimii de praguri de binarizare frunză pentru arbore, adăugarea unui root, returnarea f-measure-ului pentru o imagine, formatarea output-ului, etc.) fiind bine delimitate, interfețele de lucru fiind izolate pentru software developeri.

Totodată, și împărțirea etapizată, generativă, asemănătoare unui pipeline, asociate stagiile de antrenare, validare respectiv testare, au înlesnit scrierea individuală a programelor din ierarhia de fișiere.



Comunicarea săptămânală din cadrul întâlnirilor de colectare a progresului software iterativ a fost localizată la laboratoarele de MPS, pe 14, 21 Noiembrie, 5 Decembrie 2022. S-au putut explicita modalitățile de combinare ale interfețelor pentru integritatea arhitecturală.

Prin urmare, izolabilitatea conferită la un timp fixat de metodologia Waterfall, aplicat succesiv (spiralat) din săptămână în săptămână, a conferit dezvoltarea prezentată la acest milestone 2, reușindu-se implementarea binarizării globale.

### Feedback acordat membrilor echipei (aspecte de îmbunătățit)

- Team Leader / Liderul echipei: [Marii Hristofor 344C4](#)
  - Necesitate de a începe contribuirea la documentația proiectului (în special la milestone 3, pentru raportul de dezvoltare: ex. plan de testare și rezultate)
  - Comunicare mai eficientă cu dezvoltatorii software și testerul
- Software Developer / Dezvoltator software: [Andrei Alin-Ionuț 344C3](#)
  - Nevoie eficientizare scriere cod, întârziat datorită unor probleme tehnice cu mediul de lucru (pe laptop, defecțiuni)