Graph Builder

1. Introducere

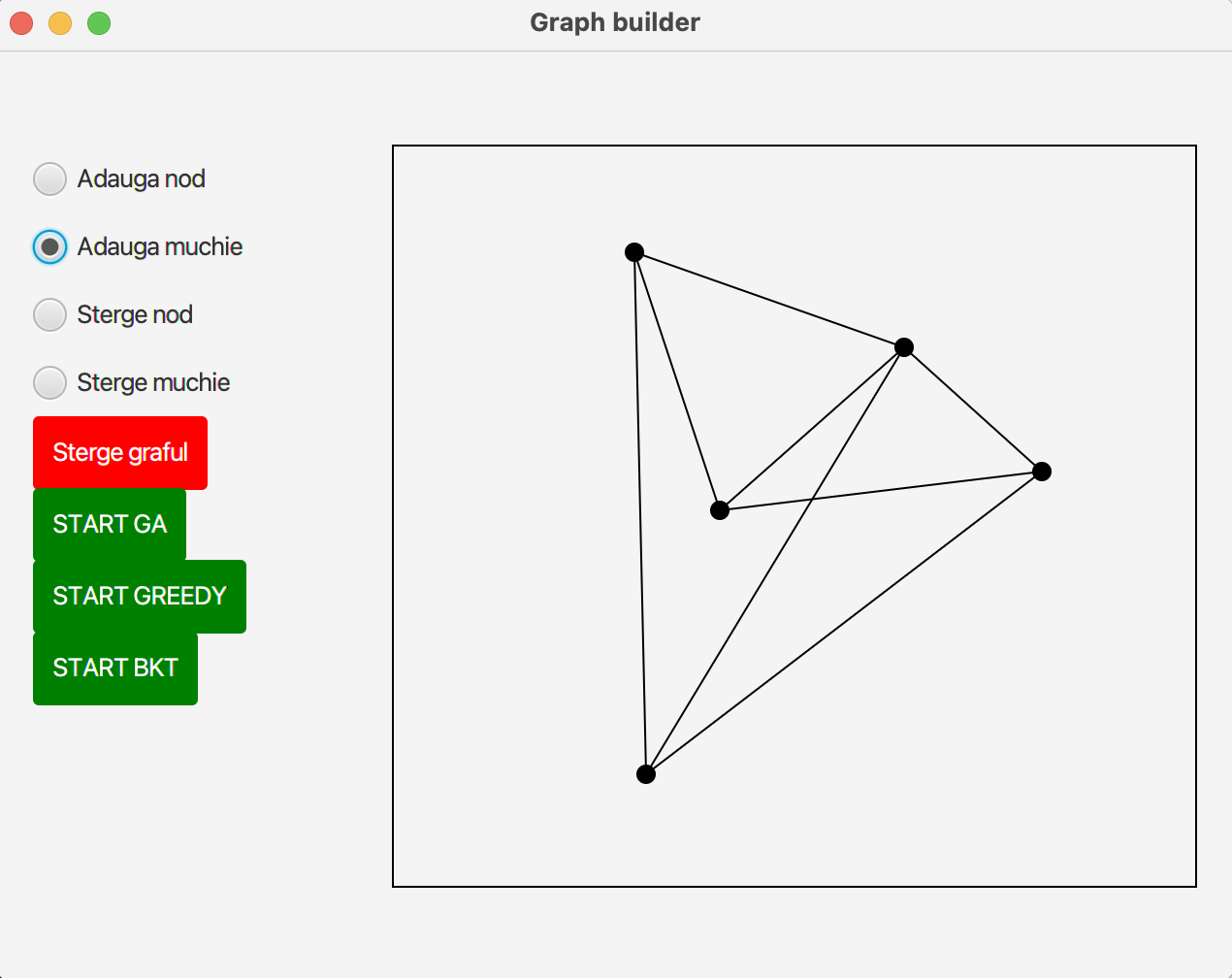
Graph Builder este o aplicatie ce permite utilizatorului sa construiasca un graf printr-o interfata grafica, si ulterior poate genera o colorare a grafului dat de catre utilizator. Colorarea grafului poate fi generata cu ajutorul a 3 algoritmi, unul greedy, unul de tip back tracking si un algoritm genetic, fiecare avand avantaje si dezavantaje specifice.

1. Construirea unui graf

Consturirea grafurilor se realizeaza prin intermediul interfetei grafice, utilizatorul avand posibilitatea alegerii unei actiuni din cele 4 disponibile:

* Adauga nod
* Adauga muchie
* Sterge nod
* Sterge muchie

Graful este afisat intr-un Pane, acelasi Pane in care interpreteaza comenzile utilizatorului pentru actualizarea grafului. Astfel, un click cu optiunea adauga nod selectata va genera un nod nou in graf, la pozitia indicate de cursor in momentul actiunii. Pentru a sterge un nod, utilizatorul trebuie sa selecteze cu un click nod-ul ce va fi sters. Pentru adaugarea / stergerea unei muchii, doua noduri trebuie selectate succesiv, pentru a genera / elimina o munchie dintre ele. Dupa selectarea primului nod, culoarea acestuia se modifica, in asteptarea selectiei nr 2.



1. Gasirea unei colorari optime

In procesul de generare a unei colorari minimale, am implementat 3 algoritmi, un Backtracking, un Greedy si un algoritm genetic.

1. **Algoritmul greedy**

Acest algoritm asigneaza culorile grafului in functie de gradul ficarui nod. Avantajul unui astfel de algorim este viteza de executie, fiind necesar un numar minim de parcurgeri ale grafului, si poate oferi solutia optima in unele cazuri.

Totusi, in cadrul unor configurari mai complexe, solutia optima nu este garantata.

1. **Algoritmul back tracking**

Spre deosebire de algoritmul greedy, back tracking-ul asigura acuratetea prin solutia optima, facand un compromise la capitlul viteza de exectuie. Acesta este executat in doi pasi, afla numarul minim de culori necesare, si ulterior genereaza o colorare pentru acesta. Dezavantajul este generarea tuturor posibilitatilor, si nu este o solutie fezabila pentru grafuri mari / foarte complexe.

1. **Algoritmul genetic**

Ultima solutie este un compromise intre viteza si acuratete. Aceasta dureaza semnificativ mai putin decat back tracking-ul, oferind o acuratete crescuta fata de algoritmul greedy (gaseste solutia optima in majoritatea cazurilor)

In implementarea acestuia generam o populatie de 200 indivizi si executam maxim 2000 generatii. De obicei solutia optima este gasita in primele generatii, restul fiind doar de validare, adaucand un beneficiu real doar in cadrul problemelor complexe.

Acest algoritm imita evolutia biologica, cu scopul de a “evolua” o solutie optima, incepand de la o multime randomizata. In acest sens vom introduce notiunea de fitness, ce defineste calitatea unui individ din populatie. Pentru noi, individul este reprezentat de o colorare, iar calitatea acesteia este invers proportionala cu numarul de noduri adiacente ce au culori identice.

Pentru a optimiza executia, algoritmul suporta “specii” diferite. O specie este definite prin numarul de culori utilizate. Astfel, incepem cu numarul de culori egal cu gradul maxim al unui nod din graf.

Cand optimizarea unei specii s-a finalizat, algoritmul incearca generarea unei specii noi, reducand numarul de culori disponibile. In cazul in care aceasta actiune nu poate duce la o solutie corecta dupa procesul optimizare, vom considera ca numarul de culori necesare este numarul precedent de culori disponibile, iar solutia optima este individul cu fitness-ul maxim din specia precedenta.

1. **Bibliografie**

<https://ro.wikipedia.org/wiki/Algoritm_genetic>

<https://profs.info.uaic.ro/~marta/ga/L3/>

<https://profs.info.uaic.ro/~acf/graph4j/>

<https://ro.wikipedia.org/wiki/Colorarea_grafurilor>