

Algoritmi evolutivi

1 CERINȚE

1. Să se implementeze **un algoritm evolutiv pentru problema rucsacului**.
 - a. Codificare binară
 - b. Operatori specifici (încrucișare, mutație)
 - c. Algoritm și parametrizare
 - d. Experimente pe cele două instanțe primite la Tema 1

2. Să se implementeze **un algoritm evolutiv pentru problema comis-voiajorului**.
 - a. Codificare prin permutări
 - b. Operatori specifici (încrucișare, mutație)
 - c. Algoritm și parametrizare
 - d. Experimente pe instanța primită la Tema 2

Observații (neîndeplinirea cerințelor din observații conduce la scăderea punctajului acordat):

- ✓ Aplicația trebuie să fie modularizată, să permită parametrizarea algoritmului și afișarea soluției.
- ✓ Testați algoritmul în diverse variante (minim 2 variante de algoritm) pentru comparații.
- ✓ Rezultatele experimentelor trebuie salvate (cu indicarea setărilor folosite: algoritmul, valori parametri, număr rulări – minim 10 rulări/configurație, calitatea soluției – best/avg).

2 TERMEN DE PREDARE

- **Lab 4**

Total Punctaj Tema 3 = 150p

3 PREDAREA TEMEI PRIN MS TEAMS

Încarcăți următoarele fișiere **ÎNAINTE** de a începe lab-ul în care este setat termenul de predare:

1. O arhivă cu codul sursă
2. Un document (Word/PDF) care să conțină:

- ✓ Descrierea pe scurt a algoritmului implementat (pseudocod) si principalelor componente (reprezentare solutie, functie de fitness, operatori, etc)
- ✓ Indicarea parametrilor algoritmului
- ✓ Tabele și grafice cu rezultatele obtinute (comparatii pentru cel puțin 3 seturi de valori ale parametrilor pentru fiecare instanta de problema)
- ✓ Analiza rezultatelor

4 ALGORITM EVOLUTIV

Algoritmul evolutiv pentru problema rucsacului va fi implementat pe baza unei reprezentări binare, repararea indivizilor care reprezintă soluții nevalide (pentru a respecta capacitatea rucsacului) și o funcție de fitness ce calculează valoarea totală pusă în rucsac (ce trebuie maximizată).

Algoritmul evolutiv pentru problema TSP va fi implementat pe baza unei reprezentări prin permutări, folosirea unor operatori de încrucișare și mutație care generează soluții valide și o funcție de fitness ce calculează distanța totală parcursă (ce trebuie minimizată).

Componentele algoritmului sunt:

- Generare populație inițială aleator (populație P).
- Selecție părinți.
- Generare descendenți: încrucișare și mutație (populație O).
- Selecție supraviețuitori: cei mai buni size(P) din P+O.
- Parametri algoritm: mărimea populației, numărul de generații, probabilitatea de încrucișare, probabilitatea de mutație.

```
BEGIN
  INITIALISE population with random candidate solutions;
  EVALUATE each candidate;
  REPEAT UNTIL ( TERMINATION CONDITION is satisfied ) DO
    1 SELECT parents;
    2 RECOMBINE pairs of parents;
    3 MUTATE the resulting offspring;
    4 EVALUATE new candidates;
    5 SELECT individuals for the next generation;
  OD
END
```