**Tema:** Algoritmi genetici

## Introducere

Algoritmii genetici fac parte din clasa mai generală a algoritmilor evolutivi. Algoritmii evolutivi sunt metode de rezolvare a problemelor bazate pe o căutare în spaţiul soluţiilor, în care se folosesc populaţii supuse unui proces de evoluţie similar într-o măsură oarecare cu cel întâlnit în natură.

Principalele noţiuni care permit analogia între rezolvarea problemelor de căutare şi evoluţia în natură sunt următoarele:

- *Populaţie*. O populaţie este formată din indivizi care trăiesc într-un mediu la care trebuie să se adapteze.
- *Fitness.* Fiecare individ din populaţie este adaptat mai mult sau mai puţin la mediu. Fitness-ul este o măsură a gradului de adaptare la mediu. Scopul evoluţiei este ca toţi indivizii să ajungă la un fitness cât mai bun.
- *Cromozom.* Este o mulţime ordonată de elmente, numite *gene* ale căror valori determină caracteristicile unui individ. În genetică poziţiile pe care se află genele în cadrul cromozomului se numesc *loci*, iar valorile pe care le pot lua se numesc *alele.*
- *Generaţie.* Este o etapă în evoluţia unei populaţii. Dacă privim evoluţia ca pe un proces iterativ în care o populaţie se transformă în alta atunci generaţia este o iteraţie în cadrul acestui proces.
- *Selecţie.* Procesul de selecţie naturală are ca efect supravieţuirea indivizilor cu grad ridicat de adaptare la mediu.
- *Reproducere.* Este procesul prin care se trece de la o generaţie la alta. Indivizii noii generaţii moştenesc caracteristici de la precursorii lor dar pot dobândi şi caracteristici noi ca urmare a unor procese de mutaşie care au un caracter aleator. În cazul în care în procesul de reproducere intervin cel puţin doi pîrinţi caracteristicile moştenite ale urmaşului se obţin prin combinarea caracteristicilor părinţilor.

## 1 Structura generală a unui algoritm genetic

Algoritmii genetici sunt procese iterative prin care o populaţie iniţializată aleator este transformată succesiv prin selecţie,

încrucişare şi mutaţie până la atingerea unui număr anumit de iteraţii sau până la îndeplinirea unui alt criteriu de oprire.

Există un număr foarte mare de variante de algoritmi genetici datorită flexibilităţii lor. În general pentru fiecare problemă concretă apar elemente particulare în cadrul algoritmului. Cu toate acestea, majoritatea algoritmilor genetici se pot încadra în următoarea structură generală, unde $P(t)= (x1(t), x2(t),..., xm(t))$ reprezintă populaţia corespunzătoare generaţiei $t$ iar $f(x_i)$ reprezintă gradul de adaptare la mediu al elementului $x_i$:

***genetic algoritm* (*P*(*t*))**

1: *Iniţializare*: $P(0)= (x1(0), ..., xm(0))$, $t = 0$

2: **repeat**

3: *evaluarea populaţiei curente:* calcul $f(x_i(t))$ pentru $i = 1, 2, ..., m$

4: *selecţia părinţilor*: $P(t) \rightarrow P^1$

5: *generarea urmaşilor prin încrucişare*: $P^1 \rightarrow P^2$

6: *modificarea urmaşilor prin mutaţie*: $P^2 \rightarrow P^3$

7: *evaluarea populaţiei de urmaşi*: calcularea valorilor lui $f$ pentru elementele lui $P^3$

8: *selecţia supravieţuitorilor*: $\{P(t), P^3\} \rightarrow P(t + 1)$

9: *incrementarea contorului de generaţii*: $t = t + 1$

10: **until** $t < t_{max}$

La proiectarea unui algoritm genetic trebuie să se stabilească:

- *Modul de codificare.* Se specifică modul în care fiecărei configuraţii din spaţiul de căutare i se asociază un cromozom.
- *Funcţia de adaptare.* Se construieşte funcţia care exprimă gradul de adaptare la mediu pornind de la restricţiile şi funcţia scop a problemei.
- *Modul de iniţializare şi dimensiunea populaţiei.* Se pot utiliza populaţii de dimensiune fixă sau de dimensiune variabilă. De cele mai multe ori populaţia se iniţializează cu elemente aleatoare din spaţiul de căutare.
- *Mecanismul de selecţie* a părinţilor şi a supravieţuitorilor.
- *Mecanismul de combinare* a părinţilor pentru a genera urmaşi.
- *Mecanismul de mutaţie* prin care se asigură modificarea

cromozomilor generaţi.

- *Criteriul de oprire*. Atunci când nu se cunoaşte un criteriu specific problemei se optează pentru un număr maxim de iteraţii.

### 1.1 Codificarea cromozomilor

Codificarea spaţiului de căutare este una dintre cele mai importante etape în proiectarea unui algoritm genetic deoarece determină modul în care se va desfăşura procesul evolutiv. Această etapă se referă la următoarele aspecte:

- structuri de date folosite;
- regula de codificare;
- regula de decodificare.

**Structuri de date.** În algoritmii genetici cromozomii sunt reprezentaţi prin structuri liniare cu număr fix de elemente (tablouri) cel mai des folosite sau cu număr variabil de elemente (liste înlănţuite).

**Reguli de codificare.** Modul în care sunt codificaţi cromozomii depinde de problema concretă. Există următoarele variante de codificare:

- *Codificare binară.* Varianta clasică de codificare a cromozomilor. Într-o astfel de codificare cromozomii sunt vectori cu elementele din mulţimea $\{0, 1\}$ iar spaţiul de căutare este $S\{0, 1\}^n$, cu $n$ gene (numărul de gene este corelat cu dimensiunea problemei). Codificarea binară este utilizată în problemele de optimizare combinatorială în care configuraţiile pot fi specificate ca vectori binari.
- *Codificarea reală.* Este adecvată pentru problemele de optimizare pe domenii continue. În acest caz cromozomii sunt nişte vectori cu elemente reale, fiind chiar elementele domeniului de definiţie al funcţiei scop. Avantajul codificării reale este faptul că este naturală şi nu necesită proceduri speciale de codificare şi decodificare.
- *Codificarea specifică.*Fiecare cromozom este reprezentat sub forma unei permutări:

| | |
|---|---|
| Cromozomul A | 1 5 3 4 6 3 4 8 9 2 |

| | |
|---|---|
| Cromozomul B | 9 2 4 3 3 6 1 5 4 8 |

Această metodă de codificare se foloseşte în cazul problemei voiajorului comercial. O permutare reprezintă ordinea în care voiajorul comercial vizitează oraşele.

**Reguli de decodificare.** Decodificarea asigură pregătirea evaluării configuraţiei. Ea este necesară doar pentru codificarea binară şi pentru codificarea specifică.

**1.2 Populaţia iniţială**

Populaţia iniţială este o mulţime de cromozomi care trebuie să fie suficient de mare pentru a se putea realiza un număr suficient de combinaţii între cromozomii componenţi, dar nu prea mare, deoarece aceasta poate încetini algoritmul. De obicei populaţia iniţială este una aleatoare, dar pentru o mai bună performanţă, o parte din cromozomi pot fi generati prin metode euristice. În acest mod s-ar putea ajunge la soluţie mai repede.

**Funcţia de adecvare (fitness-ul)**

Fitness-ul unui cromozom este speranţa lui de viaţă sau gradul de adaptare la mediu. Scopul este de a obţime cromozomi cu fitnnes cât mai mare sau cât mai mic, adică un fitness la extremă. Fitness-ul este exprimat de o funcţie ce urmează a fi optimizată minimizată sau maximizată.

**Selecţia cromozomilor**

Cromozomii sunt selectaţi din populaţie pentru a fi părinţi într-o încrucişare.

Există mai multe metode de selecţie, printre care:

***1. Selecţie aleatoare***. Doi părinţi sunt aleşi aleator din populaţie. Aceasta se face generând două numere aleatoare între 1 şi dimensiunea populaţiei.

***2. Selecţie cu ajutorul ruletei***. Părinţii sunt selectaţi în conformitate cu fitness-ul lor. Cu cât sunt mai buni, cu atât şansa lor de a fi aleşi este mai mare. Ne imaginăm o ruletă în care sunt dispuşi toţi cromozomii din populaţie. Fiecărui cromozom îi corespunde un sector al ruletei, direct proporţional, ca mărime, cu fitness-ul cromozomului respectiv. În acest fel cromozomii cu fitness mare au

ataşate sectoare mai mari, iar cei cu fitness mic au ataşate sectoare mai mici. La aruncarea bilei pe ruletă există mai multe şanse de alegere pentru cromozomii cu fitness mai mare.

Simularea roţii de ruletă se face în felul următor:

- Se calculează suma tuturor fitness-urilor cromozomilor (*S*).
- Se generează un număr aleator *r* între 1 şi *S*.
- Se parcurge populaţia şi se calculează suma fitness-urilor cromozomilor până ajungem la valoarea *r*. Acel cromozom îl alegem.

***3. Selecţie aranjată (Selecţie pe baza rangurilor).*** Se ordonează crescător valorile funcţiei de adecvare pentru toţi cromozomii. Se renumerotează cu numere întregi din intervalul [1, ..., *dimensiunea populaţiei*]. Cromozomul cu fitness-ul cel mai mic are numărul 1etc., iar cel mai mare are numărul egal cu dimensiunea populaţiei. Aceste numere sunt considerate fitness-uri şi pe ele se aplică selecţia proporţională. Se observă că cromozomii cu fitness mai mare au la fel cele mai mari şanse de a fi aleşi, dar de data aceasta nu mai sunt aşa de mari în comparaţie cu şansele celorlalţi.

***4. Selecţie de tip turneu.*** Se selectează uniform aleator câte *k* cromozomi din populaţie iar dintre aceştea se alege cel care are cea mai mare valoare pentru funcţia de adecvare. Cel mai frecvent se utilizează *k*=2.

***5. Seleţie prin trunchiere.*** Se foloseşte atunci când dintr-o populaţie sau o reuniune de populaţii se cere selectarea unei subpopulaţii. Sunt selectate atâtea elemente câte sunt necesare pentru a completa subpopulaţia în ordinea descrescătoare a valorii funcţiei de adecvare.

***6. Elitism.*** Prin elitism se înţelege supravieţuirea celui mai bun dintre elementele generate până la un moment dat. Nu toate variantele de selecţie satisfac această proprietate. De exemplu, selecţia prin trunchiere este elitistă, însă cea de tip turneu nu este elitistă. O metodă de selecţie poate fi transformată în una elitistă prin amplasarea explicită a celui mai bun element al populaţiei curente în populaţia corespunzătoare generaţiei următoare.

**Încrucișarea (crossover)**

Încrucișarea va depinde de tipul de codificare a cromozomilor și de particularitățile problemei pe care o rezolvăm. Vom analiza cîteva metode de încrucișări:

- ***Încrucișare într-un singur punct***. Se alege un punct de încrucișare. Din primul părinte se copie prima secvență, din al doilea părinte a doua secvență în raport cu punctul de încrucișare.

**10100**1011 + 11011**1111** = **101001111**

- ***Încrucișare în două sau mai multe punct***. Sunt alese mai multe puncte de încrucișare. Secvențele dintre cele puncte alese sunt luate alternativ, din primul părinte apoi din al doilea s.a.m.d.

**10**100**10**11 + 110**111**1**1** = **101111011**

- ***Încrucișare uniformă***. Biții sunt copiați aleator din primul și al doilea părinte.

1**0**100**1011** + **1**10**1**1111 = **101111011**

- ***Încrucișare aritmetică.*** Pentru crearea de noi urmați putem folosi operațiile precum și funcții aritmetice (**and, or, not** etc)

101001011 + 110111111 = **100001011 (and)**

**Mutația**

Asigura alterarea valorii unor gene pentru a evita situatiile in care o anumita gena nu apare in populatie fiindca nu a fost generata de la inceput. In acest fel se asigura diversitatea populatiei. Prin mu- tație se evită căderea soluțiilor problemei într-un optim local. De regulă această operație se efectuiază de nu prea multe ori, cu o probabilitate de circa 0,5%. În dependență de codificarea cromo- zomilor avem diferite metode de mutație.

**Codificarea binară**

***Inversarea biților*** Biții selectați sunt inversați 1↔0.

11**0**010011→11**1**010011

***Interschimbarea biților*** Biții selectați își schimbă valorile între ei.

11**0**010011→11**1**010011

**Codificarea sub formă de permutare**

***Transpoziție*** Asemănător interschimbării biților de la codificarea binară. Se efectuiază o transpoziție asupra permutării respective.

(12**3**4678**9**5)→(12**9**4678**3**5)

**Codificarea sub formă de valori**

Se adaugă sau se scade un număr mic la valorile selectate.
(**4.63** 5.32 2.89 **3.41** 9.67)→( **4.85** 5.32 2.89 **3.63** 9.67)

**Criteriul de oprire**

Algoritmii genetici se termină de obicei, după ce s-au creat un număr predefinit de populații noi. În alte cazuri se poate determina dacă soluția s-a îmbunătățit de la o populație la alta. În caz afirmativ se continua cu crearea de noi populații, iar în caz negative se afișează cel mai bun cromozom din populația curentă.

**Întrebări de control:**

1. De unde este inspirată ideea algoritmilor genetici? Cu ce noțiuni se lucrează?
2. Ce tip de probleme putem rezolva cu acești algoritmi?
3. Descrieți structura generală a unui algoritm genetic?
4. Ce este asemanator și diferit de tehnica greedy?
5. Care sunt condițiile de oprire a unui algoritm genetic?