**Funcția Schwefel**

**  
**

****

f(x)=-n·418.9829; x(i)=420.9687, i=1:n.

**Problema 1 Algoritm evolutiv**

1. Generare populație inițială aleatorie (populație P).
2. Evaluarea populației pentru a obține best, avg și worst
3. Selecție părinți.
4. Formarea copiilor prin încrucișare (C)
5. Transformarea copiilor print mutație (M)
6. Selecție supraviețuitori: cei mai buni mărime(P) din P+M. Populația nouă se reține în P
7. Evaluarea poluției noi pentru a obține best, avg și worst
8. Cât timp număr de generației e mai mic decât numărul de generații totale se execută totul începând de la pasul 3
9. Se returnează cel mai bun individ(best) din ultima generație

**Parametri algoritm: mărimea populației, numărul de generații, rata de încrucișare, rata de mutație.**

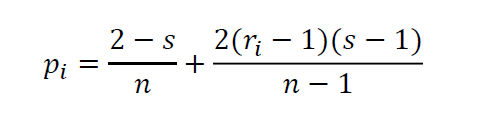
func algoritmEvolutiv(mărimePopulație, ratăMutație, ratăÎncrucișare, generații):P = inițializarePopulație(mărimePopulație)  
 b, a, w = evaluare(P) # best, avg, worst  
 nrÎncrucișări = int((mărimePopulație \* ratăÎncrucișare / 100) / 2)  
 pentru i între (0, generații - 1):

părinți = selecție(P)  
 C = încrucișare(nrÎncrucișări, părinți)  
 M = mutații(C)  
 P = populațieNouă(P, M)  
 b, a, w = evaluarePopulație(P)  
 returnează b

**Selecție**

**Selecția prin ordonare(Rank)**

* ri – rangul individului i (ri =1 pentru cel mai slab individ)
* s – presiunea de selecție
* Probabilitatea de selecție depinde de poziția relativa a cromozomilor:



**Selecția de tip turneu(Turnir)**

* Se aleg in mod aleatoriu k cromozomi
* Se calculează fitness-ul cromozomilor selectați
* Cromozomul mai performant este selectat



**Încrucișare**

**Încrucișarea discreta**

* Pentru fiecare poziție i a primului descendent se alege (cu o probabilitate fixata p) părintele a cărui gena (din poziția i) va fi transmisa acestui descendent
* p=0.5 rolul celor doi părinți este simetric

**Încrucișarea continuă (sau medie)**

* Se aleg aleatoriu (cu o probabilitate fixată p) anumite poziții
* Genele corespunzătoare în descendenți vor fi media aritmetica a genelor corespunzătoare ale părinților

**Mutație**

**Mutația uniformă**

* Înlocuiește o singura gena a cromozomului cu un număr real generat aleatoriu
* Părinte x = (x 1 , x 2 , . . . , x n)
* După mutație pentru poziția i

x’= (x 1 , x 2 , . . . , x i ’, . . . , x n ) , 1 <= i <= n ,

x i ’ aparține LI i , LS i ], uniform aleatoriu selectat

* Var : gena i determinată aleatoriu pe baza rate de mutație
* Var : fiecare poziție suferă mutație cu probabilitatea pm

**Mutația neuniformă**

La fiecare generație t se generează aleatoriu doi parametri :

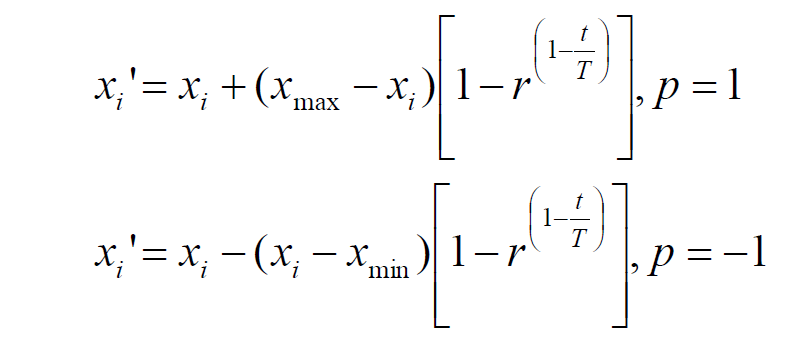
* p indică natura schimbării

p=1 indică o creștere a valorii unei gene

p= -1 indică o descreștere

* r determină amplitudinea schimbării
* r este un număr aleatoriu în intervalul [0,1], urmând o distribuție uniformă

T reprezintă numărul maxim de generații.



Xmin și Xmax reprezintă valorile maxime care pot fi luate de către Xi

**Tabele de date**

**Testele pentru algoritm au fost făcute începând de la un caz de bază pentru care am tot modificat parametri.**

**Pentru compararea cu alte variante ale algoritmului au fost schimbate doar încrucișarea, mutația și selecție pentru cazul de bază.**

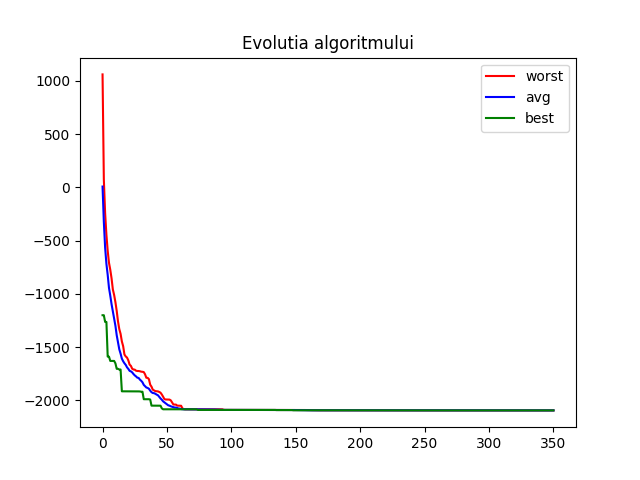
**Rezultatele optime ale problemei sunt:**

**Mărime problemă = 5; rezultat optim = -2094.9145**

**Mărime problemă = 10; rezultat optim = -4189.8290**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Mărime problemă | Nr rulări | Medie best | Medie average | Medie worst | Cel mai bun best | Mărime populație | Număr generații | Operator de încrucișare | Rata de încrucișare | Prob | Operator de mutație | Rata de mutație | Operator selecție | Mărime turneu/presiunea de selecție | Timpul mediu de execuție |
| 5 | 10 | -2094.5148 | -2094.5050 | -2094.4744 | -2094.8851 | 70 | 350 | Medie | 60 | 0.3 | Uniformă | 0.4 | Turnir | 4 | 2.63 |
| -2073.2957 | -2072.1894 | -2070.8168 | -2089.0765 | 10 | 350 | 60 | 0.3 | 0.4 | 4 | 0.37 |
| -2031.4980 | -1895.8222 | -1800.2890 | -2071.9218 | 70 | 50 | 60 | 0.3 | 0.4 | 4 | 0.40 |
| -2094.4285 | -2094.0931 | -2094.0085 | -2094.8690 | 70 | 350 | 40 | 0.3 | 0.4 | 4 | 2.08 |
| -2094.5101 | -2094.4468 | -2094.3675 | -2094.8138 | 70 | 350 | 60 | 0.1 | 0.4 | 4 | 2.79 |
| -2094.7421 | -2094.7415 | -2094.7406 | -2094.9005 | 70 | 350 | 60 | 0.3 | 0.1 | 4 | 2.80 |
| -2094.3900 | -2094.2002 | -2094.1627 | 2094.8696 | 70 | 350 | 60 | 0.3 | 0.4 | 2 | 2.45 |
| -2071.5435 | -1920.7578 | -1866.7443 | -2092.8345 | 70 | 350 | 60 | 0.3 | 0.4 | Rank | 1.75 | 12.41 |
| -2062.7603 | -1929.5265 | -1879.2218 | -2094.1628 | 70 | 350 | 60 | 0.3 | 0.4 | 1.5 | 12.29 |
| -2094.8979 | -2094.8249 | -2094.7577 | -2094.9141 | 70 | 350 | 60 | 0.3 | Neuniformă | 0.4 | Turnir | 4 | 2.72 |
| -2094.5247 | -2093.1173 | -2092.4992 | -2094.8846 | 70 | 350 | 60 | 0.3 | 0.4 | Rank | 1.75 | 12.35 |
| -2094.1075 | -2094.0699 | -2094.0456 | -2094.5159 | 70 | 350 | Discretă | 60 | 0.5 | Uniformă | 0.4 | Turnir | 4 | 2.67 |
| -2094.2979 | -2094.2809 | -2094.2778 | -2094.8682 | 70 | 350 | 60 | 0.2 | 0.4 | 4 | 2.68 |
| -2082.9460 | -2045.3837 | -2027.3405 | -2089.9193 | 70 | 350 | 60 | 0.2 | 0.4 | Rank | 1.75 | 12.78 |
| -2094.8642 | -2094.8391 | -2094.8215 | -2094.9135 | 70 | 350 | 60 | 0.2 | Neuniformă | 0.4 | Turnir | 4 | 2.88 |
| -2094.5168 | -2093.6594 | -2093.2394 | -2094.7880 | 70 | 350 | 60 | 0.2 | 0.4 | Rank | 1.75 | 13.18 |
| 10 | 10 | -3902.3879 | -3638.5121 | -3517.2543 | -4057.6643 | 70 | 350 | Medie | 60 | 0.3 | Uniformă | 0.4 | Turnir | 4 | 4.40 |
| -3488.8290 | -3376.6881 | -3309.0389 | -3711.9501 | 10 | 350 | 60 | 0.3 | 0.4 | 4 | 0.61 |
| -2967.1254 | -2449.7915 | -2251.3912 | -3334.7626 | 70 | 50 | 60 | 0.3 | 0.4 | 4 | 0.59 |
| -3806.4690 | -3436.5658 | -3294.8257 | -4007.6718 | 70 | 350 | 40 | 0.3 | 0.4 | 4 | 3.20 |
| -3953.6836 | -3753.3294 | -3641.4505 | -4108.1183 | 70 | 350 | 60 | 0.1 | 0.4 | 4 | 4.04 |
| -4188.9504 | -4188.9256 | -4188.9118 | -4189.6816 | 70 | 350 | 60 | 0.3 | 0.1 | 4 | 4.05 |
| -3756.9497 | -3385.9480 | -3247.0221 | -3992.8345 | 70 | 350 | 60 | 0.3 | 0.4 | 2 | 3.46 |
| -3053.4228 | -2613.3230 | -2470.4265 | -3179.1316 | 70 | 350 | 60 | 0.3 | 0.4 | Rank | 1.75 | 24.98 |
| -3121.4403 | -2619.6293 | -2467.7322 | -3250.9678 | 70 | 350 | 60 | 0.3 | 0.4 | 1.5 | 22.51 |
| -4177.6184 | -4177.1576 | -4176.8926 | -4189.6452 | 70 | 350 | 60 | 0.3 | Neuniformă | 0.4 | Turnir | 4 | 4.11 |
| -4182.8715 | -4176.1428 | -4173.3878 | -4185.5324 | 70 | 350 | 60 | 0.3 | 0.4 | Rank | 1.75 | 23.36 |
| -4075.8538 | -3949.2643 | -3857.3072 | -4122.8818 | 70 | 350 | Discretă | 60 | 0.5 | Uniformă | 0.4 | Turnir | 4 | 4.21 |
| -4058.9793 | -3923.7185 | -3832.1543 | -4119.0798 | 70 | 350 | 60 | 0.2 | 0.4 | 4 | 3.92 |
| -3854.5508 | -3467.9058 | -3318.7380 | -4023.8243 | 70 | 350 | 60 | 0.5 | 0.4 | Rank | 1.75 | 22.52 |
| -4177.7297 | -4177.3370 | -4177.1156 | -4189.7927 | 70 | 350 | 60 | 0.5 | Neuniformă | 0.4 | Turnir | 4 | 4.02 |
| -4183.8760 | -4176.6031 | -4173.4716 | -4186.4387 | 70 | 350 | 60 | 0.5 | 0.4 | Rank | 1.75 | 23.12 |

**Reprezentări grafic**



Mărime problemă: 5

Mărime populație: 70

Număr generații: 350

Operator încrucișare: Medie

Rata de încrucișare: 60

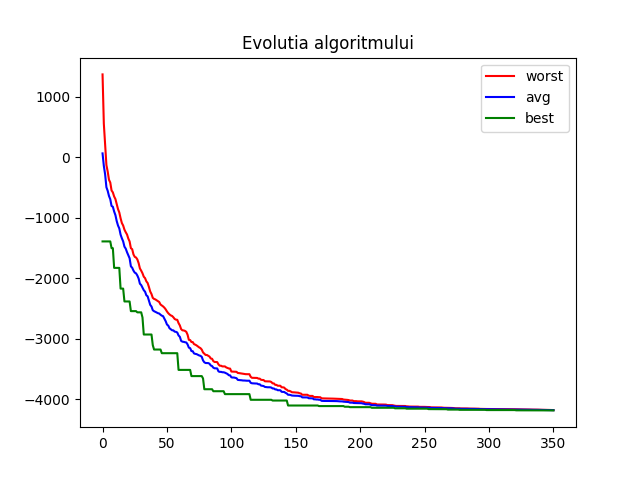
Probabilitate: 0.3

Operator mutație: Uniformă

Rata de mutație: 0.3

Operator selecție: Turnir

Mărime turneu: 4



Mărime problemă: 5

Mărime populație: 70

Număr generații: 350

Operator încrucișare: Discretă

Rata de încrucișare: 60

Probabilitate: 0.5

Operator mutație: Neuniformă

Rata de mutație: 0.3

Operator selecție: Rank

Presiunea de selecție: 1.35

**Observații**

Observațiile următoare sunt făcute pe baza datelor din tabel și a reprezentărilor grafice:

* Din datele prezentate în tabel se poate observă că parametrul cu cel mai mare efect negativ asupra rezultatelor obținute este numărul de generații parcurse;
* Se poate că selecția de tip rank oferă rezultate mai slabe pentru problemă;
* Problema dă rezultate mai slabe pentru o populație mai mică;
* Indiferent de operatorul de încrucișare, acesta dă rezultate mai bune pentru o rată de încrucișare mai mare și o probabilitate mai mare;
* Pentru o rată de mutație mai mică se obțin rezultate mai bune;
* Selecția de tip rank afectează cel mai mult timpul de execuție într-un mod negativ;
* Pentru dimensiunea 5 a problemei cel mai apropiat rezultat de cel optim a fost -2094.9141;
* Pentru dimensiunea 10 a problemei cel mai apropiat rezultat de cel optim a fost -4189.7927;
* Din reprezentările grafice se poate observa cum a evoluat cea mai bună soluție și cum pentru o dimensiune mai mare a luat mai multe generații să se ajungă la o soluție mai bună.

**Concluzie**

Din cele notate anterior se poate observa că algoritmi folosiți anterior au dat rezultate foarte apropiate de cele optime. Cele mai slabe rezultate au fost obținute pentru un număr mai mic de generații.

**Problema 2 PSO**

Fiecare particulă are asociată

* poziție –potențială soluție a problemei
* viteză –modifică o poziție în altă poziție
* funcție de calitate (fitness)

Fiecare particulă trebuie să poată:

* interacționa (schimba informații) cu vecinii ei
* memora o poziție precedentă
* utiliza informațiile pentru a lua decizii

Inițializarea particulelor

* poziții aleatorii
* viteze aleatorii

func PSO(mărimeProblemă, nrParticele, nrIterații, w, c1, c2):particule = generareParticule(mărimeProblemă, nrParticule)  
 g\_best = getGlobalBest(particule, mărimeProblemă, nrParticule)  
 pentru k până la nrIterații:  
 pentur i până la nrParticule:  
 particule[i].update\_p\_best()  
 particule[i].update\_velocity(w, c1, c2, g\_best)  
 particule[i].update\_positions()  
 g\_best = getGlobalBest(particles, problem\_size, no\_particles)  
 returnează g\_best

Modificarea vitezei v

**Eq. (a):**

v[] = w \* v[] +

c1 \* rand() \* (pbest[] -x[]) +

c2 \* rand() \* (gbest[] -x[])

•Modificarea poziţiei

**Eq. (b):**

x[] = x[] + v[]

**Tabele de date**

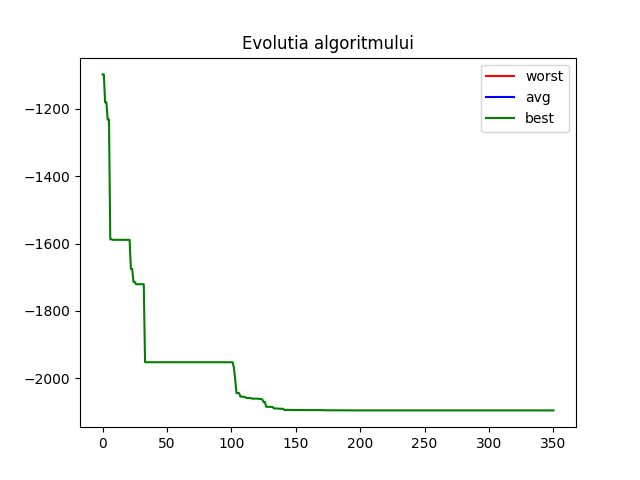
**Testele pentru algoritm au fost făcute începând de la un caz de bază pentru care am tot modificat parametri.**

**Mărime problemă = 5; rezultat optim = -2094.9145**

**Mărime problemă = 10; rezultat optim = -4189.8290**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Mărime problemă | Nr rulări | Medie best | Cel mai bun best | Număr particule | Număr iterații | Factorul de inerție | F. învățare cognitiv | F. învățare social | Timpul mediu de execuție |
| 5 | 10 | -2094.7399 | -2094.9144 | 70 | 350 | 0.75 | 1 | 2 | 1.50 |
| -1787.9319 | -1976.4760 | 10 | 350 | 0.75 | 1 | 2 | 0.23 |
| -1760.8353 | -1894.8165 | 70 | 50 | 0.75 | 1 | 2 | 0.23 |
| -2094.8423 | -2094.9144 | 70 | 350 | 0.25 | 1 | 2 | 1.57 |
| -1985.0921 | -2094.8962 | 70 | 350 | 0.75 | 2 | 2 | 1.52 |
| -1900.4732 | -2051.4070 | 70 | 350 | 0.75 | 1 | 3 | 1.57 |
| -2094.7961 | -2094.9144 | 70 | 350 | 0.75 | 2 | 1 | 1.57 |
| 10 | 10 | -3541.2710 | -3935.2754 | 70 | 350 | 0.75 | 1 | 2 | 2.86 |
| -2921.1499 | -3465.6751 | 10 | 350 | 0.75 | 1 | 2 | 0.42 |
| -2571.7309 | -2925.3915 | 70 | 50 | 0.75 | 1 | 2 | 0.44 |
| -3720.5523 | -4011.3848 | 70 | 350 | 0.25 | 1 | 2 | 3.07 |
| -2744.7221 | -3334.6693 | 70 | 350 | 0.75 | 2 | 2 | 2.86 |
| -2839.0890 | -3335.4330 | 70 | 350 | 0.75 | 1 | 3 | 3.00 |
| -3692.0825 | -4021.0077 | 70 | 350 | 0.75 | 2 | 1 | 2.88 |

**Reprezentări grafic**



Mărime problemă: 5

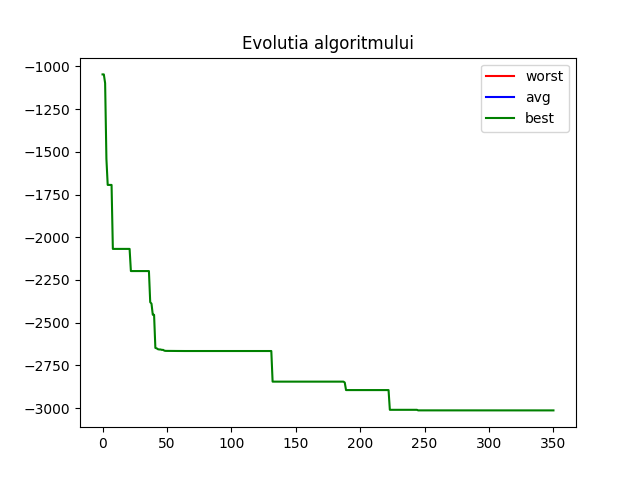
Număr particule: 70

Număr iterații: 350

Factorul de inerție: 1.75

Factorul de învățare cognitiv: 1

Factorul de învățare social: 2



Mărime problemă: 5

Număr particule: 70

Număr iterații: 350

Factorul de inerție: 0.5

Factorul de învățare cognitiv: 2

Factorul de învățare social: 2

**Observații**

Observațiile următoare sunt făcute pe baza datelor din tabel și a reprezentărilor grafice:

* Se poate observa că în ambele dimensiuni ale problemei au fost obținute rezultate mai bune pentru un factor de inerție mai mic;
* Pentru ambele dimensiuni ale problemei au fost obținute rezultate mai bune când problema a avut factorul de învățare social mai mic și factorul de învățare cognitiv mai mare;
* Pentru un număr de particule mai mic sau un număr de iterații mai mic, au fost obținute rezultate mai proaste;
* Pentru dimensiunea 5 a problemei, algoritmul a dat rezultate cu foarte puțin mai bune. Cel mai apropiat rezultat de cel optim a fost -2094.9144;
* Pentru dimensiunea 10 a problemei algoritmul a dat rezultate cu mult mai slabe în general. Cel mai apropiat rezultat de cel optim a fost -4021.0077;
* Din reprezentarea grafică se poate observa cum se modifică cea mai bună valoare până se ajunge la un punct de stop în care putem considera că algoritmul se ajunge la o valoare optimă(mai precis se observă evoluția global best).

**Concluzie**

Din cele notate anterior se poate observa că acest algoritm a dat rezultate mai slabe când a venit vorba de dimensiunea mai mare a problemei. Trebuie totuși notat că acest lucru se poate datora parametrilor dați pentru învățarea socială și învățarea cognitivă.