UNIVERSITATEA TRANSILVANIA

FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

Automate, calculabilitate si complexitate

Grey Wolf Optimizer (GWO)

Nume student: **CHIȚU RALUCA-OANA**

Program licență: **Informatică ID, anul III, an univ. 2023-2024**

Grupa: **10LD511**

CUPRINS

Capitolul 1. Descrierea algoritmului. Evaluarea avantajelor și dezavantajelor

1.1 Introducere

Gray wolf optimization este una dintre cele mai recente metode de optimizare bio-inspirate. Aceasta imită procedura de vânătoare a unei haite de lupi gri**.** Totodată, este un algoritm de tip metaeuristic care face parte din clasa celor bazate pe roiuri (*swarm technique)* sau pe populații (*population-based).*

Acest algoritm s-a inspirat din comportamentul lupului cenușiu, care vânează prada mare în haite și se bazează pe cooperarea între lupi. Există două aspecte interesante ale acestui comportament:

- ierarhia socială
- mecanismul de vânătoare

Lupul cenușiu este un animal foarte social, ceea ce face ca acesta să aibă o ierarhie socială complexă. Acest sistem ierarhic, în care lupii sunt clasificați în funcție de forță și putere, se numește „ierarhie de dominanță”. Prin urmare, există alfa, beta, delta și omega.

Masculul și femelele alfa se află în vârful ierarhiei și conduc haita. Toți membrii haitei au ordonat în cadrul unui anumit rang. Sistemul ierarhic al lupului nu se referă doar la dominație și agresiune; acesta oferă, de asemenea, asistență membrilor vulnerabili ai haitei care nu pot vâna singuri.

După aceea este lupul beta care sprijină deciziile lupului alfa și ajută la menținerea disciplinei în cadrul haitei.De asemenea, este cel mai bun candidat pentru poziția de alpha.

Lupul delta se află sub lupul beta ca rang. Ei sunt adesea puternici, dar nu au abilități de conducere sau încredere în ei înșiși pentru a-și asuma responsabilități de conducere. Ei rebuie să se supună alfa și beta, dar îl domină pe omega. Există diferite categorii de lupi delta, cum ar fi Cercetașii, Santinelele, Bătrânii, Vânătorii, Îngrijitorii etc. De obicei, ei iau poziția de detectiv, de observator, de paznic, de hoinar, de prădător și curator.

În cele din urmă, lupul omega nu are nicio putere, iar ceilalți lupi îl vor alunga rapid. Lupul omega este, de asemenea, responsabil de supravegherea lupilor mai tineri. Sunt considerați țapul ispășitor în haită, sunt cei mai puțin importanți indivizi din haită și nu au voie să mănânce decât la sfârșit.

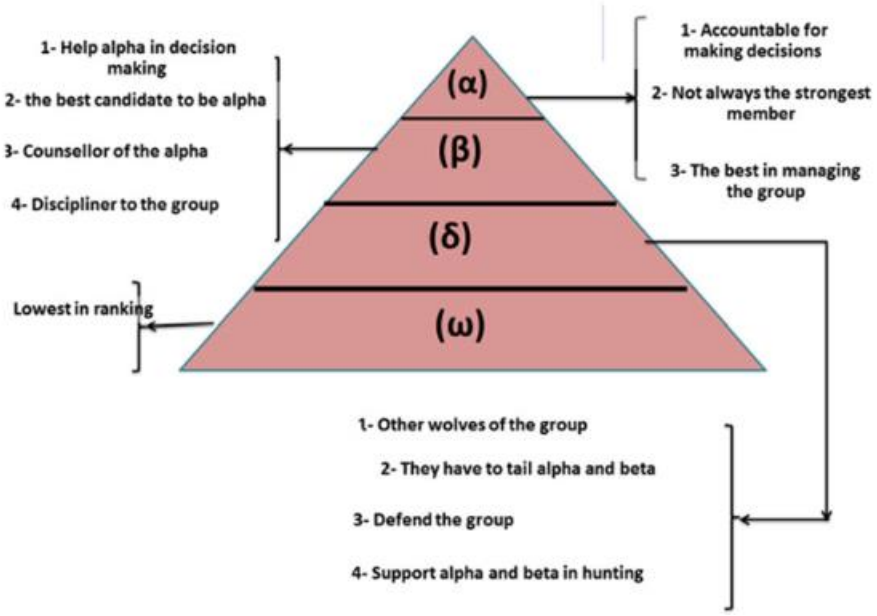

Fig.1 Ierarhia socială într-o haită de lupi cenușii

(Metaheuristic Optimization: Nature-Inspired Algorithms Swarm and Computational Intelligence, Theory and Applications Modestus O. Okwu • Lagouge K. Tartibu – page 44)

Pe lângă ierarhia socială, lupii cenușii au un mod foarte specific de a vâna, cu o strategie unică. Ei vânează în haite și colaborează în grupuri pentru a separa prada din turmă, apoi unul sau doi lupi urmăresc și atacă prada în timp ce ceilalți îi alungă pe cei rămași.

Principalele faze ale vânătorii de lupi cenușii sunt:

- urmărirea și apropierea de pradă.
- încercuirea și hărțuirea prăzii până când aceasta nu se mai mișcă.
- atacul spre pradă.

1.2. Principii de funcționare

În ceea ce privește modul de funcționare a algoritmului GWO (*Grey Wolf Optimizer)* putem identifica următoarele principii:

- **<u>Principiul ierarhiei sociale</u>**.

  Algoritmul GWO este inspirat de structura socială a haitelor de lupi gri. În aceste haite:

  - Lupii alfa (liderii) iau principalele decizii privind vânătoarea și alte activități.

- Lupii beta (al doilea la comandă) îi asistă pe lupii alfa și intervin dacă este necesar.
- Lupii delta (al treilea la comandă) îi sprijină pe lupii alfa și beta.
- Lupii omega (urmăritori) sunt membrii cu cel mai mic rang și îi urmează pe ceilalți.

Această ierarhie este utilizată pentru a ghida procesul de căutare, lupii alfa, beta și delta reprezentând cele mai bune soluții găsite până în acel moment.

- **Principiul vânătorii**

Algoritmul GWO modelează comportamentul cooperativ de vânătoare al lupilor gri. În timpul vânătorii, lupii alfa, beta și delta ghidează restul haitei. În mod similar, în cadrul algoritmului, pozițiile acestor trei cele mai bune soluții influențează actualizările tuturor celorlalte soluții candidate. Astfel, se asigură că căutarea este direcționată către zonele cele mai promițătoare.

- **Principiul explorării și exploatării**

Algoritmul GWO echilibrează între explorarea de noi domenii (explorare) și concentrarea asupra celor mai bine cunoscute domenii (exploatare). La începutul procesului de căutare, algoritmul explorează pe scară largă pentru a acoperi o suprafață mare a spațiului de căutare. Pe măsură ce căutarea avansează, acesta se concentrează treptat mai mult pe cele mai bune soluții găsite, rafinându-le pentru a găsi soluția optimă.

- **Principiul încercuirii prăzii.** Lupii gri își înconjoară prada înainte de a ataca. Algoritmul GWO imită acest lucru prin faptul că soluțiile candidate (lupii) se apropie de cele mai bune soluții (prada) identificate până în prezent. Acest lucru ajută la rafinarea zonei de căutare și la concentrarea asupra regiunilor promițătoare din spațiul soluțiilor.

- **Principiul atacului la pradă**

Atunci când lupii sunt gata să facă mișcarea finală, ei converg spre pradă. În algoritm, acest lucru înseamnă că, pe măsură ce se apropie de sfârșitul procesului de căutare, soluțiile candidate converg mai mult spre cele mai bune soluții, rafinându-le pentru a obține rezultate optime sau aproape optime.

1.3 Strategii de căutare

Referitor la strategiile de căutare, se pot sublinia următoarele aspecte:

- ✓ modul în care este realizat procesul de căutare depinde în primul rând de ierarhie, membrii alpha, beta și delta fiind cei care conduc acest proces. Aceste trei tipuri de lupi ghidează mișcarea tuturor celorlalți lupi.
- ✓ inițial, spațiul de căutare este relativ extins, însă pe măsură ce căutarea avansează lupii converg către pradă. Acest lucru se realizează prin reducerea treptată a zonei de căutare, permițând lupilor să își ajusteze pozițiile și să se concentreze pe cele mai bune zone găsite.
- ✓ practic, pozițiile lupilor sunt actualizate astfel încât aceștia să se apropie de pradă și să o înconjoare.

1.4 Model matematic

Pașii principali ai algoritmului GWO, bazat pe modul de vânătoare al lupilor cenușii sunt: **căutare, urmărire, înconjurarea și atacare**.

Astfel, modelul matematic al acestui algoritm presupune găsirea celor mai potrivite trei soluții:

- ➢ prima soluție: este echivalentă cu lupul Alpha
- ➢ cea de-a doua soluție: reprezintă lupul Beta
- ➢ cea de-a treia soluție: reflexctă lupul Delta.

Restul soluțiile sunt, în fapt, lupii care aparțin categoriei Omega.

Totodată, **soluția optimă** în cazul acestui algoritm este **prada** (cea mai bună pradă găsită în urma căutării), iar **lupii** sunt **soluțiile candidat** (variante potențiale în spațiul de căutare, pozițiile lor fiind actualizate pentru a se apropia de pradă). Practic, algoritmul încearcă să găsească cea mai bună pradă și poziția exactă a acesteia (sau o poziție cât mai apropiată).

a. <u>modelul matematic pentru înconjurarea prăzii:</u>

(1)

$$\vec{D} = \left| \vec{C}.\overrightarrow{X_p} - \vec{X}(t) \right|$$

$$\overrightarrow{C} = 2.\overrightarrow{r_2}$$

Unde:

vector D = distanța dintre lup și prada sa
vector C = vector de coeficient
vector Xp = poziția prăzii
vector X = poziția lupului cenușiu
t = iterația curentă
vector r_2 = vector aleator cu valori între [0,1]

(2)

$$\vec{X}(t+1) = \left| \overrightarrow{X_p}(t) - \vec{A}.\vec{D} \right|$$

$$\overrightarrow{A} = 2.\overrightarrow{a}.\overrightarrow{r_1} - \overrightarrow{a}$$

Unde:

vector D = distanța dintre lup și prada sa
vector A = vector de coeficient
vector Xp = poziția prăzii
vector X = poziția lupului cenușiu
t = iterația curentă
vector r_1 = vector aleator cu valori între [0,1]
vector a = componentă care descrește de la 2 la 0

Pe baza acestor ecuații, poziția lupului cenușiu este actualizată în funcție de poziția prăzii. Vectorii de coeficient sunt utilizați pentru evitarea blocării algoritmului în **optime locale** (**optim local** = prădă care este găsită, dar focusarea pe aceasta poate duce la negăsirea unei prăzi mai bune; **optim global** = prada cea mai bună găsită în zona de căutare). Astfel, vector A controlează compromisul dintre explorare și exploatare, în timp ce vectorul C permite un anumit grad de alegere aleatoare. Cu alte cuvinte, descreșterea vectorului A spre 0, înseamnă trecerea de la eplorare (căutare generală) la eploatare (căutarea într-o arie țintă).

b. Modelul matematic pentru vânătoare:

În general, procesul de vânătoare este ghidat de Alpha. Acest proces presupune că Alpha, Beta și Delta au cunoștințe mai bune despre locația prăzii (sau a soluției optime). Ceilalți lupi își vor actualiza pozițiile pe baza poziției lui Alpha, Beta și Delta.

(3)

$$\vec{D_\alpha} = \left|\vec{C_1}.\vec{X_\alpha} - \vec{X}(t)\right|$$

(4)

$$\vec{D_\beta} = \left|\vec{C_2}.\vec{X_\beta} - \vec{X}(t)\right|$$

(5)

$$\vec{D_\delta} = \left|\vec{C_3}.\vec{X_\delta} - \vec{X}(t)\right|$$

$$\vec{X_1} = \left|\vec{X_\alpha} - \vec{A_1}.\vec{D_\alpha}\right|$$

$$\vec{X_2} = \left|\vec{X_\beta} - \vec{A_2}.\vec{D_\beta}\right|$$

$$\vec{X_3} = \left|\vec{X_\delta} - \vec{A_3}.\vec{D_\delta}\right|$$

Poziția lupilor cenușii va fi actualizată cu ajutorul următoarei ecuații:

$$\vec{X}(t+1) = \frac{\vec{X_1} + \vec{X_2} + \vec{X_3}}{3}$$

1.5 Avantaje și dezavantaje

Avantaje

- **Simplicitate și ușurință în implementare**: algoritmul GWO este relativ simplu de implementat și nu necesită parametri complecși, ceea ce îl face accesibil și ușor de utilizat pentru cercetători și ingineri.
- **Convergență rapidă** : GWO are o capacitate bună de a converge rapid către soluții optime, datorită mecanismului său de actualizare a pozițiilor, care este inspirat de comportamentul natural al lupilor gri în timpul vânătorii.
- **Eficiență în probleme de optimizare multimodală**:algoritmul poate gestiona eficient problemele cu multiple minime și maxime locale, evitând să fie blocat în puncte locale datorită mecanismului său de explorare și exploatare echilibrat.
- **Flexibilitate:**GWO poate fi aplicat la o gamă largă de probleme de optimizare, inclusiv optimizarea funcțiilor continue, discrete, și combinatorii.
- **Evitarea stagnării**:prin utilizarea unui echilibru între explorare și exploatare, GWO are capacitatea de a evita stagnarea într-un punct local, ceea ce îl face robust în căutarea soluțiilor globale.

Dezavantaje

- **Scalabilitate limitată**: performanța GWO poate scădea semnificativ odată cu creșterea dimensiunii problemei sau a spațiului de căutare, ceea ce îl face mai puțin eficient pentru probleme de optimizare foarte mari sau complexe.
- **Dependința de parametrii inițiali**:eficiența algoritmului poate fi influențată de setările inițiale ale parametrilor, iar alegerea incorectă a acestora poate duce la performanțe suboptime.
- **Explorare/exploatare inadecvată**: în unele cazuri, GWO poate suferi de un echilibru inadecvat între explorare și exploatare, ceea ce poate duce la convergență prematură sau la o explorare insuficientă a spațiului de căutare.
- **Necesitatea adaptării pentru probleme specifice**: în anumite probleme specifice, GWO poate necesita modificări sau adaptări pentru a funcționa optim, ceea ce poate implica un efort suplimentar din partea utilizatorilor.
- **Convergența către soluții locale**:deși GWO are mecanisme pentru a evita convergența prematură, în practică, există încă riscul ca algoritmul să se blocheze într-un minim local, mai ales în problemele de optimizare foarte complexe.

Capitolul 2. Investigarea ariei de aplicabilitate

Principalele aplicații ale GWO aparțin domeniilor optimizării globale, ingineriei energetice, bioinformaticii, aplicațiilor de mediu, învățării automate, rețelelor și procesării imaginilor etc.

Optimizare în Design Engineering

- **Proiectare structurală**: GWO este utilizat pentru minimizarea greutății elementelor structurale, menținînd în același timp rezistența.
- **Sisteme de control**: reglarea parametrilor regulatorilor, cum ar fi regulatorii PID, pentru a obține performanțe optime în ceea ce privește stabilitatea și timpul de răspuns.

Sisteme electrice și de putere

- **Dispecerizarea economică a sarcinii (ELD)**: minimizarea costului de generare a energiei electrice, respectând în același timp cererea și constrângerile operaționale.
- **Fluxul optim de putere (OPF):** găsirea celor mai bune niveluri de funcționare pentru generarea de energie, minimizarea costului combustibilului și îmbunătățirea fiabilității sistemului.

Prelucrarea imaginilor și viziunea computerizată

- **Segmentarea imaginilor**: segmentarea automată a imaginilor în părți semnificative.
- **Selectarea caracteristicilor**: selectarea celor mai relevante caracteristici pentru sarcinile de clasificare a imaginilor.

Învățare automată și extragerea datelor

- **Selectarea caracteristicilor**: selectarea celor mai relevante caracteristici dintr-un set mare de date pentru a îmbunătăți performanța modelelor de învățare automată.
- **Reglarea hiperparametrilor**: optimizarea hiperparametrilor algoritmilor de învățare automată pentru a îmbunătăți precizia și eficiența.

Robotică și planificarea traiectoriei

- **Planificarea traiectoriei**: găsirea celei mai scurte sau mai eficiente căi pe care un robot trebuie să o parcurgă de la un punct de plecare la o destinație.
- **Planificarea mișcării**: optimizarea mișcărilor roboților pentru a evita obstacolele și a minimiza consumul de energie

Sisteme de energie regenerabilă

- **Optimizarea amenajării parcului eolian**: aranjarea turbinelor eoliene într-o fermă pentru a maximiza producția de energie și a minimiza pierderile de energie.
- **Amplasarea panourilor solare**: amplasarea optimă a panourilor solare pentru a maximiza captarea energiei.

Economie și finanțe

- **Optimizarea portofoliului**:selectarea celei mai bune combinații de active financiare pentru a maximiza rentabilitatea și a minimiza riscul.
- **Managementul lanțului de aprovizionare**: optimizarea rețelelor logistice și a lanțului de aprovizionare pentru reducerea costurilor și eficiență.

Sisteme de comunicații

- **Proiectarea rețelelor**: optimizarea proiectării rețelelor de comunicații pentru eficiență maximă și interferențe minime.
- **Alocarea resurselor**:distribuirea resurselor, cum ar fi lățimea de bandă și puterea în rețelele fără fir, pentru a îmbunătăți performanța.

Capitolul 3. Exemplificarea algoritmului pentru un caz concret de problema

Problema: maximizare a unei funcții obiectiv de două variabile. Funcția propusă:

$$f(x) = x^2 + y^2 \quad \textit{unde } x, y \in [-5, 5]$$

În contextul rezolvării acestei probleme, este nevoie de o aborbare din perspectiva algorimtului GWO. Astfel spus, deoarece acest algoritm este unul de minimizare, este necesară adaptarea funcției astfel încât să se poată implementa acest algoritm.

Pe baza materialului **METAHEURISTICS FROM DESIGN TO IMPLEMENTATION, El-Ghazali Talbi, Editura Wiley, pagina 3,** care spune că a maximiza o funcție este echivalent cu a minimiza negativul ei. (,,*Maximizing an objective function f is equivalent to minimizing −f*”).

Astfel, funcția devine:

$$f(x) = - x^2 - y^2 \quad \textit{unde } x, y \in [-5, 5]$$

Elemente:

- număr maxim de iterații: 2
- populație: 6
- coeficient r_1 – se va alege o singură valoare pentru a concentra vânătoarea (menținerea unui echilibru între explorarea spațiului de căutare și exploatarea lui)

În continuare, are loc implementarea algoritmului în cazul primului lup, pentru restul calculele fiind realizate în fișierul Excel **GWO_data.**

Pasul 1. Alocare valori random pentru cele două variabile x și y

Număr lup	x	y
1	-2.54	0.87
2	0.54	4.09
3	4.10	3.44
4	1.55	3.72
5	-4.84	4.58
6	3.22	4.04

Tabel 1. Date inițiale

Pasul 2. Calcularea funcției de fitness

Număr lup	x	y	x^2	y^2	$f(x,y) = - x^2 - y^2$
1	-2.54	0.87	6.45	0.76	-7.21
2	0.54	4.09	0.29	16.73	-17.02
3	4.10	3.44	16.81	11.83	-28.64
4	1.55	3.72	2.40	13.84	-16.24
5	-4.84	4.58	23.43	20.98	-44.40
6	3.22	4.04	10.37	16.32	-26.69

Tabel 2. Calcul funcție fitness

Pasul 3. Stabilire Alpha, Beta, Delta – cele mai mici valori ale funcției fitness

Categorie	x	y	f(x,y) = - x^2 - y^2	Vectori
Alpha(α)	-4.84	4.58	-44.40	X_α
Beta (β)	4.10	3.44	-28.64	X_β
Delta (δ)	3.22	4.04	-26.69	X_δ

Tabel 3. Calcul funcție fitness

Pasul 4. Calcularea $\vec{a}$

$$\vec{a} = 2 - 2\left(\frac{\text{iter}}{\text{max iter}}\right)$$

iter – numărul iterației, care în acest moment este **1**

max iter – numărul maxim de iterații este **3**

$$\vec{a} = 2 - 2 \text{ x}(\frac{1}{3}) = \mathbf{1.33}$$

Pasul 5. Aleger valori pentru $\vec{r_1}, \vec{r_2}$

- **valori în intervalul [0, 1]**

	r_{11}	r_{12}
r_1	**0.67**	**0.53**
	r_{21}	r_{22}
$r_{2\alpha}$	**0.42**	**0.81**
$r_{2\beta}$	**0.56**	**0.44**
$r_{2\delta}$	**0.71**	**0.29**

Pasul 6. Calculare $\vec{A} = 2.\vec{a}.\vec{r_1} - \vec{a}$

- **calcul** $\vec{a}.\vec{r_1}$ - deoarece atât vectorul a, cât și r1 au valori unice calculul este același pentru toate valorile A_{alpha}, A_{β}, A_{δ}

  [1.33, 1.33] x [0.67, 0.53] = **[0.8933, 0.7067]**

- **calcul** $2.\vec{a}.\vec{r_1}$

  Identic ca la pasul anterior, vom avea aceleași valori pentru A_{alpha}, A_{β}, A_{δ}

  2 x [0.8933, 0.7067] = **[1.7867, 1.4133]**

- calcul final $\vec{A} = 2.\vec{a}.\vec{r_1} - \vec{a}$

  Identic, aceeiași valoare pentru A_{α}, A_{β}, A_{δ}

  [1.7867, 1.4133] – [1.3333, 1.3333] = **[0.4533, 0.0800]**

Pasul 7. Calculare $\vec{C} = 2.\vec{r_2}$

C_{α} = 2 x [0.42, 0.81] = **[0.84, 1.62]**

C_{β} = 2 x [0.56, 0.44] = **[1.12, 0.88]**

C_{δ} = 2 x [0.71, 0.29] = **[1.42, 0.58]**

Pasul 8. Calculare $\vec{D} = \left|\vec{C}.\vec{X_p} - \vec{X}(t)\right|$

- $\vec{X}(t)$ **– valorile x și y din tabelul inițial**

$D_{alpha} = |C_{\alpha} \times \mathbf{X_{\alpha}} - \vec{\mathbf{X}}_{(t)}|$ = [0.84, 1.62] x [-4.84, 4.58] – [-2.54, 0.87] = **[-1.53, 6.55]**

$D_{\beta} = |C_{\beta} \times \mathbf{X_{\beta}} - \vec{\mathbf{X}}_{(t)}|$ = [1.12, 0.88] x [4.10, 3.44] – [-2.54, 0.87] = **[7.13, 2.16]**

$D_{\delta} = |C_{\delta} \times \mathbf{X_{\delta}} - \vec{\mathbf{X}}_{(t)}|$ = [1.42, 0.58] x [3.22, 4.04] – [-2.54, 0.87] = **[7.11, 1.47]**

Pasul 9. Calcularea $\vec{X}(t+1) = \left|\vec{X_p}(t) - \vec{A}.\vec{D}\right|$

$X_1 = X_\alpha(t) - A_\alpha \times D_\alpha = |[-4.84, 4.58] – [0.45, 0.08] \times [-1,53, 6.55]| =$ **[-4,15, 4.06]**

$X_2 = X_\beta(t) - A_\beta \times D_\beta = |[4.10, 3.44] – [0.45, 0.08] \times [7,13, -2,16]| =$ **[0.87, 3.27]**

$X_3 = X_\delta(t) - A_\delta \times D_\delta = |[3.22, 4.04] – [0.45, 0.08] \times [7.11, 1,47]| =$ **[-0.043, 3.92]**

Pasul 10. Calcularea $\vec{X}(t+1) = \frac{\vec{X_1} + \vec{X_2} + \vec{X_3}}{3}$

X(t+1) = (X1 + X2 + X3) /3 = ([-4.15, 4.06] + [0.87, 3.27] + [-0.043, 3.92])/3 = ([3.32, 11.25])/3 = **[-1.10, 3.75]**

Pasul 11. Calcularea noii valori a funcției obiectiv (fitness)

Noile valori ale celor două variabile x și y sunt, în fapt, valorile aferente vectorului X(t+1). Deci, valorile vor fi: x = -1.10 și y = 3.75

Astfel, funcția de fitness va fi: $f(x,y) = - X^2 – y^2 = (-0,10)^2 – (3.75)^2 =$ **-15.25**

Pasul 11. Comparăm noua valoare de fitness cu cea inițială

Finițial(x,y) = -7.21

Fnou(x,y) = -15.25

Rezultă că noua valoare de fitness este mai bună și astfel x = -1.10 și y = 3.75 vor fi valorile pentru iterația a II-a aferente primului lup.

PAȘII 1-11 SE VOR RELUA PENTRU TOȚI LUPII. CEEA CE SE SCHIMBĂ VOR FI VARIABILELE CARE DEPIND DE X ȘI Y. TOTODATĂ, PENTRU ITERAȚIILE 2 ȘI 3 SE VA SCHIMBA ȘI VECTORUL a, ACESTA DEVENIND 2/3 ȘI RESPECTIV 3/3. TOT ÎN URMĂTOARELE ITERAȚII SE VOR SCHIMBA ȘI VECTORII r. DATELE SUNT

Iterația 2 – bazată pe valori obținute după prima iterare

Număr lup	x	y	x^2	y^2	$f(x,y) = - x^2 - y^2$
1	-1.10	3.75	1.20	14.06	-15.27
2	0.54	4.09	0.29	16.73	-17.02
3	4.10	3.44	16.81	11.83	-28.64
4	1.55	3.72	2.40	13.84	-16.24
5	-4.84	4.58	23.43	20.98	-44.40
6	3.22	4.04	10.37	16.32	-26.69

Iterația 3 – bazată pe valori obținute după a doua iterare

Număr lup	x	y	x^2	y^2	$f(x,y) = - x^2 - y^2$
1	1.85	3.90	3.41	15.20	-18.62
2	0.54	4.09	0.29	16.73	-17.02
3	4.10	3.44	16.81	11.83	-28.64
4	1.55	3.72	2.40	13.84	-16.24
5	-4.84	4.58	23.43	20.98	-44.40
6	3.22	4.04	10.37	16.32	-26.69

Capitolul 4. Concluzii și recomandări

În urma analizei datelor obținute, prezentate pe larg în fișierul GWO_data, se pot concluziona următoarele:

- Alegerea unei singure valori per iterație pentru vectorul r_1 nu este o alegere neapărat bună, deoarece rezultate funcției de fitness nu se îmbunătățesc de la o iterație la alta
- Vectorul A rămâne constant pentru principalii lupi ai haitei, datorită vectorului r_1, de care depinde
- Doar pentru primul lup există îmbunătățiri de la o iterație la alta

Pentru o analiză mai în detaliu a algoritmului prezentat, se recomandă testarea lui într-o etapă ulterioară, având valori diferite pentru vectorului r_1 aferente lupilor alpha, beta, delta, la fiecare iterație.

Capitolul 5. Webografie și bibliografie

https://www.baeldung.com/cs/grey-wolf-optimization

https://www.geeksforgeeks.org/grey-wolf-optimization-introduction/

https://www.youtube.com/watch?v=CQquzq24BPc&t=335s

Metaheuristic Optimization: Nature-Inspired Algorithms Swarm and Computational Intelligence, Theory and Applications - Modestus O. Okwu • Lagouge K. Tartibu – Springer, 2021