**Punctul 1 – funcție nouă de fitness**

double Individual::Evaluate()

{

double maximSimulatedStress = SimulateAndGetMaximStress();

double fitness = 0.0;

double stressPenalty = 0.0;

double removalPenalty = 0.0;

double stressBonus = 0.0;

if (maximSimulatedStress > m\_maximStress || maximSimulatedStress < EPSILON\_STRESS)

{

stressPenalty = std::numeric\_limits<double>::max();

}

else

{

stressPenalty = m\_maximStress - maximSimulatedStress;

}

removalPenalty = pow(GetNumberOfRemovedElements(), 2);

stressBonus = 1.0 / maximSimulatedStress;

fitness = stressPenalty + removalPenalty + stressBonus;

return fitness;

}

Fitness = StressPenalty + RemovalPenalty + StressBonus, unde:

StressPenalty = penalizarea pentru depășirea stresului maxim admis.

Dacă maximSimulatedStress > m\_maximStress sau maximSimulatedStress < EPSILON\_STRESS:

StressPenalty = -∞, semnalează o soluție invalidă

altfel:

StressPenalty = m\_maximStress – maximSimulatedStress

RemovalPenalty = penalizarea pentru eliminarea elementelor.

RemovalPenalty = (GetNumberOfRemovedElements())^2

StressBonus = bonusul pentru minimizarea stresului simulat.

StressBonus = 1 / maximSimulatedStress

**De ce?**

* **Penalizare pentru stres:** Funcția de fitness originală verifica doar dacă stresul maxim simulat depășește limita maximă admisă. În această funcție de fitness, penalizarea pentru depășirea stresului maxim admis este proporțională cu diferența dintre stresul maxim admis și stresul maxim simulat. Acest lucru oferă o penalizare mai bună. Funcția de fitness alternativă poate ghida căutarea către soluții cu niveluri de stres mai apropiate de limitele admisibile.
* **Introducerea penalizărilor pentru eliminarea elementelor și bonusul pentru stres**: Funcția de fitness alternativă introduce atât o penalizare pentru eliminarea elementelor, cât și un bonus pentru stres. Penalizarea pentru eliminarea elementelor descurajează eliminarea excesivă a elementelor, promovând design-uri mai compacte și structural stabile. Pe de altă parte, bonusul pentru stres favorizează soluțiile cu niveluri de stres mai scăzute.

**Punctul 2 – metode noi pentru pașii de selecție și încrucișare**

Tournament selection

* selectează cel mai bun individ dintr-un subgrup aleatoriu, ceea ce face ca indivizii cu fitness extrem de mare sau extrem de mic să nu aibă o influență disproproționată asupra selecției. Acest lucru ajută la menținerea diversității în populație și poate preveni stagnarea într-un minim local.
* dimensiunea turneului poate fi ajustată. Prin utilizarea unor turnee cu candidati mai puțini se promovează diversitatea genetică și se evită convergența prematură către o soluție suboptimală. În schimb, turnee mai mari cresc presiunea selectivă, permițând selecția părinților cu fitness mai ridicat.
* selecția prin turneu poate fi mai rezistentă la fluctuațiile temporare ale fitnessului indivizilor din populație. Deoarece se alege cel mai bun individ dintr-un subgrup, este posibil ca un individ cu un fitness temporar scăzut să fie selectat într-un turneu, dacă ceilalți indivizi din subgrup au un fitness și mai mic.

Metodele de crossover alese: Two Point Crossover și Uniform Crossover With Crossover Mask.

Metodele initiale:

EPOCH,VALUE

1,4.482e+08

2,4.4849e+08

3,4.48365e+08

4,4.4895e+08

5,4.48167e+08

Noua functie de fitness:

EPOCH,VALUE

1,4.4808e+08

2,4.4858e+08

3,4.48482e+08

4,4.48079e+08

5,4.47892e+08

Metoda Tournament selection cu crossover initial nemodificat

EPOCH,VALUE

1,4.49359e+08

2,4.47923e+08

3,4.4775e+08

4,4.47961e+08

5,4.48967e+08

Two Point Crossover

EPOCH,VALUE

1,4.49088e+08

2,4.48155e+08

3,4.481e+08

4,4.48477e+08

5,4.48104e+08

Uniform Crossover With Crossover Mask

EPOCH,VALUE

1,4.4893e+08

2,4.49014e+08

3,4.48795e+08

4,4.49022e+08

5,4.48855e+08