Tema 1

Maciuc Mihai 2B2

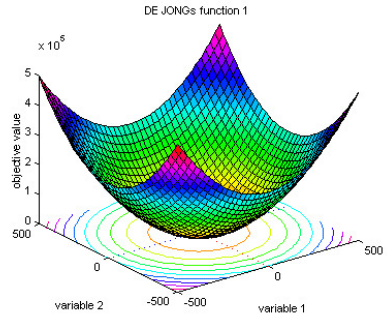
26 octombrie 2024

1 Enunt

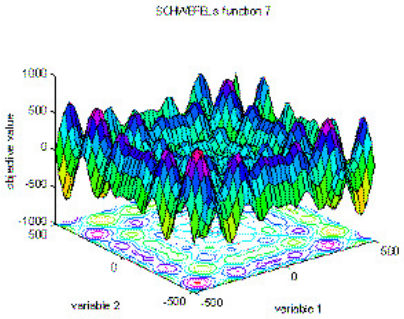
Tema H1: Să se determine minimul pentru următoarele funcții: De Jong 1, Schwefel's, Rastrigin's, Michalewicz's implementand algoritmii Hill Climbing (variantele first improvement, best improvement și worst improvement) și Simulated Annealing (hibridizant una dintre variantele de Hill Climbing, la alegere). Analizați versiunile 5, 10 și 30-dimensionale ale funcțiilor (poate încercați și versiunea 100-dimensională, ca o margine superioară). Folosiți o precizie de reprezentare de minim 5 zecimale după 0.

1.1 Motivatie

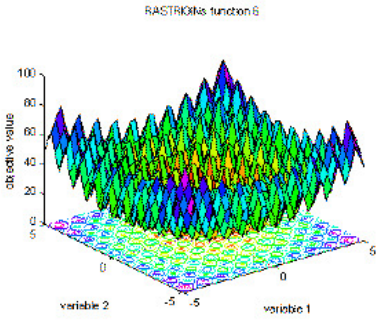
Acest proiect a fost făcut pentru a întelege mai bine cele două euristici învațate: Hill Climbing și Simulated Annealing. Pe baza celor două metode am putut face observații despre comportamentul, rezultatul și complexitatea implementărilor pentru seturi mici de date, cât și pentru seturi mari. Pentru testarea algoritmului s-au folosit cele patru funcții menționate în enunt: De Jong, Schwefel, Rastrigin si Michalewicz.



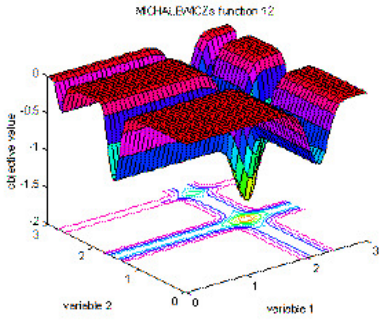
(a) Funcția De Jong 1



(b) Funcția Schwefel



(c) Funcția Rastrigin



(d) Funcția Michalewicz

Figura 1: Funcțiile De Jong 1, Schwefel, Rastrigin și Michalewicz

2 Metode

Metodele aplicate în implementare sunt: Hill Climbing (în cele trei forme: first improvment, best improvment și worst improvment) și Simulated Annealing. Variantele de Hill Climbing diferă doar prin schimbarea functiei "improve", functie folosită și pentru a implementa Simulated Annealing hibridizant. Am folosit 2 mapari pentru a schimba usor funcția pentru care aplicam metodele. O mapare retine domeniul de definitie al functiei și a doua numele si functia potrivita acesteia. Solutiile sunt procesate sub forma de bitstrings si decodate ca numere reale pentru a compara soluția mai bună. Inițializarea este asemanatoare în cele două euristici. Este generată o solutie random (fiind și cea mai buna la momentul 0), după care, în funcție de forma de Hill Climbing aleasă, este cautat vecinul care se apropie de punctul de minim. Daca folosim best-improvment vom selecta cel mai bun vecin, mai bun decat bitstringul initial, daca este worst improvment selectam cel mai rau vecin dintre vecinii ce au o valoare mai buna decat bitstringul initial iar in final, pentru cazul first-improvment, selectam primul vecin mai bun decat bitstringul intial. Condiția de oprire pentru Hill Climbing este dacă numarul de iteratii (initializat înainte) este depasit. Pentru Simulated Annealing metoda este asemanatoare cu o excepție. Folosim un indice de temperatură care ne permite sa alegem puncte mai rele din punct de vedere al castigului de informatie, pentru a evita eventuale blocaje pe minime locale. Indicele de temperatura scade, făcând mai puțin probabilă acceptarea solutilor proaste odată cu progresul spre minim. Conditia de oprire este și ea adaptata in functie de indicele de temperatura. Parametrii celor doua functii sunt "nume" un string in care ii dam numele functiei pe baza careia aplicam operatiile si "dimensiune" in care oferim o valoare numerica, numarul dimensiunilor pentru valorile functiei.

3 Experiment

Structura algoritmului este urmatoarea: Avem functiile pe care efectuam operatiile separate, 2 mapari in care facem legatura (nume-functie) si (nume-intervalul functiei), o functie cu ajutorul careia decodam un string din binar in valoare reala, o functie ce genereaza un string de lungime determinata, o formula pentru a determina lungimea pe care ar trebui sa o aiba un nr din intervalul [a,b]. In final avem functiile de HillClimbing si Simulated Annealing, iar partea de worst/best/first improvment am implementat-o separat. Am testat fiecare metodă pe funcțiile De Jong, Schwefel, Rastrigin și Michalewicz, utilizând dimensiuni de 5, 10 și 30 pentru a observa performanța în funcție de complexitatea dimensională pe 30 de iteratii. Rezultatele obținute au fost analizate și comparate pentru a evalua eficiența fiecărei metode în identificarea minimelor funcțiilor.

4 Rezultate

Funcție	**Timp (s)**	**Minim obținut**
De Jong 1	1479	0.00000
Schwefel	2742	-10881.25170
Rastrigin	1206	35.76472
Michalewicz	915	-24.98664

Tabela 1: Rezultatele obținute pentru dimensiune 30, 30 iteratii si metoda Hill Climbing best-improvment.

Funcție	**Timp (s)**	**Minim obținut**
De Jong 1	1769	0.00000
Schewefel	2632	-9686.79664
Rastrigin	1896	52.13907
Michalewicz	1554	-22.44330

Tabela 2: Rezultatele obținute pentru dimensiune 30 si metoda Simulated Annealing (best-improvment).

Nr dimensiuni	Timp (s)	Minim obținut	Metoda
30	1769	0.00000	SA
30	1479	0.00000	HC-best
10	347	0.00000	SA
10	82	0.00000	HC-best
5	12	0.00000	SA
5	29	0.00000	HC-best

Tabela 3: Rezultatele obținute pentru functia DeJong

Nr dimensiuni	Timp (s)	Minim obținut	Metoda
30	2632	-9686.79664	SA
30	2742	-10881.25170	HC-best
10	461	-3364.46012	SA
10	141	-3799.78262	HC-best
5	12	-1500.01955	SA
5	68	-1963.58395	HC-best

Tabela 4: Rezultatele obținute pentru functia Schewefel

Nr dimensiuni	Timp (s)	Minim obținut	Metoda
30	1896	52.12907	SA
30	1206	35.76472	HC-best
10	63	17.37494	SA
10	224	7.47168	HC-best
5	8	11.83675	SA
5	6	3.23079	HC-best

Tabela 5: Rezultatele obținute pentru functia Rastrigin

Nr dimensiuni	Timp (s)	Minim obținut	Metoda
30	1554	-22.44330	SA
30	915	-24.98664	HC-best
10	59	-7.32275	SA
10	179	-7.99397	HC-best
5	7	-3.06107	SA
5	6	-3.66941	HC-best

Tabela 6: Rezultatele obținute pentru functia Michalewicz

Functie	Media	Deviatia standard	Minim	Maxim
DeJong	0.00000	0.00000	0.00000	0.00000
Schewefel	-5199.31576	4047.2712	-10881.25170	-1500.01955
Rastrigin	21.30132	18.88034	3.23079	52.12907
Michalewicz	-11.57952	9.63205	-24.98664	-3.06107

Tabela 7: Rezultatele obținute total

4.1 Interpretare

Rezultatele obținute din experimentele efectuate sugerează că fiecare metodă are punctele sale bune si rele. Metoda Hill Climbing a arătat viteza mai bună în cazul funcției De Jong 1, Michalewicz si Rastrigin obținând minimul mai repede.

*(Mentiune) La momentul efectuarii testelor nr de iteratii pentru Simulated Annealing era ceva mai mare de 30 de iteratii. Astfel, chiar daca punctele de minim sunt relativ aceleasi, Simulated Annealing are timpi de executie mai buni in unele cazuri si se poate apropia mai mult de minimul global.

5 Bibliografie

1 https://youtu.be/3SiWtAnUROs?si=zVECPiPkLa61TKt7

2 https://www.geeksforgeeks.org/introduction-hill-climbing-artificial-intelligence/

3 https://medium.com/@tahsinsoyakk/hill-climbing-algorithm-a-comprehensive-guide-46e33f1ecc02

4 https://www.geeksforgeeks.org/simulated-annealing/

5 https://machinelearningmastery.com/simulated-annealing-from-scratch-in-python/

6 https://stackoverflow.com/questions/13445688/how-to-generate-a-random-number-in-c

7 https://stackoverflow.com/questions/73347075/c-get-a-substring-from-a-vector-of-chars

8 https://www.geeksforgeeks.org/measure-execution-time-function-cpp/

9 https://github.com/stratzilla/hill-climbers/blob/master/hillclimb.cpp

10 https://stackoverflow.com/questions/33631418/simulated-annealing-algorithm

11 https://stackoverflow.com/questions/11040203/map-of-functions-c

12 https://www.khanacademy.org/math/statistics-probability/summarizing-quantitative-data/variance-standard-deviation-sample/a/population-and-sample-standard-deviation-review

13 https://github.com/Kumar-laxmi/Algorithms/blob/main/C/Optimization/Hill$_c$*limbing.c*