Inteligenta Artificiala

Algoritm genetic modificat pentru numere reale

Apopei Roxana Manoleasa Tudor - grupa IIIA4

March 23, 2020

1 Algoritm

1.1 Reprezentare

Fiecare dimensiune a functiei data pentru optimizare va fi reprezentata printr-un vector de numere reale. O solutie candidat va lua forma unei matrici de astfel de vectori iar populatia este un vector de solutii candidat.

1.2 Implementare

La nivel de implementare, am folosit limbajul Python. Desi restrictiv din punctul de vedere al structurilor de date oferite, acesta ofera o modalitate simpla de manipulare a informatiei. In cadrul unui modul am creat o clasa ce pune la dispozitie metodele necesare pentru lucrul cu algoritmi genetici. La instantiere, i-am oferit utilizatorului posibilitatea de a rula algoritmul cu diferiti parametri:

- numarul de dimensiuni al functiei, in cazul in care aceasta permite
- dimensiunea populatiei
- probabilitatea de mutatie
- probabilitatea de incrucisare
- functia dorita, din cele patru puse la dispozitie

1.2.1 Functia fitness

Intrucat scopul proiectului este de a calcula minimul functiilor iar algoritmul genetic este construit pentru maximizare, functia fitness a fost modificata astfel:

- $fitness(x) = \frac{1}{f(x)}$

Functiile pe care am testat acuratetea algoritmului genetic sunt:

Dejong

$$f(x_1 \cdots x_n) = \sum_{i=1}^{n} x_i^2$$

$$-5.12 \leq x_i \leq 5.12$$

Rastrigin

$$f(x_1 \cdots x_n) = 10n + \sum_{i=1}^{n} (x_i^2 - 10cos(2\pi x_i))$$

Rosenbrock

$$-5.12 \leq x_i \leq 5.12$$

$$f(x_1 \cdots x_n) = \sum_{i=1}^{n-1} (100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2)$$

$$-2.048 \leq x_i \leq 2.048$$

Spring design

$$f(x_1, x_2, x_3) = (x_3 + 2)x_1 x_2^2$$

$$0.25 \leq x_1 \leq 1.3$$

$$0.05 \leq x_2 \leq 2.0$$

$$2 \leq x_3 \leq 15$$

$$1 - \frac{x_1^3 x_3}{71785 x_2^4} \leq 0$$

$$\frac{4x_1^2 - x_1 x_2}{12566(x_1 x_2^3 - x_2^4)} + \frac{1}{5108 x_2^2} - 1 \leq 0$$

$$1 - \frac{140.45 x_2}{x_1^2 x_3} \leq 0$$

$$\frac{x_1 + x_2}{1.5} - 1 \leq 0$$

1.2.2 Mutatie

Pentru mutatia unui cromozom ne-am folosit de patru dintre variantele prezentate in cadrul lucrarii [2]:

- **random mutation (Michalewicz, 1992)**
- **non-uniform mutation (Michalewicz, 1992)**
- **real number creep (Davis, 1991)**
- **Muëhlenbein's mutation (1993)**

1.2.3 Incrucisare

Similar, am utilizat sase din functiile de incrucisare prezentate in [2]:

- **flat crossover**
- **simple crossover**
- **linear crossover**
- **extended line crossover**
- **extended intermediate crossover**
- **Wright's heuristic crossover**

Intrucat dorim sa pastram constanta dimensiunea populatiei, am modificat cateva dintre functii astfel incat dupa aplicarea operatiei asupra a doi cromozomi, sa avem drept rezultat tot doua solutii candidat.

1.2.4 Selectie

Pentru selectie, ne-am documentat din [3] si am implementat trei strategii:

- **roata norocului**
- **elitism**
- **selectie bazata pe rang**

Dintre acestea, ultimele doua sunt variatii ale primei. Elitismul se va asigura ca la fiecare iteratie cei mai buni k-cromozomi nu se vor pierde. In cadrul selectiei bazate pe rang, probabilitatile cumulate vor fi calculate in functie de fitnessul fiecarui cromozom.

2 Rezultate experimentale

2.1 Rastrigin pe 5 dimensiuni - 15 executii ale algoritmului x 1000 generatii folosind selectia bazata pe roata norocului

Mutatie	**Incrucisare**	**Min**	**Avg**	**Max**	**Real**
random	simple	10.55927	17.37034	22.11691	0
	flat	8.42332	9.86524	11.48934	0
	linear				
	extended line	13.09767	15.84324	19.33007	0
	extended intermediate	16.15982	17.37746	18.19127	0
	Wright's heuristic	3.32732	7.91542	10.24979	0
non-uniform	simple	23.35495	28.97677	33.69945	0
	flat	16.16271	18.42929	21.92780	0
	linear				
	extended line	18.27103	25.18364	32.19834	0
	extended intermediate	21.82914	27.88930	32.79082	0
	Wright's heuristic	15.43263	22.85134	31.65727	0
real number creep	simple	0.99523	1.74559	4.11292	0
	flat	1.03258	1.46895	2.04551	0
	linear				
	extended line	0.99543	1.47971	2.38413	0
	extended intermediate	0.11299	3.04455	8.60465	0
	Wright's heuristic	0.04073	2.08103	5.37343	0
Muehlenbein	simple	15.42625	20.43995	28.88456	0
	flat	11.07226	24.42353	32.55621	0
	linear				
	extended line	13.73031	25.02244	36.12217	0
	extended intermediate	16.43196	24.42148	34.27309	0
	Wright's heuristic	21.66934	27.14133	32.61502	0

2.2 Rastrigin pe 10 dimensiuni - 15 executii ale algoritmului x 1000 generatii folosind selectia bazata pe elitism

Mutatie	Incrucisare	Min	Avg	Max	Real
random	simple	51.38033	56.81199	61.39079	0
	flat	49.60214	54.451294	60.87059	0
	linear				
	extended line	54.34754	59.83693	65.07675	0
	extended intermediate	41.18055	56.07034	62.64513	0
	Wright's heuristic	52.94609	55.96038	61.71274	0
non-uniform	simple	74.15846	82.04575	91.37403	0
	flat	74.58602	83.5955	93.50109	0
	linear				
	extended line	79.17203	83.81176	91.79295	0
	extended intermediate	78.61998	85.95045	92.09711	0
	Wright's heuristic	81.80072	88.95196	103.79711	0
real number creep	simple	20.60739	24.10652	30.40111	0
	flat	11.76325	16.88401	22.94909	0
	linear				
	extended line	20.0084	22.86519	26.72591	0
	extended intermediate	15.71603	22.08159	31.15445	0
	Wright's heuristic	9.97013	19.64347	30.85044	0
Muehlenbein	simple	7.53809	14.92296	21.35862	0
	flat	8.52547	13.95505	22.25569	0
	linear				
	extended line	9.71658	11.44463	16.43537	0
	extended intermediate	8.67639	17.0666	22.67373	0
	Wright's heuristic	15.00219	20.13901	28.15664	0

2.3 Rastrigin pe 30 dimensiuni - 15 executii ale algoritmului x 1000 generatii folosind selectia bazata pe rang

Mutatie	Incrucisare	Min	Avg	Max	Real
random	simple	350.34228	380.38617	393.67695	0
	flat	333.57219	355.84407	389.96079	0
	linear				
	extended line	344.94969	354.86405	365.85257	0
	extended intermediate	351.6823	369.82678	391.05048	0
	Wright's heuristic	343.25924	370.61708	388.96301	0
non-uniform	simple	379.42185	386.62069	397.72782	0
	flat	328.00394	360.62421	390.10821	0
	linear				
	extended line	328.9893	371.58867	399.13294	0
	extended intermediate	341.79539	371.41751	400.76697	0
	Wright's heuristic	384.65226	393.59163	414.43137	0
real number creep	simple	234.31917	266.85575	300.4297	0
	flat	217.45273	253.87486	280.19172	0
	linear				
	extended line	259.34481	266.95948	276.61062	0
	extended intermediate	223.93174	251.48965	284.0488	0
	Wright's heuristic	228.91286	262.45278	288.22641	0
Muehlenbein	simple	322.91589	351.29262	376.82596	0
	flat	263.03169	346.40802	384.74642	0
	linear				
	extended line	332.73746	359.57611	375.4838	0
	extended intermediate	297.28975	359.26298	388.43141	0
	Wright's heuristic	321.3142	341.84877	361.76155	0

2.4 Spring Design - 15 executii ale algoritmului x 1000 generatii folosind selectia bazata pe elitism

Mutatie	Incrucisare	Min	Avg	Max	Real
random	simple	0.00338	0.00363	0.00424	0.0025
	flat	0.0031	0.00387	0.00466	0.0025
	linear				
	extended line	0.00333	0.00443	0.00519	0.0025
	extended intermediate	0.00323	0.0039	0.00477	0.0025
	Wright's heuristic	0.00361	0.00402	0.00445	0.0025
non-uniform	simple	0.01289	0.01401	0.01528	0.0025
	flat	0.01272	0.014	0.01537	0.0025
	linear				
	extended line	0.01333	0.01425	0.01508	0.0025
	extended intermediate	0.00485	0.01117	0.0147	0.0025
	Wright's heuristic	0.00285	0.05117	0.0647	0.0015
real number creep	simple	0.0026	0.00303	0.00352	0.0025
	flat	0.00253	0.00277	0.00295	0.0025
	linear				
	extended line	0.0025	0.00328	0.00555	0.0025
	extended intermediate	0.00251	0.0029	0.0034	0.0025
	Wright's heuristic	0.00263	0.00293	0.00357	0.0025
Muehlenbein	simple	0.00564	0.0085	0.01154	0.0025
	flat	0.00463	0.00713	0.01185	0.0025
	linear				
	extended line	0.00598	0.00752	0.01039	0.0025
	extended intermediate	0.0058	0.00865	0.01154	0.0025
	Wright's heuristic	0.00452	0.00681	0.00832	0.0025

3 Influenta parametrilor

Valorile obtinute in tabelele anterioare au fost obtinute pentru o populatie de 50 de indivizi, cu o probabilitate de mutatie (pm) < 0.01 si probabilitatea de crossover (pc) < 0.25. Odata cu scaderea populatiei am observat o crestere a minimului (pentru functia Rastrigin pentru 5 dimensiuni si populatie de 25 am obtinut o medie de 7.75). Asadar, obtinem valori mai bune pentru o populatie mai mare. Cu toate acestea, pentru un pop_size mai mare de 1000, imbunatatirile pe care le putem obtine sunt minore raportat la timpul necesar de executie a algoritmului.
Pe de alta parte, cand am crescut valoarea pm la 0.5 si cu *pop_size* de 50, putem obtine un minim mai bun: la Rosenbrock o medie de 7.27.

4 Concluzie

In urma rezultatelor obtinute, am observat ca nu putem avea o combinatie de mutatie si incrucisare care sa produca un rezultat optim pentru fiecare functie. Cu o mutatie de tip *real number creep* obtinem valori foarte apropiate de minimul global pentru *Rastrigin* pe 5 dimensiuni. Pe de alta parte, pentru 10 dimensiuni, mutatia *Muehlenbein* produce rezultate mai bune. De asemenea, cresterea numarului de dimensiuni afecteaza drastic optimul obtinut intrucat, se faciliteaza mai multe posibilitati de randomificare, incrucisare, mutatie si selectie. Totodata, selectia bazata pe elitism produce valori mult mai bune pentru functii de o dimensionalitate redusa, esuand la generalizarea pe functii mari. Am observat ca *flat crossover* este o alegere inspirata pentru majoritatea mutatiilor intrucat produce la fiecare iteratie un numar real nou din intervalul corespunzator cromozomilor parinti. Astfel, este simulat un comportament specific mutatiei, in care sunt explorate mai multe solutii din spatiul problemei si, in acelasi timp, datorita selectiei, intervalul pentru fiecare numar este filtrat. Algoritmul genetic obtine rezultate care converg rapid spre valoarea minima in cazul *Spring Design*. Argumentul in vederea sustinerii acestei idei este faptul ca functia nu are foarte multe optime locale.

References

[1] Tapabrata Ray, K. M. Liew, *Society and civilization: An optimization algorithm based on the simulation of social behavior*

[2] F. Herrera, M. Lozano, J.L. Verdegay, *Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis*

[3] Prof. Dr. Henri Luchian, Prep. Drd. Mihaela Breabăn , *ALGORITMI GENETICI*