Dokumentacja do

WSI - ćwiczenie 2

Algorytmy ewolucyjne i genetyczne

Wykonano przez Marfenko Mykhailo

Struktura programu Program jest podzielony na 2 pliki:

genetic_algorithm.py – Ten plik zawiera główną funkję genetic_algorithm() która jest wyłowana z main.

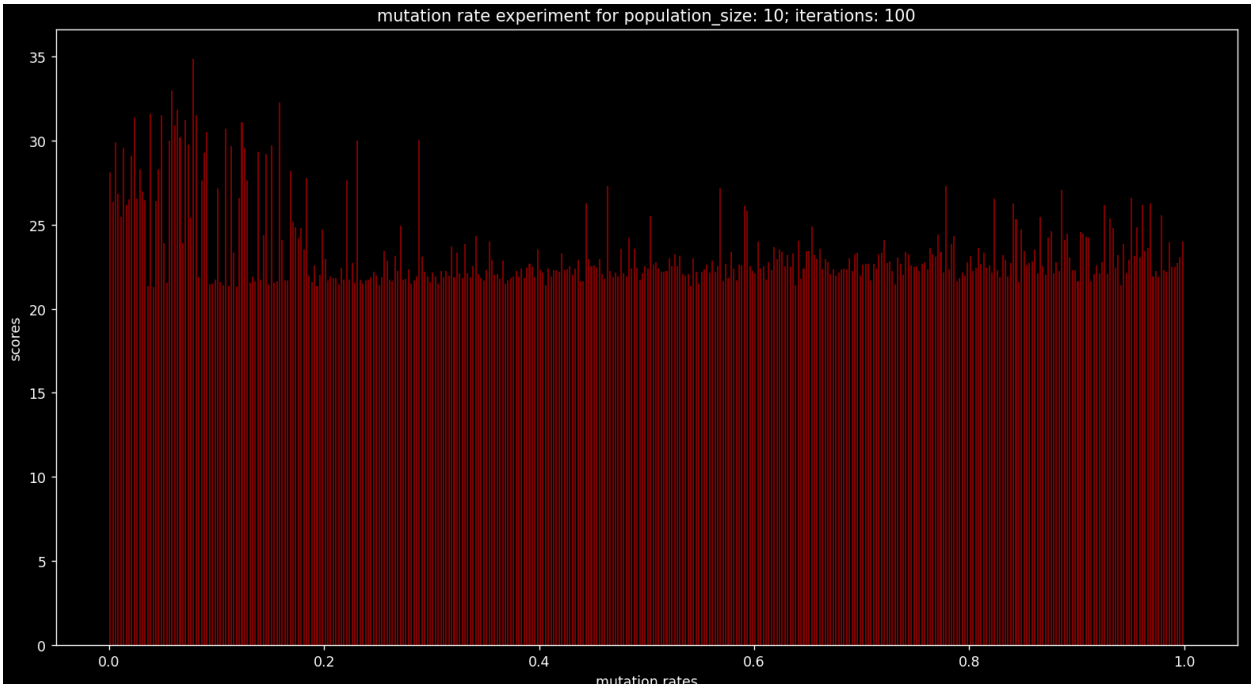
experiments.py – tutaj znajdują się 3 funckje: mutation_rate_experiment(); population_size_experiment() oraz iterations_experiment() które badają wpływ odpowiedniego hiperparametru: mutation_rate; population_size; iterations.

Rozpatrzymy działanie algorytmu dla iterations = 500; population_size = 500; mutation_rate = 0.25:

*wszystkie eksperymenty będą mieli punkt początkowy rodziców w przydziale od (-1;-1) do (1; 1)

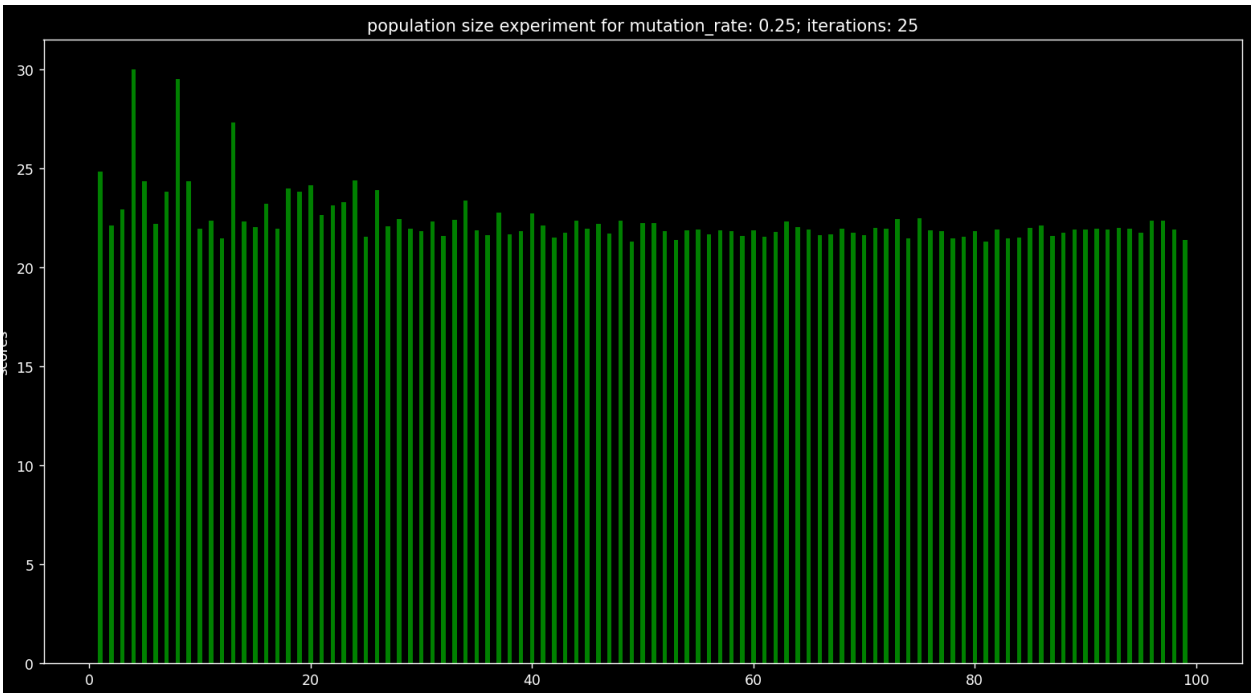
```
Iteration: 0   with new best score: 27.018439863011913   with coords: (0.9912867130824008, 0.6722914799301964)
Iteration: 0   with new best score: 26.88783024284716   with coords: (0.7510658077921726, 0.9021920194230075)
Iteration: 0   with new best score: 25.783096536913416   with coords: (0.9030013205213687, 0.8411209290536155)
Iteration: 0   with new best score: 24.371513660053004   with coords: (0.8773150803993859, 0.9479169759928034)
Iteration: 1   with new best score: 23.49606749786951   with coords: (1.0368849764054398, 0.9284379723397213)
Iteration: 1   with new best score: 23.011490212496703   with coords: (1.074331976038326, 0.9975648593360181)
Iteration: 3   with new best score: 22.479060199922124   with coords: (1.0384813809729594, 1.0140508577842187)
Iteration: 9   with new best score: 21.956316847163922   with coords: (0.9913590268405337, 0.9739187933301197)
Iteration: 10   with new best score: 21.925431195486812   with coords: (0.9904166739667982, 1.020603733728776)
Iteration: 11   with new best score: 21.81394162414876   with coords: (0.9855578595636616, 1.0119214687214053)
Iteration: 19   with new best score: 21.71723950339098   with coords: (0.9906975112297598, 0.986410146561525 7)
Iteration: 20   with new best score: 21.3292407269414   with coords: (0.9967176132357436, 0.9998674740690914)
Iteration: 126   with new best score: 21.321837682002382   with coords: (1.0000078478879195, 0.9972314272289753)
Iteration: 148   with new best score: 21.30125724338008   with coords: (0.9996858418696264, 0.9985056125170472)
Iteration: 350   with new best score: 21.28268479163135   with coords: (0.99974675123701, 0.9993472126426927)
Best score is: 21.28268479163135  with coordinates: (0.99974675123701, 0.9993472126426927)
21.26435864903798
```

Teraz zbadamy wpływ mutation_rate która przyjmuje wartości od 0.001 do 1 z krokiem w 0.0025:



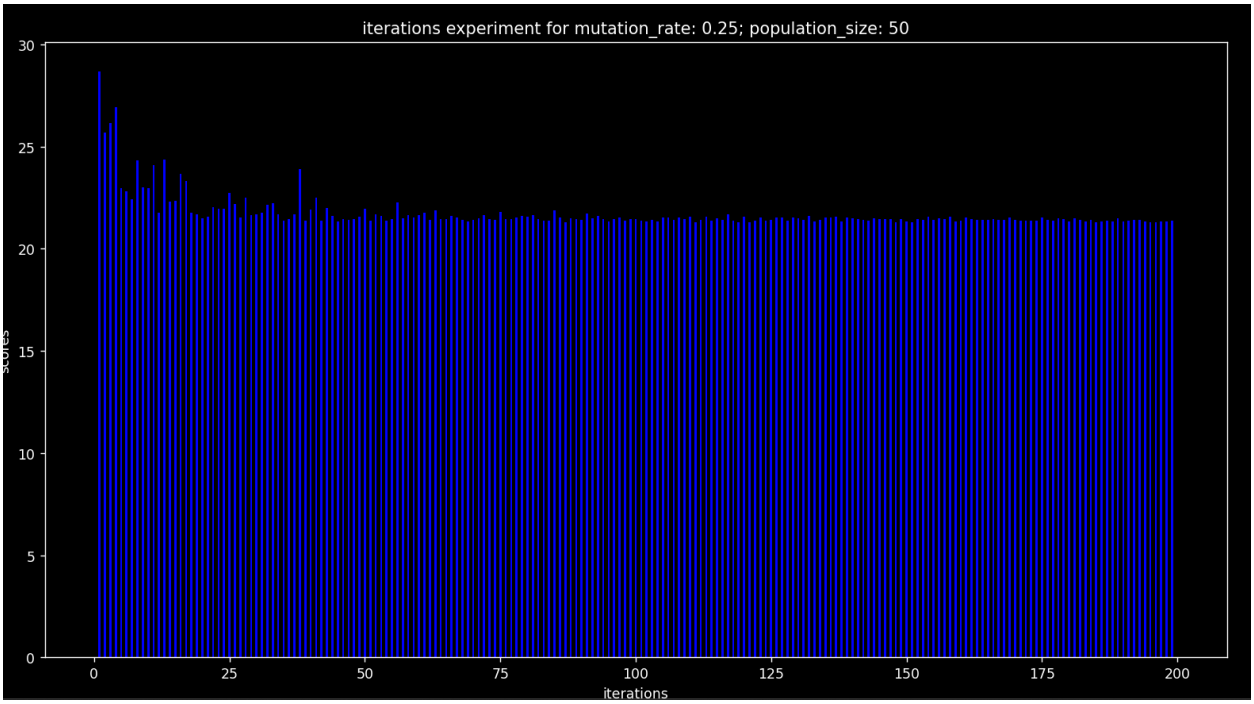
Komentarz: mutation_rate nie ma wielkiego wpływu. Można zauważyć że im większy mutation rate, tym mniej poprawny wynik śridnio. Jets tak ponieważ nawet jeżeli algorytm znajdzie bardzo dobry wynik, to po mutacji on bardzo zmieni się. Też jest ważne żeby nie wybrać za małą liczbę ponieważ wtedy algorytm może nie zdążyć wystarczająco zmodyfikować dzieci jeżeli będzie za mało iteracji, ale to raczej nie wielki problem kiedy rodzice mają losowy punkt początkowy.

Teraz zbadamy wpływ population_size która przyjmuje wartości od 1 do 100 z krokiem w 1:



Komentarz: Nie wątpliwie im większa populacja początkowa, tym lepiej będą wyniki. Ale nawet przy małej populacji czasem można otrzymać dobry wynik, ale to już bardziej kwestia losowości.

Teraz zbadamy zależność parametru iterations która przyjmuje wartości od 1 do 200 z krokiem w 1



Komentarz: Nie wątpliwie im większa liczba iteracji, tym lepiej będą wyniki...