**Raport**

**Wprowadzenie do Sztucznej Inteligencji - Ćwiczenie 2.**

*Algorytmy genetyczne i ewolucyjne*

Autor: Aleksandra Jamróz, nr albumu: 310 708

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Treść zadania**: Pan Mateusz kupił szczepionki na Kolanowirusa dla mieszkańców Wolski i musi objechał n=30 wielkich miast (wierzchołki w grafie) tego wspaniałego kraju, by dostarczyć je dla wszystkich. Niestety budżet pana Mateusza jest ograniczony i musi rozliczyć się z kosztów za paliwo z własnej kieszeni. Dlatego też chciałby on przejechać przez wszystkie miasta dokładnie raz jak najkrótszą / najszybszą drogą zaczynając i kończąc w tym samym mieście X (szukamy najkrótszego cyklu). Wskaż sekwencję tych miast (travelling salesman problem). Sugerowane rozkłady miast: jednorodny (taka szachownica), duże skupiska grup oraz losowy

**Sugerowany pseudokod:**

P\_t = **init**()  
t = 0  
**ocena**(P\_t)  
**while**!**stop**  
    Tt = **selekcja**(Pt)  
    Ot = **mutacja**(Tt)  
**ocena**(Ot)  
    Pt = Ot  
    t=t+1

Zgodnie ze wskazówką, stworzyłam 3 funkcje generujące współrzędne miast. Pierwsza generuje podaną liczbę współrzędnych losowo, druga układa współrzędne na siatce, trzecia formuje je w podaną liczbę aglomeracji o podanej liczbie miast.

Algorytm ewolucyjny składa się z następujących części:

1. Generowanie populacji – na podstawie listy współrzędnych miast tworzy podaną ilość osobników. Osobniki są w tym przypadku różnymi permutacjami zbioru współrzędnych.
2. Selekcja turniejowa – wykonywanych jest n turniejów, przy czym n to liczebność populacji, w każdym z nich startuje dwoje osobników. Lepszy osobnik wybierany jest na podstawie funkcji oceniającej – dla każdego osobnika liczę sumaryczną odległość między kolejnymi miastami i osobnik, którego odległość jest mniejsza, trafia do nowej listy.
3. Mutacja – z populacji po selekcji losuje się podany % osobników. Ulegają one mutacji, tj. zamieniona zostaje w nich kolejność niektórych punktów.
4. Sukcesja – zmodyfikowana populacja staje się populacją bazową i rozpoczyna się kolejny cykl.

W rozwiązaniu korzystałam z następujących bibliotek:

* Math – do liczenia pierwiastków potrzebnych do obliczenia odległości między miastami
* Matplotlib – efekt końcowy działania programu zwizualizowany jest na wykresie stworzonym za pomocą tej biblioteki
* Random – do generowania listy miast, losowania pozycji z listy i mieszania zawartości listy miast
* Timeit – mierzenia czasu wykonania programu

Obserwacje z przeprowadzonych prób:

* W przypadu małej liczby miast i liczebności populacji zbliżonej do liczby permutacji zbioru miast (liczby wszystkich możliwych ścieżek), duży współczynnik mutacji (% osobników, które miały przejść mutację) pogarszał wynik końcowy. Kiedy miast było więcej, a populacja nie mogła być aż tak ogromna, lepiej sprawdza się większy współczynnik mutacji.
* Za każdym razem zwiększenie liczby iteracji oraz populacji ma pozytywny wpływ na precyzję i optymalność rozwiązania, jednak wydłuża jego czas.
* Program szybciej znajduje rozwiązania dla miast ułożonych w szachownicę

SZACHOWNICA:

LOSOWE:

GRUPY: