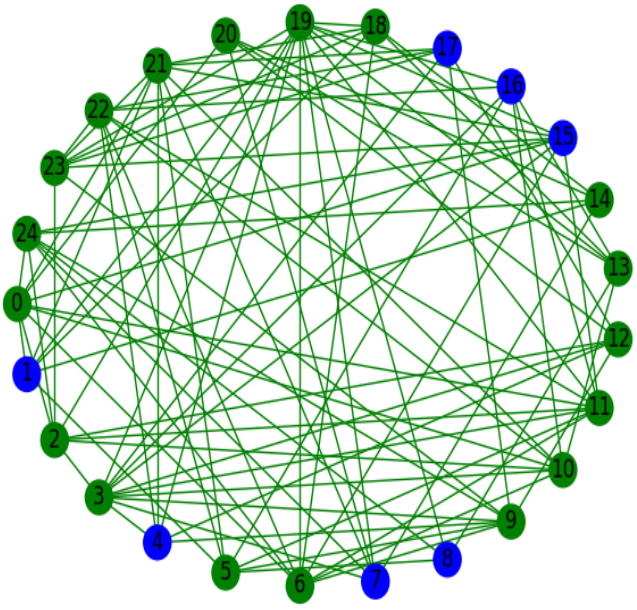
ALGORYTM GENETYCZNY – VERTEX COVER PROBLEM

Do rozwiązania problemu został użyty algorytm genetyczny z selekcją turniejową 2 osobnikową, losową mutacją i bez krzyżowania.

Algorytm na początku tworzy mapę połączeń między “latarniami” oraz populację osobników, z których każdy jest listą 0 i 1 (0 – latarnia zgaszona, 1- zapalona).

Następnie, każdy z osobników zostaje porównany z bazą wymaganych połączeń i wstępnie oceniany. Wtedy zaczyna się pętla, która powtarza proces oceny, dokonując dodatkowo mutacji z określonym prawdopodobieństwem w osobnikach, które wygrały porównania z losowo dobranym “przeciwnikiem”.




```
# ******************number of nodes (problem size) ************
s = 10
# ******************number of individuals in population *******
n = 100
# ******************number of iterations **********************
iterations = 40
# ******************filling the graph *************************
fill = 0.6
# ******************filling start population with ones *********
fill_pop = 0.3
# ******************probability of mutation *******************
mut = 0.1
# ******************loss multiplier ***************************
loss_multiplier = 100
# ******************probability of better solution winning ****
good_win_prob = 1
```

Pierwszy zrzut ekranu przedstawia wizualizację efektów pracy algorytmu.

Niebieski punkt – latarnia zgaszona, zielony – latarnia zapalona.

Wszystkie połączenia są zielone, więc ścieżki są oświetlone zgodnie z założeniem.

Jeśli któreś połączenie byłoby czerwone, oznaczałoby to że jest niepokryte.

Pomiary były wykonywane dla dwóch różnych ustawień parametrów (wynik to ilość zgaszonych node, cała reszta zapalona):

a) Ilość node: 25, rozmiar populacji: 100, iteracje: 40, wypełnienie połączeniami: 60%

   Wyniki:
   najlepszy wynik: 7 , mediana: 7, odchylenie standardowe: 0,516
   średni czas: 0,66s

6	0,644			
6	0,655			
6	0,629			
6	0,647		standard deviation:	
7	0,671		0,516397779	
7	0,642			
7	0,653			
7	0,657		average time:	
7	0,678		0,66	
7	0,659			

b) Ilość node : 60, rozmiar populacji: 200, iteracje:100, wypełnienie połączeniami: 15%

   Wyniki:
   najlepszy wynik: 14, mediana: 13, odchylenie standardowe: 0,75
   średni czas: 0,91s

12	0,895			
14	0,92			
14	0,913		standard deviation:	
14	0,93		0,752772653	
12	0,901			
12	0,928			
13	0,899		average time:	
13	0,912		0,911666667	
13	0,921			
14	0,909			

Parametry były dobierane wcześniej, na bazie tego, kiedy algorytm zaczyna dawać zbieżne wyniki po minimalnym czasie, były to jednak oszacowania.

Jak widać, przy większych rozmiarach danych wymagana jest większa liczba iteracji, więc większy czas na wykonanie algorytmu.

Biblioteki wykorzystane w projekcie:

networkx, matplotlib.pyplot- do tworzenia grafów

numpy - do przetwarzania macierzy

time – do pomiaru czasu wykonania

datetime – do tworzenia znaczników czasu wytworzenia matryc połączeń