Ćwiczenie nr 2
Wprowadzenie do sztucznej inteligencji

Autor
Maciej Prostak

1 Wprowadzenie

Zadanie polega na rozwiązaniu problemu pokrycia wierzchołkowego za pomocą algorytmu genetycznego. Algorytm zostanie zaimplementowany bez krzyżowania z selekcją turniejową i sukcesją generyczną. Następnie sprawdzę, jak zmiana liczby osobników w populacji wpływa na jakość uzyskanych rozwiązań.

1.1 Technologia

Rozwiązanie zostało zaimplementowane w języku Python (3.8.0) z wykorzystaniem bibliotek matplotlib (3.4.3) oraz numpy (1.21.2). Do generowania i rysowania grafów została użyta biblioteka networkx (2.6.3). Dodatkowo dla poprawy czytelności kodu zostały dołączone biblioteki black (21.9b0), flake8 (4.0.1) oraz isort (5.9.3).

1.2 Osobnik

Ważnym elementem w algorytmie jest reprezentacja pojedynczego osobnika. Została wybrana reprezentacja wektorowa dla osobnika, w której wartość 1 oznacza, że wierzchołek jest zaznaczony a wartość 0 jako wierzchołek niezaznaczony.

0	1	0	0	1

Fig. 1 Wizualizacja pojedynczego osobnika.

1.3 Populacja

Populacja jest to zbiór osobników i będzie reprezentowany jako lista osobników. Poprzez zmianę wielkości populacji zobaczymy, jaki będzie miało to wpływ na algorytm genetyczny pod względem czasowym oraz dokładności.

0	1	0	0	1
1	1	0	1	0
...	...	...	...	...
0	0	0	1	1

Fig. 2 Wizualizacja populacji złożonej z osobników.

1.4 Mutacja

Podczas mutacji pod uwagę będziemy brali dwa prawdopodobieństwa. Pierwsze z nich to to, czy u osobnika zajdzie mutacja, a druga ile wartości będziemy chcieli zamienić. W naszym problemie liczba zaznaczonych wierzchołków jest ustalona na początku działania algorytmu, więc należało zadbać o to, że liczba zaznaczonych wierzchołków pozostanie bez zmian. Znajdujemy dwa losowe indeksy, w którym jeden określa zaznaczony wierzchołek, a drugi niezaznaczony, a następnie następuje prosta zamiana tych wartości.

1.5 Selekcja

Selekcję w algorytmie będziemy odbywać metodą turniejową. Ważnym założeniem w wybranej selekcji jest, że do turnieju będziemy losować tylko dwóch osobników i wybierać lepszego. Losowanie odbywa się ze zwracaniem, więc ten sam osobnik może pojawić się populacji kilkukrotnie.

1.6 Sukcesja generacyjna

Na etapie selekcji i mutacji zadbałem o to, żeby wielkość populacji była niezmienna. Sukcesja w naszym przypadku jest prosta i polega na podmianie starej populacji nową po wykonaniu mutacji.

1.7 Ocena osobnika

Osobniki będą ze sobą porównywane za pomocą specjalnej funkcji oceny, która pomoże określić nam odpowiednią selekcję osobników w populacji oraz odpowiedzieć nam na jakim etapie jest algorytm. Funkcja oceny w naszym przypadku jest to suma ilości krawędzi, które nie są pokrywane za pomocą zaznaczonych wierzchołków. W takim przypadku otrzymujemy rozwiązanie, gdy wartość oceny będzie równa 0. Na odpowiednich wykresach zamieszczonych w wynikach zauważymy, że funkcja oceny będzie odpowiednio minimalizowana.

2 Wyniki

W zadaniu określone jest, że algorytm ma być wykonywany na grafach, w których liczba wierzchołków jest równa 25. W większości przypadków eksperymenty będą przeprowadzane na dwuwymiarowych grafach siatkowych. Wierzchołki zaznaczone, czyli te na których stoją "policjanci", są oznaczone kolorem czerwonym. Jeżeli istnieje krawędź grafu, która nie wychodzi z żadnego zaznaczonego wierzchołka, oznaczymy ją kolorem czerwonym.

2.1 Siatka dwuwymiarowa

Przykładowe działanie algorytmu genetycznego można zaobserwować na rysunku (Fig. 3), w którym widoczny jest graf końcowy oraz wykres jakości populacji względem kolejnych generacji. Widoczne są "schodki" idące w dół, które oznaczają, że algorytm znajduje kolejne lepsze rozwiązania. W generacji około 500 widzimy, że zaszła mutacja która odniosła lepszy wynik niż pozostałe osobniki, lecz w selekcji turniejowej nie został on dalej wylosowany.

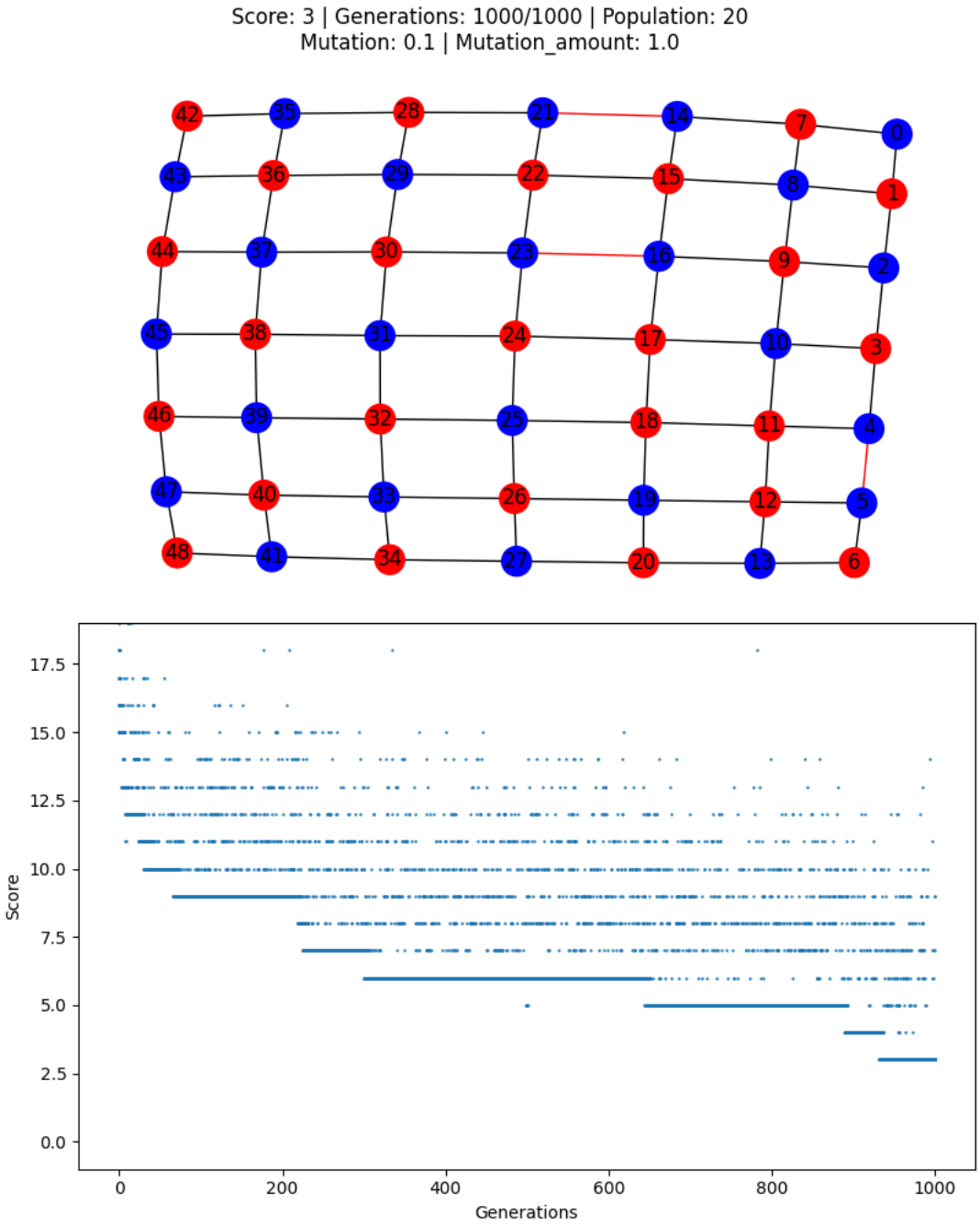

Fig. 3 Wizualizacja grafu końcowego oraz wykres oceny populacji do każdej generacji.

Na podanych poniżej przykładach (Fig 4. oraz Fig. 5) widać, jak wielkość populacji wpływa na jakość wyników. Przy wielkości populacji równej 100 optymalne rozwiązanie zostaje znalezione dość szybko (33 generacje), a w przypadku małej populacji równej 20, rozwiązanie nie zostaje znalezione (nawet przy większej ilości generacji). Należy również dodać, że rozwiązanie w drugim przypadku jest możliwe do znalezienia, jeżeli wylosowujemy odpowiednią początkową populację.

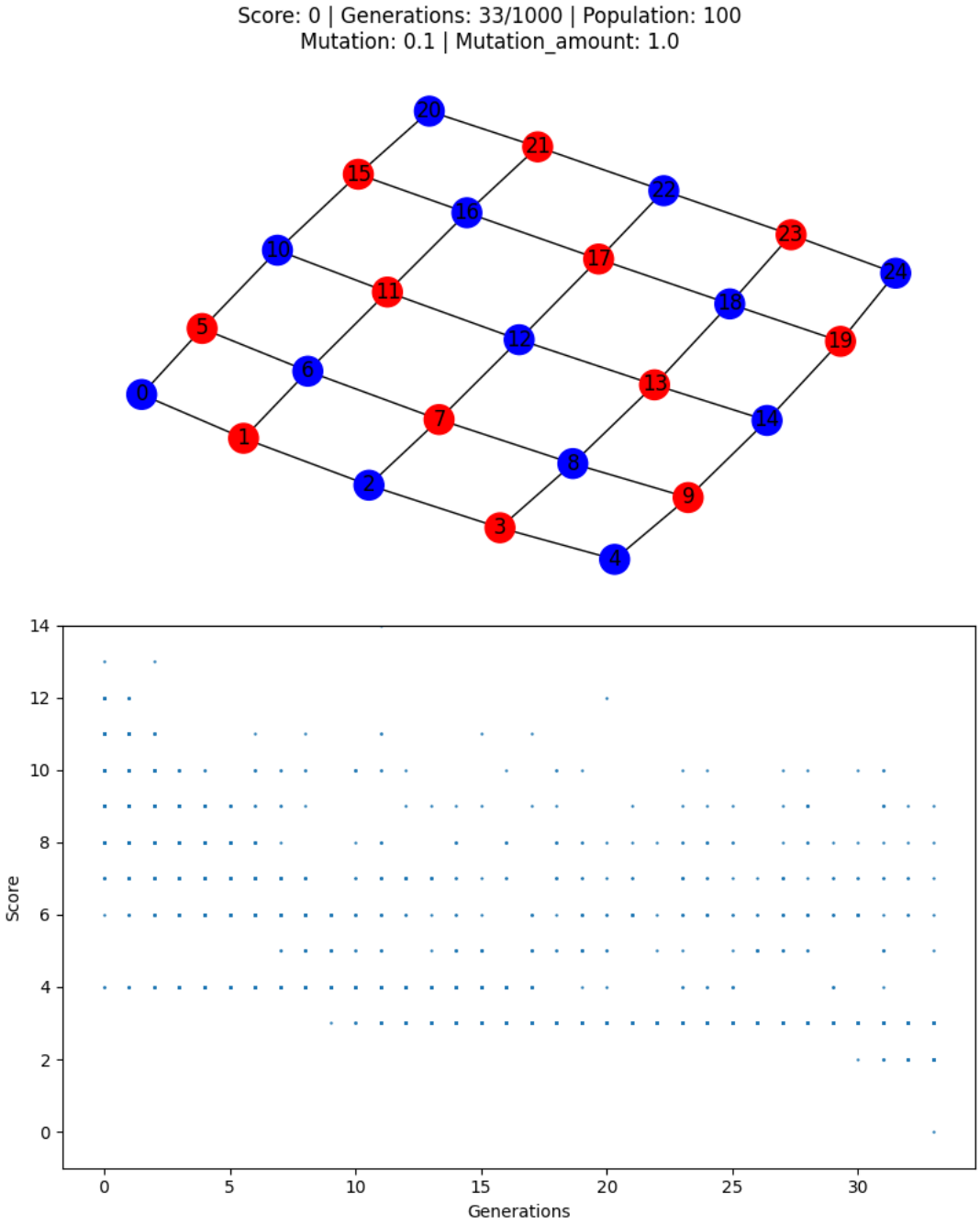

Fig. 4 Wizualizacja grafu końcowego oraz wykres oceny populacji do każdej generacji. Liczba wierzchołków=25, liczba zaznaczonych wierzchołków=12.

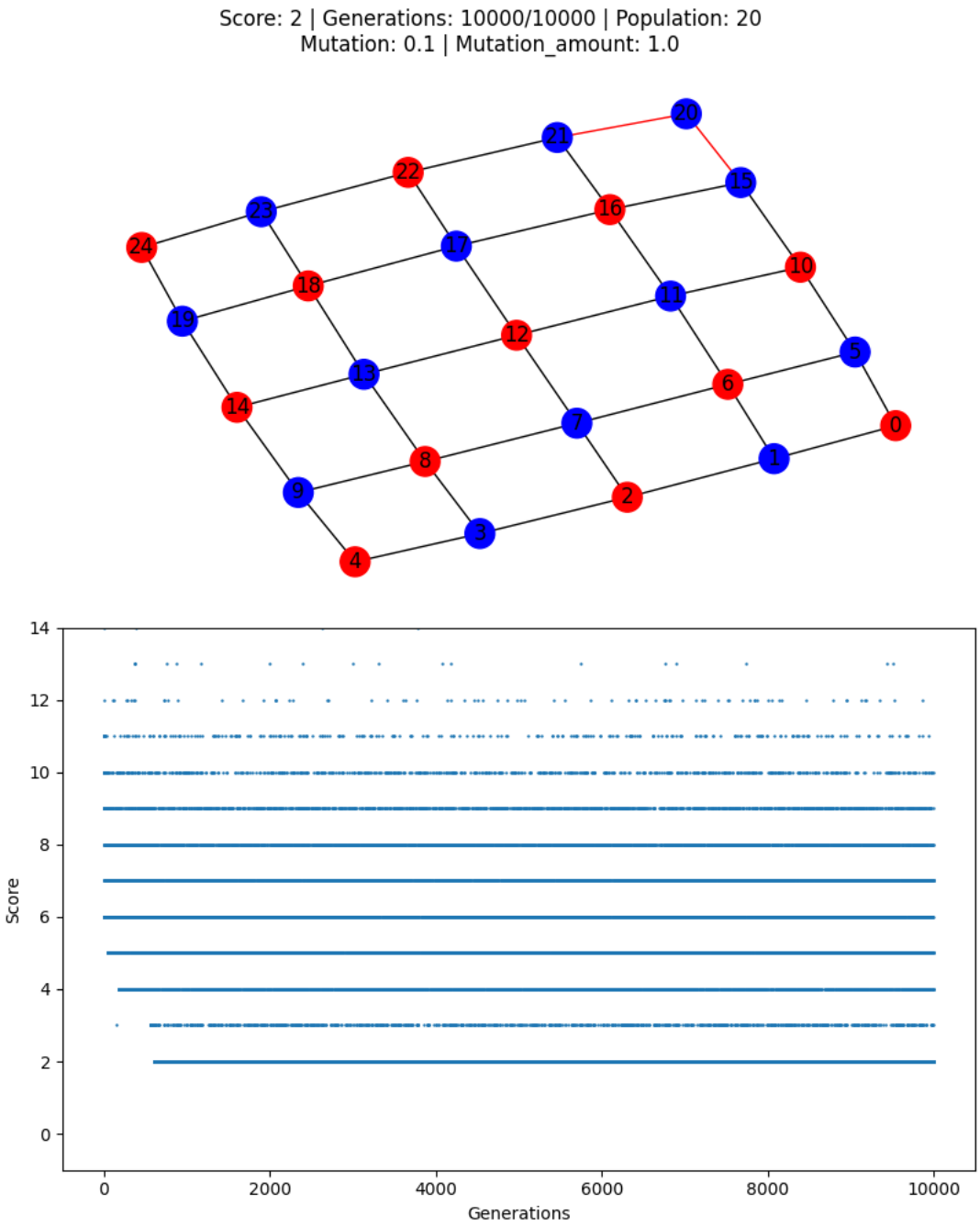

Fig. 5 Wizualizacja grafu końcowego oraz wykres oceny populacji do każdej generacji. Liczba wierzchołków=25, liczba zaznaczonych wierzchołków=12.

2.2 Czas wykonywania

Podane wyniki w tabeli (Tabela 1) ilustrują mocną korelację pomiędzy wielkością populacji a czasem wykonania. Czas podany w mikrosekundach odpowiada czasu wykonania obliczeń na jedną iterację generacji, dlatego, że czasem algorytm mógł skończyć się wcześniej jeżeli znajdziemy rozwiązanie w trakcie działania algorytmu (nie po warunku stopu na maksymalną liczbę generacji). Widzimy również, że po prawie dwukrotnym zwiększeniu ilości wierzchołków czas wykonania nie zmienia się dwukrotnie, tylko o około 30 procent.

Ilość wierzchołków	Wielkość populacji	Czas w milisekundach / generacje
25	5	0.22
25	10	0.43
25	20	0.87
25	100	4.28
25	500	21.50
49	5	0.40
49	10	0.78
49	20	1.55
49	100	7.66
49	500	38.55

Tabela 1 Wyniki czasowe dla różnych parametrów dla dwuwymiarowego grafu siatkowego. mutation probability=0.1, mutation amount probability=1.0

2.3 Graf pełny

W podanych przykładach końcowych grafów pełnych (Fig. 6, Fig. 7), w którym wybrany jest najlepszy osobnik widzimy, że algorytm daje optymalne rozwiązanie dość szybko jeżeli szukana liczba zaznaczonych wierzchołków jest równa n-1, gdzie n to liczba wierzchołków.

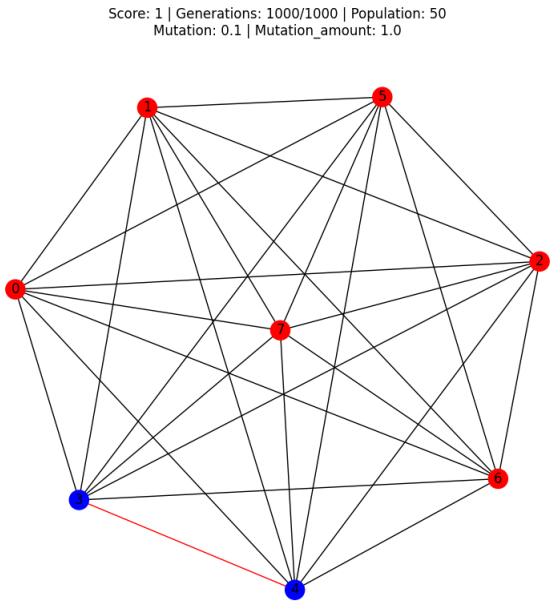

Fig. 6 Wizualizacja grafu pełnego (8 wierzchołków) z 6 zaznaczonymi wierzchołkami.

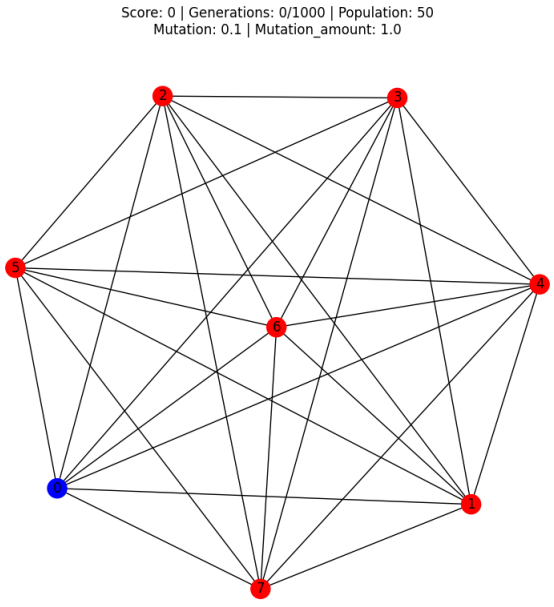

Fig. 7 Wizualizacja grafu pełnego (8 wierzchołków) z 7 zaznaczonymi wierzchołkami.

2.4 Graf losowy

Graf losowy widoczny na rysunku (Fig. 8) nie znalazł rozwiązania dla podanych parametrów. Widzimy jednak ciekawe zachowanie jakim jest unikanie zaznaczania wierzchołków przy liściach. Wyjątkiem jest para wierzchołków z numerami 11 oraz 21, w którym nie ma znaczenia który dokładnie jest zaznaczony.

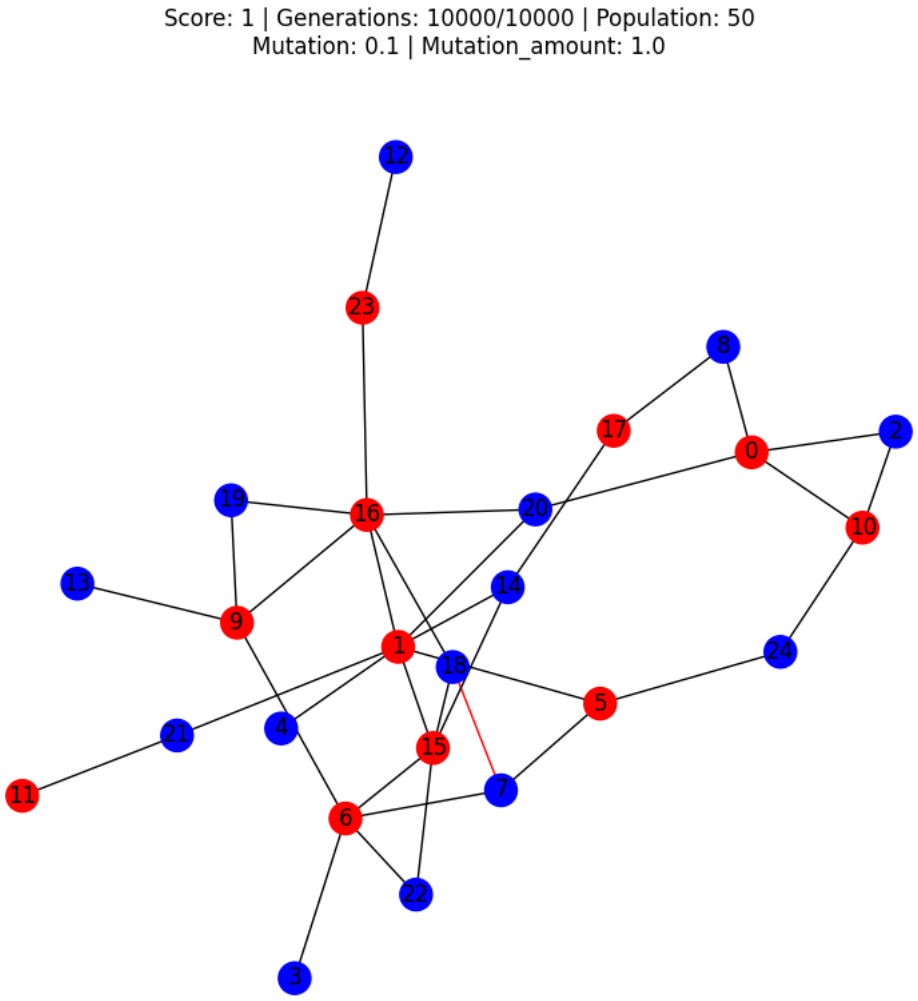

Fig. 8 Wizualizacja grafu losowego (25 wierzchołków) z 11 zaznaczonymi wierzchołkami.

2.5 Mutacja

Zwiększenie wartości losowania osobnika w mutacji oraz wartości losowania ilości zmian znacząco wpływa na działanie algorytmu. Przedstawione wykresy (Fig. 9, Fig. 10) pokazują, że zwiększenie siły mutacji utrudnia dojścia do minimum oraz że najlepsze osobniki mutacji zostają często modyfikowani, tracąc przy tym możliwość dojścia do najlepszych wyników.

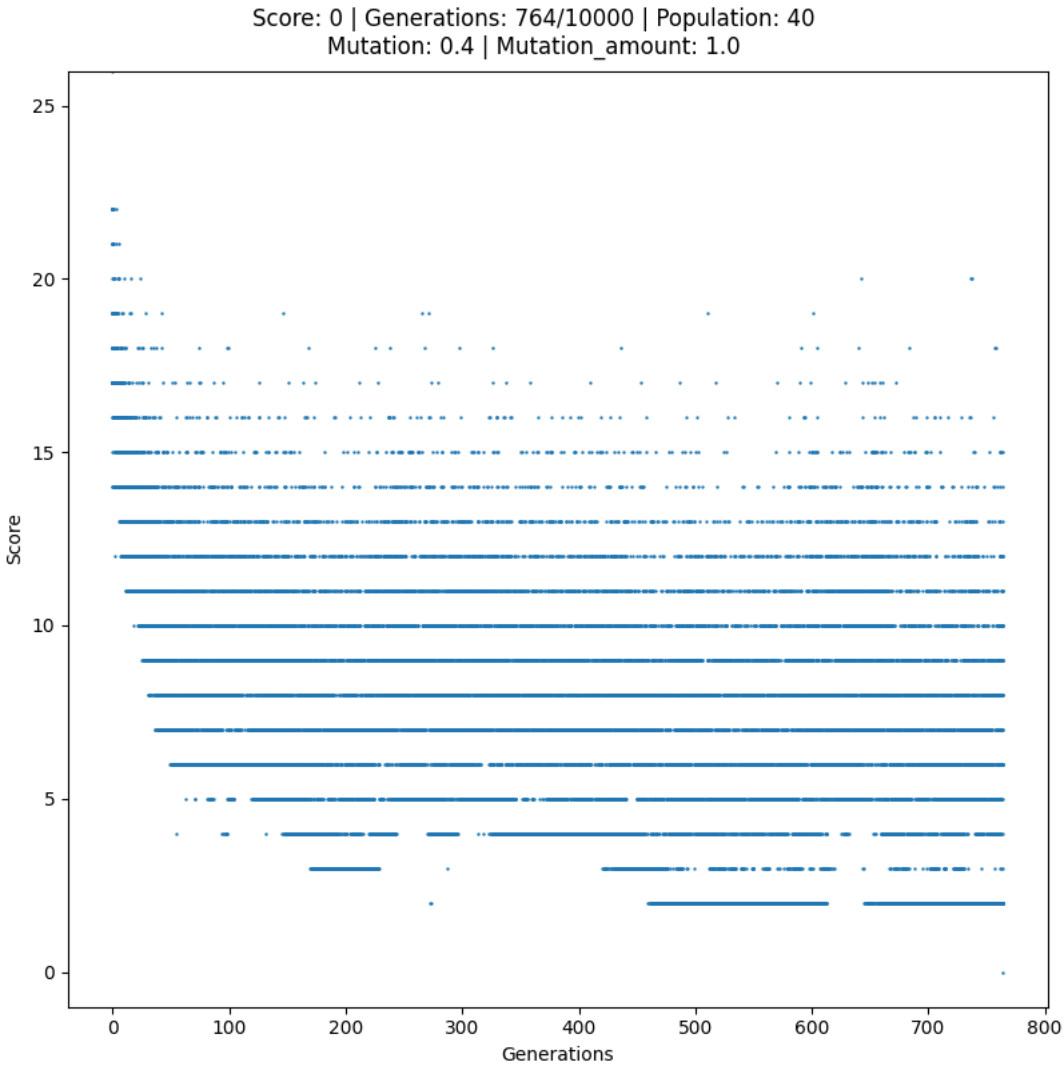

Fig. 9 Wizualizacja grafu losowego (25 wierzchołków) z 11 zaznaczonymi wierzchołkami.

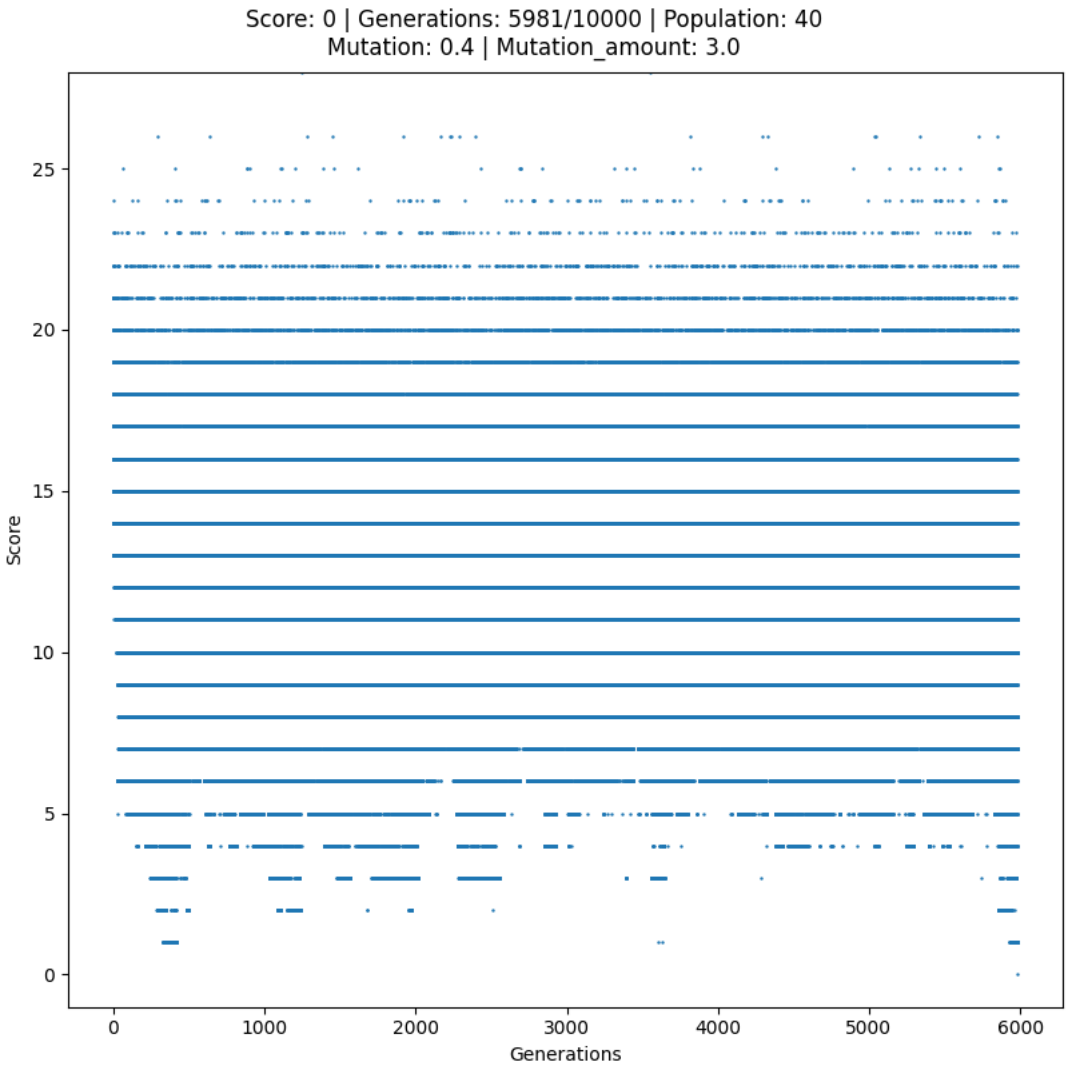

Fig. 10 Wizualizacja grafu losowego (25 wierzchołków) z 11 zaznaczonymi wierzchołkami.

2.6 Większa ilość prób

Przedstawiony algorytm wymaga wprowadzenia ile wierzchołków zamierzamy pokryć na początku. Oznacza to, że jeżeli badamy dany graf musimy wielokrotnie sprawdzić wyniki, podając coraz mniejszą lub większą ilość pokrytych wierzchołków. Dla dwuwymiarowego grafu siatkowego o wymiarach 7x7 i uruchomiłem wielokrotnie program, sprawdzając czasy znalezienia rozwiązań. Maksymalna liczba generacji to 10000 w naszym przypadku. W wynikach widzimy, że tylko 2 na 10 prób zakończyły się poprawnym wynikiem. Prawdopodobieństwo w inicjalizacji populacji początkowej, selekcji oraz mutacji mają duży wpływ na wynik końcowy. Należy więc podjąć większą ilość prób w celu znalezienia najlepszego wyniku. Ilość wierzchołków ile zamierzamy pokryć to 25. Liczba populacji to 100, czynnik mutacji: 0.1

Wywołanie	Najlepszy wynik	Czas w milisekundach
1	2	77384
2	0	34390
3	2	77589
4	3	77626
5	2	77379
6	0	2912
7	2	78388
8	2	77531
9	2	77381
10	2	77212

Tabela 2 Wyniki dla 10 prób. Tabela ukazuje numer próby, najlepszy wynik oraz całkowity czas działania programu.

3 Wnioski

Przedstawiony i zaimplementowany algorytm genetyczny na podanych przykładach pozwolił na możliwość wyszukiwania zadowalających rozwiązań dla problemu pokrycia wierzchołkowego. Ważne jest odpowiednie dobranie parametrów oraz cierpliwości, żeby znaleźć odpowiednie rozwiązanie dla badanego grafu. Przy niepoprawnej dobranej wartości populacji możemy dojść do sytuacji, w której pokrywamy mały zbiór możliwości i znalezione rozwiązania są nieoptymalne. Wartość mutacji jest równie ważna, ponieważ za duża wartość nie pozwoli nam się skupić na lokalnym minimum, ponieważ duża część populacji będzie się zmieniać (w tym najlepsze osobniki), a jeśli mutacja będzie za mała to algorytm będzie bardzo wolno (lub wcale) poszukiwać nowych, lepszych rozwiązań.