Semestr zimowy 2021

Drugie laboratorium z przedmiotu Wprowadzenie do sztucznej inteligencji

Autor
Łukasz Jaremek
310710

Data
październik 2021

Praca wykonana samodzielnie.

Użyte technologie

Język:
- Python 3.8.8

Biblioteki ponad standardowe:
- pygame 2.0.2
- matplotlib 3.4.3

Znajdowanie najkrótszego cyklu

Problem:

Znalezienie najlepszego cyklu ułożonego z podanej grupy punktów z użyciem algorytmu ewolucyjnego. Punkty są umieszczone w kartezjańskim układzie współrzędnych na przedziałach x, y $\in$ <-200, 200>.

Najważniejsze pojęcia:

- Osobnik – cykl złożony z punktów
- Populacja – grupa osobników
- Selekcja turniejowa – wybieranie cyklu z najkrótszą trasą
- Mutacja populacji – zamiana kolejności punktów w populacji
- Generacja – selekcja a następnie mutacja populacji

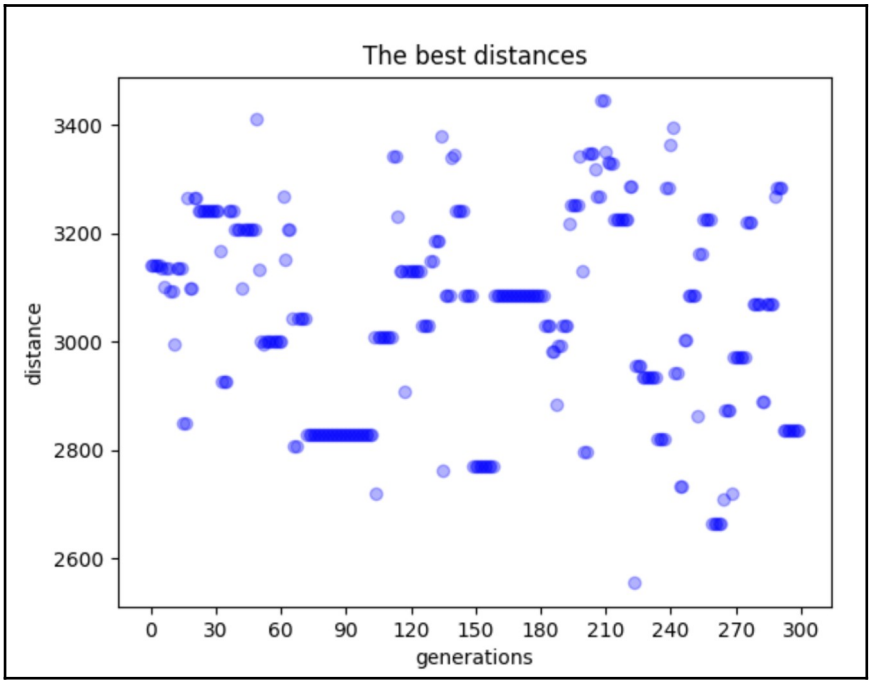
Pseudokod algorytmu ewolucyjnego:

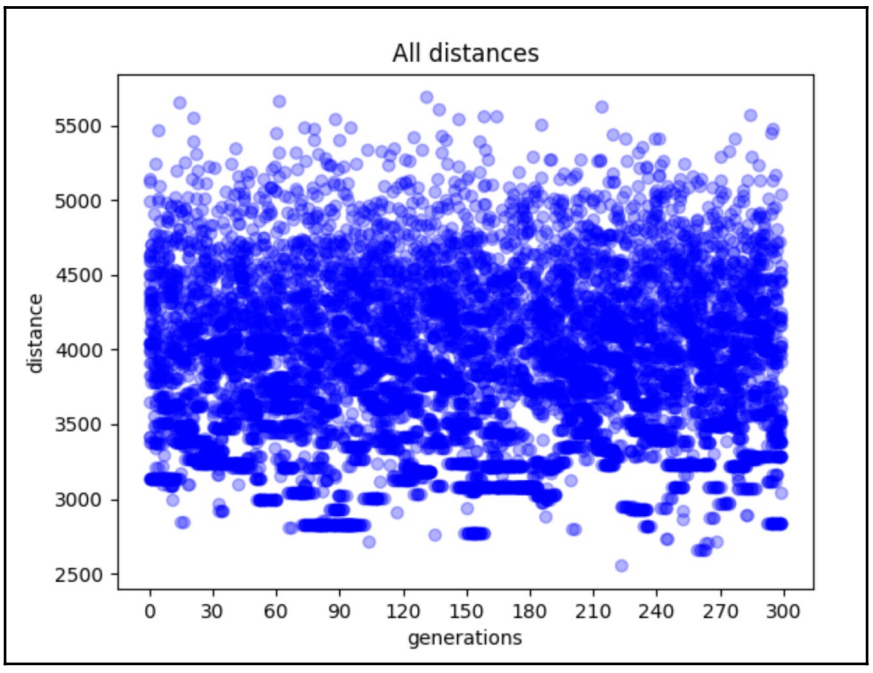
```
proc_populacji = <szansa na mutację populacji>
proc_osobnika = <szansa na mutację osobnika>
populacja = [cykl1, cykl2, cykl3...]
iteracje = X
while iteracje != 0:
    populacja = selekcja_turniejowa(populacja)
    populacja = mutuj_populacje(populacja, proc_populacji, proc_osobnika)
    iteracje -= 1
```

Sprawdzanie algorytmu na grafach losowych

Sprawdźmy, jak zadziała algorytm dla takich parametrów:

- szans na zmutowanie populacji: 50%
- szansa na zmutowanie osobnika: 50%
- liczba osobników = liczba populacji = 30
- liczba generacji: 300

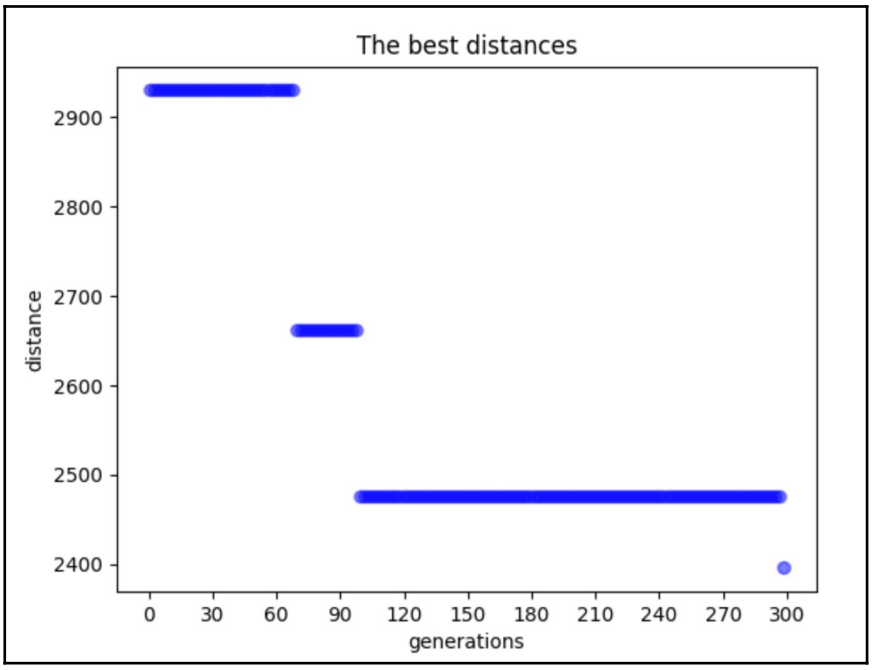


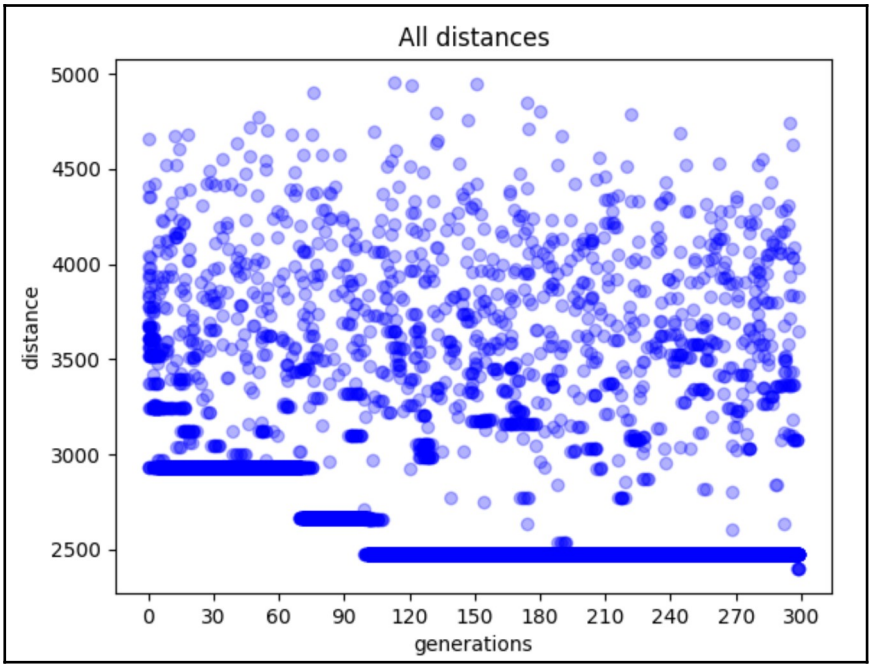


Jak widać, wyniki są mocno losowe, powinniśmy dążyć do sytuacji, w której im dalsza generacja, tym krótszy dystans, natomiast tak nie jest. Spróbujmy to zmienić.

Zmniejszam szansę, na zmutowanie populacji:

- szans na zmutowanie populacji: **10%**
- szansa na zmutowanie osobnika: 50%
- liczba osobników = liczba populacji = 30
- liczba generacji: 300

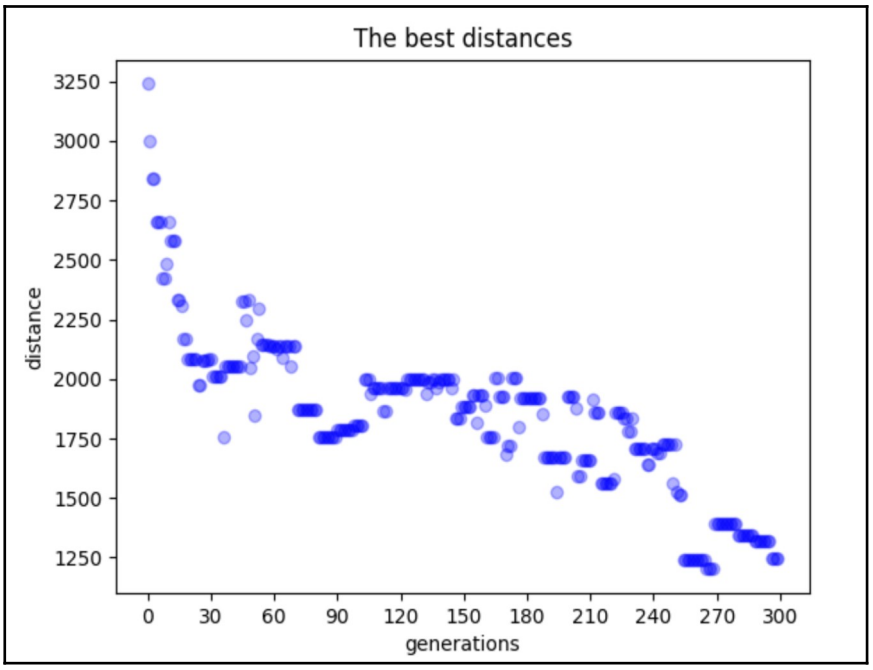


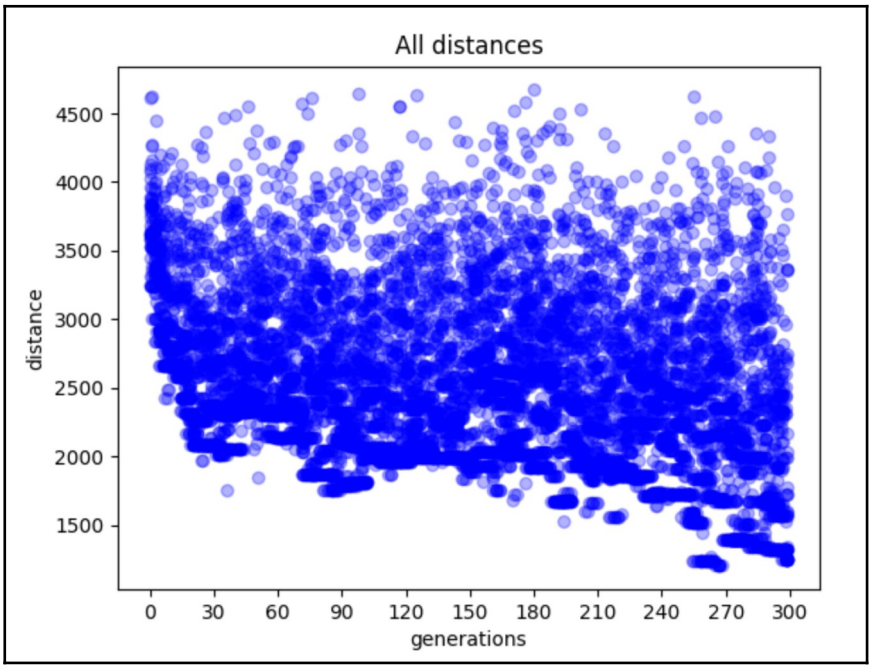


Widzimy wyraźny postęp, im dalsza generacja tym faktycznie wyraźnie krótsza trasa. Nawet w ostatnich generacjach powstała jeszcze lepsza trasa, więc możliwe że przy większej liczbie iteracji wynik byłby jeszcze lepszy.

Sprawdźmy więc jak wpłynie zmniejszenie szansy na zmutowanie osobnika:

- szans na zmutowanie populacji: 50%
- szansa na zmutowanie osobnika: **10%**
- liczba osobników = liczba populacji = 30
- liczba generacji: 300

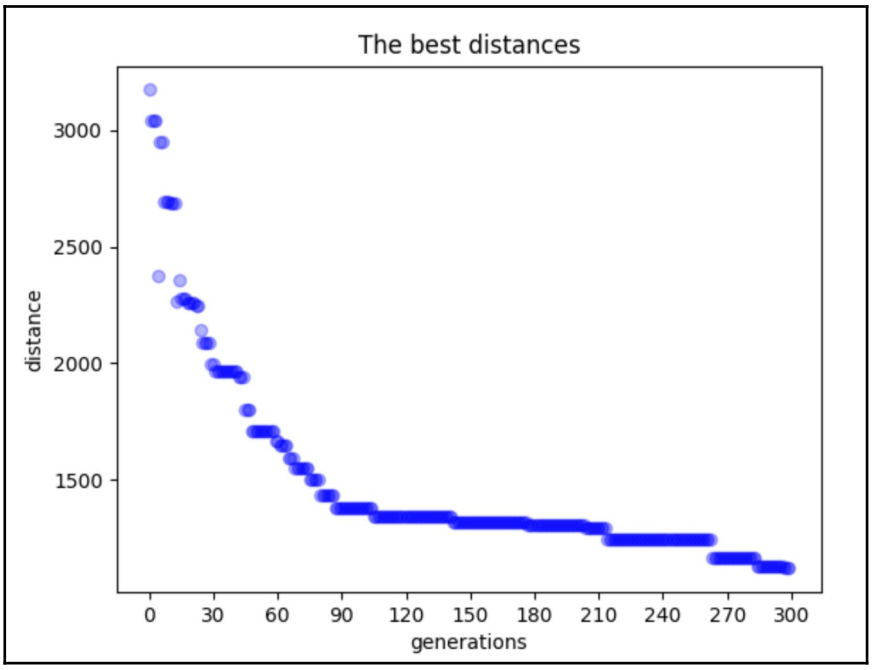


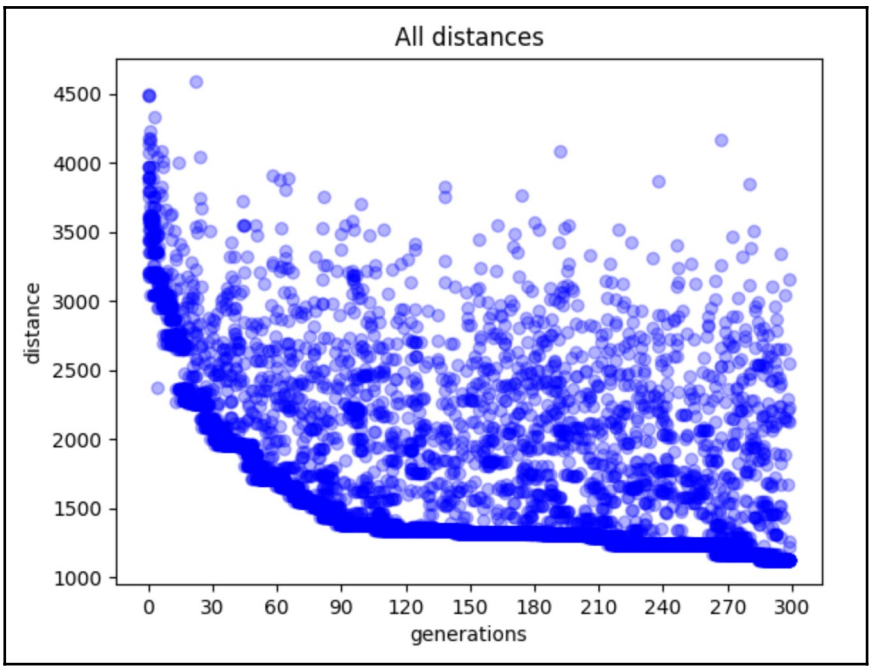


Widzimy ciekawe zachowanie, najkrótsza trasa się **regularnie** zmniejsza w porównaniu do **skokowej** zmiany w poprzednim przykładzie. Można też zauważyć bardziej równomierne rozłożenie wszystkich osobników i lepszy wynik najkrótszego cyklu (może to być przypadek).

Skoro mutacja osobnika jest bardziej istotna, zostawmy ją tak i zwiększy drugą mutację:

- szans na zmutowanie populacji: **20%**
- szansa na zmutowanie osobnika: **10%**
- liczba osobników = liczba populacji = 30
- liczba generacji: 300





Rozwiązanie jest jeszcze lepsze od poprzednich: logarytmicznie znajdujemy coraz lepsze trasy – w bardzo małej ilości iteracji jesteśmy w stanie szybko dojść do lepszego wyniku oraz mamy coraz mniej „złych” tras. Najlepsza trasa jest lepsza od poprzednich najlepszych.

Statystyki algorytmu na grafach losowych

Dla najlepszych znalezionych parametrów, sprawdźmy statystyki algorytmu.

- Ilość pomiarów: 30
- szansa na zmutowanie populacji: 20%
- szansa na zmutowanie osobnika: 20%
- liczba osobników: 30
- liczba populacji: 30
- liczba generacji: 300

Funkcja	Czas trwania (s)	Długość cyklu
Minimum	5.00	2396.0
Maksimum	11.65	3772.0
Średnia	7.09	3057.7(3)
Odchylenie standardowe	2.41	290.50

Wpływ wielkości populacji na wynik (30 pomiarów):

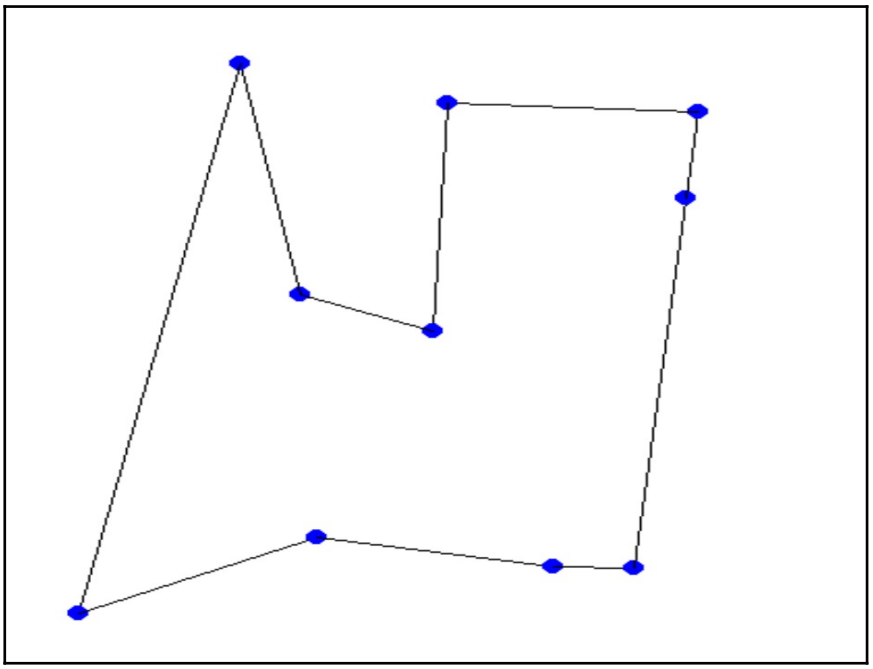
Wielkość populacji	Średni czas trwania (s)	Średnia długość cyklu
30	4.50	3075.60
60	11.29	2972.60
100	25.56	2912.73
200	93.36	2806.67

Wnioski z algorytmu ewolucyjnego na grafach losowych

Im mniejsza szansa na mutację, tym lepiej.
Im więcej generacji, tym lepiej.
Im większa liczba osobników w populacji tym lepiej, lecz biorąc pod uwagę wzrost czasu, jest to nie opłacalne.

Dlatego najlepiej wybrać małą liczbę populacji (kilkadziesiąt sztuk), małe prawdopodobieństwo na mutacje i dużo generacji (tak dużo, na ile nas stać). Jeśli generacje nie przynoszą już postępu, można spróbować zwiększyć liczbę osobników w populacji.

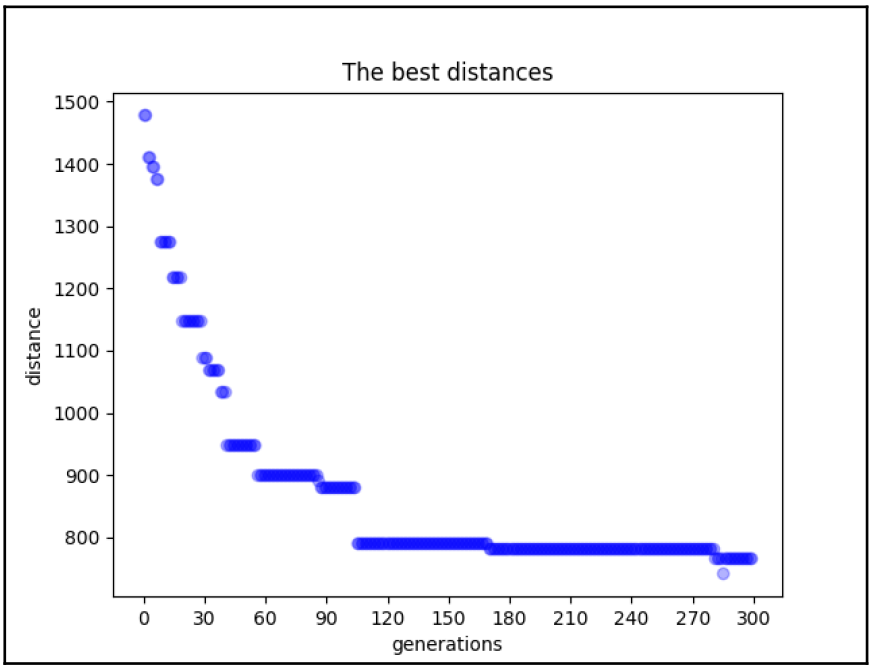
Przykładowy graf z 10 osobnikami (najkrótsza, znaleziona trasa):

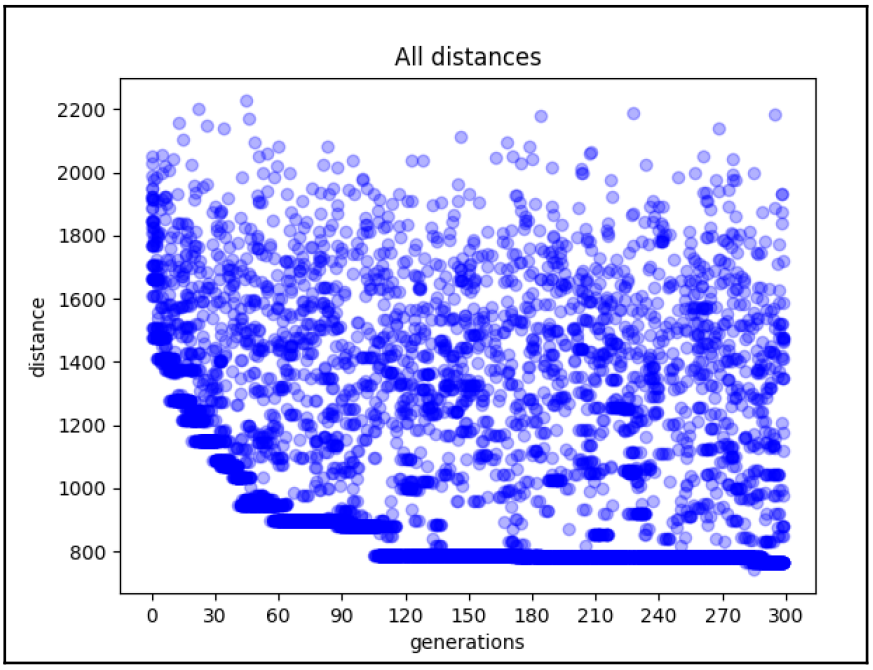


Sprawdzanie algorytmu na skupiskach

Sprawdźmy, jak zadziała algorytm dla takich parametrów:

- szans na zmutowanie populacji: 20%
- szansa na zmutowanie osobnika: 20%
- liczba osobników = liczba populacji = 30
- liczba generacji: 300





Mogłoby się wydawać, że wyniki które przyniosły sukces w poprzedniej próbie, w przypadku miast również się sprawdzą.

Statystyki algorytmu na grafach skupiskowych

Dla najlepszych znalezionych parametrów, sprawdźmy statystyki algorytmu.

- Ilość pomiarów: 30
- szansa na zmutowanie populacji: 20%
- szansa na zmutowanie osobnika: 20%
- liczba osobników: 30
- liczba populacji: 30
- liczba generacji: 300

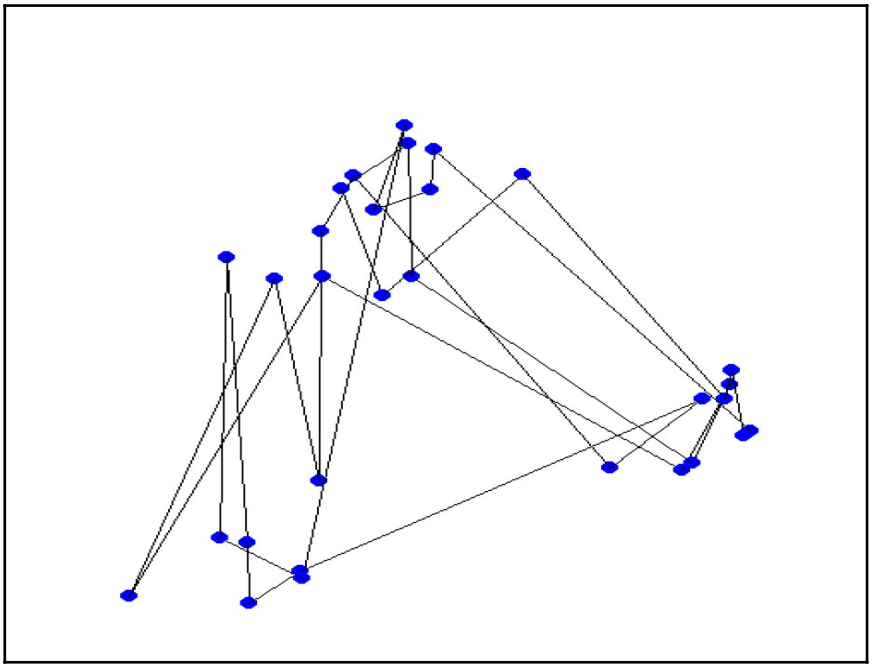
Funkcja	Czas trwania (s)	Długość cyklu
Minimum	3.88	1200.0
Maksimum	5.53	4036.0
Średnia	4.37	2394.73
Odchylenie standardowe	0.35	720.77

Wpływ wielkości populacji na wynik (30 pomiarów):

Wielkość populacji	Średni czas trwania (s)	Średnia długość cyklu
30	4.37	2448.27
60	11.04	2394.73
100	24.49	2189.20
200	85.35	2218.4

Wnioski z algorytmu ewolucyjnego na skupiskach

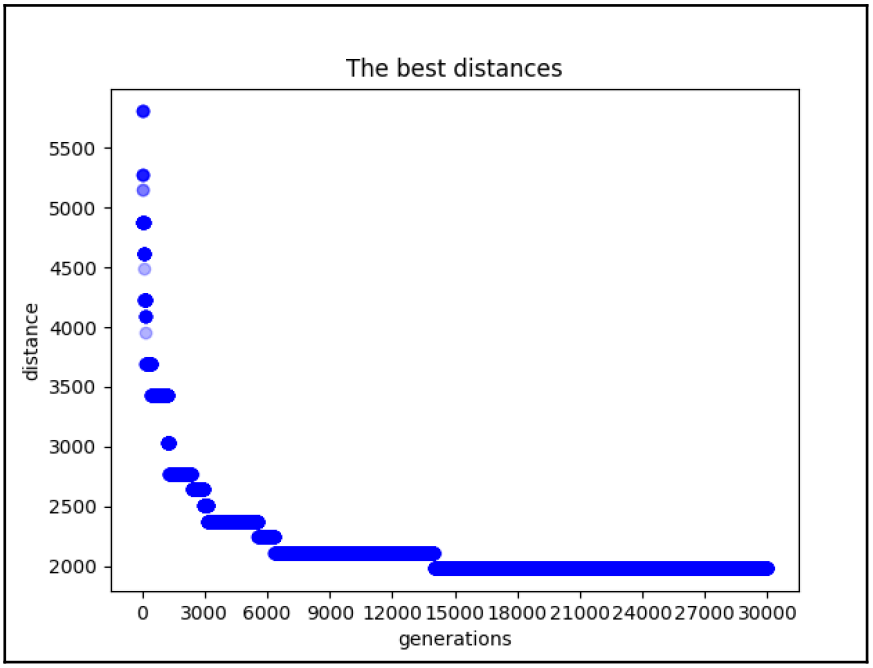
Choć wykresy pokazują, że algorytm działa dobrze, sprawdźmy jak wygląda przykładowe połączenie punktów na mapie:



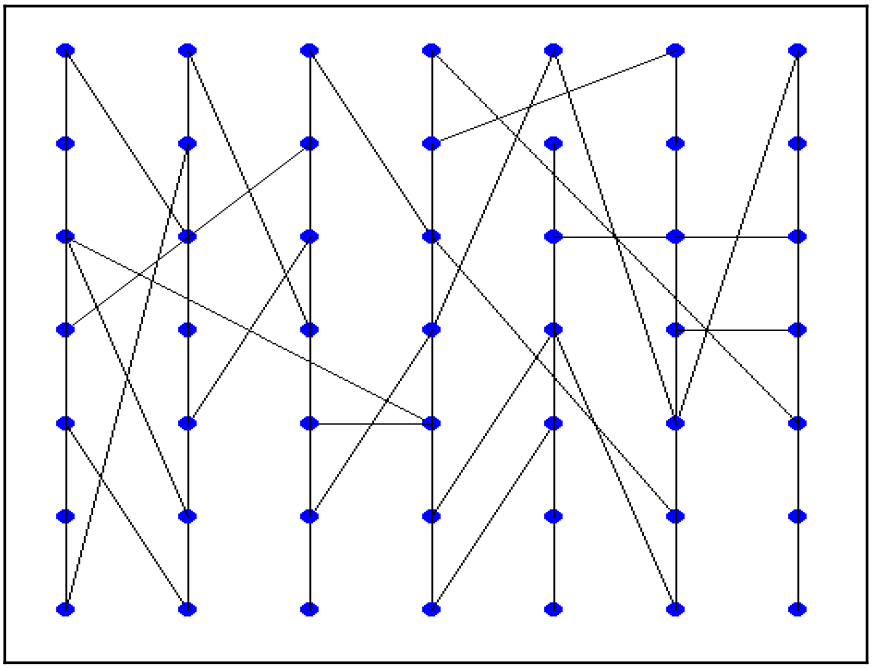
Jak widać, wynik nie jest zadowalający. Połączenia w skupiskach są przeważnie w porządku, lecz połączenia między skupiskami są nieoptymalne (powinny być najwyżej dwa: do miasta i z miasta, a jest ich wiele). Dziwne wyniki zwrócił również test różnych wielkości populacji, gdzie populacje z 200 osobnikami, dała gorszy wynik niż populacja ze 100 osobnikami.
Na podstawie tej i innych prób stwierdzam, że algorytm nie jest najlepszym rozwiązaniem do tego problemu.

Sprawdzanie i wnioski z algorytmu na grafach jednorodnych

Ustawiam 49 punktów w szachownicę, 7 punktów w 7 wierszach. Najkrótsza droga będzie wtedy, gdy jak najwięcej krawędzi będzie poziomych lub pionowych.
Wykonałem 30 000 generacji i zobaczmy jak algorytm znajduje coraz lepsze trasy:



Trasa po 30 000 generacji:



Wynik nie jest perfekcyjny, lecz z pewnością bardzo dobry. Rozłożenie punktów na wykresie wygląda jak funkcja $\frac{1}{x}$, więc żeby uzyskać wynik idealny potrzebowalibyśmy zdecydowanie więcej iteracji, które kosztują nas czas.

Podsumowanie laboratorium

Algorytm ewolucyjny daje różne wyniki zależnie od:

- rodzaju grafu, na którym operujemy
- liczby osobników w populacji, liczby populacji
- szansy na zmutowanie osobnika / populacji
- rodzaju mutacji, rodzaju selekcji
- liczby generacji
- …

Parametrów jest sporo, lecz ich zrozumienie i odpowiednie dobranie pozwala nam faktycznie otrzymać dobre wyniki, pod warunkiem że dysponujemy odpowiednią ilością czasu.

Algorytm z pewnością nie służy do znalezienia najlepszego (czy nawet jednego z najlepszych) rozwiązań, lecz jest prosty w implementacji i zrozumieniu działania co w kryzysowych sytuacjach może znaleźć zastosowanie.

Najlepiej sprawuje się na grafach losowych, nieco gorzej na jednorodnych, i niezadowalająco na skupiskowych.

Gdybym dla tych ostatnich miał zaproponować rozwiązanie, wyglądałoby ono tak:

Wykrywam skupiska punktów, szukam między nimi najkrótszego cyklu, a następnie szukam najlepszej drogi pomiędzy cyklami. Można by tutaj zastosować rekurencję, która by znajdowała coraz to większe skupiska, składające się z mniejszych.