AEM - Lokalne przeszukiwanie

Dawid Białek 131731, Bartosz Mila 131804

22 kwietnia 2020

1. Krótki opis zadania.

Zadanie polega na implementacji lokalnego przeszukiwania w wersjach stromej (steepest) i zachłannej (greedy) z dwoma różnym rodzajami sąsiedztwa. W sumie 4 kombinacje - wersje lokalnego przeszukiwania.

Sąsiedztwa:

W naszym przypadku będą potrzebne dwa typy ruchów: ruchy zmieniające zbiór wierzchołków i ruchy wewnątrztrasowe, które jedynie zmieniają kolejność wierzchołków na trasie.

Jako ruch zmieniający zbiór wierzchołków wykorzystujemy wymianę dwóch wierzchołków, z których jeden jest w trasie, a drugi po nią.

Stosujemy dwa rodzaje ruchów wewnątrztrasowych (jeden albo drugi, stąd dwa rodzaje sąsiedztwa). jeden to wymiana dwóch wierzchołków wchodzących w skład trasy, drugi to wymiana dwóch krawędzi.

Implementacje musi wykorzystywać obliczanie delty funkcji celu.

Sąsiedztwo składa się więc z ruchów dwóch typów. W wersji stromej przeglądamy wszystkie ruchy obu typów i wybieramy najlepszy. W wersji stromej należy zrandomizować kolejność przeglądania. W sprawozdaniu proszę opisać sposób randomizacji.

Każdy algorytm na każdej instancji uruchamiany 100 razy startując z rozwiązań losowych i raportujemy wartości min, max i średnią dla funkcji celu i czasu. Wszystkie wyniki zbieramy w jednej tabeli.

Poza tym zawartość sprawozdania podobnie jako poprzednio. Pseudokody algorytmów, wyniki liczbowe i wizualizacje najlepszych rozwiązań, wnioski, link do kodu.

2. Opis zaimplementowanych algorytmów w pseudokodzie.

Algorytm zachłanny wymieniający wierzchołki przebiega w następujący sposób:

- Wylosuj nową ścieżkę.
- Dopóki znajdujesz lepsze rozwiązania:
    - Dla każdego punktu:
        - Wylosuj punkt spoza ścieżki jako kandydata do wymiany.
        - Wymień punkt w krawędzi z punktem spoza niej.

            - Zaktualizuj zbiory wierzchołków i zapisz otrzymaną ścieżkę w miejsce aktualnie wykorzystywanej, jeśli ma mniejszą od niej długość.
    - Zwróć najkrótszą otrzymaną ścieżkę.

Algorytm zachłanny wymieniający krawędzie przebiega w następujący sposób:
- Wylosuj nową ścieżkę.
- Dopóki znajdujesz lepsze rozwiązanie:
    - Dla każdej krawędzi:
        - Wylosuj drugą krawędź do wymiany, inną niż rozpatrywana.
        - Zamień miejscami punkt końcowy pierwszej krawędzi oraz punkt początkowy krawędzi drugiej.
        - Zapisz otrzymaną ścieżkę w miejsce aktualnie wykorzystywanej, jeśli ma mniejszą od niej długość.
- Zwróć najkrótszą otrzymaną ścieżkę.

Algorytm stromy wymieniający wierzchołki przebiega w następujący sposób:
- Wylosuj nową ścieżkę.
- Dla każdego punktu ze ścieżki (w losowej kolejności):
    - Dla każdego punktu spoza ścieżki:
        - Wymień punkt w krawędzi z punktem spoza niej.
        - Zaktualizuj zbiory wierzchołków i zapisz otrzymaną ścieżkę w miejsce aktualnie wykorzystywanej, jeśli ma mniejszą od niej długość.
- Zwróć najkrótszą otrzymaną ścieżkę.

Algorytm stromy wymieniający krawędzie przebiega w następujący sposób:
- Wylosuj nową ścieżkę.
- Dla każdej krawędzi w ścieżce (w losowej kolejności):
    - Dla każdej innej krawędzi w ścieżce:
        - Zamień miejscami punkt końcowy pierwszej krawędzi oraz punkt początkowy krawędzi drugiej.
        - Zapisz otrzymaną ścieżkę w miejsce aktualnie wykorzystywanej, jeśli ma mniejszą od niej długość.
- Zwróć najkrótszą otrzymaną ścieżkę.

3. Opis sposobu randomizacji kolejności przeglądania w wersji stromej.

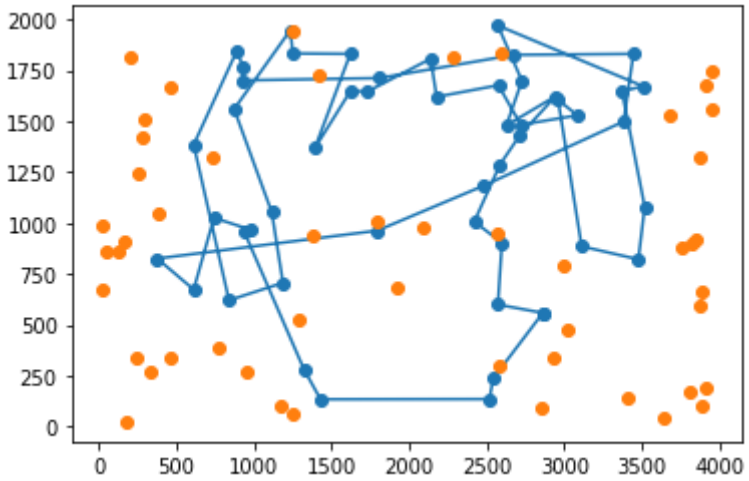
Do losowania kolejności przeglądanych wierzchołków wykorzystana została funkcja *np.random.shuffle()* z biblioteki numpy. W pierwszej kolejności tworzona jest tablica kolejnych liczb całkowitych od 0 do n-1, gdzie n jest rozmiarem tablicy zawierającej punkty ścieżki. W następnej kolejności z pomocą funkcji *shuffle()* pseudolosowo zmieniana jest kolejność liczb w tablicy. Ostatecznie przestawione liczby określają kolejność, w jakiej wierzchołki ścieżki będą rozpatrywane.

4. Wyniki eksperymentu obliczeniowego.

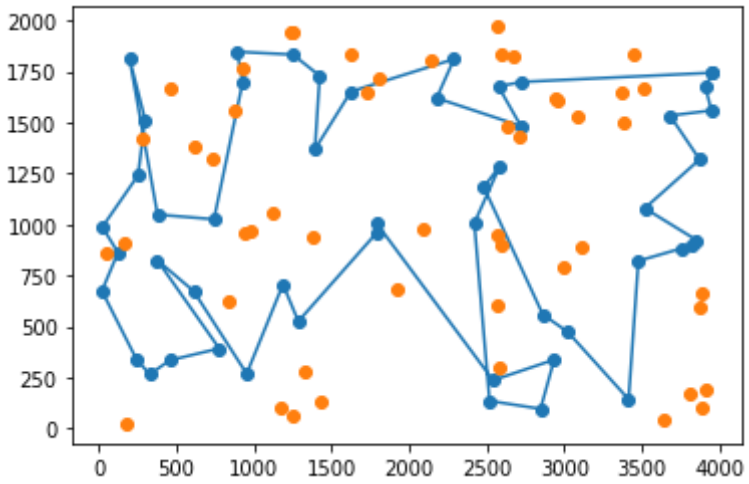
	Algorytm zachłanny 1	Algorytm zachłanny 2	Algorytm stromy 1	Algorytm stromy 2
Najkrótsza ścieżka	21677	18972	22122	18694
Średnia ścieżka	30083.47	26665.23	34453.95	25981.18
Najdłuższa ścieżka	39536	37696	47370	32709
Minimalny czas [ms]	89	110	690	711
Średni czas [ms]	239	254	798	854
Maksymalny czas [ms]	488	501	1269	1610

Tabela 1. Wyniki długości ścieżek i czasu działania na wykonywanych algorytmów

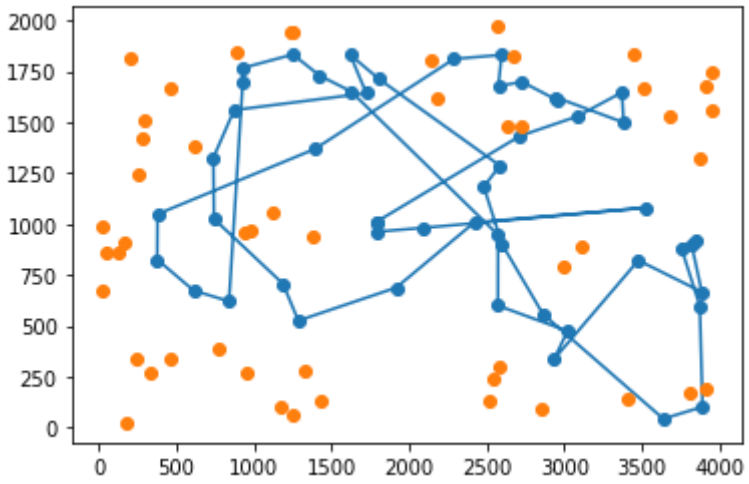
5. Wizualizacje najlepszych rozwiązań dla każdej kombinacji.



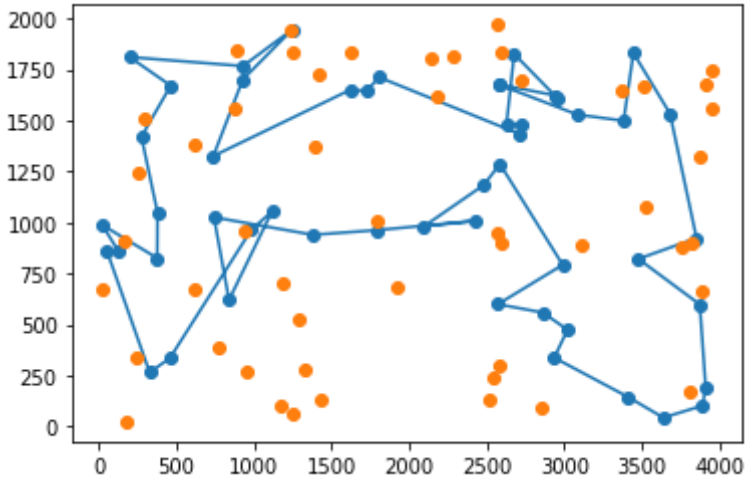
Rysunek 1. Wizualizacja dla pierwszego algorytmu zachłannego



Rysunek 2. Wizualizacja dla drugiego algorytmu zachłannego



Rysunek 3. Wizualizacja dla pierwszego algorytmu stromego



Rysunek 4. Wizualizacja dla drugiego algorytmu stromego

6. Wnioski.

Z przeprowadzonych testów wynika, iż algorytm zachłanny jest średnio o wiele szybszy niż algorytm stromy. Średnia długość ścieżki dla rozwiązań tej samej wersji jest porównywalna, zwłaszcza dla zachłannego i stromego wariantu drugiego. Otrzymane dane pozwalają stwierdzić, iż najlepszym algorytmem z przedstawionych jest algorytm zachłanny w wersji drugiej.

7. Kod programu (np. w postaci linku).

https://github.com/Kurkum/AEM