Algorytmy Metaheurystyczne
Lista 0 - TSP

Rafał Włodarczyk

10.03.2026

Spis treści

1 Model

Model TSP zakłada graf pełny $G = (V, E)$ z wagami. Wiemy, że G ma $H = (n-1)!$ cykli Hamiltona, ponieważ mamy $n!$ możliwych permutacji wierzchołków, po których chodzimy - a każdy cykl możemy zacząć z dowolnego z nich - zatem ostatecznie mamy $H = \frac{n!}{n} = (n-1)!$ cykli Hamiltona.

2 Źródło dla danych

Dane do programu są wzięte ze strony internetowej:

`https://www.math.uwaterloo.ca/tsp/world/countries.html`

oraz są dostępne do pobrania w formacie `.tsp` za pomocą skryptu `get_data.sh`.

3 Zadanie 1 - Metoda Losowa

Zdefiniujmy odległość Euklidesową jako:

$$d((x_1, y_1), (x_2, y_2)) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

W zadaniu użyłem `std::shuffle` do losowania permutacji wierzchołków.
W pliku `task1.cpp` zapisany został kod źródłowy programu wczytującego z dane grafów oraz wyznaczającego statystyki 1000 permutacji, które prezentują się następująco:

Graf	Min	Min10Avg	Min50Avg
western_sahara.tsp	77764	91586	82601
djibouti.tsp	20884	23528	21335
qatar.tsp	81614	86907	81237
uruguay.tsp	1555856	1581006	1495132
zimbabwe.tsp	2243300	2283067	2160078

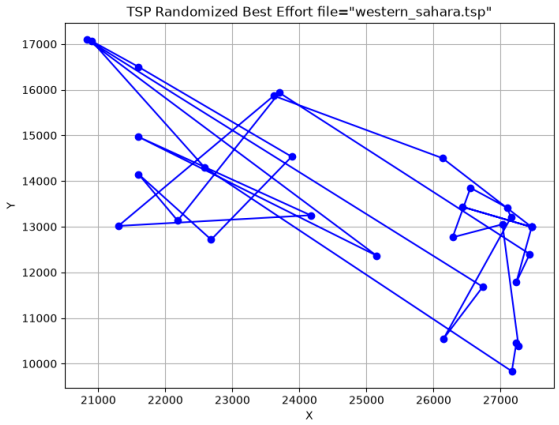
Wartości zostały wyznaczane w precyzji `double` zaokrąglone do najbliższej liczby całkowitej. Wyjaśnienie kolumn (wynik w jednostce wagi krawędzi - odległości w płaszczyźnie euklidesowej):

- **Min** oznacza najmniejszą całkowitą wagę spośród 1000 permutacji
- **Min10Avg** średnia z liczb - najmniejsza całkowita waga w kolejnych 10 losowaniach.
- **Min50Avg** średnia z liczb - najmniejsza całkowita waga w kolejnych 50 losowaniach.

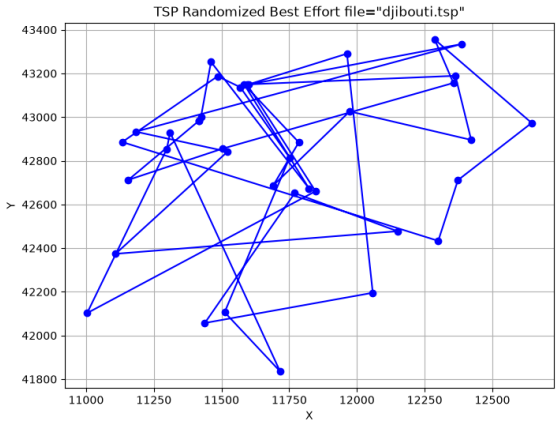
4 Zadanie 2 - Wykresy

Rozwiązanie graficzne obejmuje stworzenie wykresu w płaszczyźnie Euklidesowej, w której poszczególne punkty reprezentują wierzchołki grafu, a linie łączące te punkty reprezentują krawędzie, łączymy krawędzie w kolejności permutacji uzyskanej w zadaniu 1, tworząc w ten sposób trasę.

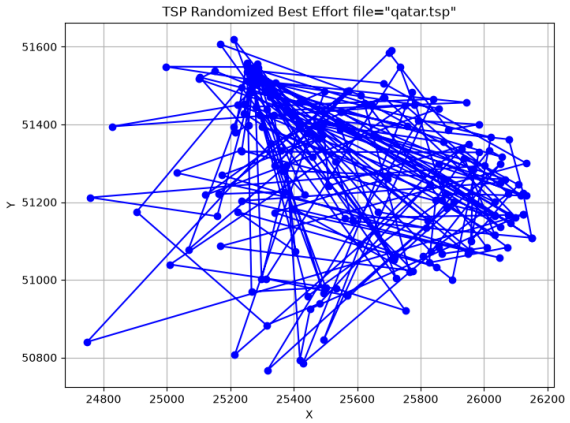
Przedstawiam wykresy dla każdego z grafów, które zostały wygenerowane w pliku `task2.cpp`:



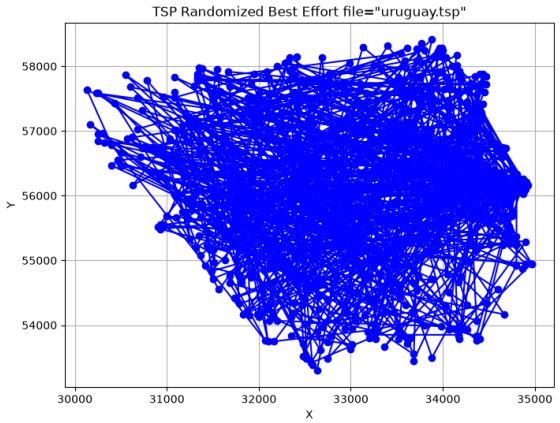
Rysunek 1: Trasa dla grafu western_sahara.tsp



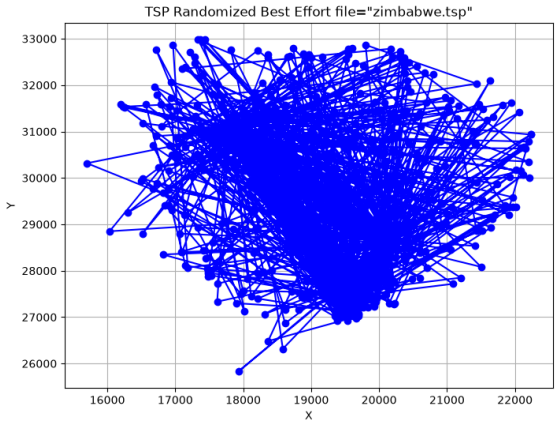
Rysunek 2: Trasa dla grafu djibouti.tsp



Rysunek 3: Trasa dla grafu qatar.tsp



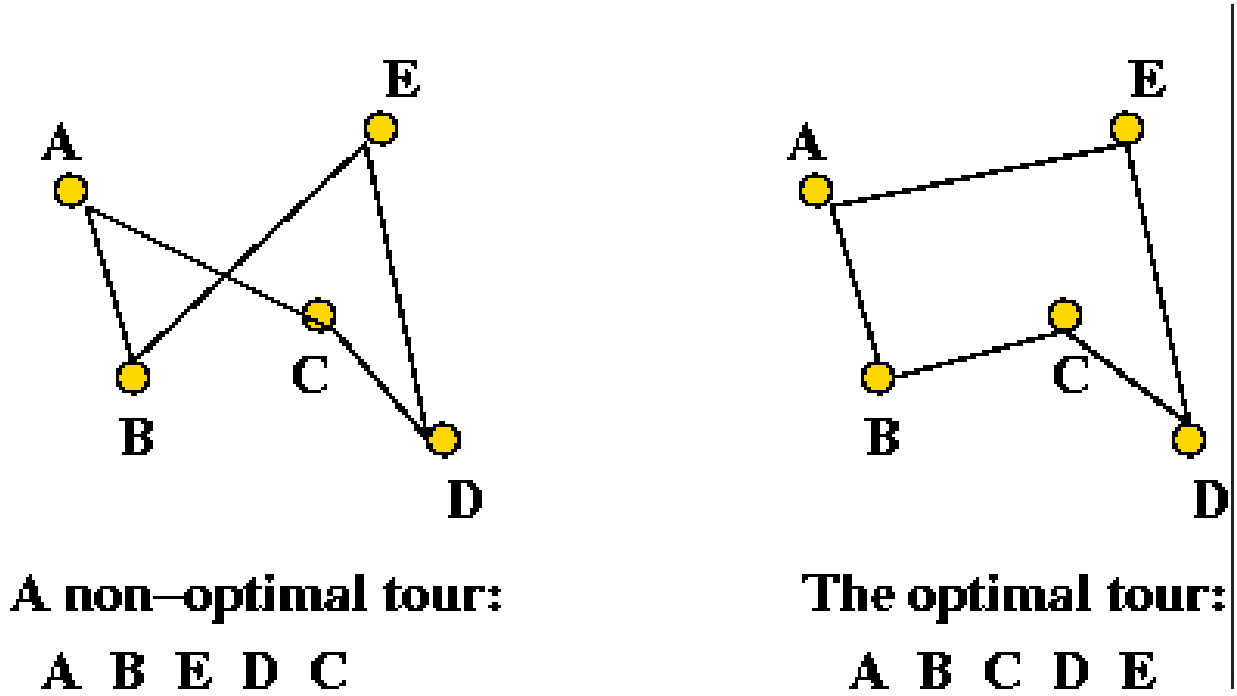
Rysunek 4: Trasa dla grafu uruguay.tsp



Rysunek 5: Trasa dla grafu zimbabwe.tsp

4.1 Wniosek

Zobaczmy, że jeżeli bierzemy wierzchołki losowo to pomijamy trywialny fakt, iż para krzyżujących się krawędzi może zostać zastąpiona przez parę nie krzyżującą się, o mniejszej całkowitej wadze.



Rysunek 6: Przykład ścieżki

Najprościej można to uzasadnić z nierówności trójkąta - ścieżka $(a, c), (c, b)$ przechodząca przez dodatkowy wierzchołek na pewno będzie większy niż bezpośrednie przejście (a, b).

5 Zadanie 3 - MST

Do wykonania zadanie posłużyłem się algorytmem Kruskala, który był szerzej omawiany na przedmiocie Algorytmy i Struktury Danych. Algorytm Kruskala działa w złożoności $\mathcal{O}(E \cdot \log V)$, oraz jest opisywany w następujących krokach:

1. Utwórz L - docelowa struktura MST, oraz ustal każdy wierzchołek jako osobne drzewo.
2. Utwórz zbiór S zawierający wszystkie krawędzie grafu G.
3. Dopóki S nie jest pusty oraz L nie jest MST
4. Wybierz i usuń z S krawędź o minimalnej wadze.
5. Jeżeli krawędź łączyła dwa różne drzewa to dodają ją do lasu L, aby połączyła dwa poddrzewa w jedno.
6. W przeciwnym przypadku odrzuć, wróć do punktu 3-go.

W pliku `task3.cpp` znajduje się kod źródłowy programu, który wczytuje dane grafów, wyznacza minimalne drzewa rozpinające (MST) oraz oblicza wagi tych drzew.

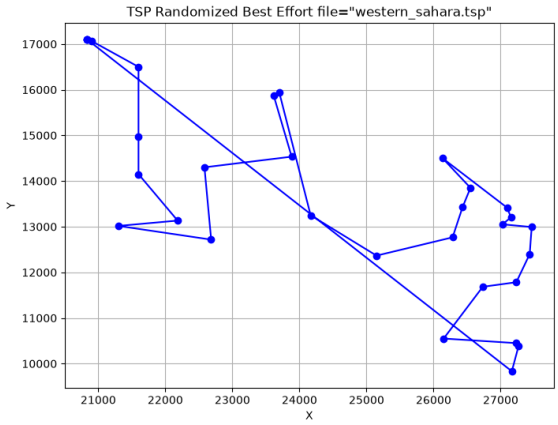
Wyniki prezentują się następująco:

Graf	**Liczba wierzchołków (n)**	**Waga MST**
western_sahara.tsp	29	21751
djibouti.tsp	38	5831
qatar.tsp	194	8028
uruguay.tsp	734	69876
zimbabwe.tsp	929	82873

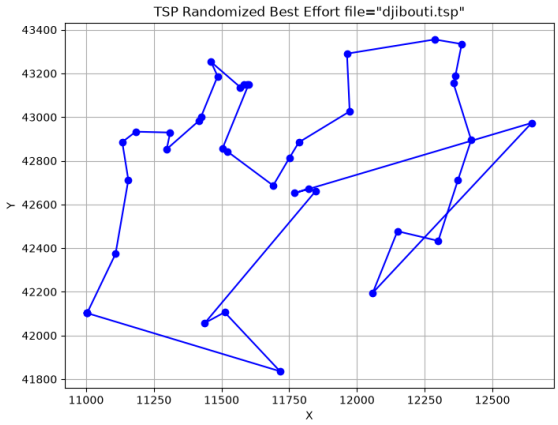
6 Zadanie 4 - MST do TSP z Wykresami

Możemy wykonać DFS na MST, aby uzyskać trasę rozwiązującą TSP, uzyskując w ten sposób 2-aproksymację rozwiązania. W pliku `task4.cpp` znajduje się kod źródłowy programu.

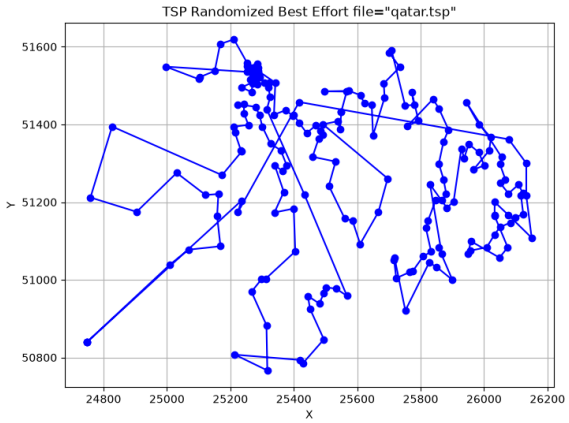
Cykl komiwojażera powstaje w wyniku wypisania wierzchołków w kolejności odwiedzania podczas przeszukiwania w głąb. To znaczy że wykorzystujemy MST do stworzenia grafu, po którym następnie wykonujemy DFS, a kolejność odwiedzania wierzchołków podczas DFS tworzy trasę TSP.



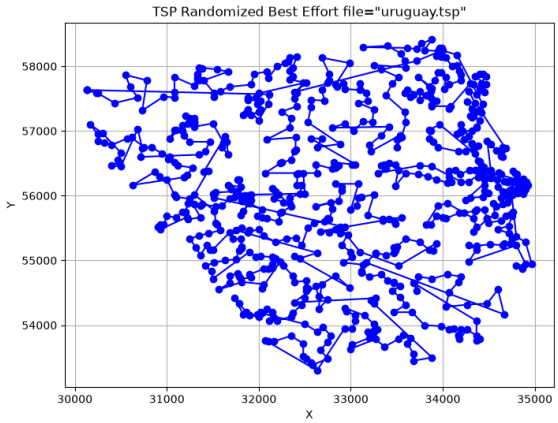
Rysunek 7: Trasa dla grafu western_sahara.tsp



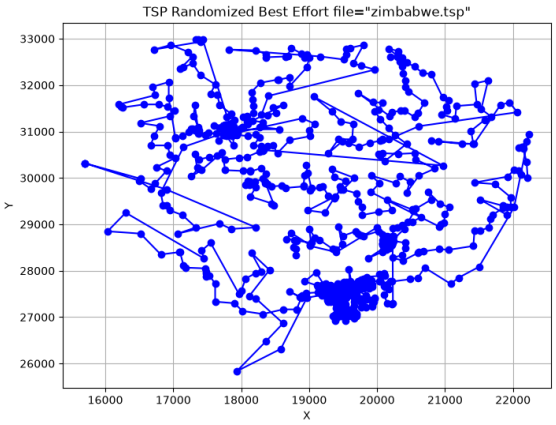
Rysunek 8: Trasa dla grafu djibouti.tsp



Rysunek 9: Trasa dla grafu qatar.tsp



Rysunek 10: Trasa dla grafu uruguay.tsp



Rysunek 11: Trasa dla grafu zimbabwe.tsp

7 Porównanie wyników

Porównanie następuje pomiędzy metodą **Random** (zadanie 1), **MST 2-approx** (zadanie 4), **OPT** - informacje na stronie internetowej.

Graf	**Random**	**MST 2-approx**	**OPT**
western_sahara.tsp	77764	35009	27603
djibouti.tsp	20884	8739	6656
qatar.tsp	81614	12510	9352
uruguay.tsp	1555856	107340	79114
zimbabwe.tsp	2243300	131828	95345

Porównanie wyników pokazuje, że metoda MST 2-aproksymacji daje wyniki znacznie bliższe wartościom optymalnym (OPT) w porównaniu do metody losowej. Widzimy również doświadczalnie, że dla każdego przypadku metody **MST 2-approx** wynik nie przekracza dwu-krotności wartości optymalnej, co jest zgodne z teoretyczną gwarancją tej aproksymacji.