

Zadanie 1. Heurystyki konstrukcyjne

Oskar Kiliańczyk 151863 & Wojciech Kot 151876

1 Opis zadania

Podczas zajęć rozważamy zmodyfikowany problem komiwojażera. Początkowo, obliczamy macierz odległości pomiędzy danymi miastami. Obliczona macierz odległości między wierzchołkami grafu będzie podstawą dla każdego algorytmu, a celem jest wyznaczenie dwóch rozłącznych zamkniętych ścieżek (cykli), z których każda zawiera 50% wierzchołków. Jeśli liczba wierzchołków jest nieparzysta, jedna ścieżka zawiera jeden wierzchołek więcej. Kryterium optymalizacji jest minimalizacja łącznej długości obu cykli.

Rozważane instancje problemu pochodzą z biblioteki TSPLib, a są to kroa200 oraz krob200. Są to instancje dwuwymiarowe euklidesowe, w których każdemu wierzchołkowi przypisane są współrzędne na płaszczyźnie. Odległość między wierzchołkami liczona jest jako odległość euklidesowa, zaokrąglana do najbliższej liczby całkowitej. W implementacji algorytmów wykorzystywana będzie wyłącznie macierz odległości, co zapewnia możliwość zastosowania kodu do innych instancji problemu.

2 Zaimplementowane algorytmy

2.1 Algorytm zachłanny - metoda najbliższego sąsiada

Algorytm ten wykorzystuje funkcję znajdującą najbliższego sąsiadę dla danego wierzchołka (miasta). Działa ona w następujący sposób:

dla każdego miasta, sprawdza czy zostało już odwiedzone

jeśli nie, to sprawdza czy dystans jest mniejszy od dystansu z danego miasta do obecnie zapamiętanego jako najbliższe

jeśli jest bliższe niż obecnie pamiętane jako najbliższe, zapamiętuje je jako najbliższe

jeśli nie ma żadnego miasta obecnie pamiętanego jako najbliższe, przypisuje to miasto

Główny algorytm natomiast, wygląda następująco:

Przydziela wierzchołki startowe do cykli pierwszego i drugiego

Tworzy tablice indeksów miast, zaznaczając wszystkie poza startowymi jako nieodwiedzone dopóki istnieją jakieś nieodwiedzone miasta, powtarza następujące 4 kroki:

 Znajduje najbliższego nieodwiedzonego sąsiadę do ostatniego wierzchołka cyklu 1.

 dodaje go do cyklu1 oraz zapisuje w tablicy jako odwiedzony

 Znajduje najbliższego nieodwiedzonego sąsiadę do ostatniego wierzchołka cyklu 2.

 dodaje go do cyklu2 oraz zapisuje w tablicy jako odwiedzony

Kiedy już nie ma żadnych nieodwiedzonych miast, dopisuje na koniec cykli ich wierzchołki startowe

Zwraca oba cykle jako znalezione ścieżki

2.2 Algorytm zachłanny - metoda rozbudowy cyklu

Algorytm ten korzysta z funkcji znajdowania najlepszego wstawienia, a działa ona w następujący sposób:

Przyjmuje jako argumenty obecny cykl, macierz dystansów, oraz tablice indeksów odwiedzonych miast

Tworzy listę możliwości (nieodwiedzonych wierzchołków)

ustawia najtańszy koszt na maksymalnie dużą wartość

dla każdej możliwości z listy możliwości:

 dla każdego możliwego wstawienia w cykl:

 - oblicza wzrost dystansu, jaki spowoduje wstawienie (a więc przy wstawianiu między a

 - zapisuje najmniejszy znaleziony wzrost dystansu oraz miejsce jego wstawienia

 jeśli koszt wstawienia obecnie znalezionego wierzchołka jest mniejszy niż obecnie pamiętany

Po przejrzeniu wszystkich możliwości zwraca parę <wierzchołek, miejsce wstawienia>

Dwukrotnie przydziela wierzchołki startowe do cykli pierwszego i drugiego jako początek i koniec

Tworzy tablice indeksów miast, zaznaczając wszystkie poza startowymi jako nieodwiedzone

dopóki istnieją jakieś nieodwiedzone miasta, powtarza następujące 7 kroków:

 Znajduje najtańsze wstawienie, czyli parę <wierzchołek, miejsce w cyklu> dla cyklu 1

Wstawia w odpowiednie miejsce cyklu 1 znaleziony wierzchołek
Zapisuje wierzchołek jako już odwiedzony
Znajduje najtańsze wstawienie dla cyklu 2
Jeśli ono nie istnieje (np. bo ostatni wierzchołek został już wstawiony) to kończy petle
Wstawia w odpowiednie miejsce cyklu 2 znaleziony wierzchołek
Zapisuje wierzchołek jako już odwiedzony
Kiedy już nie ma żadnych nieodwiedzonych miast zwraca oba cykle jako znalezione ścieżki

2.3 Algorytm z żalem - metoda rozbudowy cyklu

kodekoden

2.4 Algorytm z żalem - metoda rozbudowy cyklu z żalem ważonym

kodekoden

2.5 Własny jakiś dziki algorytm, jak nam się będzie chciało, bo imo bym zrobił coś

kodekoden

3 Wyniki eksperymentu obliczeniowego

3.1 Algorytm zachłanny - metoda najbliższego sąsiada

Graficzna reprezentacja wyników dla algorytmu zachłannego metoda najbliższego sąsiada przedstawiono na ??

3.2 Algorytm zachłanny - metoda rozbudowy cyklu

Graficzna reprezentacja wyników dla algorytmu zachłannego metoda rozbudowy cyklu przedstawiono na ??

4 Wnioski

Żal nie działa, chociaż byśmy chcieli, ale żal rozwiązuje zwykły TSP. Może w połączeniu z algorytmem klastrowym, jakimś k-means zadziałałoby lepiej, ale strasznie krzywdzące dla niego jest też to że ścieżki MUSZA być równej długości

5 Link do repo

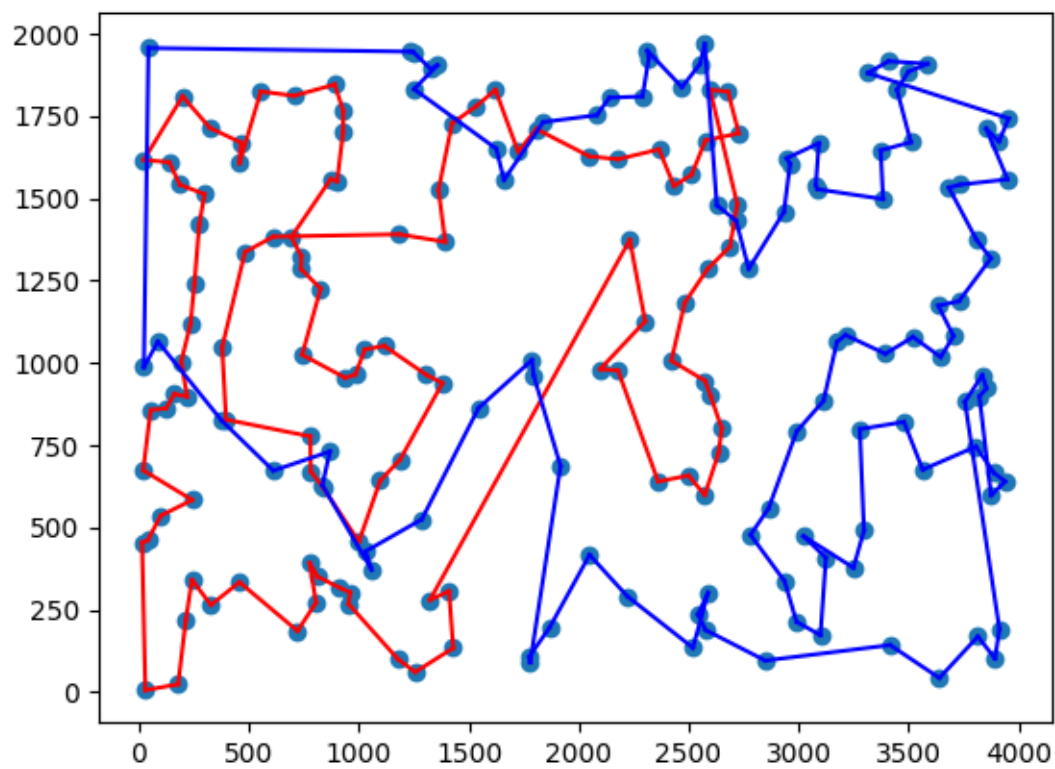


Figure 1: Instancja kroA200

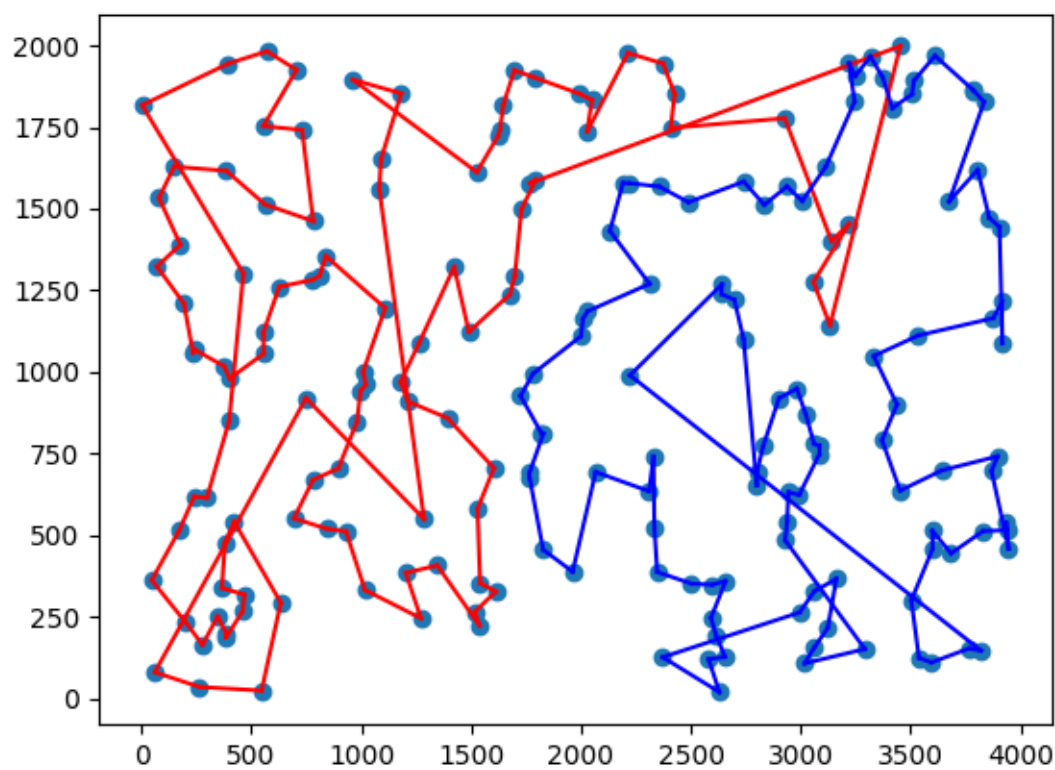


Figure 2: Instancja kroB200

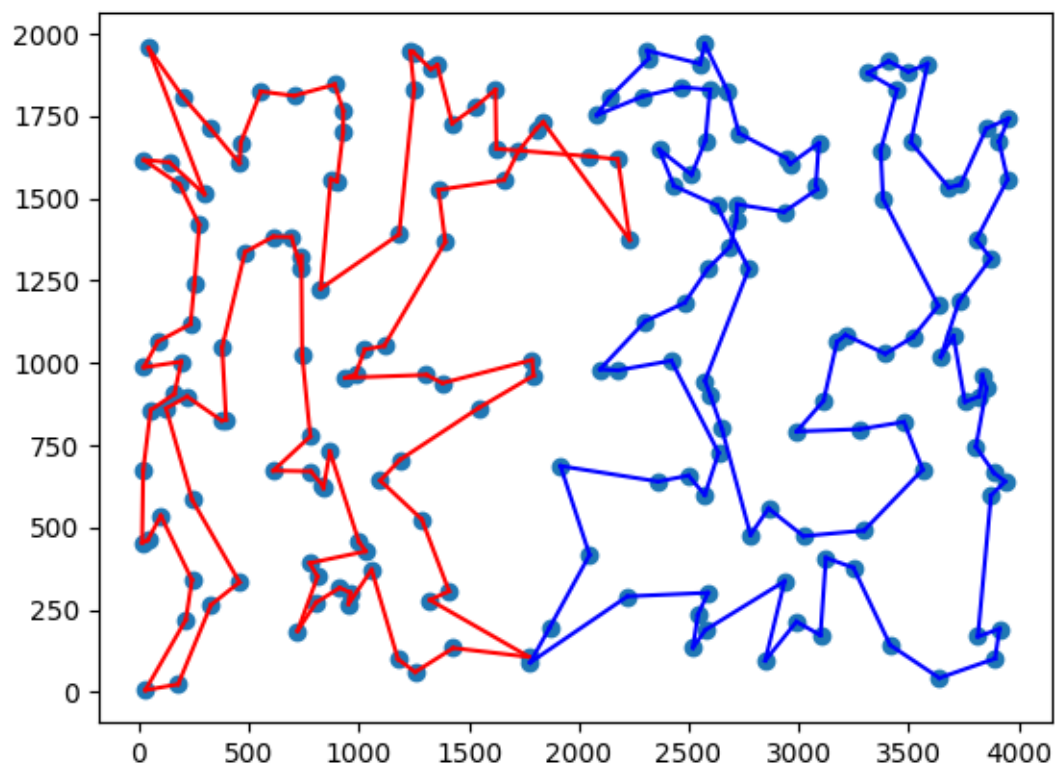


Figure 3: Instancja kroA200

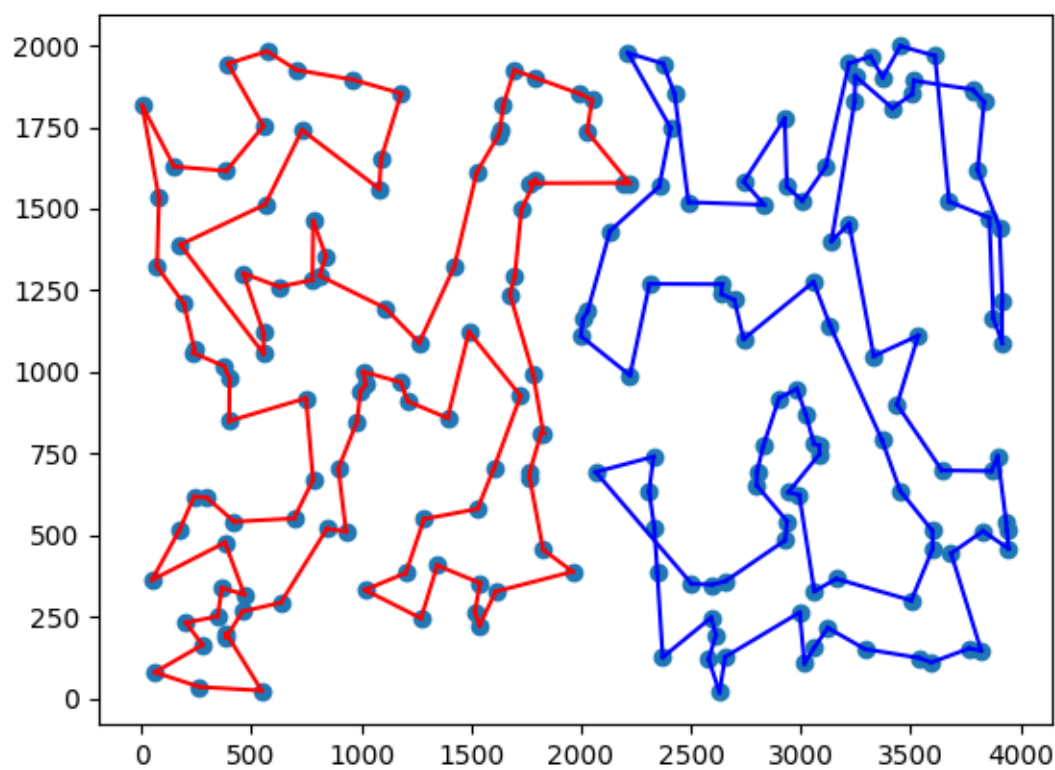


Figure 4: Instancja kroB200