Zadanie 1. Heurystyki konstrukcyjne

Oskar Kiliańczyk 151863 & Wojciech Kot 151876

1 Opis zadania

Podczas zajeć rozważamy zmodyfikowany problem komiwojażera. Poczatkowo, obliczamy macierz odległości pomiedzy danymi miastami. Obliczona macierz odległości miedzy wierzchołkami grafu bedzie podstawa dla każdego algorytmu, a celem jest wyznaczenie dwóch rozłacznych zamknietych ścieżek (cykli), z których każda zawiera 50% wierzchołków. Jeśli liczba wierzchołków jest nieparzysta, jedna ścieżka zawiera jeden wierzchołek wiecej. Kryterium optymalizacji jest minimalizacja łacznej długości obu cykli.

Rozważane instancje problemu pochodza z biblioteki TSPLib, a sa to kroa200 oraz krob200. Sa to instancje dwuwymiarowe euklidesowe, w których każdemu wierzchołkowi przypisane sa współrzedne na płaszczyźnie. Odległość miedzy wierzchołkami liczona jest jako odległość euklidesowa, zaokraglana do najbliższej liczby całkowitej. W implementacji algorytmów wykorzystywana bedzie wyłacznie macierz odległości, co zapewnia możliwość zastosowania kodu do innych instancji problemu.

2 Zaimplementowane algorytmy

2.1 Algorytm zachłanny - metoda najbliższego sasiada

Algorytm ten wykorzystuje funkcje znajdujaca najbliższego sasiada dla danego wierzchołka (miasta) Działa ona w nastepujacy sposób:
dla każdego miasta, sprawdza czy zostało już odwiedzone
jeśli nie, to sprawdza czy dystans jest mniejszy od dystansu z danego miasta do obecnie zapamietanego jako najbliższe
jeśli jest bliższe niż obecnie pamietane jako najbliższe, zapamietuje je jako najbliższe
jeśli nie ma żadnego miasta obecnie pamietanego jako najbliższe, przypisuje to miasto

Główny algorytm natomiast, wyglada nastepujaco:

```
Przydziela wierzchołki startowe do cykli pierwszego i drugiego
Tworzy tablice indeksów miast, zaznaczajac wszystkie poza startowymi jako nieodwiedzone
dopóki istnieja jakieś nieodwiedzone miasta, powtarza nastepujace 4 kroki:
    Znajduje najbliższego nieodwiedzonego sasiada do ostatniego wierzchołka cyklu 1.
    dodaje go do cyklu1 oraz zapisuje w tablicy jako odwiedzony
    Znajduje najbliższego nieodwiedzonego sasiada 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 star
Zwraca oba cykle jako znalezione ścieżki
```

2.2 Algorytm zachłanny - metoda rozbudowy cyklu

Algorytm ten korzysta z funkcji znajdywania najlepszego wstawienia, a działa ona w nastepujacy sposób:

```
Przyjmuje jako argumenty obecny cykl, macierz dystansów, oraz tablice indeksów odwiedzonych mi
Tworzy liste możliwości (nieodwiedzonych wierzchołków)
ustawia najtańszy koszt na maksymalnie duża wartość
dla każdej możliwości z listy możliwości:
    dla każdego możliwego wstawienia w cykl:
         - oblicza wzrost dystansu, jaki spowoduje wstawienie (a wiec przy wstawianiu miedzy a
         - zapisuje najmniejszy znaleziony wzrost dystansu oraz miejsce jego wstawienia
    jeśli koszt wstawienia obecnie znalezionego wierzchołka jest mniejszy niż obecnie pamieta
Po przejrzeniu wszystkich możliwości zwraca pare <wierzchołek, miejsce wstawienia>
```

```
Dwukrotnie przydziela wierzchołki startowe do cykli pierwszego i drugiego jako poczatek i kon
Tworzy tablice indeksów miast, zaznaczajac wszystkie poza startowymi jako nieodwiedzone
dopóki istnieja jakieś nieodwiedzone miasta, powtarza nastepujace 7 kroków:
    Znajduje najtańsze wstawienie, czyli pare <wierzchołek, miejsce w cyklu> dla cyklu 1
```

```
    Wstawia w odpowiednie miejsce cyklu 1 znaleziony wierzcho␣lek
    Zapisuje wierzcho␣lek jako juz odwiedzony
    Znajduje najtańsze wstawienie dla cyklu 2
    Jeśli ono nie istnieje (np. bo ostatni wierzcho␣lek zosta␣l juz wstawiony) to kończy petle
    Wstawia w odpowiednie miejsce cyklu 2 znaleziony wierzcho␣lek
    Zapisuje wierzcho␣lek jako juz odwiedzony
Kiedy juz nie ma zadnych nieodwiedzonych miast zwraca oba cykle jako znalezione ścieżki
```

2.3 Algorytm z żalem - metoda rozbudowy cyklu

```
kodkodkod
```

2.4 Algorytm z żalem - metoda rozbudowy cyklu z żalem ważonym

```
kodkodkod
```

2.5 Własny jakiś dziki algorytm, jak nam sie bedzie chciało, bo imo bym zrobił coś

```
kodkodkod
```

3 Wyniki eksperymentu obliczeniowego

ccccccc i wizualizacje

4 Wnioski

Żal nie działa, chociaż byśmy chcieli, ale żal rozwiazuje zwykły TSP. Może w połaczeniu z algorytmem klastrujacym, jakimś k-means zadziałałby lepiej, ale strasznie krzywdzace dla niego jest też to że ścieżki MUSZA być równej długości

5 Link do repo