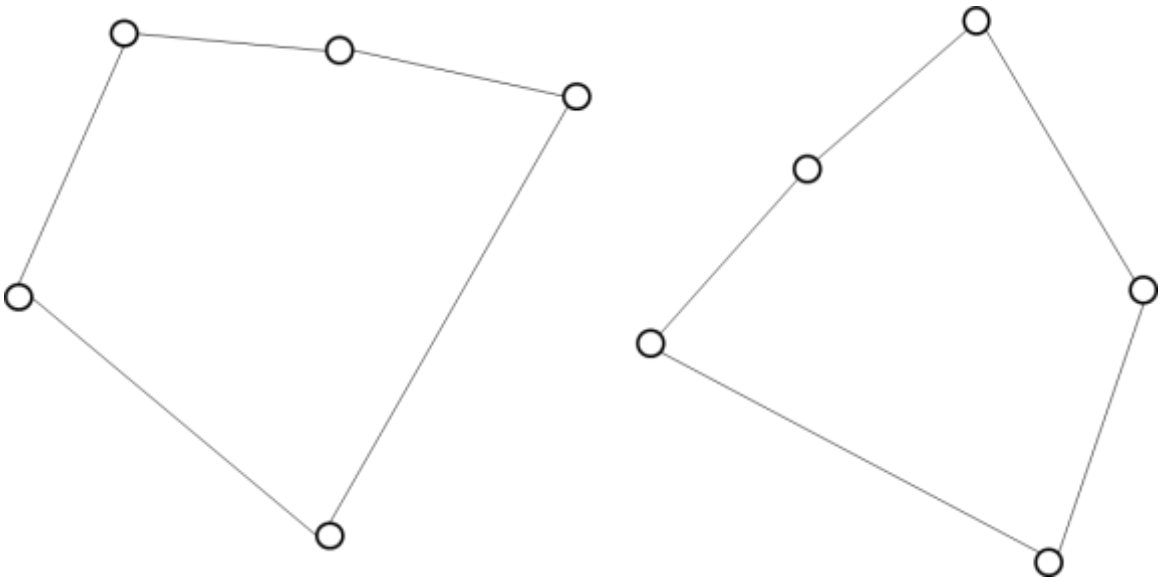
## Opis problemu optymalizacji

Podczas zajęć laboratoryjnych będziemy wykorzystywać zmodyfikowany problem komiwojażera. Dany jest zbiór wierzchołków i symetryczna macierz odległości pomiędzy dowolną parą wierzchołków. Należy ułożyć dwie rozłączne zamknięte ścieżki (cykle), każdą zawierającą 50% wierzchołków (jeżeli liczba wierzchołków w instancji nie jest parzysta, to pierwsza ścieżka zawiera jeden wierzchołek więcej), minimalizując łączną długość obu ścieżek.

Przykład rozwiązania:



### Instancje

Rozważamy instancje kroa100 i krob100 z biblioteki TSPLib. Są to dwuwymiarowe instancje euklidesowe, tj. dla każdego wierzchołka podane są dwie współrzędne, a odległość pomiędzy wierzchołkami jest odległością euklidesową. Ważna uwaga, odległość jest zaokrąglana do liczby całkowitej stosując zaokrąglanie matematyczne. Proszę jednak, aby dalszy kod wykorzystywał jedynie macierz odległości, tj. aby był w pełni stosowalny do innych instancji, które będą zdefiniowane jedynie przez macierze odległości.

# Zadanie 1. Heurystyki konstrukcyjne

## Opis zadania

W ramach zadania należy:

- Zaimplementować wczytywanie instancji kroa100 i krob100 (w jednym z formatów w jakim są dostępne) i obliczanie macierzy odległości.
- Zaimplementować algorytm zachłanny (greedy) inspirowany metodą najbliższego sąsiada (nearest neighbor) dla klasycznego problemu komiwojażera dostosowując go do rozważanego problemu. Stosujemy wersję, w której rozważa się wszystkie możliwe miejsca wstawienia kolejnych wierzchołków (nie tylko na końcu). Np. wybieramy losowo dwa startowe wierzchołki dla dwóch cykli i rozbudowujemy je zgodnie z zasadami dla klasycznego problemu komiwojażera aż do osiągnięcia cykli o zadanej liczbie wierzchołków. Można też np. pierwszy wierzchołek wybrać losowo, a drugi najodleglejszy od pierwszego.
- Zaimplementować algorytm zachłanny (greedy) inspirowany metodą rozbudowy cyklu (greedy cycle) dla klasycznego problemu komiwojażera dostosowując go do rozważanego problemu.
- Zaimplementować algorytm typu regret heuristics (z żalem) na bazie algorytmu inspirowanego metodą rozbudowy cyklu – stosujemy 2-regret (2-żal).
- Opcjonalnie można zaimplementować jeszcze inną heurystykę konstrukcyjną.
- Wykonać eksperymenty obliczeniowe. Na każdej instancji każdy algorytm należy uruchomić 100 razy wykorzystując różne wierzchołki startowe.

Heurystyka najbliższego sąsiada (nearest neighbor) dla problemu komiwojażera

wybierz (np. losowo) wierzchołek startowy

**powtarzaj**

dodaj do rozwiązania wierzchołek (i prowadzącą do niego krawędź) najbliższy ostatnio dodanemu

**dopóki** nie zostały dodane wszystkie wierzchołki

Dodaj krawędź z ostatniego do pierwszego wierzchołka

Istnieje też inna wersja, w rozważa się wszystkie możliwe miejsca wstawienia kolejnych wierzchołków (nie tylko na końcu)

Metoda rozbudowy cyklu (greedy cycle)

wybierz (np. losowo) wierzchołek startowy

wybierz najbliższy wierzchołek i stwórz z tych dwóch wierzchołków niepełny cykl

**powtarzaj**

wstaw do bieżącego cyklu w najlepsze możliwe miejsce wierzchołek powodujący najmniejszy wzrost długości cyklu

**dopóki** nie zostały dodane wszystkie wierzchołki

Heurystyki zachłanne oparte na żalu – regret heuristics

- Załóżmy, że element możemy wstawiać do rozwiązania w różne miejsca (np. wierzchołki w różne miejsca trasy TSP)
- Np. dwa elementy a i b:
- Koszty wstawienia a:
- 1, 2, 3, 4, 5
- Koszty wstawienia b:
- 5, 9, 10, 12, 15
- Zgodnie z regułą zachłanną wybralibyśmy a (1 < 5)
- Wtedy jednak możemy zablokować sobie możliwość wstawienia b przy koszcie 5 a zostanie nam jedynie miejsce z kosztem 9
- k-żal (k-regret) to suma różnic pomiędzy najlepszym, a k kolejnymi opcjami wstawienia
- 2-żal różnica pomiędzy pierwszą a drugą opcją
- Wybieramy element o największym żalu i wstawiamy go w najlepsze miejsce
- Możemy też ważyć żal z regułą zachłanną (oceną pierwszej opcji)

## Sprawozdanie

W sprawozdaniu należy umieścić:

- Krótki opis zadania.
- Opis wszystkich zaimplementowanych algorytmów w pseudokodzie. Uwaga pseudokod to nie jest jednozdaniowy, deklaratywny opis. Nie jest to także docelowy kod.
- Wyniki eksperymentu obliczeniowego. Dla każdej kombinacji instancja/algorytm należy podać wartość średnią, minimalną i maksymalną. Spodziewamy się, że czasy wykonania tych algorytmów powinny być pomijalne, więc nie musimy ich raportować. Sugerowany format tabeli:

| | Instancja 1 | Instancja 2 |
|---|---|---|
| Metoda 1 | średnia (min – max) | średnia (min – max) |
| Metoda 2 | średnia (min – max) | średnia (min – max) |

- Wizualizacje najlepszych rozwiązań dla każdej kombinacji podobne do powyższego przykładu rozwiązania
- Wnioski.
- Kod programu (np. w postaci linku).