Kierunek: Informatyka, semestr V
Specjalność: Aplikacje Internetowe i Mobilne
Rok akademicki: 2022/2023

Inteligencja obliczeniowa
LABORATORIUM

Zajęcia 4.
Algorytmy przeszukiwania lokalnego (*local search*)

CEL ZAJĘĆ

Zapoznanie studentów z grupą algorytmów przeszukiwania lokalnego, ich implementacja dla wybranych problemów optymalizacji ciągłej oraz dyskretnej, eksperymenty obliczeniowe.

PROBLEM 1

Definicja problemu

Problem komiwojażera (*travelling salesman problem*)

Definicja jak w konspekcie z Zajęć 3 (Problem 2):

Graf $G = (V, E)$, gdzie:

- $V = \{c_1, ..., c_n\}$ – zbiór miast,
- E – zbiór krawędzi odpowiadających połączeniom między miastami, gdzie określona jest odległość $d_{ij} \in N$ między każdą parą miast c_i, $(c_j \in V)$.

Pytanie:
Znaleźć najkrótszą drogę łączącą wszystkie miasta należące do V tak, aby każde miasto było odwiedzone dokładnie jeden raz.

Algorytm przeszukiwania lokalnego i jego implementacja

Opis
Wykorzystamy algorytm przeszukiwania lokalnego (*local search*) o następującym pseudokodzie:

```
algorytm LocalSearch_1
{
    local = FALSE
    vc = wybierz_rozwiazanie_poczatkowe()
    ocen(vc)
    do
    {
        vn = wybierz_rozwiazanie_z_otoczenia(vc)
        If (f(vn) jest lepsze od f(vc)
            vc = vn;
        else
            local = TRUE
    }
    while not local
}
```

Kodowanie rozwiązania

Wektor zmiennych $x = (x_0, x_1, ..., x_{n-1})$, $x_i \in [0, n)$ tworzący permutację (reprezentacja ścieżkowa). Przykładowo dla grafu z 6 miastami x = [0, 2, 4, 3, 1, 5] oznacza znalezioną trasę 0-2-4-3-1-5-0.

Funkcja celu

Suma odległości między odwiedzanymi miastami (min).

Ograniczenia

Wszystkie miasta muszą zostać odwiedzone i każde miasto odwiedzamy dokładnie jeden raz.

Eksperyment

Celem eksperymentu jest znalezienie rozwiązania przybliżonego z wykorzystaniem powyższego algorytmu heurystycznego. Eksperyment przeprowadź dla zbioru danych testowych zawartych w pliku `berlin52.tsp` z biblioteki TSPLIB (http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsp/index.html). (podobnie jak na Zajęciach 2, 3).

Uwaga! Zakładamy, że graf jest pełny. Co to oznacza w sensie struktury danych do jego reprezentowania (przypomnij sobie metody reprezentowania grafu z przedmiotu *Algorytmy i struktury danych*).

Przyjmij następujące postacie funkcji:

- `wybierz_rozwiazanie_poczatkowe()` – powinna zwracać rozwiązanie początkowe:
  - Uzyskane z miast rozpatrywanych po kolei: 0-1-2-3-…-
  - Losowo wygenerowane rozwiązanie.
- `wybierz_rozwiazanie_z_otoczenia(vc)` – powinna zwracać poprawione rozwiązanie z sąsiedztwa vc, gdzie sąsiedztwo/otoczenie danego rozwiązania definiujemy jako zbiór rozwiązań (tras) powstałych poprzez zmianę kolejności odwiedzania poszczególnych miast (dokładniej: mając permutację *n* liczb dokonujemy zamiany miast na pozycjach *i* oraz *j*, gdzie *i*, *j* = 0, …, *n*-1):
  - Pierwsze uzyskane rozwiązanie lepsze od vc
  - Najlepsze rozwiązanie spośród wszystkich sąsiadów vc

Określ:

a) Rozpatrz powyższe 4 przypadki i określ: jaką trasę znalazłeś i ile wynosi długość tej trasy?
b) Jak jest jakość tego rozwiązania, czyli o ile procent jest ono gorsze od rozwiązania wskazanego jako rozwiązanie najlepsze znane?
c) Porównaj uzyskane rozwiązanie z rozwiązaniem uzyskanym na Zajęciach 2, 3 za pomocą algorytmu konstrukcyjnego najbliższego sąsiada.
d) Wykorzystaj rozwiązanie uzyskane za pomocą algorytmu konstrukcyjnego najbliższego sąsiada jako rozwiązanie początkowe (`wybierz_rozwiazanie_poczatkowe()` i sprawdź jakość uzyskiwanych rozwiązań dla obu przypadków działania funkcji `wybierz_rozwiazanie_z_otoczenia(vc)`.
e) W jaki inny sposób moglibyśmy zdefiniować sąsiedztwo rozwiązania?