Kierunek: Informatyka, semestr V
Specjalność: Aplikacje Internetowe i Mobilne
Rok akademicki: 2023/2024

Inteligencja obliczeniowa
LABORATORIUM

Zajęcia 2, 3.
Algorytmy konstrukcyjne, przeszukiwania zachłannego (przybliżone) c.d.

CEL ZAJĘĆ

Zapoznanie studentów z wybranymi algorytmami przeszukiwania wyczerpującego oraz zachłannego, ich implementacja dla wybranych problemów optymalizacji ciągłej oraz dyskretnej, eksperymenty obliczeniowe.

PROBLEM 1

Definicja problemu

Problem pakowania (*bin packing*) – odmiana problemu plecakowego (*knapsack problem*)

Dane:
- Nieskończona liczba pojemników, każdy o pojemności B (w praktyce będzie to M pojemników, gdzie M – bardzo duża liczba),
- n przedmiotów, każdy o wielkości s_i ($i = 1…n$),

Pytanie:
Jak zapakować wszystkie przedmioty do jak najmniejszej liczby pojemników, nie przekraczając pojemności żadnego pojemnika i zakładając, że każdy element jest przypisany do jednego pojemnika? Określ sposób rozlokowania przedmiotów oraz liczbę wykorzystanych pojemników.

Algorytm zachłanny i jego implementacja

Opis słowny
Algorytm zachłanny zaimplementujemy w pięciu wersjach:

a) *Next-Fit* (NF) – zakłada jeden otwarty pojemnik w danym czasie. Algorytm rozważa przedmioty do zapakowania w pewnym porządku L. Jeżeli aktualnie rozważany przedmiot mieści się do pojemnika, zapakuj go do pojemnika. W przeciwnym przypadku, aktualny pojemnik jest zamykany, otwierany jest następny pojemnik i do niego pakowany jest rozważany przedmiot.
b) *First-Fit* (FF) – umieszcza przedmioty w pewnym porządku L. Dla każdego przedmiotu z L, próbujemy go umieścić w pierwszym pojemniku, który jest w stanie go pomieścić. Jeśli nie znajdę takiego pojemnika, otwieramy nowy pojemnik i przedmiot umieszczamy w nim.
c) *Best-Fit* (BF) – podobny do FF, ale zamiast kolejny przedmiot umieszczać w pierwszym pojemniku, który może go pomieścić, umieszczamy go w pojemniku najbardziej załadowanym, który jednak byłby w stanie pomieścić jeszcze ten przedmiot.
d) *Worst-Fit* (WF) – podobny do BF, przy czym rozważam pojemnik o najmniejszym wypełnieniu (a nie o największym).
e) *First-Fit-Decreasing* (FFD) – podobny do FF, jednakże zanim wystartujemy z procedurą umieszczania obiektów w pojemnikach, obiekty są sortowane w porządku nierosnącym ich wielkości.

Kodowanie rozwiązania

Wektor zmiennych $x = (x_0, x_1, ..., x_{n-1})$, $x_i \in [0, \infty)$, gdzie liczba zapisana na określonej pozycji będzie oznaczała nr pojemnika do którego przydzielamy przedmiot. Przykładowo: x = [3, 2, 4, 4, 1, 0] oznacza, że przedmiot 0 jest w pojemniku nr 3, przedmiot 1 jest w pojemniku nr 2, przedmioty 2 i 3 są w pojemniku nr 4, itp.

Funkcja celu

Liczba wykorzystanych pojemników (min).

Ograniczenia

- Nie przekraczamy pojemności żadnego pojemnika
- Nie dopuszczamy do zapakowana jednego obiektu do więcej niż jednego pojemnika – zapewnione przez kodowanie rozwiązania.

Eksperyment

Celem eksperymentu jest znalezienie rozwiązania przybliżonego z wykorzystaniem algorytmu heurystycznego.
Eksperyment przeprowadź dla zbioru danych testowych zawartych w pliku `binpack1.txt`.
Dane testowe zawarte w tym pliku zapisane są w formacie:
Pierwsza linia: `B n Popt`
gdzie: *B* – pojemność pojemnika, *n* – liczba obiektów, *Popt* – liczba pojemników wykorzystana w najlepszym znanym rozwiązaniu
Kolejne linie (jest ich *n*): s_i (*i* = 1, …, *n*)
gdzie s_i – wielkość każdego obiektu

Określ:

a) Jakie znalazłeś rozwiązanie i ile wykorzystałeś pojemników?
b) Jak jest jakość tego rozwiązania, czyli o ile procent jest ono gorsze od rozwiązania wskazanego jako rozwiązanie najlepsze znane dla tej instancji (48)?

Pytania dodatkowe

a) Jakie inne kodowanie rozwiązania mógłbyś przyjąć? Wskaż jego zalety i wady.

PROBLEM 2

Definicja problemu

Problem komiwojażera (*travelling salesman problem*)

Dane:
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 zachłanny i jego implementacja

Opis słowny
Wykorzystamy algorytm *najbliższego sąsiada* (*best neighbor*), polegający na stopniowym konstruowaniu trasy, poprzez dołączanie kolejnych odwiedzanych miast:

a) Startujemy z dowolnego/ustalonego miasta
b) Każde kolejne miasta dołączamy zgodnie z regułą najbliższego sąsiada, tj. wybieramy najbliższej położone osiągalne bezpośrednio miasto, nieodwiedzone wcześniej.
c) Algorytm kończy się, gdy rozważymy wszystkie miasta lub gdy nie jest możliwe skonstruowanie takiej trasy (brak rozwiązania).

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). Obejrzyj w jakim formacie są zapisane dane.
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*).

Określ:

c) Jaką trasę znalazłeś i ile wynosi długość tej trasy?
d) Jak jest jakość tego rozwiązania, czyli o ile procent jest ono gorsze od rozwiązania wskazanego jako rozwiązanie najlepsze znane? (załączony plik `berlin52.opt.tour` pokazuje optymalną trasę).