Kierunek: Informatyka, semestr V
Specjalność: Aplikacje Internetowe i Mobilne
Rok akademicki: 2021/2022

Inteligencja obliczeniowa
LABORATORIUM

Zajęcia 10, 11.
Algorytmy optymalizacji mrowiskowej (*ant colony optimization*)

CEL ZAJĘĆ

Zapoznanie studentów z algorytmem optymalizacji mrowiskowej, ich implementacja dla wybranych problemów optymalizacji ciągłej oraz dyskretnej, eksperymenty obliczeniowe.

PROBLEM 1

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 optymalizacji mrowiskowej i jego implementacja

Opis słowny
Wykorzystamy algorytm optymalizacji mrowiskowej o następującym pseudokodzie

```
Algorytm mrówkowy wersja 1
inicjalizuj feromony τij dla każdej krawędzi (i,j) ∈ E
for k = 1 to K do
   umieść mrówkę k w pewnym węźle początkowym
endfor
for i = 1 to MAX_ITER do
   for k = 1 to K do
      Dla k wybierz probabilistycznie następny wierzchołek na ścieżce
   endfor
   uaktualnij zawartość feromonu na ścieżkach
endfor
return najlepsze znalezione rozwiązanie
```

Zakładamy, że:

a) Mamy *K* mrówek, gdzie *K* odpowiada liczbie miast w problemie TSP,
b) Początkowo, każdą mrówkę umieszczamy w pewnym węźle początkowym, przyjmij, że każdą w innym mieście, np. mrówka nr 0 – w mieście nr 0, mrówka nr 1 – w mieście nr 1, …, mrówka nr *K*-1 – w mieście nr *K*-1.
c) W kolejnej iteracji każda mrówka będzie systematycznie budowała rozwiązanie zgodnie z zasadą algorytmu konstrukcyjnego, tj. w każdej iteracji trasa każdej mrówki będzie stopniowo rozbudowywana o kolejne odwiedzone miasto.
d) Każde kolejne miasto wybierane jest probabilistycznie, tj. … wybieramy najbliższej położone osiągalne bezpośrednio miasto, nieodwiedzone wcześniej.

$$p_j^k = \begin{cases} \dfrac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{m \in N_i^k} \tau_{im}^{\alpha} \cdot \eta_{im}^{\beta}} & if\ j \in N_i \\ 0 & if\ j \notin N_i \end{cases}$$

- N_i - sąsiedzi wierzchołka v, czyli zbiór wierzchołków, do których mrówka k może przejść z v
- η_{ij} - parametr określający heurystycznie atrakcyjność wyboru krawędzi (i, j), załóżmy: $\eta_{ij} = 1/d_{ij}$
- τ_{ij} - ilość feromonu na krawędzi (i, j), załóżmy wartość początkową $\tau_{ij} = 1$, dla każdej krawędzi (*i*, *j*).
- α, β – parametry określające wpływ τ_{ij} oraz η_{ij} odpowiednio na działanie algorytmu, załóżmy: $\alpha = 1, \beta = 1$ (zwykle dobre wyniki daje zestaw $\alpha = 1, \beta = 2$ do 5).

e) Ilość feromonu na każdej krawędzi jest aktualizowana wg:

$$\tau_{ij} = (1 - \rho) \cdot \tau_{ij} + \Delta\tau_{ij}$$

- ρ – współczynnik wyparowania 'starego' feromonu, załóżmy: $\rho = 0.5$.
- $\Delta\tau_{ij}$ - 'nowy' feromon, który jest zdeponowany przez wszystkie mrówki na krawędzi (*i*, *j*) obliczany wg poniższego wzoru, załóżmy Q=100:

$$\Delta\tau_{ij} = \sum_{k=0}^{m} \Delta\tau_{ij}^{k}$$

$$\Delta\tau_{ij}^{k} = Q/d_{ij}$$

f) Algorytm kończy się, gdy rozważymy wszystkie miasta lub gdy nie jest możliwe skonstruowanie takiej trasy (brak rozwiązania).

Kodowanie rozwiązania

Wektor zmiennych $x = (x_0, x_1, \ldots, x_{n-1})$, $x_i \in [0, n)$ 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 optymalizacji mrowiskowej. 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) – przypomnij sobie format tego pliku z zajęć 2, 3. Przypominam również, iż zakładamy, że graf jest pełny.

Określ:

a) 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? (załączony plik `berlin52.opt.tour` pokazuje optymalną trasę).
c) Porównaj znalezione rozwiązanie z rozwiązaniami uzyskanymi wcześniej za pomocą innych algorytmów dla tego zestawu danych.

Modyfikacje algorytmu

Opis słowny

```
Algorytm mrówkowy wersja 2
inicjalizuj feromony τij dla każdej krawędzi (i,j) ∈ E
for i = 1 to MAX_ITER do
   for k = 1 to K do
      umieść mrówkę k w pewnym węźle początkowym
      while mrówka k nie zbudowała rozwiązania do
         dla k wybierz probabilistycznie nast. wierzchołek na ścieżce
      endwhile
   endfor
   uaktualnij zawartość feromonu na ścieżkach
endfor
return najlepsze znalezione rozwiązanie
```

Modyfikacje w stosunku do algorytmu w wersji 1

a) W każdej iteracji i = 1 … *MAX_ITER* konstruujemy pełne trasy dla każdej mrówki,

b) *MAX_ITER* nie jest bezpośrednio związany liczbą węzłów, załóżmy *MAX_ITER* = 1000.

c) Aktualizacja feromonu następuje dopiero po zbudowaniu pełnych ścieżek przez każdą mrówkę, zgodnie ze wzorem:

$$\Delta\tau_{ij}^{k} = \begin{cases} Q/L_k & if\ (i,j) \in T_k \\ 0 & otherwise \end{cases}$$

gdzie:

- Q – pewien parametr heurystyki, załóżmy w algorytmie Q=1
- T_k – ścieżka tworzona przez mrówkę k
- L_k – długość T_k (suma długości wszystkich krawędzi z skład T_k).

Eksperyment

Cel eksperymentu - jw.

Eksperyment przeprowadź dla wybranych zbiorów danych testowych zawartych w bibliotece TSPLIB (link jw.):

Problem	**Rozwiązanie optymalne**
kroa100.tsp - 100-city problem A (Krolak/Felts/Nelson)	21282
kroa150.tsp - 150-city problem A (Krolak/Felts/Nelson)	26524
kroa200.tsp - 200-city problem A (Krolak/Felts/Nelson)	29368
krob100.tsp - 100-city problem B (Krolak/Felts/Nelson)	22141
krob150.tsp - 150-city problem B (Krolak/Felts/Nelson)	26130
krob200.tsp - 200-city problem B (Krolak/Felts/Nelson)	29437
kroc100.tsp - 100-city problem C (Krolak/Felts/Nelson)	20749
krod100.tsp - 100-city problem D (Krolak/Felts/Nelson)	21294
kroe100.tsp - 100-city problem E (Krolak/Felts/Nelson)	22068

a) Porównaj wyniki uzyskane przez algorytm w wersji 1 i 2.

b) Zastanów się nad możliwością hybrydyzacji ww. algorytmów z innymi wcześniej poznanymi.