Algorytmy metaheurystyczne

Problem komiwojażera euklidesowego. Local Search.

Karol Janic

16 listopada 2023

Spis treści

1 Cel zadania

Celem zadania jest sprawdzenie skuteczności heurystyki Local Search na przykładzie euklidesowego problemu komiwojażera oraz zbadanie wpływu wyboru rozwiązania początkowego i metody generowania otoczenia na jakość rozwiązania.

2 Algorytm Local Search

2.1 Otoczenie invert

Otoczeniem rozwiązania reprezentowanego przez permutację π jest zbiór rozwiązań uzyskanych przy użyciu pojedynczego ruchu $invert(\pi, i, j)$, który zamienia kolejność wierzchołków od i-tego do j-tego.

2.1.1 Moc otoczenia

Można łatwo zauważyć, że:

- $invert(\pi, i, i) = \pi$
- $invert(\pi, i, j) = invert(\pi, j, i)$

Zatem ruchy powodujące powstawanie różnych rozwiązań z rozwiązania π można opisać zbiorem (wierzchołki numerujemy liczbami od 1 do n):

$$INV = \{invert(\pi, i, j) : 1 \leq i < j \leq n\}$$

Aby obliczyć moc zbioru INV ustalamy i kolejno na $1, 2, ..., (n-1)$ oraz dobieramy odpowiednie j, czyli $i < j \leq n$. Wtedy:

$$|INV| = (n-1) + (n-2) + ... + 1 = \frac{(n-1)n}{2} = \frac{n^2 - n}{2}$$

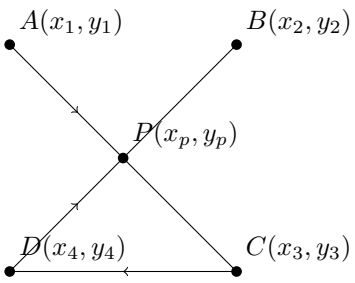
Interpretując permutację π jako ciąg wierzchołków do odwiedzenia można zauważyć, że $invert(\pi, 1, n)$ nie wyprowadza nowego rozwiązania, ponieważ nie ma znaczenia czy odwiedzanie rozpoczniemy od pierwszego czy ostatniego wierzchołka, więc całe otoczenie ma rozmiar $\frac{n^2-n}{2} - 1$.

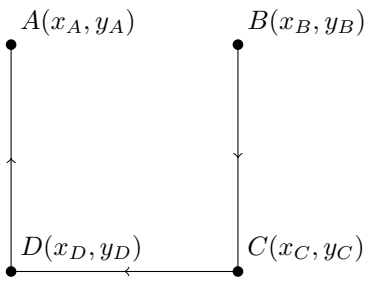
2.1.2 Własność ruchu invert

Cechą ruchu typy invert jest "rozplątywanie" pętli w cyklu. Pętle takie są nieoptymalne. Można to pokazać korzystając z nierówności trójkata, która na przestrzeni euklidesowej zachodzi:

$$d(A, D) \leq d(A, P) + d(D, P) \qquad d(B, C) \leq d(B, P) + d(P, C)$$

$$\begin{aligned} &d(A, C) + d(C, D) + d(D, B) = \\ &= d(A, P) + d(P, C) + d(C, D) + d(D, P) + d(P, B) \geq \\ &\geq d(A, D) + d(D, C) + d(C, B) \end{aligned}$$





2.1.3 Problem nierówności trójkąta przy zaokrąglaniu

W realizacji modelu odległości pomiędzy wierzchołkami wyrażane są liczbami całkowitymi. Konwersja dokładnej odległości następuje poprzez zaokraglenie jej do najbliższej liczby całkowitej. W takim modelu nierówność trójkąta nie zawsze zachodzi. Weźmy dla przykładu powyżej punkty $A(0, 0.5)$, $B(0.5, 0.5)$, $C(0.5, 0)$, $D(0, 0)$. Wówczas punktem przecięcia jest $P(0.25, 0.25)$. Rzeczywiste odległości prezentują się następująco:

$$d(A, D) = d(B, C) = 0.5$$

$$d(A, P) = d(D, P) = d(B, P) = d(C, P) \approx 0.35$$

$$d(A, C) = d(B, D) \approx 0.7$$

Odległości w modelu prezentują się natomiast tak jak zapisano poniżej:

$$d'(A, D) = d'(B, C) = 1$$

$$d'(A, P) = d'(D, P) = d'(B, P) = d'(C, P) = 0$$

$$d'(A, C) = d'(B, D) = 1$$

Wtedy nierówności w $\triangle APC$ oraz $\triangle BPC$ nie zachodzą a obie skonstruowane drogi mają długość 3.

2.2 Wybór rozwiązania początkowego

1. rozwiązanie zbudowane na podstawie MST
2. rozwiązanie wygenerowane w sposób losowy

2.3 Przegląd sąsiedztwa

1. przeglądanie całego sąsiedztwa(rozmiar: $O(n^2)$)
2. przeglądanie n losowo wybranych sąsiadów(rozmiar: $O(n)$)

2.4 Generowanie najlepszego kandydata

Gdy rozwiązaniem początkowym jest rozwiązanie generowane na podstawie MST to $\sqrt{n}$ razy generujemy takie rozwiązanie zaczynając od losowego wierzchołka drzewa a następnie używamy go jako startowego w algorytmie Local Search.
Gdy rozwiązaniem początkowym jest rozwiązanie losowe to n razy powtarzamy losowanie rozwiązania oraz poprawianie go algorytmem Local Search. W przypadku $n > 1000$ procedurę powtarzamy tylko 100 razy.

W każdym przypadku jako wynik wybieramy rozwiązanie o najmniejszej wadze.

3 Wyniki

Oznaczenia metod generujących kandydatów:

- LS1 - algorytm Local Search startujący z rozwiązania wygenerowane z losowego wierzchołka MST i przeglądający całe otoczenie w każdej iteracji
- LS2 - algorytm Local Search startujący z losowego rozwiązania i przeglądający całe otoczenie w każdej iteracji
- LS3 - algorytm Local Search startujący z losowego rozwiązania i przeglądający n losowych sąsiadów w każdej iteracji

Zaimplementowane metody zostały porównane na przykładach z `https://www.math.uwaterloo.ca/tsp/vlsi/index.html`. Wyniki prezentują się następująco:

Przykład	Suma wag rozwiązania optymalnego	Suma wag kandydata opartego o MST	Suma wag najlepszego rozwiązania LS1	Suma wag najlepszego rozwiązania LS2	Suma wag najlepszego rozwiązania LS3
xqf131					
xqg237					
pma343					
pka379					
bcl380					
pbl395					
pbk411					
pbn423					
pbm436					
xql662					
xit1083					
icw1483					
djc1785					
dcb2086					
pds2566					

Tabela 1: Porównanie metod generowania kandydatów dla problemu komiwojażera.

4 Wnioski

-