

Techniki optymalizacji

Łukasz Wojnarowski (80164)

Tomasz Kujawa (75909)

9 listopada 2010

Spis treści

1	Opis problemu	3
2	Generowanie rozwiązania początkowego (<i>RP</i>)	3
2.1	Opis metody	3
2.1.1	Słowny	3
2.1.2	Pseudokod	5
2.2	Wyniki	5
3	Local search (LS)	6
3.1	Opis metody	6
3.2	Opis słowny metody	6
3.3	Pseudokod	6
3.4	Wyniki	7
3.5	Rysunki najlepszych rozwiązań	7

1 Opis problemu

Rozwiązywany problem jest rozwinięciem *problemu komiwojażera* (TSP - ang. traveling salesman problem), który polega na znalezieniu 4 cykli hamiltona w pełnym grafie ważonym o minimalnej sumie wag.

Dane wejściowe składają się z grafu pełnego $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, gdzie \mathcal{V} to zbiór wierzchołków (można go interpretować jako zbiór punktów na płaszczyźnie), a \mathcal{E} to zbiór krawędzi. Dla każdej z krawędzi $\{v_i, v_j\}$: $v_i, v_j \in \mathcal{V}$ znana jest waga, będąca odległością pomiędzy wierzchołkami v_i, v_j . Rozwiązaniem problemu są cztery cykle proste

$$w_1, w_2, w_3, \dots, w_{n-1}, w_n ; x_1, x_2, x_3, \dots, x_{n-1}, x_n ; y_1, y_2, y_3, \dots, y_{n-1}, y_n \text{ oraz } z_1, z_2, z_3, \dots, z_{n-1}, z_n, \quad (1)$$

które spełniają następujące ograniczenia:

- $w_i \in \mathcal{V}'$,
- $x_j \in \mathcal{V}''$,
- $y_k \in \mathcal{V}'''$,
- $z_l \in \mathcal{V}''''$,
- $\mathcal{V}' \cup \mathcal{V}'' \cup \mathcal{V}''' \cup \mathcal{V}'''' = \mathcal{V}$,
- $\mathcal{V}' \cap \mathcal{V}'' \cap \mathcal{V}''' \cap \mathcal{V}'''' = \emptyset$,
- $|\mathcal{V}'| = |\mathcal{V}''| = |\mathcal{V}'''| = |\mathcal{V}''''| = n$, przy założeniu, że $\mathcal{V} = 4n$.

Niech $|v_i, v_j|$ oznacza wagę (koszt przebycia drogi) krawędzi pomiędzy wierzchołkami v_i, v_j . Dla tak zdefiniowanego modelu funkcja celu została określona w następujący sposób:

$$\min C = \sum_{i < n}^{i=1} |w_i, w_{i+1}| + |w_n, w_1| + \sum_{i < n}^{i=1} |x_i, x_{i+1}| + |x_n, x_1| + \sum_{i < n}^{i=1} |y_i, y_{i+1}| + |y_n, y_1| + \sum_{i < n}^{i=1} |z_i, z_{i+1}| + |z_n, z_1| \quad (2)$$

gdzie $|\mathcal{V}| = 4n$.

2 Generowanie rozwiązania początkowego (RP)

2.1 Opis metody

Analizowana metoda generowania rozwiązania początkowego to *grupowanie* i następnie *poszukiwanie najbliższego sąsiada*.

2.1.1 Słowny

Metoda rozpoczyna się od losowego wybrania wierzchołka początkowego, na którego podstawie stworzone zostaną grupy. Grupy mają najpierw przydzielane z puli dostępnych wierzchołków elementy początkowe - takie, że środek ciężkości od punktów już wcześniej przydzielonych jest największy. Następnie na podstawie wybranych "liderów" budowane są grupy - tak, że każdy kolejny element dodawany do grupy będzie miał najmniejszą odległość od środka ciężkości grupy. Należy zaznaczyć, że przydział po grupach odbywa się iteracyjnie - tzn. najpierw przydzielamy jeden element do grupy pierwszej, potem jeden element do grupy drugiej i iteracyjnie aż do wyczerpania się elementów nieprzydzielonych do żadnej grupy. Przydział ten jest powtarzany, aż stworzone zostaną 4 grupy o równych licznosciach.

Następnie w każdej grupie następuje budowanie ścieżki (cyklu) tak, że przy każdym kroku wybierany jest taki wierzchołek, że jego odległość od środka ciężkości dotychczas wybranych wierzchołków jest

najmniejsza. Algorytm zatrzymuje się, jeśli w grupie nie będzie już nieodwiedzonych wierzchołków. Należy pamiętać, by rozwiązanie uzupełnić o krawędź pomiędzy ostatnim a pierwszym wierzchołkiem - tzn. by waga zwracana uwzględniała połączenie pomiędzy ostatnim, a pierwszym elementem cyklu.

2.1.2 Pseudokod

Poniżej zaprezentowano pseudokod algorytmu opisanego w części 2.1.1.

GENERUJ ROZWIĄZANIE POZĄTKOWE(\mathcal{V})

```

1  GENERUJ PODZIAŁ NA GRUPY()
2
3   $i \leftarrow 1$ 
4  for  $\forall i$  in  $\{1, 2, 3, 4\}$ 
5      do
6           $v \leftarrow$  POBIERZ LOSOWY Z GRUPY( $i$ )
7          PRZYDZIEL WIERZCHOŁEK DO ŚCIEŻKI W GRUPIE( $v$ )
8
9  while  $\exists$  grupa z nieprzydzielonymi wierzchołkami
10     do
11          $next \leftarrow$  NAJBLIŻSZY NIEPRZYDZIELONY WIERZCHOŁEK DLA GRUPY( $i$ )
12         PRZYDZIEL WIERZCHOŁEK DO ŚCIEŻKI W GRUPIE( $next$ )
13          $i = (i + 1) \% 5$ 
14
```

W kodzie wykorzystano metode przygotowania grup, która została zaprezentowana poniżej:

GENERUJ PODZIAŁ NA GRUPY(\mathcal{V})

```

1   $v1 \leftarrow$  POBIERZ LOSOWO( $\mathcal{V}$ )
2   $v2 \leftarrow$  POBIERZ NAJDALSZY( $\mathcal{V} \setminus \{v1\}$ )
3   $v3 \leftarrow$  POBIERZ NAJDALSZY( $\mathcal{V} \setminus \{v1, v2\}$ )
4   $v4 \leftarrow$  POBIERZ NAJDALSZY( $\mathcal{V} \setminus \{v1, v2, v3\}$ )
5  UMIEŚĆ WIERZCHOŁEK W GRUPIE( $v1, 1$ )
6  UMIEŚĆ WIERZCHOŁEK W GRUPIE( $v2, 2$ )
7  UMIEŚĆ WIERZCHOŁEK W GRUPIE( $v3, 3$ )
8  UMIEŚĆ WIERZCHOŁEK W GRUPIE( $v4, 4$ )
9   $i \leftarrow 1$ 
10 while  $\exists \mathcal{U} \leftarrow$  wierzchołki nieumieszczone w żadnej grupie
11     do
12          $closest\_v \leftarrow$  POBIERZ NAJBLIŻSZY DO  $i$ -TEJ GRUPY ( $\mathcal{U}$ )
13         UMIEŚĆ W GRUPIE( $closest\_v, i$ )
14          $i = (i + 1) \% 5$ 
15
```

2.2 Wyniki

W tabeli 1 zostały przedstawione uśrednione wyniki dla opracowywanej metody.

instancja	metoda	śr. wart. roz. z 10 pomiarów	mediana	odch. std.	najlepsza wartość
kroA100.txt	NS G	36305,1	35181	5171,589	30000
kroB100.txt	NS G	36434,7	37189	3499,752	30973

Tabela 1: Uśrednione wyniki pomiarów.



Rysunek 1: Rozwiązanie początkowe dla *kroA100.txt*

3 Local search (LS)

3.1 Opis metody

Analizowana metoda generowania rozwiązania to *rozrywanie (1 ruch) w wersji stromej*.

3.2 Opis słowny metody

Proces poszukiwania lokalnego optimum rozpoczyna się od wykonania kroków z opisanego w rozdziale *Generowanie rozwiązania początkowego*. Następnie na takim rozwiązaniu dokonywane jest lokalne przeszukiwanie.

Kroki metody:

1. Wybierz wierzchołek i $k - 1$ mu najbliższych wierzchołków.
2. Rozerwij łuki wokół tych wierzchołków.
3. Rozważ wszystkie możliwe sposoby naprawy do rozwiązania tego problemu.
4. Wykonaj ruch, który przynosi najwięcej zysku.

Parametr $k \in 2, 3, 4$ jest definiowany na wejściu programu.

3.3 Pseudokod

Algorytm generowania rozwiązania można zapisać przy pomocy poniższego pseudokodu.

LOKALNE PRZESZUKIWANIE(\mathcal{V}, k)

```

1  rozwiązanie ← GENERUJ ROZWIĄZANIE POCZĄTKOWE( $\mathcal{V}$ )
2  while (TRUE)
3      do
4          zysk ← 0
5          wybrani ← WYBIERZ ŁUKI( $k, \mathcal{V}$ )
6          możliwe_przydziały ← GENERUJ MOŻLIWE PRZYDZIAŁY( $wybrani$ )
7          wartość ← OBLICZ WARTOŚĆ ROZWIĄZANIA( $rozwiązanie$ )
8          for  $\forall$  ruch in możliwe_przydziały
9              do
10                 aktualne_rozwiązanie ← WYKONAJ RUCH( $rozwiązanie, ruch$ )
11                 aktualna_wartość ← OBLICZ WARTOŚĆ ROZWIĄZANIA( $aktualne_rozwiązanie$ )
12                 aktualny_zysk ← wartość − aktualna_wartość
13                 if aktualny_zysk  $\geq$  zysk
14                     then
15                         zysk ← aktualny_zysk
16                         ZAPAMIĘTAJ RUCH( $ruch$ )
17                     else
18                         return
19
20         if zysk > 0
21             then
22                 WYKONAJ ZAPAMIĘTANY RUCH( $rozwiązanie$ )
23             else
24                 return
25
```

3.4 Wyniki

W tabeli przedstawione zostały zbiorcze wyniki pomiarów:

- RP - metoda z pierwszego ćwiczenia - generowanie rozwiązania początkowego,
- $RP + LS$ - metoda lokalnego przeszukiwania rozpoczynająca się od wygenerowania rozwiązania początkowego zgodnie z zasadami z ćwiczenia numer 1.

instancja	metoda	śr. jakość (odch. standardowe)	śr. czas [ms]	jakość najlepszego przeszukiwania
kroA100.txt	RP	36305 (4995)	0	25893
kroA100.txt	$RP + LS$	23096 (1121)	123	21359
kroB100.txt	RP	36434 (3452)	0	25421
kroB100.txt	$RP + LS$	24604 (665)	125	22869

Tabela 2: Uśrednione wyniki pomiarów.

3.5 Rysunki najlepszych rozwiązań

Na poniższych rysunkach przedstawione zostały rozwiązania wygenerowane przy pomocy metody $RP + LS$.

Rysunek 2: Rozwiązanie RP+LS dla *kroA100.txt*Rysunek 3: Rozwiązanie RP+LS dla *kroB100.txt*