

Techniki Optymalizacji

Laboratorium nr 2

Sprawozdanie

Paulina Sadowska, Rafał Araszkiewicz

21 października 2016

1. Wprowadzenie

Celem ćwiczenia było poprawienie wyników otrzymanych przez algorytmy Nearest Neighbour i Greedy Cycle poprzez zastosowanie lokalnego przeszukiwania w wersji stromej. Otrzymane wyniki porównano z otrzymanymi po wyznaczeniu losowej ścieżki i następnie zastosowaniu lokalnego przeszukiwania.

2. Local search

W algorytmie tym trasa budowana jest w taki sposób, że dla wygenerowanego wcześniej rozwiązania szukamy takiej pary wierzchołków lub pary krawędzi które po zamianie pozwolą na otrzymanie krótszej ścieżki. Następnie wykonujemy zamianę, która pozwoli na uzyskanie największego zysku. Kroki te należy powtarzać tak długo, jak możliwe jest znalezienie takiej pary wierzchołków lub krawędzi których zamiana skróci ścieżkę.

2.1. Implementacja w pseudokodzie

```
dopoki zmiana trasy zmniejsza jej koszt

  dla kazdego z punktow na sciezce
    staryKoszt = koszt przejścia do punktu + koszt przejścia z
    ↪ punktu do następnego
    dla kazdego punktu spoza sciezki
      nowyKoszt = koszt przejścia do nowego punktu + koszt
      ↪ przejścia z nowego punktu do następnego
      jezeli nowyKoszt < minKoszt oraz nowyKoszt <
      ↪ staryKoszt
        minKoszt = nowyKoszt

  dla kazdej pary punktow na sciezce
    dla kazdej kolejnej pary punktow na sciezce
      stayKoszt = koszt przejścia między para nr 1 + koszt
      ↪ przejścia między para nr 2
      nowyKoszt = koszt przejścia między pierwszymi punktami
      ↪ z kazdej z par + koszt przejścia między drugimi
      ↪ punktami z kazdej z par
      jezeli nowyKoszt < minKoszt oraz nowyKoszt < koszt
        minKoszt = nowyKoszt

  jezeli zysk z zamiany wierzchołkow jest wiekszy niz z zamiany krawedzi
```

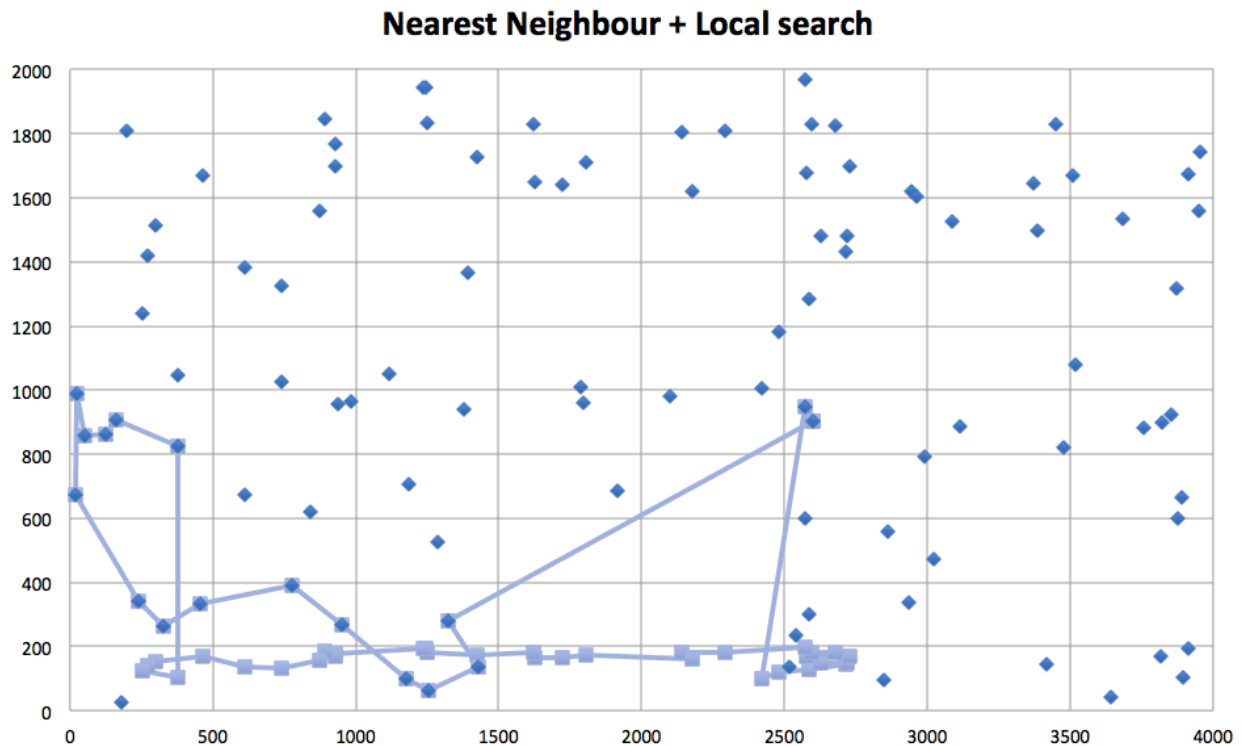
zamien punkt sciezki na ten dajacy lepszy zysk
jezeli zmiana krawedzi przyniesie zysk
zamien krawedzie miejscami
w przeciwnym razie
opusc petle

koniec

3. Najlepsze ścieżki

3.1. Nearest Neighbour + Local Search

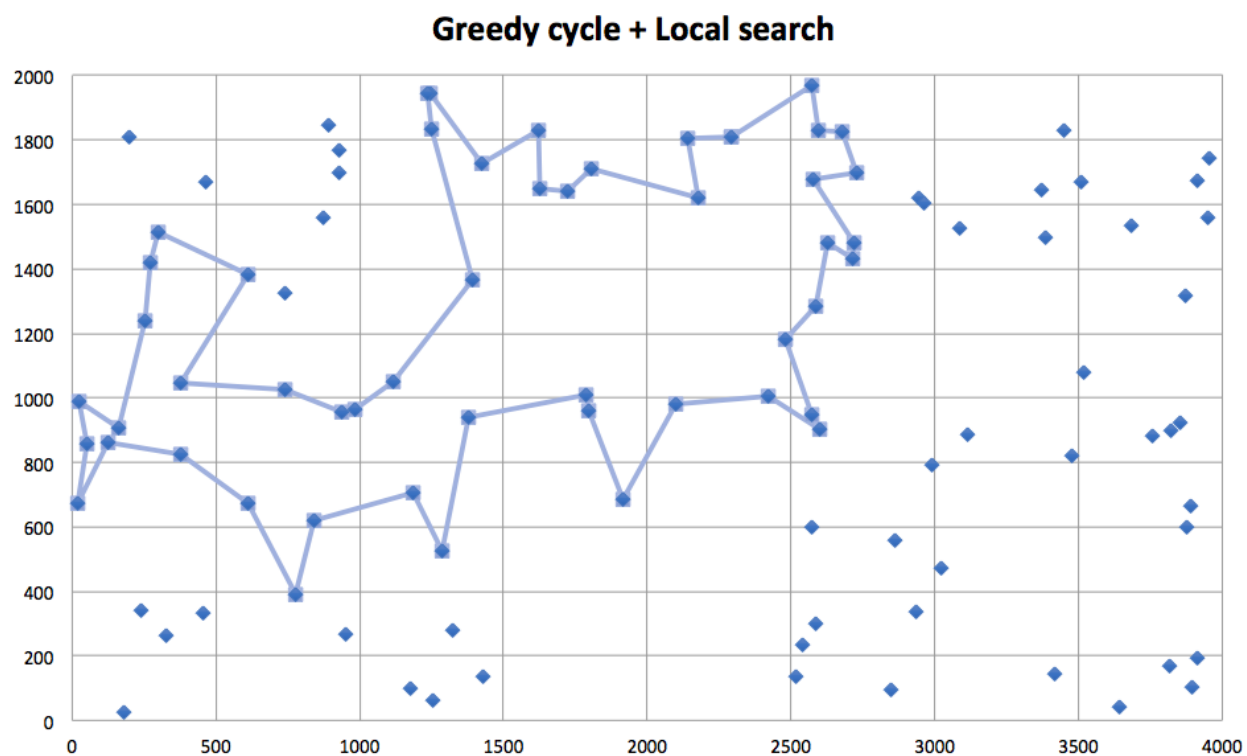
Najlepsza trasa: 50, 86, 8, 6, 56, 19, 11, 26, 85, 34, 61, 59, 76, 22, 97, 90, 44, 31, 10, 14, 16, 58, 73, 20, 71, 9, 83, 35, 37, 23, 17, 78, 52, 87, 15, 93, 21, 69, 65, 64, 3, 96, 55, 79, 30, 88, 41, 27, 92, 57, 50



Rysunek 1: Najlepsza trasa - Nearest Neighbour + Local Search

3.2. Greedy Cycle + Local Search

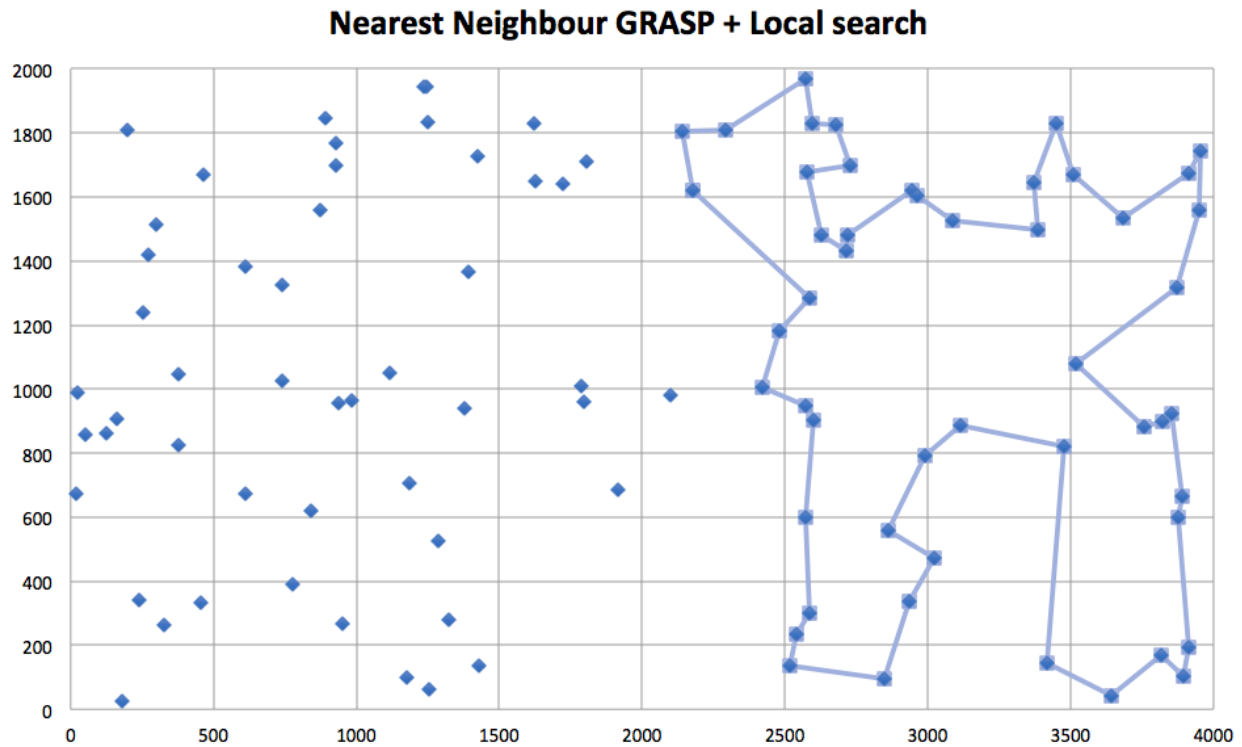
Najlepsza trasa: 61, 34, 85, 26, 11, 19, 6, 8, 56, 86, 50, 24, 80, 60, 57, 66, 27, 92, 0, 91, 7, 55, 96, 18, 52, 87, 15, 69, 21, 93, 17, 23, 37, 83, 9, 89, 48, 5, 62, 46, 10, 16, 14, 31, 44, 90, 97, 22, 76, 59, 61



Rysunek 2: Najlepsza trasa - Greedy Cycle + Local Search

3.3. Nearest Neighbour Grasp + Local Search

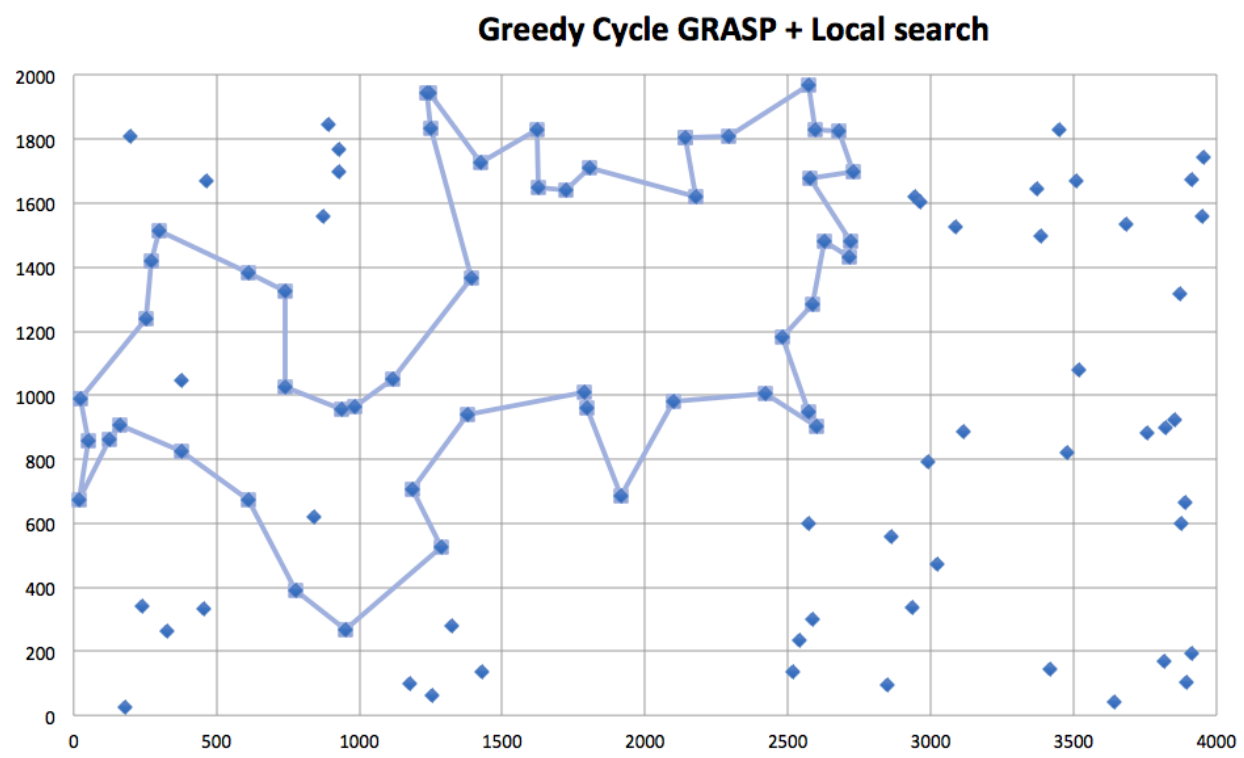
Najlepsza trasa: 0, 5, 48, 89, 78, 17, 23, 83, 9, 20, 71, 73, 58, 14, 10, 31, 90, 22, 44, 16, 97, 76, 19, 26, 85, 34, 61, 59, 6, 11, 56, 50, 86, 8, 33, 82, 45, 28, 42, 13, 2, 99, 47, 77, 95, 51, 29, 38, 84, 67, 0



Rysunek 3: Najlepsza trasa - Nearest Neighbour + Local Search

3.4. Greedy Cycle Grasp + Local Search

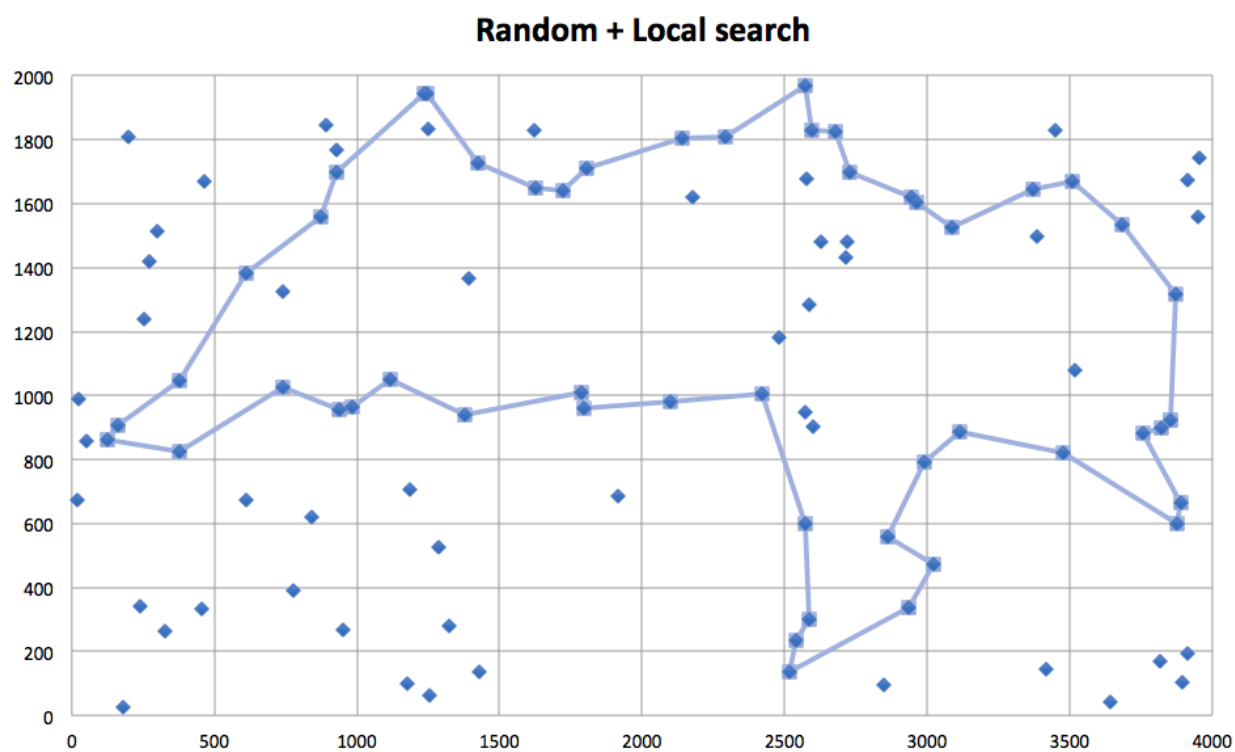
Najlepsza trasa: 59, 61, 34, 85, 26, 11, 19, 6, 8, 56, 86, 50, 24, 80, 60, 57, 66, 27, 92, 0, 91, 7, 55, 96, 18, 52, 87, 15, 69, 21, 93, 17, 23, 37, 83, 9, 89, 48, 5, 62, 46, 10, 16, 14, 31, 44, 90, 97, 22, 76, 59



Rysunek 4: Najlepsza trasa - Greedy Cycle + Local Search

3.5. Random + Local search

Najlepsza trasa: 51, 47, 13, 2, 45, 33, 82, 54, 11, 26, 85, 34, 61, 59, 22, 97, 90, 31, 14, 16, 20, 71, 83, 78, 87, 15, 52, 89, 48, 5, 62, 0, 92, 27, 57, 60, 68, 63, 39, 53, 43, 49, 72, 67, 84, 38, 36, 4, 95, 77, 51



Rysunek 5: Najlepsza trasa - Random + Local Search

4. Otrzymane wyniki

Tablica 1: Otrzymane wyniki

	NN + LS	GC + LS	NN Grasp + LS	GC Grasp + LS	Random + LS
min cost	9870	10548	10289	10548	11092
average cost	11002	11962	11622	11966	12615
max cost	12693	12629	13505	12629	14833
best time	5.32ms	1.95ms	20.45ms	1.95ms	49.00ms
average time	11.76ms	7.46ms	32.00ms	8.63ms	66.68ms
worst time	19.10ms	16.10ms	78.63ms	17.57ms	86.03ms