

Sprawozdanie z projektu TSP

Jakub Zięba, Albert Pieniądz, Łukasz Pyrek

Spis treści

[Solver 2](#_Toc155811367)

[Zagadnienie Optymalizacyjne Komiwojażera 2](#_Toc155811368)

[NN (algorytm najbliższego sąsiada) 3](#_Toc155811369)

[IHC (iteracyjna wspinaczka z multistartem) 3](#_Toc155811370)

[SA (symulowane wyżarzanie) 3](#_Toc155811371)

[TS (przeszukiwanie Tabu) 4](#_Toc155811372)

[GA (algorytmy genetyczne) 4](#_Toc155811373)

[Podsumowanie 4](#_Toc155811374)

# Zagadnienie Optymalizacyjne Komiwojażera

Zagadnienie optymalizacyjne komiwojażera jest klasycznym problemem w dziedzinie badań operacyjnych. Jego nazwa wywodzi się z wyobrażenia sprzedawcy podróżującego (komiwojażera), który ma za zadanie odwiedzić zestaw miast, każde tylko raz, i powrócić do miasta początkowego. Kluczowym elementem problemu jest istnienie n miast oraz konieczność znalezienia najkrótszej możliwej ścieżki łączącej je wszystkie, z punktu startowego do punktu końcowego. W analizowanym przypadku dostępne są również informacje o odległościach pomiędzy miastami. Projekt dotyczy symetrycznej wersji problemu komiwojażera, co oznacza, że odległość z miasta A do B jest równa odległości z B do A. Głównym wyzwaniem jest ogromna liczba możliwych kombinacji do rozważenia, która rośnie wykładniczo w zależności od liczby miast (n=48, 76, 127). Do tej pory nie znaleziono algorytmu, który w sposób efektywny i w czasie wielomianowym gwarantowałby znalezienie idealnego rozwiązania.

W ramach projektu wykorzystane zostaną różne algorytmy, takie jak: algorytm najbliższego sąsiada (NN), iteracyjna wspinaczka (IHC), symulowane wyżarzanie (SA), przeszukiwanie Tabu (TS) oraz algorytmy genetyczne (GA). Te metody nie gwarantują znalezienia rozwiązania optymalnego, ale pozwalają na uzyskanie wyników wystarczająco dobrych, będących kompromisem pomiędzy dokładnością a efektywnością obliczeniową. Ponadto, wykorzystanie metod lokalnej optymalizacji może prowadzić do ulepszenia wyników uzyskanych za pomocą tych algorytmów niedokładnych.

# Solver

Dla uproszczenia analizy efektywności badanych algorytmów, początkowy etap realizacji projektu skupił się na wykorzystaniu narzędzia Solver w Excelu do ustalenia optymalnych rozwiązań. To podejście umożliwiło porównanie wyników najlepszych rozwiązań uzyskanych przez różne algorytmy z tymi otrzymanymi za pomocą Solvera. Osiągnięte w ten sposób rezultaty zostały zaprezentowane poniżej:

* 48 miast – 11643
* 76 miast - 118893,5944
* 127 miast - 144030,6

# NN (algorytm najbliższego sąsiada)

W przypadku algorytmu najbliższego sąsiada, jedynym parametrem jest miasto, od którego rozpoczynamy budowanie trasy. Nie ma tutaj możliwości uwzględnienia różnych rodzajów ruchu, ponieważ nie zamieniamy ze sobą miast, a jedynie na podstawie miasta startowego wybieramy kolejnych, najbliższych sąsiadów. Uzyskane w ten sposób wyniki są następujące:

* 48 miast – 13251
* 76 miast - 145385,2632
* 127 miast - 145272,8249

# IHC (iteracyjna wspinaczka z multistartem)

Dla algorytmu IHC sprawdzaliśmy parametry takie jak rodzaj ruchu, liczba iteracji ruchu w każdym wykonaniu algorytmu, liczba wykonań algorytmu. Najlepsze wyniki uzyskaliśmy dla zamiany, czyli ruchu który wprowadza największe zmiany w wylosowanej trasie. Ponadto im wyższe wartości obydwu parametrów, tym lepsze wyniki. Jest to wynik którego można było się spodziewać, ponieważ mimo tego że w algorytmie iteracyjnej wspinaczki można wpaść w minimum lokalne, to zwiększenie liczby wykonanych prób zwiększa prawdopodobieństwo losowania coraz lepszych wyników. Najlepsze wyniki uzyskane dla każdego rodzaju ruchu są następujące:

* Swap:
  + 48 miast – 11955
  + 76 miast - 162847,413
  + 127 miast - 221146,1986
* Insercja:
  + 48 miast – 11083
  + 76 miast - 142793,6866
  + 127 miast - 201552,6997
* Zamiana:
  + 48 miast – 10653
  + 76 miast - 117385,3998
  + 127 miast - 164749,4253

Widać tutaj, że ze wszystkich ruchów, najgorsze wyniki otrzymaliśmy dla swapu, a najlepsze dla zamiany, niezależnie od długości trasy.

# SA (symulowane wyżarzanie)

Dla algorytmu symulowanego wyżarzania sprawdziliśmy parametry takie jak rodzaj ruchu, liczba iteracji ruchu, liczba iteracji całego algorytmu, temperatura startowa, rodzaj chłodzenia i parametr chłodzenia. Podobnie jak poprzednio, najlepsze wyniki uzyskaliśmy dla ruchu zamiana. Również analogicznie do poprzedniego przypadku, im większa liczba wykonań ruchów, tym lepsze otrzymujemy wyniki. Na podstawie topowych wyników z różnych zbiorów danych można zobaczyć, że dla krótszych tras lepsze wyniki otrzymaliśmy dla temperatur startowych niższych, a dla dłuższych lepsze są wyższe temperatury startowe, ale nie jest to regularne. Dla długich tras niższe temperatury również są w stanie dać dobre wyniki i vice versa. Jeśli chodzi o parametr chłodzenia, to dla najdłuższej trasy najlepiej sprawdziły się wartości niskie, natomiast dla krótszych tras topowe wartości częściej powiązane były z wyższymi wartościami parametru.  
Najlepsze wyniki znalezione z pomocą tej metody:

* 48 miast – 10663
* 76 miast - 127211,1054
* 127 miast - 182377,9445

# TS (przeszukiwanie Tabu)

W przypadku przeszukiwania Tabu sprawdziliśmy parametry takie jak rodzaj ruchu, liczba iteracji ruchu, liczba iteracji algorytmu oraz długość samej listy. Podobnie jak w przypadku wszystkich poprzednich algorytmów, większa liczba iteracji ruchu pozytywnie wpływa na uzyskiwane wyniki. Natomiast w przeciwieństwie do dwóch poprzednich algorytmów, najlepszym ruchem jest tutaj insercja. Wynika to z różnic w samych ruchach, gdzie dla poprzednich algorytmów losowaliśmy dwa miasta na których wykonywaliśmy ruch, tutaj natomiast losujemy miasto i wykonujemy ruch z każdym pozostałym miastem. Jeśli chodzi o długość listy Tabu, to w przypadku naszych obliczeń najlepsze wyniki otrzymywaliśmy dla krótkich list, o długości 3 lub 5. Mimo tego dłuższe listy nie odstają od nich znacząco, plasując się na trzecim i dalszych miejscach rankingu. Można z tego wywnioskować, że długość listy Tabu powinna znajdować się w pewnym przedziale, w którym mogły równie dobrze znaleźć się wszystkie podane przez nas wartości, z lekką przewagą wartości niskich.  
Najlepsze znalezione wyniki z pomocą tego algorytmu:

* 48 miast – 10711
* 76 miast - 116210,2381
* 127 miast - 142209,4776

# GA (algorytmy genetyczne)

# Podsumowanie

Patrząc na wszystkie zaimplementowane i wykorzystane do rozwiązania problemu algorytmu bardzo łatwo jest zauważyć, że ich skuteczność faktycznie jest zależna od zastosowanych wartości parametrów. Mimo tego, że algorytmy różnią się od siebie, najlepiej wśród nich wszystkich wypadają algorytmy genetyczne, dając najlepsze wyniki. Algorytmy mają też oczywiście różną efektywność jeśli chodzi o czas. Algorytm najbliższego sąsiada mimo tego że nie daje zbyt dobrych wyników, to tutaj działał najszybciej. Jest to spowodowane tym że nie musimy wykonywać go wielokrotnie, ponieważ nie ma znaczenia ile razy go wykonamy, dla danego miasta startowego wyniki zawsze będą takie same. Algorytm listy Tabu liczy się natomiast bardzo długo, ponieważ sprawdzane są w nim wszystkie możliwe kombinacje miast dla ruchu. Algorytmy IHC oraz SA z powodu swojej podobnej budowy mają również podobne czasy wykonywania.