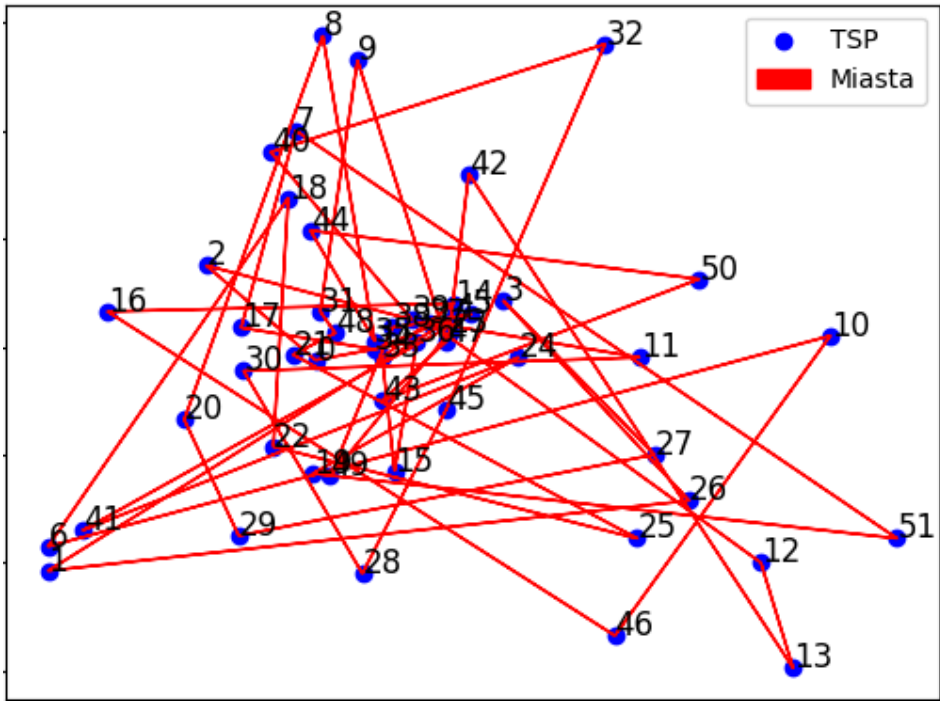
Wykorzystanie metaheurystyki w rozwiązaniu problemu TSP

Aleksander Kamiński inf155840
Mateusz Puk inf155864

Algorytm - Simulated Annealing:

1. **Inicjalizacja**



Zachłanna inicjalizacja

2. **Opis Algorytmu**

Zaimplementowaliśmy metaheurystykę symulowanego wyżarzania. Algorytm rozpoczyna od generacji zachłannego cyklu komiwojażera jako punktu startowego. Następnie, w każdej iteracji, zamienia losowo wybrane 2 krawędzie cyklu i akceptuje nową trasę z prawdopodobieństwem zależnym od różnicy długości tras oraz temperatury. Proces ten powtarzany jest przez określoną liczbę iteracji lub czas, stopując się po upływie limitu czasu. Odległość między każdą parą miast jest zapisywana w tablicy dwuwymiarowej. Prawdopodobieństwo otrzymujemy ze wzoru $P = T * e^{-c*i}$ gdzie T - temperatura początkowa, c - stała chłodzenia, i - iteracja w której się znajdujemy. W naszej implementacji skupiliśmy się na dużej temperaturze początkowej i liczbie iteracji, a nie zwiększyliśmy ilości zmienianych wierzchołków naraz.

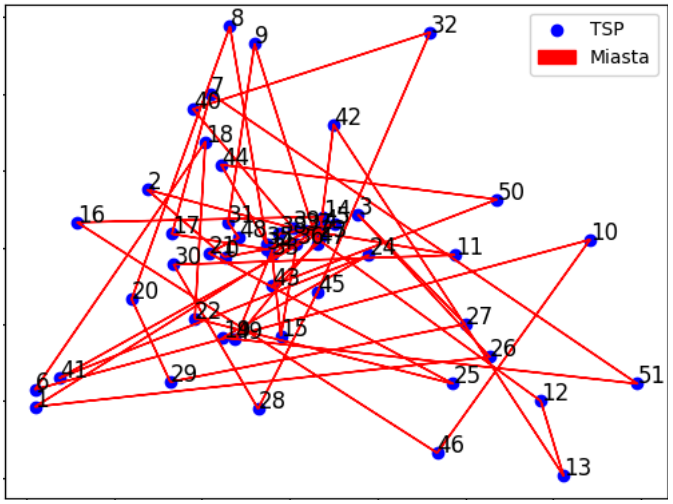
3. Pseudokod

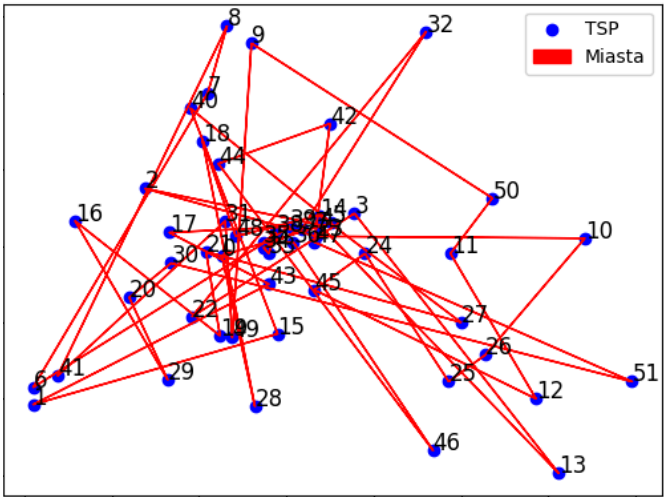
Funkcja main():
Wczytaj dane miast z pliku.
Ustaw parametry początkowej temperatury, współczynnika chłodzenia i maksymalnej liczby iteracji.
Wywołaj funkcję simulatedAnnealing.
Wyświetl najlepszą trasę i jej długość.

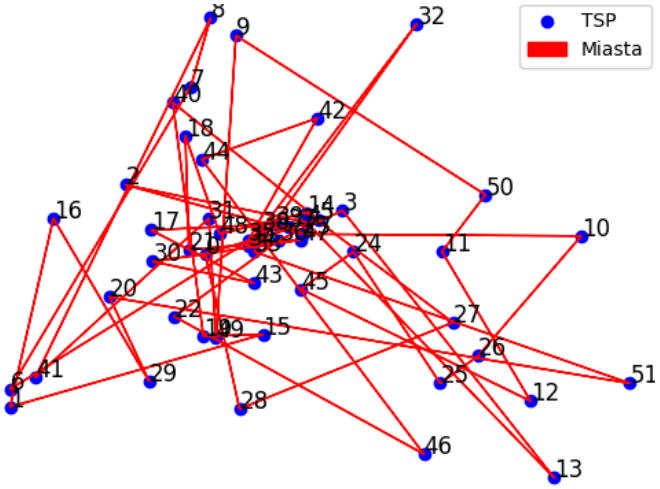
Funkcja simulatedAnnealing(cities, initialTemperature, coolingRate, maxIterations):
Stwórz losową trasę początkową za pomocą funkcji generateGreedyTour.
Ustaw ją jako najlepszą trasę.
Oblicz jej długość i ustaw jako najlepszą długość.
Dla każdej iteracji w zakresie od 0 do maxIterations-1:
{Oblicz temperaturę dla bieżącej iteracji.
Wygeneruj nową trasę poprzez zamianę dwóch losowych miast w trasie.
Oblicz różnicę długości między bieżącą trasą a nową trasą.
Jeśli nowa trasa jest krótsza lub zostanie zaakceptowana z prawdopodobieństwem zależnym od temperatury:
{Ustaw nową trasę jako bieżącą trasę.
Zaktualizuj bieżącą długość.}
Jeśli bieżąca długość jest krótsza od najlepszej długości:
{Ustaw bieżącą trasę jako najlepszą trasę.
Zaktualizuj najlepszą długość.}}
Zwróć najlepszą trasę.

Funkcja generateGreedyTour(cities):
Zainicjuj listę visited o długości równej liczbie miast, wypełnioną wartościami False.
Stwórz listę tour.
Dodaj indeks pierwszego miasta (0) do listy tour.
Ustaw pierwsze miasto jako odwiedzone.
Dopóki długość listy tour jest mniejsza niż liczba miast:
{Znajdź nieodwiedzone miasto najbliższe do bieżącego miasta.
Dodaj jego indeks do listy tour.
Oznacz to miasto jako odwiedzone.}
Zwróć listę tour.

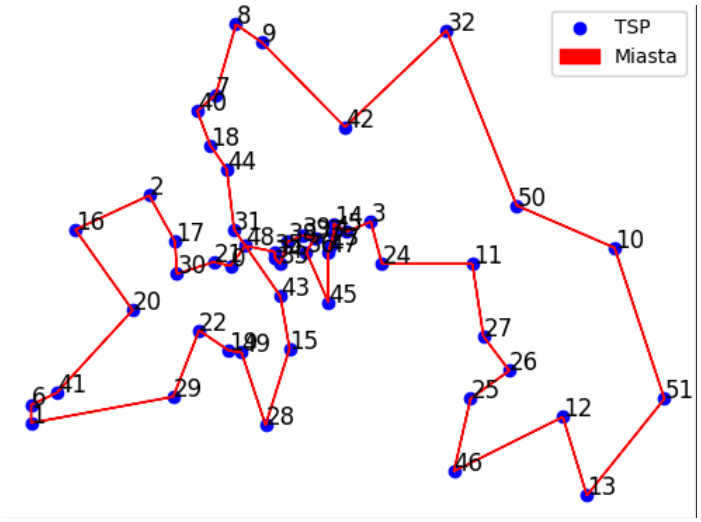
4. **Przykład obrazujący działanie**





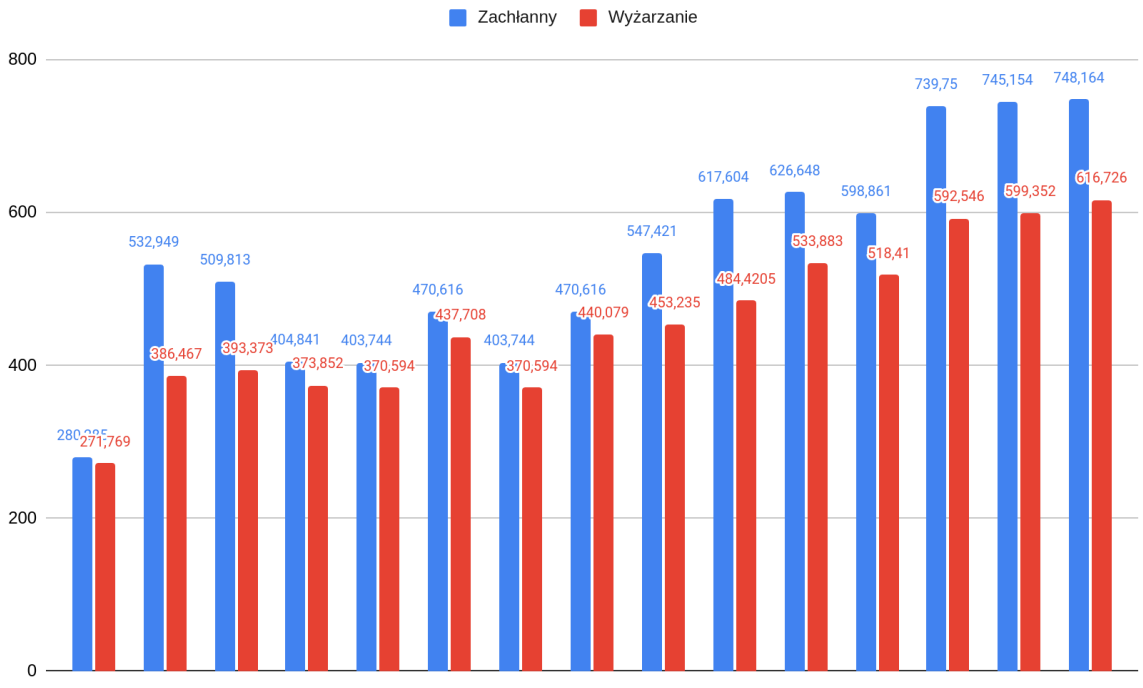


5. **Finalizacja**

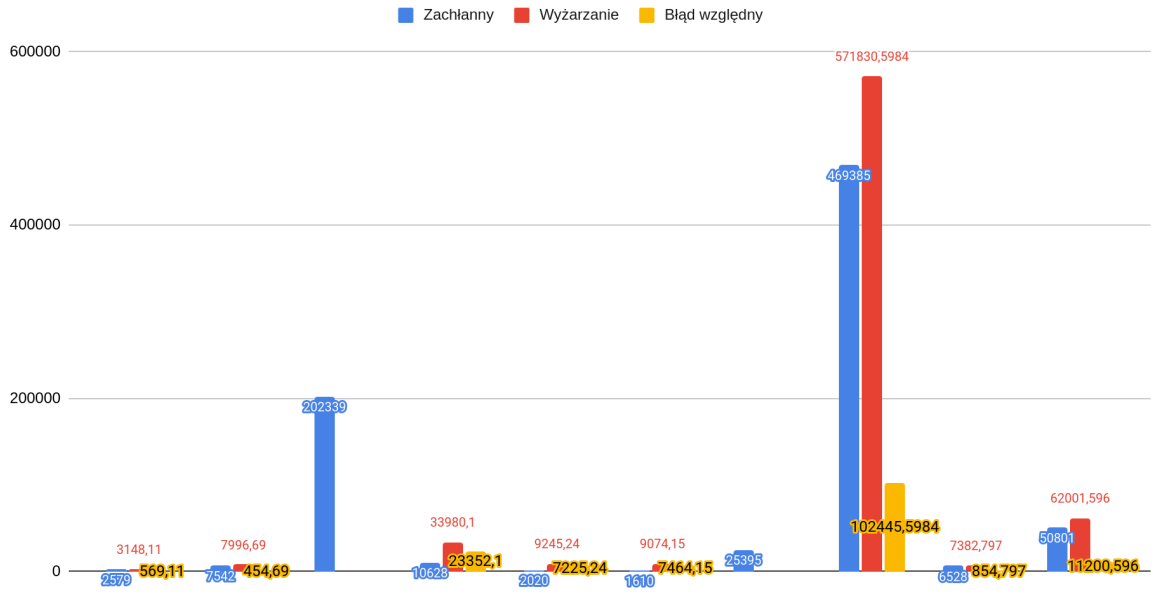


Wykresy

1. Porównanie wartości optymalnej Algorytmu z Algorytmem zachłannym.



2. Wartość błędu względnego Benchmarków w stosunku do wartości optymalnej.



Ranking

berlin52	7996.69
bier127	125760
tsp1000	29197.7
tsp500	97874.5
tsp250	14266,3