Algorytm genetyczny to metoda optymalizacyjna inspirowana mechanizmami biologicznej ewolucji. Jej główna idea polega na ewolucji populacji potencjalnych rozwiązań poprzez symulację procesu doboru naturalnego. W kontekście problemu komiwojażera, każde rozwiązanie, zwane osobnikiem, reprezentuje możliwą trasę odwiedzenia wszystkich miast, zapisaną jako permutacja liczb od 1 do n, gdzie n to liczba punktów do odwiedzenia.

Proces działania rozpoczyna się od utworzenia początkowej populacji osobników (80). Każdy osobnik jest generowany losowo, jako unikalna permutacja odwiedzin miast. Po wygenerowaniu populacji następuje etap oceny dopasowania, który polega na obliczeniu długości trasy dla każdego osobnika. W tym celu wykorzystuje się funkcję calculate\_distance, która korzysta z wcześniej przygotowanej macierzy odległości pomiędzy wszystkimi parami miast. Funkcja sumuje odległości pomiędzy kolejnymi miastami w trasie i dodaje koszt powrotu z ostatniego miasta do pierwszego, zamykając pętlę. Na tej podstawie wyliczane jest dopasowanie (fitness), które w implementacji przyjmuje postać odwrotności długości trasy – im krótsza trasa, tym wyższe dopasowanie. Taka konstrukcja zapewnia, że algorytm preferuje osobniki, które reprezentują bardziej optymalne rozwiązania.

W kolejnym etapie odbywa się selekcja osobników, które zostaną wykorzystane jako rodzice do utworzenia nowego pokolenia. W tym celu zastosowano selekcję turniejową o rozmiarze 3. W każdej selekcji losowanych jest trzech osobników z bieżącej populacji, a do dalszego etapu wybierany jest ten, którego trasa ma najniższy koszt.

Po wybraniu par rodziców następuje krzyżowanie, którego celem jest wygenerowanie potomków łączących cechy obu rodziców. W implementacji zastosowano operator Order Crossover. Losowo wybierany jest fragment trasy z pierwszego rodzica, który zostaje bezpośrednio skopiowany do potomka. Następnie pozostałe pozycje wypełniane są elementami z drugiego rodzica w kolejności ich występowania, z pominięciem już użytych miast. Dzięki temu uzyskiwana jest pełna, poprawna permutacja. Taki sposób działania umożliwia zachowanie części struktury trasy (czyli kolejności odwiedzin) z jednego rodzica oraz jej uzupełnienie o elementy drugiego, co zwiększa szansę na dziedziczenie korzystnych fragmentów rozwiązań.

Każdy nowo utworzony potomek może podlegać mutacji – czyli losowej zamianie miejsc dwóch miast w trasie. W implementacji prawdopodobieństwo mutacji wynosi 10%. Mutacja umożliwia wprowadzenie dodatkowej różnorodności do populacji i pozwala na ucieczkę z lokalnych minimów, które mogłyby zatrzymać proces optymalizacji w miejscu.

Aby zapobiec sytuacji, w której bardzo dobre rozwiązania zostałyby przypadkowo utracone, w algorytmie zastosowano mechanizm elityzmu. Polega on na tym, że dwa najlepsze osobniki z bieżącej populacji są bez zmian przenoszone do kolejnego pokolenia. Dzięki temu algorytm nie traci osiągniętego postępu i może go wykorzystać w kolejnych iteracjach.

W implementacji zdecydowałam się na 400 iteracji, co okazało się wystarczające do osiągnięcia stabilnej zbieżności w większości przypadków testowych.

**Macierz 16 x 16:**

Najlepsza długość trasy: 1981.30

Czas działania: 1.324 sekund

A map of a country with red lines and blue dots

AI-generated content may be incorrect.

*Najlepsza trasa – AG, 16 miast*

**Macierz 30 x 30:**

Najlepsza długość trasy: 2722.50

Czas działania: 1.829 sekund

A map of a map with red dots and blue points

AI-generated content may be incorrect.

*Najlepsza trasa – AG, 30 miast*

**Macierz 50 x 50:**

Najlepsza długość trasy: 4538.90

Czas działania: 2.625 sekund

A map of a country with red dots and blue points

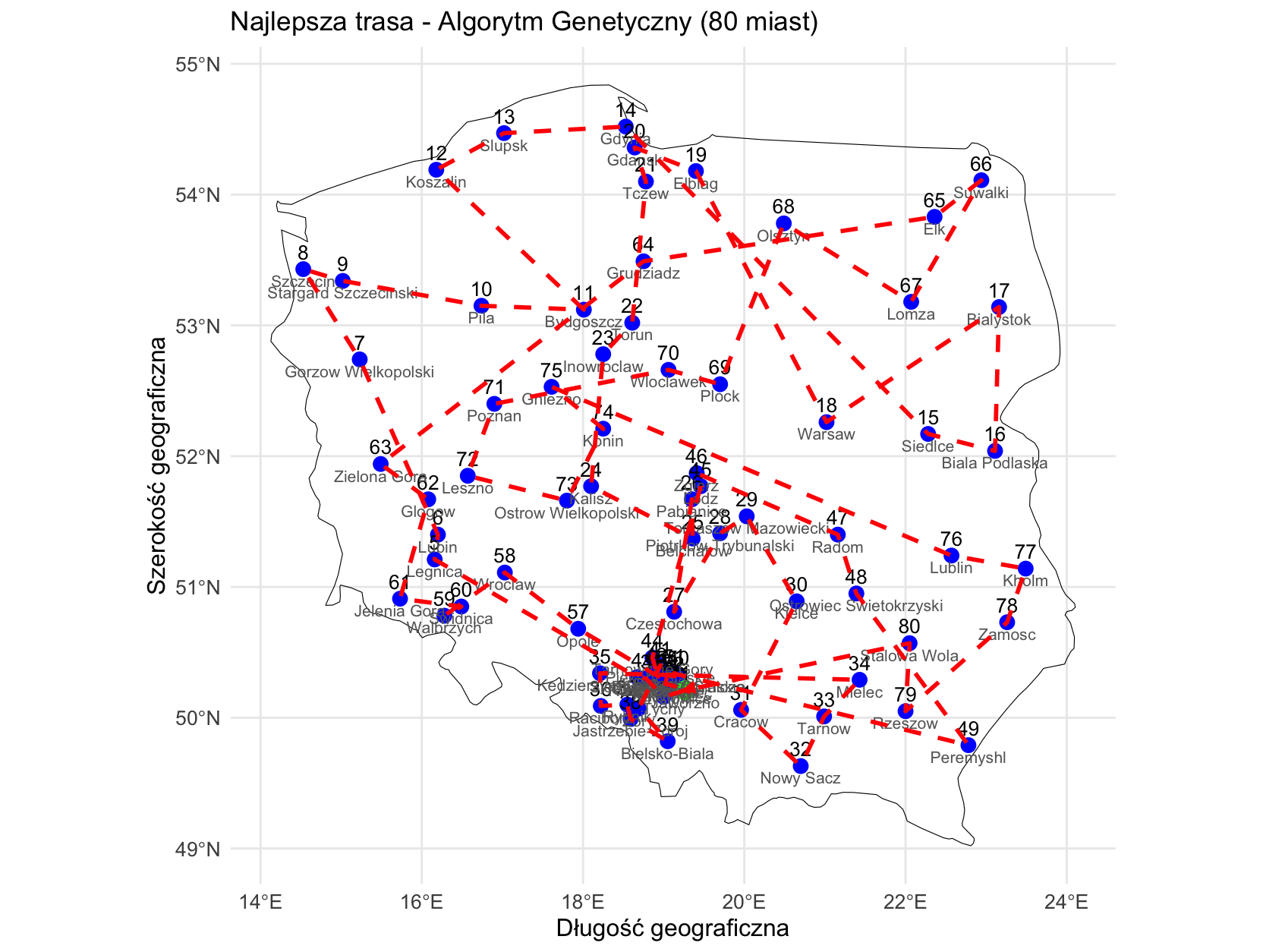
AI-generated content may be incorrect.

*Najlepsza trasa – AG, 50 miast*

**Macierz 80 x 80:**

Najlepsza długość trasy: 7364.50

Czas działania: 3.629 sekund



*Najlepsza trasa – AG, 80 miast*

A graph of different colored lines

AI-generated content may be incorrect.