`­­­­­­

Dominik Guz

Problem TSP w ujęciu dwukryterialnym oraz ewolucyjnym

2016

Spis treści

[TSP 3](#_Toc472202245)

[Biblioteka ParadisEO 4](#_Toc472202246)

[Program 4](#_Toc472202247)

[Wymagania 4](#_Toc472202248)

[Architektura 4](#_Toc472202249)

[Interfejs użytkownika 6](#_Toc472202250)

[Kod źródłowy 6](#_Toc472202251)

[referencje 14](#_Toc472202252)

# TSP

Problem komiwojażera ( ang. TSP ) jest rozwiązywalny poprzez znalezienie minimalnego cyklu Hamiltona w grafie. W ujęciu wielokryterialnym ( ang. MOTSP – Multi objective travelling salesman problem ) za dodatkowe kryterium oprócz długości przyjęto parametr kosztu.

Problem ten jest NP. trudny, dlatego też zastosowanie algorytmów ewolucyjnych do jego rozwiązania jest optymalne w przypadku dużych zestawów danych których przetworzenie w czasie rzeczywistym jest niemożliwe.

Ref1 [1]

Ref2 [2]

Ref3 [3]

Ref4 [4]

# Biblioteka ParadisEO

Biblioteka paradiseo jest zbiorem narzędzi metaheurystycznych. Została ona napisana w zgodności z ANSI C++. Autorem biblioteki jest organizacja INRIA.

W bibliotece tej znajdują się implementacje algorytmów ewolucyjnych oraz optymalizacji jednokryterialnej (EvolvingObjects) i wielokryterialnej (MultiObjectiveEvolvingObjects).

# Program

## Wymagania

Aplikacja pozwala na wczytywanie zestawu danych problemu TSP a następnie znalezienie optymalnych rozwiązań za pomocą algorytmów ewolucyjnych.

Przebieg postępu algorytmu ewolucyjnego wizualizowany jest na wykresie przedstawiającym zmianę wartości długości oraz kosztu trasy w kolejnych generacjach.

Wyniki optymalizacji przedstawiane są również na wykresie 3d w którym widoczne są wartości wszystkich osobników populacji ze względu na oba kryteria w dziedzinie czasu ( generacji ).

## Architektura

Aplikacja została stworzona z wykorzystaniem bibliotek graficznych QT Framework.

Dla celów rozwiązania problemu TSP w ujęciu wielokryterialnym, zostały stworzone implementacje operacji krzyżowania, mutacji oraz ewaluacji osobników. Dane wejściowe programu to pliki tekstowe zawierające informacje o ilości wierzchołków oraz ich współrzędne.

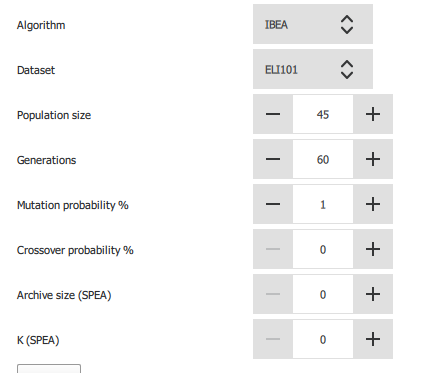
## Interfejs użytkownika

Interfejs użytkownika pozwala na definiowanie parametrów algorytmy ewolucyjnego takich jak wielkość populacji oraz ilość generacji a także prawdopodobieństwo mutacji oraz krzyżowania.

## Kod źródłowy

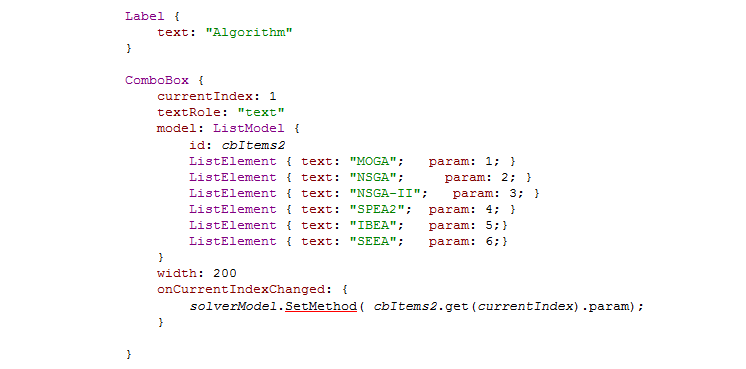
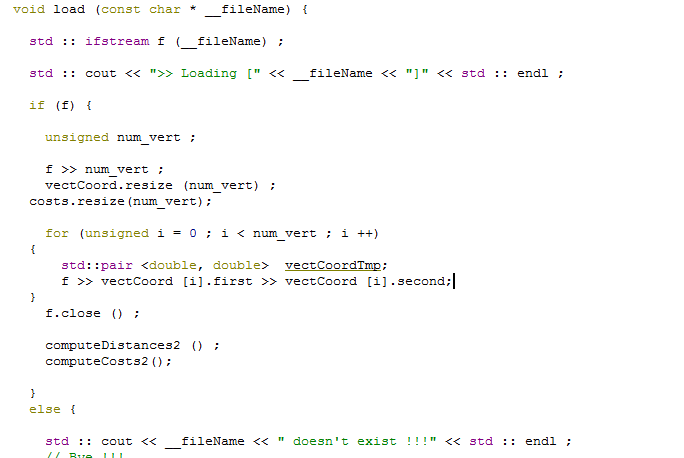
Podczas uruchamiania aplikacji rejestrowane są komponenty rozpoznawane w graficznym interfejsie użytkownika co pozwala na realizację wzorca MVC.

Po uruchomieniu program przechodzi do głównego widoku w który możliwa jest parametryzacja optymalizacji oraz wybranie zestawu danych. W tym tez widoku wyświetlane są wyniki.



Kliknięcie przycisku z napisem ‘solve’ uruchamia optymalizacje wielokryterialną z wykorzystaniem algorytmów ewolucyjnych.

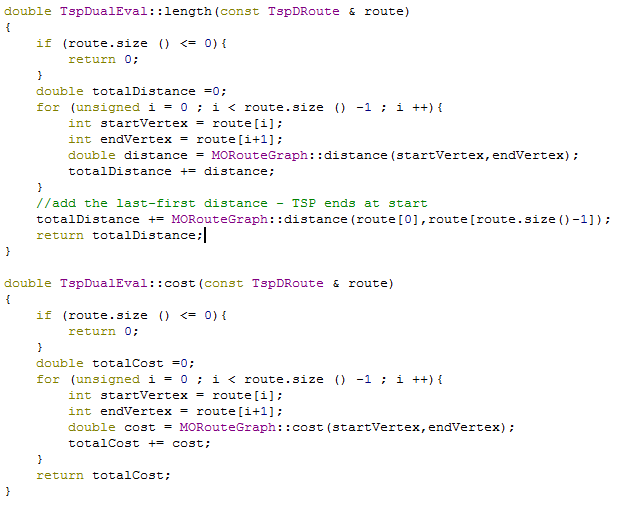
Dane zaczytywane są z pliku zlokalizowanego w folederze ‘benchs’ aplikacji. Przedstawia to poniższy kod.



Wykonanie optymalizacji następuje poprzez zainicjalizowanie populacji startowej ( obiekt TspDRouteInit ) a następnie inicjalizację algorytmów ewolucyjnych na podstawie zadanych parametrów z uwzględnieniem wielkości populacji, ilości generacji oraz typ algorytmu.



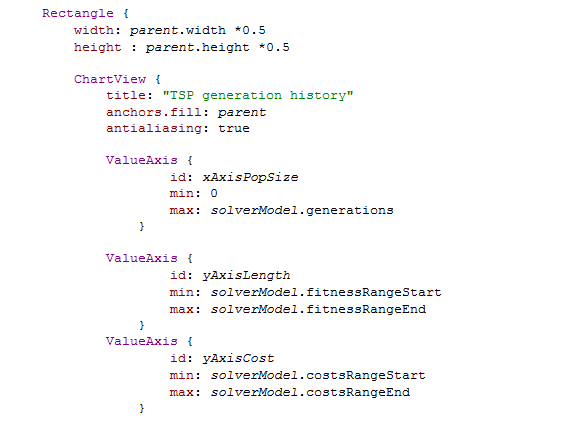
Osobniki populacji oceniane są przez algorytm ewolucyjny za pomocą obiektu TspDualEval, zawierającego funkcje ewaluacji ścieżki względem długości oraz kosztu.

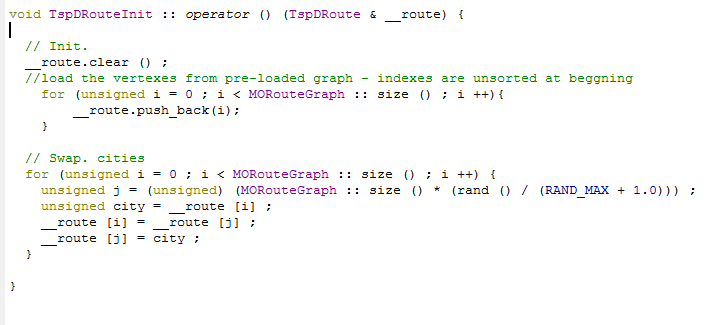


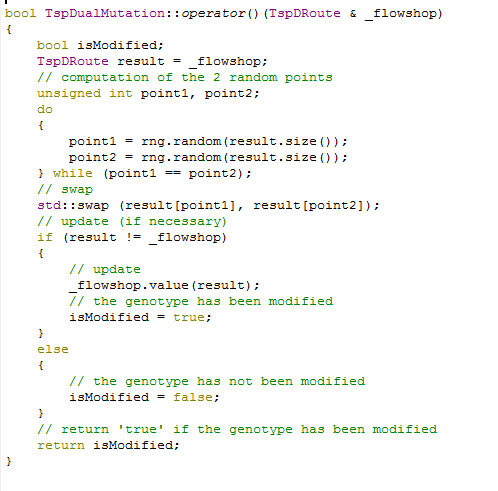
Rysowanie ma miejscie poprzez wbudowane funkcje ‘drawLine’ rysujące linie pomiędzy dwoma punktami reprezentowanymi przez obiekty QPoint.

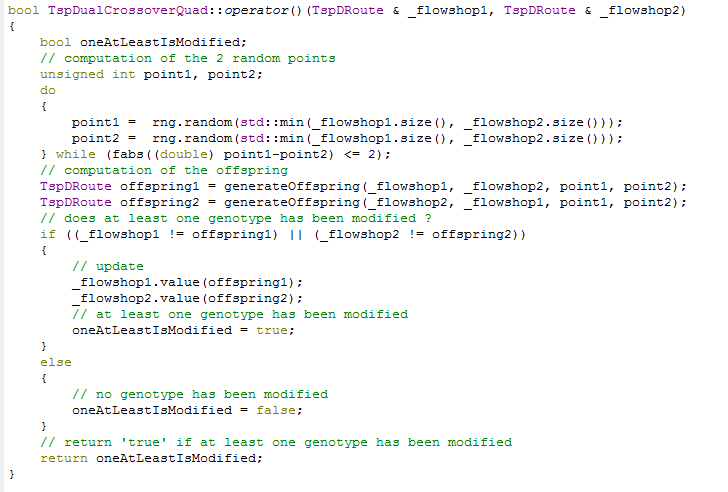
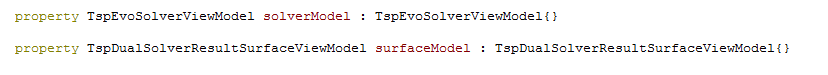
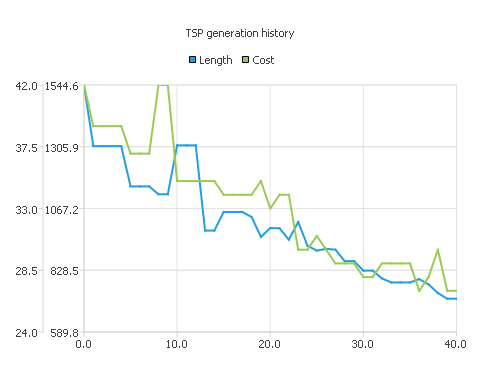


Wyświetlanie postępu algorytmy ewolucyjnego i wartości pareto-optymalnych rozwiązań wyświetlane są na wykresie zawierającym osobne osie wartości dla długości oraz kosztu trasy co przedstawia poniższy kod.









# Środowisko testowe

Testy zostały przeprowadzone na jednostce obliczeniowej o następujących parametrach :

Procesor : i7 1.5Ghz

System operacyjny : MS Windows 10

# Wyniki

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Zbiór danych /Algorytm | MOGA | NSGA | IBEA | SPEA | SPEA-II |
| ELI105 |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# referencje