`­­­­­­

Dominik Guz , Włodzimierz Kesler, Dominik Kawczak , Damian KRYSTA

Problem TSP w ujęciu dwukryterialnym oraz ewolucyjnym

2016

Spis treści

[TSP Błąd! Nie zdefiniowano zakładki.](#_Toc472202245)

[Biblioteka ParadisEO 4](#_Toc472202246)

[Program Błąd! Nie zdefiniowano zakładki.](#_Toc472202247)

[Wymagania 4](#_Toc472202248)

[Architektura 4](#_Toc472202249)

[Interfejs użytkownika 5](#_Toc472202250)

[Kod źródłowy 5](#_Toc472202251)

[referencje 14](#_Toc472202252)

# Problem

## 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 [1], dlatego też zastosowanie algorytmów ewolucyjnych do jego rozwiązania jest optymalne w przypadku dużych zestawów danych których przetworzenie w czasie rzeczywistym jest niemożliwe [2].

## Optymalizacja wielokryterialna

Ze względu na istnienie wielu kryteriów nie jest możliwe znalezienie jednego optymalnego rozwiązania ponieważ każde rozwiązanie faworyzuje jedno z kryteriów. Zbiór rozwiązań optymalnych nazywany Pareto frontem to rozwiązania które są niezdominowane.

Znanych jest wiele algorytmów optymalizacji wielokryterialnej które mogą być z powodzeniem zastosowane z algorytmami ewolucyjnymi. Przykładami takich algorytmów są NSGA [3], IBEA, MOGĄ oraz SPEA.

# Program

## Biblioteka ParadisEO

Biblioteka paradiseo [4] jest zbiorem narzędzi metaheurystycznych. Została ona napisana w zgodności z ANSI C++. Autorem biblioteki jest miedzy innymi organizacja INRIA.

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

## 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 QT Framework które są udostępniane przez organizację Digia na licencji LGPL.

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

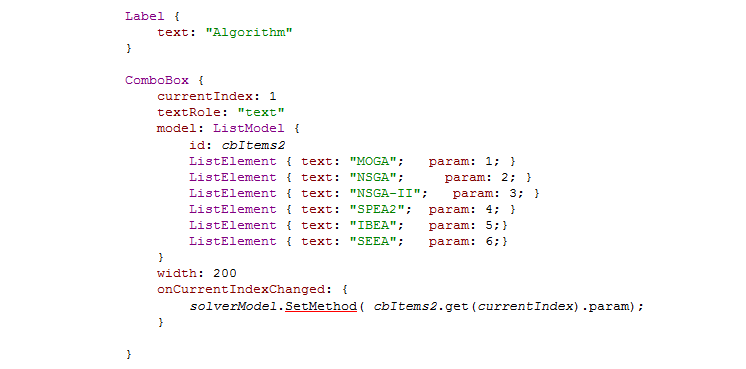
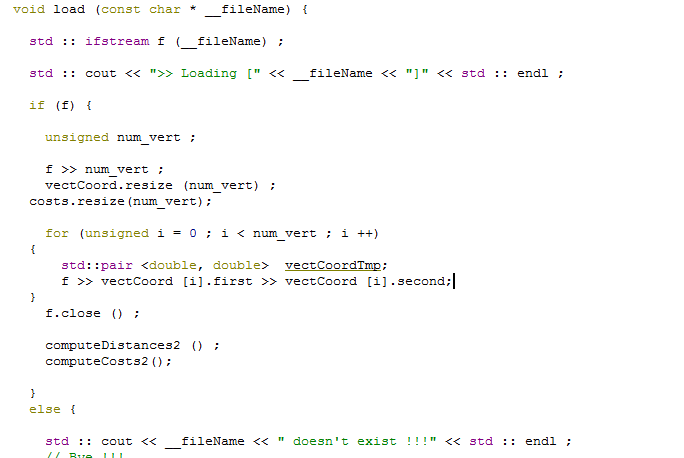
Podczas uruchamiania aplikacji rejestrowane są komponenty rozpoznawane w graficznym interfejsie użytkownika co pozwala na realizację wzorca MVC.

Po uruchomieniu program przechodzi do głównego widoku w którym 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.

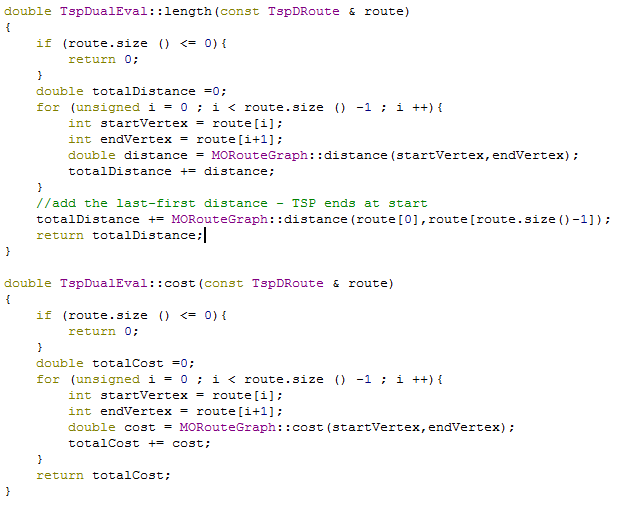
Dane zaczytywane są z pliku zlokalizowanego w folederze ‘benchs’ aplikacji. Przedstawia to poniższy kod.



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.



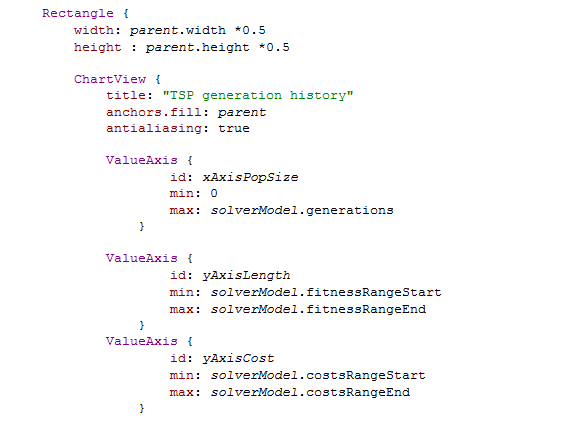
Osobniki populacji oceniane są przez algorytm ewolucyjny za pomocą obiektu TspDualEval, zawierającego funkcje ewaluacji ścieżki względem długości oraz kosztu.



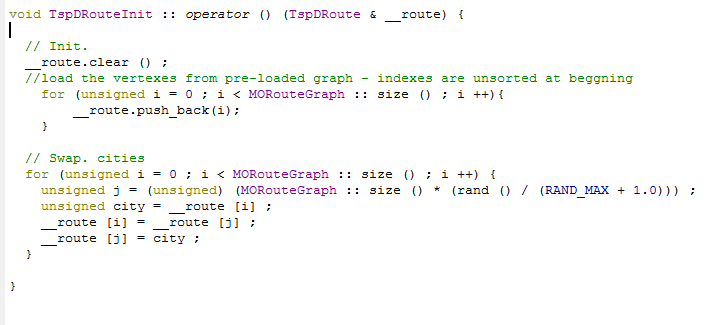
Rysowanie ma miejscie poprzez wbudowane funkcje ‘drawLine’ rysujące linie pomiędzy dwoma punktami reprezentowanymi przez obiekty QPoint.

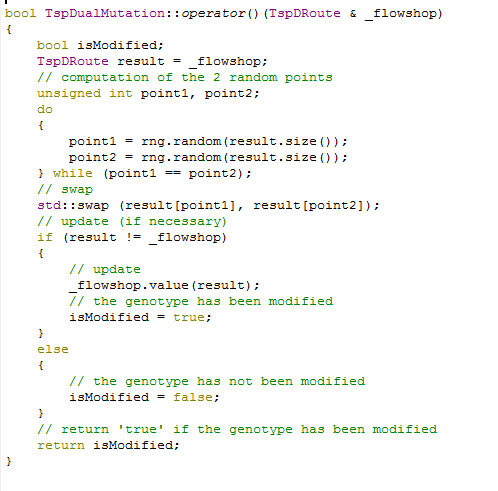


Wyświetlanie postępu algorytmy ewolucyjnego i wartości pareto-optymalnych rozwiązań wyświetlane są na wykresie zawierającym osobne osie wartości dla długości oraz kosztu trasy co przedstawia poniższy kod.

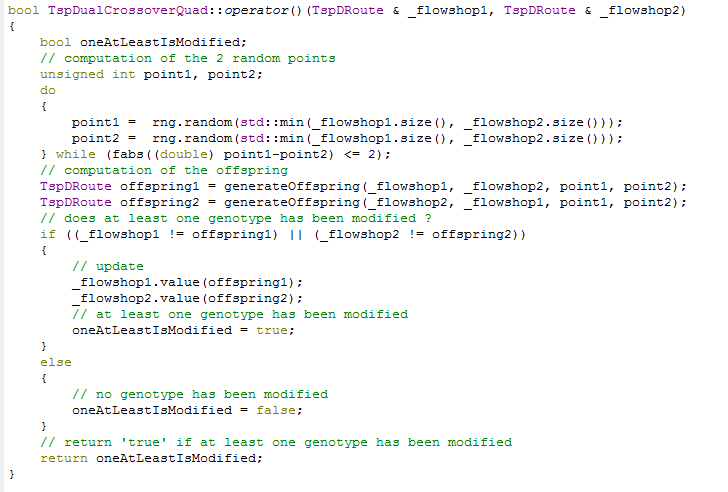
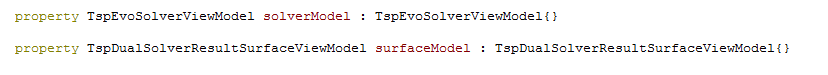


Inicjalizacja trasy następuje poprzez wczytanie listy wierzchołków a następnie losowe ich przetasowanie.





Dane aplikacji wyświetlane są za pomocą połączenia graficznego interfejsu użytkownika z modelem danych, dzięki czemu możliwe jest automatyczne aktualizowanie wyświetlanych wyników w przypadku zmiany wartości.



# Środowisko testowe

## Założenia

Testy zostały wykonane na standardowych zestawach danych o nazwie TSP95, co pozwala porównać wyniki z innymi implementacjami oraz algorytmami.

## Sprzęt

Testy zostały przeprowadzone na jednostce obliczeniowej o następujących parametrach :

Procesor : i7 1.2Ghz

System operacyjny : MS Windows 10

# Wyniki

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Zbiór danych /Algorytm | MOGA | NSGA | NSGA-II | SPEA2 | IBEA |
| ALI535 (ms) | 3592 | 2361 | 2927 | 1300 | 2184 |
| Population size | 70 | 70 | 70 | 70 | 70 |
| Generations | 150 | 150 | 150 | 150 | 150 |
| Pareto solution ( f1) | 134312 | 10654 | 9459 | 17431 | 10178 |
| Pareto solution ( f2) | 148 | 149 | 149 | 147 | 148 |

# Podsumowanie

Wyniki testów pokazują znaczące róznice w czasie wykonania oraz skuteczności poszczególnych algorytmów. Najszybszym algorytmem okazał się SPEA2, jednak przy najgorszym wyniku optymalizacji. Najlepsza optymalizacja została wykonana przez algorytm NSGA-)) który był drugim co do czasu wykonywania.

# referencje

|  |  |
| --- | --- |
| [1] | D. B. F. a. A. Gaspar-Cunha, „On Multi-Objective Evolutionary Algorithms”. |
| [2] | E. Zitzler, Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications, Zurich: Swiss Federal Institute of Technology, 1998. |
| [3] | I. Diakonikolas, Approximation of Multiobjective Optimization Problems, COLUMBIA UNIVERSITY, 2011. |
| [4] | M. L. a. L. T. Eckart Zitzler, SPEA2: Improving the Strength Pareto Evolutionary Algorithm, Swiss Federal Institute of Technology (ETH) Zurich. |
| [5] | C. A. C. Coell, Metaheuristics for Multiobjective Optimization, Av. Instituto Polit´ecnico Nacional . |
| [6] | R. Agrawal, „Application of the Modified 2-opt and Jumping Gene Operators in Multi-Objective Genetic Algorithm to solve MOTSP”. |
| [7] | A. Jaszkiewicz, „Genetic local search for multi-objective combinatorial optimization,” 2001. |