POLITECHNIKA WARSZAWSKA WYDZIAŁ EITI INSTYTUT INFORMATYKI

Dokumentacja ALHE SNDLib sieć Polska

Kacper Kula & Wojciech Sitek
25 stycznia 2021

1 ZADANIE PROJEKTOWE

Przy użyciu Algorytmu Ewolucyjnego zaprojektować sieć teleinformatyczną minimalizującą liczbę użytych systemów teletransmisyjnych o różnej modularności m (m = 1, m > 1 i m » 1). Sieć opisana za pomocą grafu G = (N, E), gdzie N jest zbiorem węzłów, a E jest zbiorem krawędzi. Funkcja pojemności krawędzi opisana jest za pomocą wzoru, określonego w zadaniu. Zbiór zapotrzebowań D, pomiędzy każdą parą węzłów opisuje macierz zapotrzebowań i jest dany. Dla każdego zapotrzebowania istnieją co najmniej 3 predefiniowane ścieżki. Sprawdzić, jak wpływa na koszt rozwiązania agregacja zapotrzebowań, tzn. czy zapotrzebowanie realizowane jest na jednej ścieżce (pełna agregacja), czy dowolnie, na wszystkich ścieżkach w ramach zapotrzebowania (pełna dezagregacja). Dobrać optymalne prawdopodobieństwo operatorów genetycznych oraz liczność populacji. Dane pobrać ze strony http://sndlib.zib.de/home.action, dla sieci polska.

2 WYJAŚNIENIE POJĘĆ

2.1 MODULARNOŚĆ

Modularność jest to ...

2.2 ALGORYTM EWOLUCYJNY

Zgodnie z wykładami prof. Jarosława Arabasa, Algorytm Ewolucyjny opisuje się za pomocą algorytmu **??**. Na algorytm składają się następujące funkcjonalności:

1. inicjalizacja populacji,

2. główna pętla algorytmu ewolucyjnego,
3. mutacja (ang. *mutation*),
4. krzyżowanie (ang. *crossover*),
5. selekcja (ang. *selection*, oznaczane jako *select*),
6. warunek zatrzymania,
7. funkcja celu.

3 Założenia projektu

Projekt wykonywany jest w ramach przedmiotu Algorytmy Heurystyczne (ALHE) w semestrze zimowym 2020 na Wydziale EiTI Politechniki Warszawskiej. Prowadzącym projekt jest dr inż. Stanisław Kozdrowski.

Implementacja projektu jest wykonywana w języku Python. Algorytm ewolucyjny posiada własną implementację i nie jest zaczerpnięty z bibliotek zewnętrznych języka Python. Biblioteki narzędziowe języka Python, którymi się wspomagano, to między innymi biblioteki:

- xml - do przeprowadzenia parsowania pliku XML do obiektów Python (rozdział 4)
- tqdm - ukazywania paska postępu
- random - do generacji liczb pseudolosowych (użyto ziarna (ang. *seed*))
- logging - do logowania informacji pomocniczych w czasie działania programu
- json - do zapisu i odczytu plików JSON.

4 Przygotowanie danych

Dla sieci „polska", pobrano plik XML oraz TXT ze wszystkimi informacjami dla danych dotyczących terytorium Polski. Następnie, przeprowadzono analizę budowy plików oraz znaczenia poszczególnych terminów. Wyeliminowano z programu dane nieistotne dla rozwiązywanego problemu. Analiza znaczenia terminów przeprowadzona jest w rozdziale 2.

Następnie, zbudowano parser, konwertujący wybrane części pliku XML do obiektów języka Python. Budowa słowników i list, w których były przechowywane informacje o sieci, została przedstawiona na poniższym przykładzie:

```
nodes = ['Gdansk', 'Bydgoszcz', ...]

link_keys = [
    (0, 1),
    (0, 2),
    (1, 3),
```

```
]

links_array = [
    Link_0_1_data,
    Link_0_2_data,
    ...,
    Link_1_3_data,
    ...,
]

Link_a_b_data = {
    'setupCost': 156.0,
    'capacity0': 155.0,
    'capacity1': 622.0,
    'cost0': 156.0,
    'cost1': 468.0,
}

demand_array = [
    Demand_0_1_data,
    Demand_0_2_data,
    ...
]

Demand_a_b_data = {
    'demand': 195.00,
    'admissiblePaths': [
        [(0,2), (1,2)],
        [(0,10), (1,10)],
        [(0,2), (2,9), (7,9), (1,7)],
        ...
    ]
}

demand_keys = [
    (0, 1),
    (0, 2),
    (1, 2),
]
```

5 Charakterystyka implementacji algorytmu

5.1 Mutacja

itp

6 Analiza wyników