



Wydział Elektroniki i Technik Informatycznych

POLITECHNIKA WARSZAWSKA

Podstawy Sztucznej Inteligencji

PROJEKT WG.AE.1

Marcin Baran 259804

Łukasz Kilaszewski 259822

Mateusz Perciński 259827

13 czerwca 2017

Spis treści

1	Definicja problemu	2
1.1	Miasta	2
1.2	Trasa	2
1.3	Zużycie paliwa	2
1.4	Algorytm ewolucyjny	3
1.4.1	Krzyżowanie	3
1.4.2	Mutacja	3
2	Implementacja	4
2.1	Struktura programu	4
2.2	Instrukcja dla użytkownika	4
3	Testy i osiągnięte rezultaty	6
3.1	Test1	6
3.2	Czas obliczeń	6
3.3	Wnioski	6

Treść zadania

WG.AE1 Rozwożenie mebli

Zaplanować trasę samochodu ciężarowego rozwożącego meble. Każdy mebel ma określoną wagę oraz miasto przeznaczenia. Zużycie paliwa przez samochód ciężarowy jest zależne od masy przewożonego ładunku. Zaplanować trasę rozwiezienia mebli która jest optymalna ze względu na zużycie paliwa. Program powinien na bieżąco prezentować jakość znalezionej rozwiązania w funkcji numeru pokolenia.

Podział pracy

Marcin Baran - Definicja zadania

Łukasz Kilaszewski - Implementacja programu

Mateusz Perciński - Raport oraz wnioski

Rozdział 1.

Definicja problemu

Projekt polega na rozwiązaniu zmodyfikowanego zadania komiwojażera. Optymalizowana jest trasa (kolejność odwiedzonych miast) ciężarówki rozwożącej meble ze względu na zużycie paliwa. Na podstawie treści zadania przyjęto, że dla każdego miasta na planowanej trasie, znana jest masa mebli, które mają być do niego dostarczone, a zużycie paliwa jest zależne masy przewożonego towaru. W kolejnych podpunktach opisano przyjęte założenia, które nie wynikają ściśle z treści zadania.

1.1. Miasta

Przyjęto, że zadanie rozwiązywane jest dla większych polskich miast, których lista wraz ze współrzędnymi geograficznymi została pobrana z Odległość między miastami zdefiniowana jest miarą euklidesową, jako odległość w linii prostej. Założono, że użytkownik będzie mógł wybrać miasta, mające się znaleźć na trasie przejazdu, i każdemu z nich przypisać masę mebli, które mają być w nim zostawione.

1.2. Trasa

Założenia dotyczące trasy przejazdu ciężarówki:

- trasa zaczyna się i kończy tym samym, określonym na początku mieście,
- każde miasto jest odwiedzane tylko raz,
- odległości między miastami są jednakowe w obydwu kierunkach (problem komiwojażera jest symetryczny),
- ciężarówka zostawia w każdym mieście wszystkie, predestynowane do niego, meble. Jej masa zmniejsza się. Ostatni, powrotny odcinek, pokonywany jest bez ładunku.

1.3. Zużycie paliwa

Przyjęto, że zużycie paliwa jest wprostproporcjonalne do masy pojazdu, a w związku z tym z masą przewożonych mebli. Koszt przejazdu pomiędzy dwoma miastami, czyli zużycie paliwa na trasie między nimi, określono wzorem

$$cost_{section} = m_a * mass + m_b) \cdot dist, \quad (1.1)$$

gdzie:

$mass$ - masa mebli znajdujących się w ciężarówce na tym odcinku drogi,

$dist$ - odległość między miastami,

m_b - współczynnik określający stałe spalanie ciężarówki (bez obciążenia) $\left[\frac{liter}{km}\right]$

m_a - współczynnik wpływu dodatkowego obciążenia na spalanie ciężarówki $\left[\frac{liter}{kg \cdot km}\right]$

1.4. Algorytm ewolucyjny

Do rozwiązania problemu zastosowano algorytm ewolucyjny ($\mu + \lambda$). Przyjęto, że osobnikiem jest wytyczona trasa, a jego genotypem wektor liczb całkowitych określający kolejność odwiedzanych miast. Numeracja odwiedzanych miast jest zgodna z kolejnością na liście definiującej problem (wypisującej wszystkie miasta, które mają zostać odwiedzone wraz z masami predestynowanych do nich mebli). Miasto początkowe (i jednocześnie końcowe) nie występuje w wektorze. Założono, że populacją dla algorytmu ewolucyjnego jest zbiór takich wektorów.

1.4.1. Krzyżowanie

krzyżowanie,

1.4.2. Mutacja

mutacja,

Rozdział 2.

Implementacja

Program rozwiązujący zdefiniowany wcześniej problem komiwojażera zaimplementowano w języku Python. Wykorzystano następujące biblioteki:

- **numpy** - umożliwiła korzystanie z macierzy,
- **geopy** - wykorzystano funkcję `great_circle` w celu obliczania odległości między dwoma miastami znając ich współrzędne geograficzne,
- **matplotlib** - posłużyła do generowania czytelnych wykresów.

2.1. Struktura programu

Napisany program ma strukturę modułową. Głównym celem było wyodrębnienie części odpowiedzialnej za realizację obliczeń według algorytmu ewolucyjnego od części definiującej problem do rozwiązania.

2.2. Instrukcja dla użytkownika

Plik *cities.csv* zawiera większe miasta wraz z ich współrzędnymi geograficznymi. Spośród nich należy wybierać te, które mają być odwiedzone przez ciężarówkę. Definiowanie problemu do rozwiązania polega na uzupełnianiu pliku *task.csv*. W pierwszym wierszu znajduje się nazwa miasta startowego (i jednocześnie końcowego). Następne wiersze zawierają nazwy miast, które mają zostać odwiedzone wraz z masą mebli, które mają być do nich dostarczone. Wiersz ma format *nazwa_miasta;masa_mebli*. Masa mebli powinna być podana jako liczba, z przynajmniej jednym miejscem dziesiętnym np. 100.0, 123.4.

Parametry algorytmu ewolucyjnego można zdefiniować jako zmieniając wartości argumentów funkcji **init** obiektu **alg**:

- **population** - ilość osobników wybieranych do populacji w każdej iteracji algorytmu,
- **cross** - ilość osobników, które mają być krzyżowane w każdym kroku iteracji algorytmu,
- **stop_iters** - ilość iteracji, po której, w przypadku braku poprawy wyniku, działanie algorytmu jest przerywane.

Po zdefiniowaniu problemu do rozwiązania oraz parametrów algorytmu, należy uruchomić plik *main.py*. W oknie konsoli, będą wypisywane genotypy najlepszych osobników z populacji danej iteracji wraz z obliczoną dla nich wartością funkcji oceny, którą jest określona

jako zużycie paliwa. Po zakończeniu działania algorytmu najlepsze wartości dla najlepszych osobników każdej iteracji prezentowane się w formie wykresu.

Rozdział 3.

Testy i osiągnięte rezultaty

3.1. Test1

3.2. Czas obliczeń

3.3. Wnioski

wnioski dotyczące osiągniętych rezultatów.

wykres dla różnej wielkości populacji, jakiś teścik dla zadania niesymetrycznego : np. dojazd do miasta z innego jest niemożliwy,

lub całkiem zablokować przejazd między dwoma miastami.