Implementacja i analiza efektywności algorytmu genetycznego (ewolucyjnego) dla wybranego problemu optymalizacji

Jakub Grzegocki (264009)

styczeń, 2024

Sprawozdanie z trzeciego zadania projektowego kursu "Projektowanie efektywnych algorytmów"

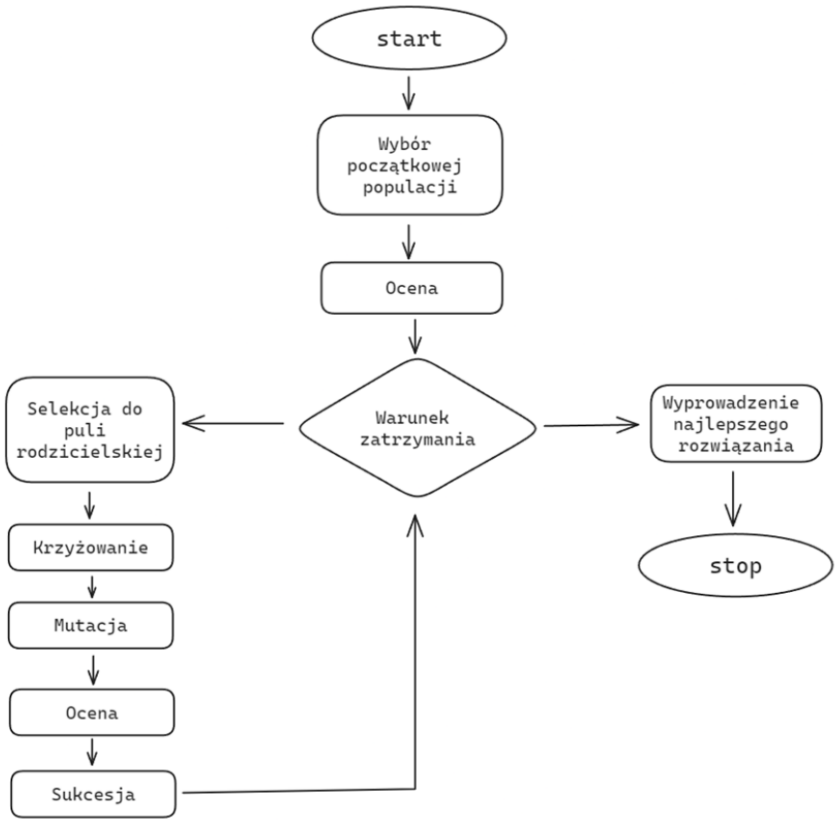
Spis treści

1 Wstęp teoretyczny

1.1 Ogólny opis algorytmu genetycznego

Algorytmy genetyczne to zaawansowana metoda heurystyczna wykorzystywana do rozwiązywania złożonych problemów optymalizacyjnych, inspirowana procesami ewolucyjnymi zachodzącymi w naturze. Kluczowym założeniem tych algorytmów jest imitowanie mechanizmów takich jak selekcja naturalna, krzyżowanie i mutacja w celu iteracyjnego poprawiania rozwiązań problemu. W kontekście ATSP, gdzie celem jest znalezienie najkrótszej możliwej trasy obejmującej wszystkie punkty (miasta), algorytm genetyczny stara się optymalizować ścieżki, które są reprezentowane przez chromosomy, czyli uporządkowane sekwencje miast.

1.2 Schemat blokowy działania algorytmu



Rysunek 1: schemat blokowy GA

1.3 Metody krzyżowania

Metoda **Position-Based Crossover** koncentruje się na zachowaniu pozycji niektórych elementów z jednego z rodziców w potomku. Proces ten wygląda następująco:

1. **Inicjalizacja Potomka**: Potomek jest inicjalizowany z wartościami -1, co oznacza, że żaden element nie został jeszcze przypisany.
2. **Wybór Pozycji z Rodzica 1**: Losowo wybierane są pozycje z pierwszego rodzica, które będą zachowane w potomku.
3. **Uzupełnienie Potomka z Rodzica 2**: Potomek jest uzupełniany elementami z drugiego rodzica na pozycjach, które pozostały puste, uwzględniając tylko elementy, które jeszcze się nie pojawiły w potomku, aby uniknąć duplikatów.

Przykład:

1) Wybór rodziców:

$$\text{Rodzic1} = [1, 2, 3, 4, 5]$$
$$\text{Rodzic2} = [3, 4, 5, 1, 2]$$

2) Wybieramy losowe pozycje np. 2 i 4. Potomek będzie miał postać :

$$[0, 2, 0, 4, 0].$$

3) Następnie uzupełniamy potomka elementami z rodzica drugiego, dając ostateczny wynik:

$$[3, 2, 5, 4, 1].$$

Metoda **Ordered Crossover** skupia się na przeniesieniu uporządkowanego segmentu z jednego rodzica do potomka, a następnie wypełnieniu reszty potomka elementami z drugiego rodzica.

1. **Inicjalizacja Potomka**: Potomek jest inicjalizowany bez wartości w .
2. **Losowe Wybranie Segmentu z Rodzica 1**: Losowo wybierany jest ciągły segment z pierwszego rodzica, który jest następnie kopiowany do potomka na te same pozycje.
3. **Wypełnienie Reszty Potomka z Rodzica 2**: Reszta potomka jest wypełniana elementami z drugiego rodzica, zaczynając od końca skopiowanego segmentu i idąc dalej w kółko, ale pomijając te elementy, które już znajdują się w potomku.

Przykład:

1) Wybór rodziców:

$$\text{Rodzic1} = [8, 4, 7, 3, 6, 2, 5, 1, 9]$$
$$\text{Rodzic2} = [9, 7, 5, 1, 6, 8, 2, 3, 4]$$

2) Wybieramy losowe pozycje np. segment od pozycji 3 do 6 i przepisujemy do potomka zawierające się w nim miasta z rodzica pierwszego. Potomek przybiera formę:

$$\text{potomek} = [0, 0, 0, 3, 6, 2, 5, 0, 0]$$

3) Następnie uzupełniamy potomka miastami z drugiego rodzica w tej samej kolejności pomijając miasta występujące w potomku dając ostateczny wynik:

$$\text{potomek} = [9, 7, 8, 3, 6, 2, 5, 1, 4]$$

1.4 Metody mutacji

Mutacja przez Swap
Metoda *swap* polega na zamianie miejscami dwóch losowo wybranych elementów w chromosomie.

- **Proces**: Wybierz dwa losowe elementy w chromosomie i zamień je miejscami.
- **Przykład**: Rozważmy chromosom [1, 2, 3, 4, 5]. Jeśli losowo wybrano elementy na pozycjach 2 i 4, po mutacji otrzymamy chromosom [1, 4, 3, 2, 5].

Mutacja przez Invert
Metoda *invert* polega na odwróceniu kolejności elementów w losowo wybranym segmencie chromosomu.

- **Proces**: Wybierz losowy segment w chromosomie i odwróć kolejność jego elementów.
- **Przykład**: Dla chromosomu [1, 2, 3, 4, 5] i wybranego segmentu od pozycji 2 do 4, po inwersji otrzymamy chromosom [1, 4, 3, 2, 5].

2 Metoda selekcji

W algorytmie została zaimplementowana selekcja rankingowa. Przypisuje ona każdemu osobnikowi rangę na podstawie funkcji fitness. Proces ten można opisać następująco:

1. **Obliczanie Fitness**: Dla każdej ścieżki w populacji obliczany jest jej koszt, który służy jako miara fitness. Niższy koszt oznacza lepsze przystosowanie.
2. **Sortowanie i Ranking**: Ścieżki są sortowane na podstawie ich kosztu, a następnie każdej ścieżce przypisywana jest ranga na podstawie jej pozycji.
3. **Przydzielenie Prawdopodobieństw**: Każdej ścieżce przypisywane jest prawdopodobieństwo selekcji na podstawie jej rangi. Wyższa ranga daje większe prawdopodobieństwo.
4. **Obliczenie Prawdopodobieństwa**: Dla każdego osobnika o randze R w populacji o rozmiarze N, prawdopodobieństwo selekcji wynosi $\frac{R}{\frac{N \times (N+1)}{2}}$.

Metoda ta promuje lepsze ścieżki, jednocześnie zachowując różnorodność w populacji przez dawanie szansy ścieżkom o niższej randze.

3 Opis klas

- **Position-Based Crossover (PBX)**
  - **Nazwa:** `PositionBasedCrossover`
  - **Parametry:** `rodzic1`, `rodzic2`
  - **Opis:** Metoda wybiera losowo pozycje z pierwszego rodzica, które są kopiowane do potomka. Następnie, pozostałe miejsca w potomku są wypełniane wartościami z drugiego rodzica, zachowując kolejność i unikając duplikatów.
- **Ordered Crossover (OX)**
  - **Nazwa:** `OrderedCrossover`
  - **Parametry:** `rodzic1`, `rodzic2`
  - **Opis:** Wybierany jest losowy segment z pierwszego rodzica, który jest kopiowany do potomka. Pozostałe miejsca są wypełniane wartościami z drugiego rodzica w oryginalnej kolejności, ale pomijając wartości już obecne w potomku.
- **Mutacja**
  - **Nazwa:** `mutacja`
  - **Parametry:** `sciezka`
  - **Opis:** Metoda polega na losowym wybraniu dwóch pozycji w ścieżce i zamianie ich miejscami, co wprowadza niewielkie losowe zmiany w genotypie osobnika.
- **Mutacja Inwersyjna**
  - **Nazwa:** `mutacjaInwersyjna`
  - **Parametry:** `sciezka`
  - **Opis:** Metoda wybiera losowo dwa punkty w ścieżce i odwraca kolejność elementów między tymi punktami, zmieniając sekwencję genów, ale zachowując informacje o ich wzajemnym położeniu.
- **Rank Selection**
  - **Nazwa:** `rankSelection`
  - **Parametry:** Brak
  - **Opis:** Osobniki są klasyfikowane według ich przystosowania, a następnie przypisywane są im rangi. Wyższa ranga oznacza większe prawdopodobieństwo wyboru do reprodukcji, promując różnorodność genetyczną.
- **Metoda** `wykonaj`
  - **Nazwa:** `wykonaj`
  - **Parametry:** `uzyjPBX`, `uzyjInwersji`
  - **Opis:** Główna funkcja algorytmu, kontrolująca proces ewolucji. Generuje nową populację przez selekcję, krzyżowanie i mutację. Umożliwia wybór między PBX/OX i standardową/mutacją inwersyjną. Monitoruje stagnację i dostosowuje współczynnik mutacji.

4 Wyniki badań

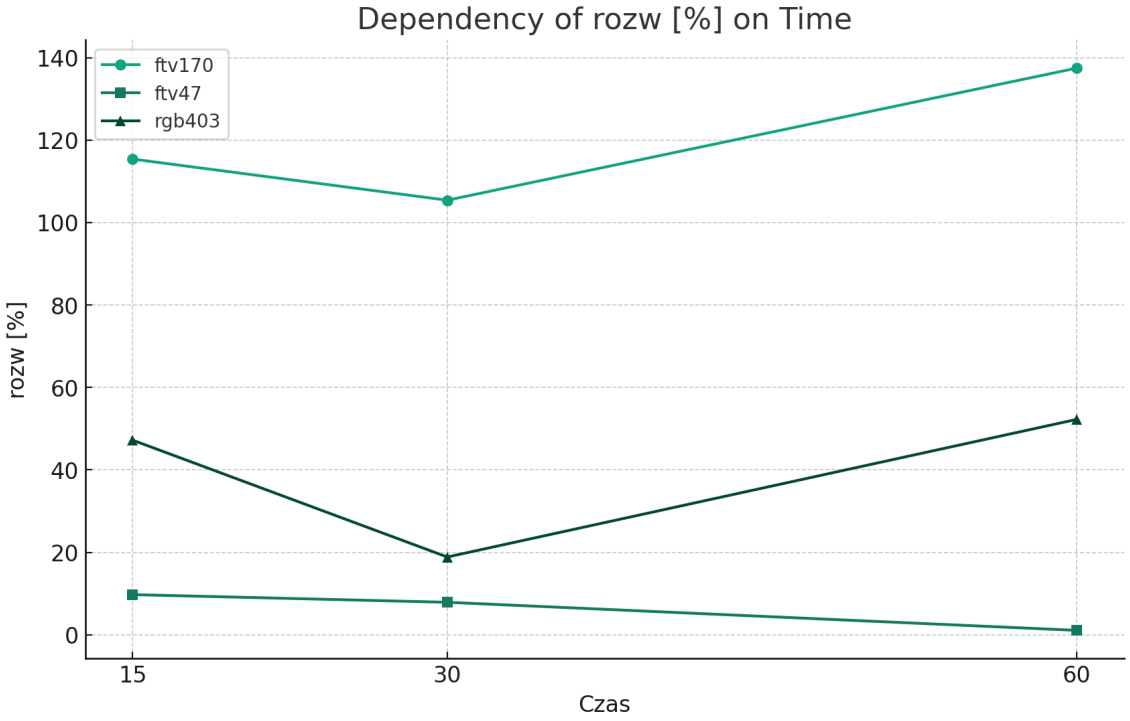
				ftv170		ftv47		rgb403	
krzyz	mut	pop	czas	rozw	b.wzgl [%]	rozw	b.wzgl [%]	rozw	b.wzgl [%]
OX	SWAP	20	15	5934	115,3902	1949	9,740991	3629	47,2211
		50	30	5659	105,4083	1916	7,8828829	2930	18,8641
		200	60	6542	137,4592	1795	1,0698198	3753	52,25152
	INVERT	20	15	6625	140,4719	1814	2,1396396	3896	58,05274
		50	30	5941	115,6443	1917	7,9391892	2890	17,24138
		200	60	7914	187,2595	1933	8,8400901	3004	21,86613

Rysunek 2: Wyniki testów dla róznych populacji czasów i współczynników

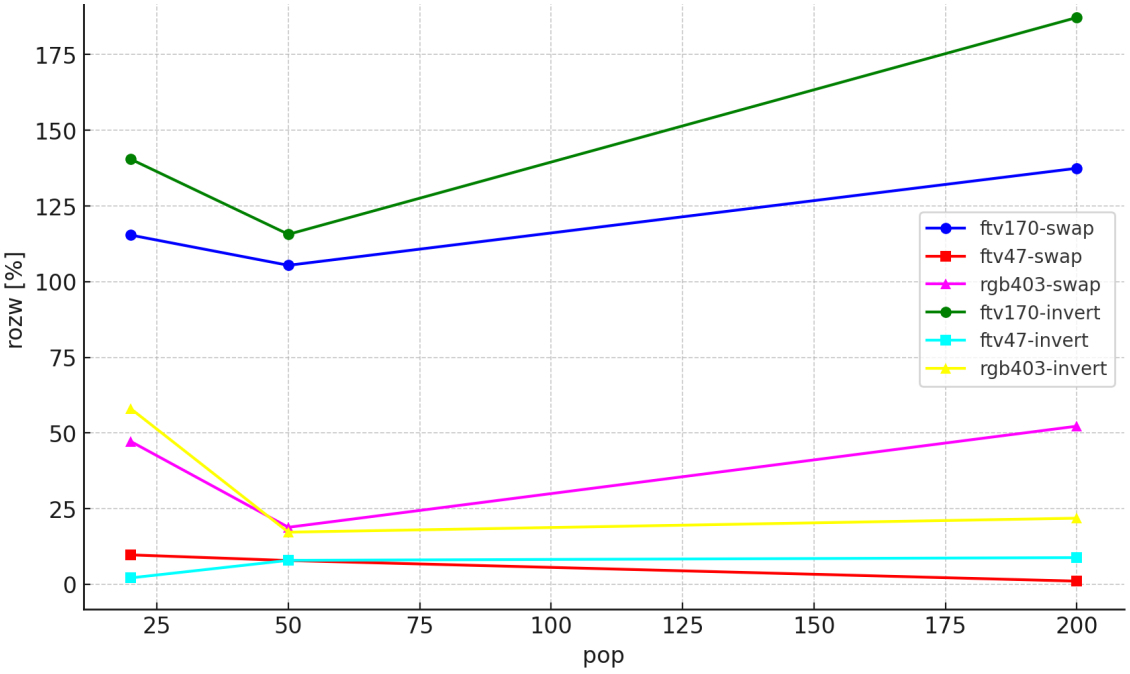
	swap					
	0.02		0.05		0.1	
	rozw	b.wzgl [%]	rozw	b.wzgl [%]	rozw	b.wzgl [%]
170	5261	90,96189	5628	104,2831	5990	117,4229
47	1789	0,731982	1846	3,941441	1841	3,65991
403	3500	41,98783	3620	46,85598	2930	18,8641

Rysunek 3: wyniki rozwiązań dla różnych wsp. mutacji w mutacji typu swap

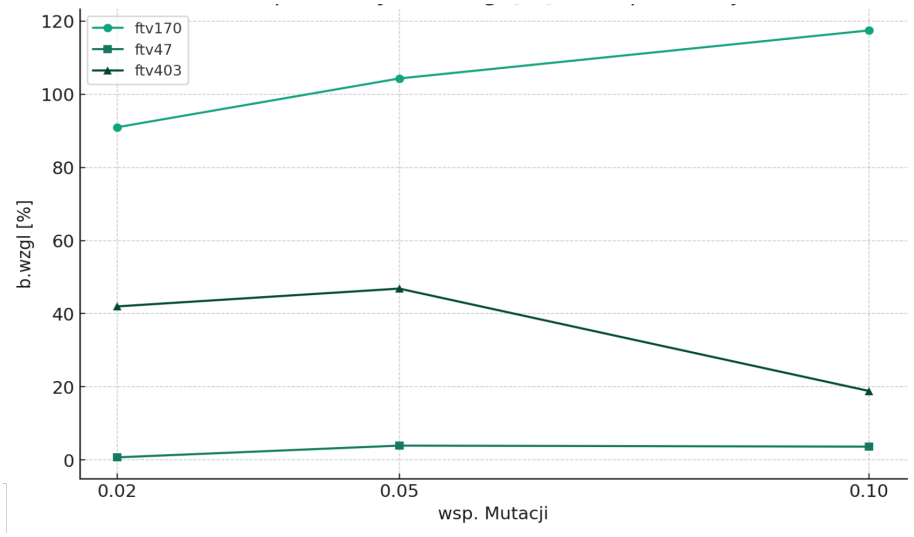
5 Wykresy



Rysunek 4: wykres zależności błędu względnego od czasu działania



Rysunek 5: wykres zależności błędu względnego od populacji dla mutacji swap i invert



Rysunek 6: wykres zależności błędu względnego wsp. mutacji dla mutacji typu swap

6 Porównanie najlepszych wyników z symulowanym wyżarzaniem

Najlepsze odnalezione ścieżki przez algorytm genetyczny dla ftv170, ftv47 i rbg403 to kolejno 5659, 1795 i 2890. Dla SA wyniki były równe kolejno 3822, 1797 i 2830 co oznacza, że poziom wydajności algorytmów jest porównywalny dla podobnych czasów działania.

7 Wnioski

Algorytm genetyczny jest skutecznie działającym algorytmem w problemie TSP. Jego działanie jest zależne od wielu czynników takich jak metody krzyżowania, mutacji, sposób selekcji, wielkość populacji lub współczynniki, które można ustawiać na różne sposoby. Jego podobieństwo i zaczerpnięcie schematów z natury upraszcza zrozumienie działania i implementacji algorytmu. Wyniki są satysfakcjonujące jak na czas przeszukiwania mapy rozwiązań przez algorytm. Niestety w porównaniu do symulowanego wyżarzania nie działa on znacząco lepiej więc nie widzę różnic w użyciu GA dla tego problemu.

8 Źródła

1. https://www.geeksforgeeks.org/traveling-salesman-problem-using-genetic-algorithm/
2. https://pl.wikipedia.org/wiki/Algorytm_genetyczny
3. slajdy z wykładu
4. Z.Michalkiewicz, D.B.Vogel, Jak to rozwiązać czyli nowoczesna heurystyka, WNT 2006