Sprawozdanie - projekt 3.

Implementacja i analiza efektywności algorytmu genetycznego dla problemu komiwojażera.

Autor: Michał Bańka, 235051
Prowadzący: mgr inż. Antoni Sterna
Termin zajęć: Czwartek, 7:30

1. OPIS PROBLEMU

Problem komiwojażera (ang. TSP – Travelling Salesman Problem) jest problemem, który polega na znalezieniu najkrótszego cyklu Hamiltona w grafie. Oznacza to, że ścieżka powinna zawierać wszystkie wierzchołki dokładnie jeden raz i przez każdą krawędź przechodzić również najwyżej jeden raz. Dodatkowo ostatni wierzchołek powinien być połączony z pierwszym, tak aby ścieżka stała się cyklem.

Dla 4 wierzchołków mamy 3*2*1 możliwości wyboru ścieżek, dla 5 wierzchołków 4*3*2*1 ścieżek itd. Tak więc dla *n* wierzchołków ilość ścieżek do sprawdzenia czy są najkrótszymi cyklami Hamiltona wyniesie (n-1)*(n-2)*…*2*1 = (n-1)! . Najprostsza metoda algorytmu rozwiązującego problem będzie miała złożoność (n-1)!.

Sprowadza to problem komiwojażera do zbioru problemów NP-trudnych, a właściwie do jego podzbioru problemów NP.-zupełnych. Oznacza to, że problem jest możliwy do sprawdzenia poprawności w czasie wielomianowym, ale nie jest możliwe znalezienie jego rozwiązania w czasie wielomianowym. Złożoność problemu komiwojażera jest wykładnicza, gdyż: $n!>2^n$ już od n=4.

2. OPIS ALGORYTMÓW

A. ALGORYTM GENETYCZNY

To rodzaj algorytmu heurystycznego przeszukującego przestrzeń problemu w celu wyszukania możliwie najlepszych rozwiązań. Sposób działania algorytmu jak sama nazwa wskazuje przypomina zjawisko ewolucji biologicznej, a sam zalicza się do grupy algorytmów ewolucyjnych. Jeden cykl Hamiltona można porównać to genotypu. Długość tego cyklu byłaby odpowiednikiem fenotypu z biologii. Na podstawie wielu osobników, czyli w przypadku problemu komiwojażera, na podstawie wielu cyklów tworzona jest populacja. Właśnie ta populacja jest przedmiotem badań algorytmu. Zbiór cykli ewoluuje tzn. elementy genotypu są krzyżowane, mutowane i selekcjonowane. Każde wywołanie tych trzech operacji tworzy nowe pokolenie rozwiązań. Populacja powinna utrzymywać stały rozmiar pomiędzy kolejnymi pokoleniami, więc aby utrzymać rozmiar z populacji usuwane są osobniki najsłabsze, a w tym wypadku te, które mają najdłuższą ścieżkę.

Problem rozwiązywany jest przez cykliczne selekcje, krzyżowanie i mutowanie rozwiązań znajdujących się w populacji. Różne metody tych trzech operacji mają największy wpływ na jakość rozwiązań.

3. OPIS IMPLEMENTACJI ALGORYTMÓW I STRUKTUR

a. PRZECHOWYWANIE GRAFU

Graf przechowywany jest w stworzonej specjalnie dla niego klasie Matrix. Klasa ta została stworzona do przechowywania macierzy kwadratowych i wykonywania prostych operacji na niej. Dodatkowo na macierzy można zastosować opisywane w punkcie 2. algorytmy.

Struktura grafu zapisana jest w postaci macierzy sąsiedztwa, gdzie jako pierwsza współrzędna macierzy podaje się wierzchołek, z którego krawędź wychodzi, a jako druga wierzchołek końcowy krawędzi.

Wszystkie struktury alokują pamięć dynamicznie, co zostało uwzględnione podczas mierzenia czasu.

b. IMPLEMENTACJA ALGORYTMU GENETYCZNEGO

Algorytm genetyczny do uruchomienia wymaga wprowadzenia kilku parametrów. Te parametry to między innymi: wielkość populacji, ilość pokoleń do wykonania, współczynnik mutacji i krzyżowania, który pokazuje z jakim prawdopodobieństwem powinny być wykonywane te operacje. Dodatkowo jeśli istnieje więcej niż jedna metoda , którejkolwiek z trzech bazowych operacji, powinien istnieć wybór tej metody.

W tym projekcie stworzone zostały:

- 1 metoda krzyżowania – metoda ta polega na dziedziczeniu informacji z dwóch cykli w populacji. Dziedziczenie polega na wycięciu z pierwszego rodzica pierwszej połowy cyklu i przepisaniu jej do potomka. Druga połowa potomka składa się natomiast z wszystkich pozostałych wierzchołków jakie mogą znaleźć się w cyklu, jednak w takiej kolejności w jakiej występują w drugim rodzicu.

Przykład:

Rodzic_1: 1,4,2,0,3,5
Rodzic_2: 5,2,3,0,1,4
Potomek: $[1,4,2]_{\text{część od Rodzic\_1}}$, $[5,3,0]_{\text{część od Rodzic\_2}}$ => 1,4,2,5,3,0

-1 metoda mutacji – jest to prosta metoda mutacji polegająca na zamianie dwóch wierzchołków w cyklu. Wierzchołki są wybierane losowo.

Przykład:

Rodzic: 0,1,2,3,4,5 => 0,[1],2,3,[4],5 => 0,4,2,3,1,5

-2 metody selekcji – selekcja czyli wybór osobników do wykonania pozostałych dwóch operacji może odbywać się na dwa sposoby. Pierwszy nazwany PROBABILITY polega na ustaleniu wspomnianego wyżej współczynnika krzyżowania i mutacji. Każdy element populacji posiada takie samo prawdopodobieństwo. Drugi nazwany TOP pomija prawdopodobieństwo i krzyżuje ze sobą początkowe x% populacji, a y% mutuje.

Przykład selekcji PROBABILITY:

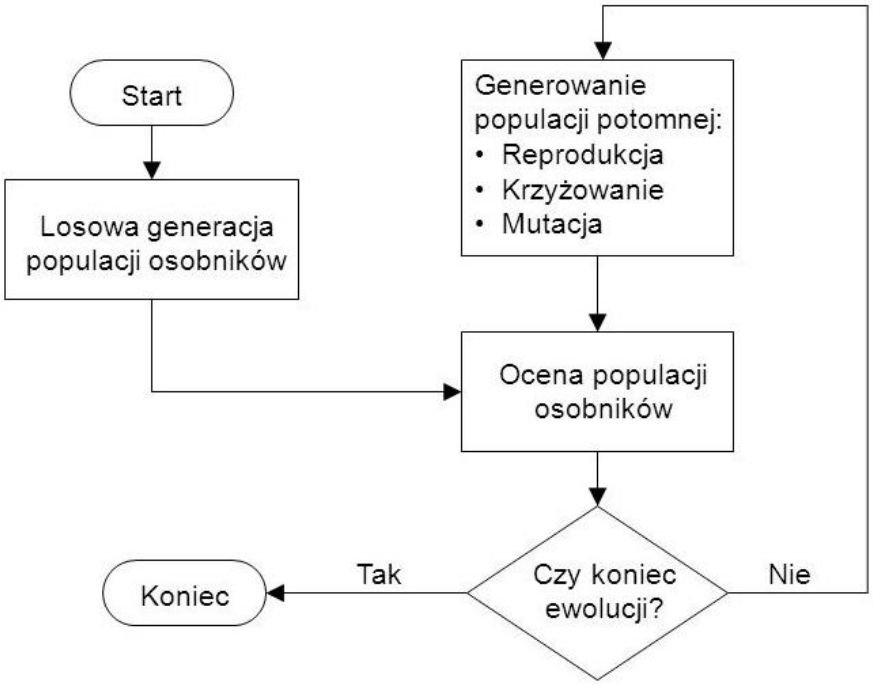
```
probability = losuj( 0 ; 1 );
jeżeli probability < współczynnik_krzyżowania/mutacji
    rodzic2 = losuj( 0 ; rozmiar_populacji )
    krzyżuj/mutuj(rodzic1, rodzic2);
```

W selekcji TOP:

Pierwszy rodzic z góry wiadomo czy jest w początkowym x/y% populacji. Wystarczy nam to do wykonania mutacji. Dla krzyżowania tak samo jak wyżej losowany jest drugi rodzic.

Algorytm może zakończyć się jeśli wykona się określona liczba pokoleń lub zostanie przekroczony czas stopu.

Schemat blokowy dla algorytmu genetycznego:



4. PLAN EKSPERYMENTU

Plan eksperymentu zakłada wykonanie algorytmu po 50 razy dla gotowych macierzy wybranych ze strony internetowej podanej przez prowadzącego. Są to odpowiednio pliki z macierzami o rozmiarach: 34 (ftv33), 70 (ftv70) i 171 (ftv170). Zapisywany jest czas wykonywania algorytmu i długość znalezionego cyklu.

Przyjęty współczynnik krzyżowania to 0,9, a mutacji 0,1. Ilość pokoleń to 100, dla ograniczenia długości wykonywania największych populacji i grafów.

Wykonywanie pomiarów czasów dla każdego grafu było wykonywane dla różnych rozmiarów populacji: 50,100,150,200,250,300.

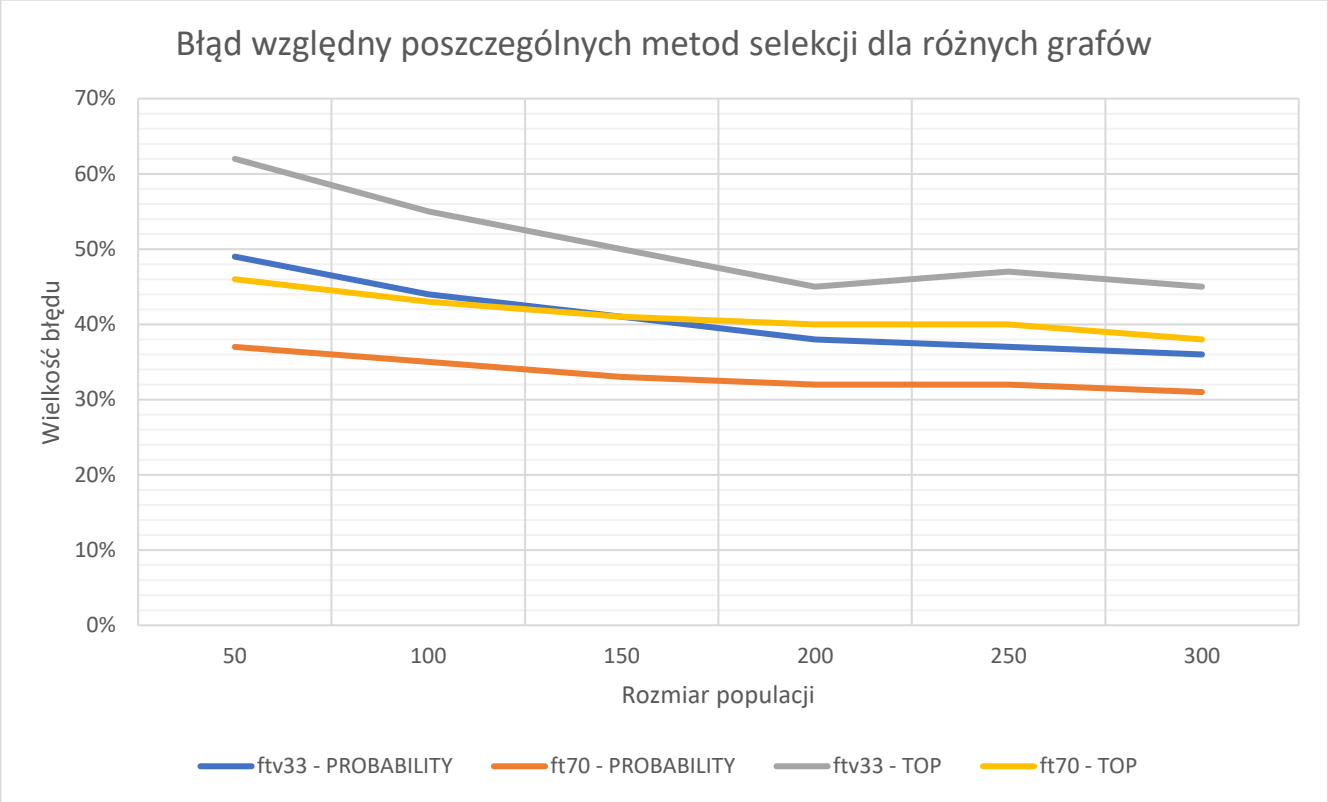
Co więcej dla każdego rozmiaru algorytm był wykonywany metodą selekcji PROBABILITY i TOP.

Pomiary czasu mierzą jedynie czas wykonania algorytmów, bez zmian elementów, wypisywania danych do konsoli i innych czynności wymaganych do obsługi programu.

Pomiar czasu powinien odbywać się przez funkcję *QueryPerformanceCounter,* zapewniający stosunkowo wysoką jakość pomiarów rzędu mikrosekund.

5. Wyniki eksperymentów

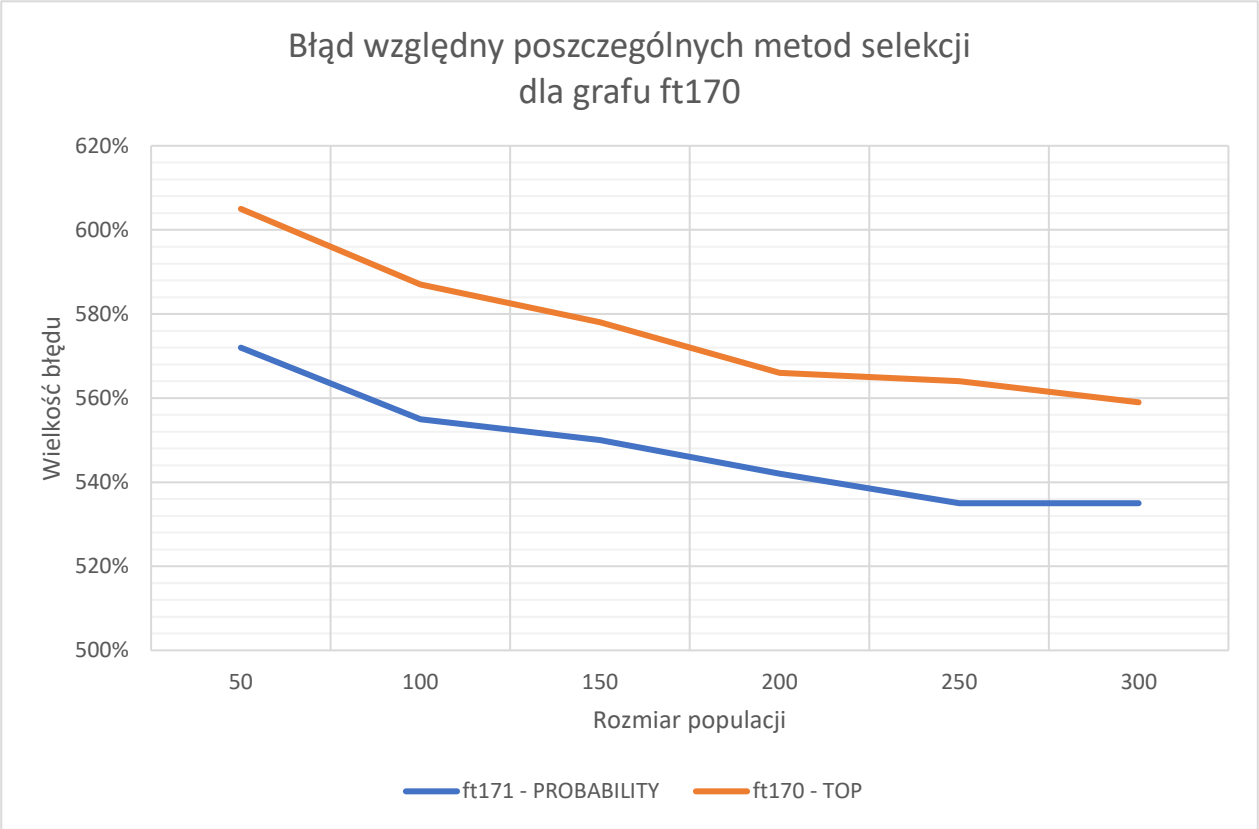
Z powodu dużych różnic błędów dla lepszej widoczności wykresów wyniki dla grafu ftv170 przedstawione są na drugim wykresie.



Wykres 1

	Błąd względny			
ROZMIAR POPULACJI	ftv33 - PROBABILITY	ftv33 - TOP	ft70 - PROBABILITY	ft70 - TOP
50	49%	62%	37%	46%
100	44%	55%	35%	43%
150	41%	50%	33%	41%
200	38%	45%	32%	40%
250	37%	47%	32%	40%
300	36%	45%	31%	38%

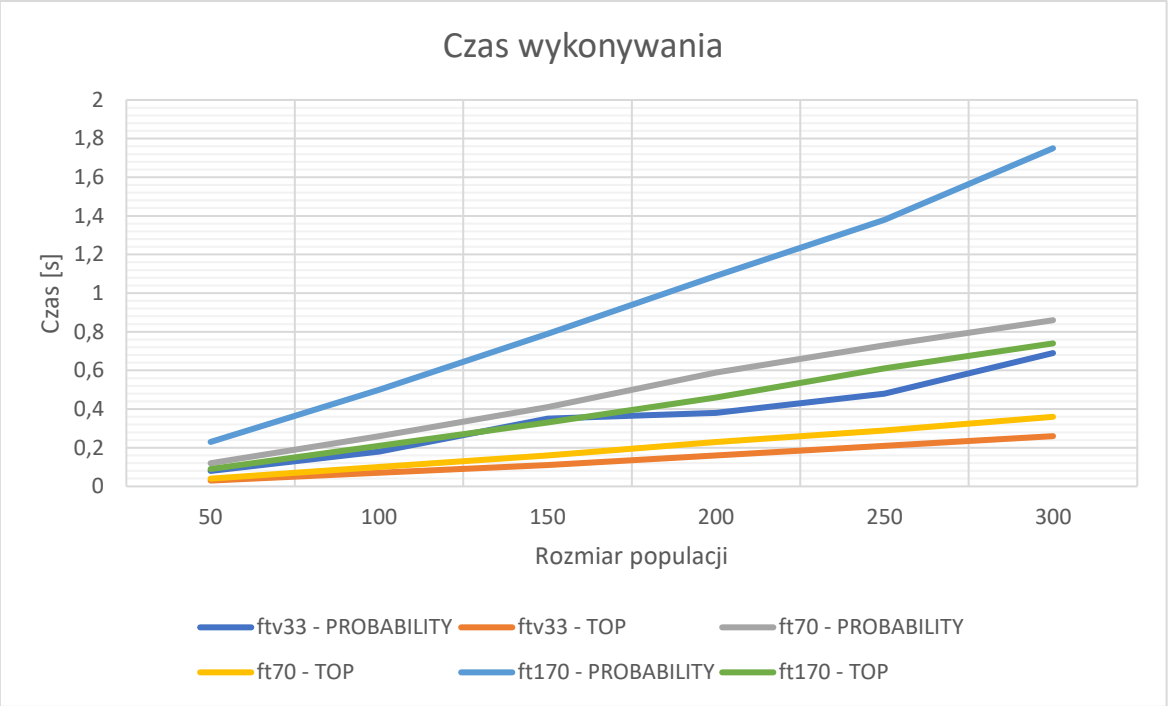
Tabela 1



Wykres 2

	Błąd względny	
ROZMIAR POPULACJI	Ft170 - PROBABILITY	Ft170 - TOP
50	572%	605%
100	555%	587%
150	550%	578%
200	542%	566%
250	535%	564%
300	535%	559%

Tabela 2



Wykres 3

	Czas wykonywania algorytmu [s]					
ROZMIAR POPULACJI	ftv33 - PROBABILITY	ftv33 - TOP	ft70 - PROBABILITY	ft70 - TOP	ft170 - PROBABILITY	ft170 - TOP
50	0,08	0,03	0,12	0,04	0,23	0,09
100	0,18	0,07	0,26	0,1	0,5	0,21
150	0,35	0,11	0,41	0,16	0,79	0,33
200	0,38	0,16	0,59	0,23	1,09	0,46
250	0,48	0,21	0,73	0,29	1,38	0,61
300	0,69	0,26	0,86	0,36	1,75	0,74

Tabela 3

6. WNIOSKI

Jakość rozwiązania zależy od rozmiaru populacji, ale razem ze zwiększaniem rozmiaru wydłuża się czas wykonywania algorytmu.

Metoda selekcji PROBABILITY odznacza się lepszą jakością rozwiązań niż metoda TOP, jednak wykonuje się dłużej. Wiąże się to z realokacją pamięci wewnątrz tej metody. Jeśli udałoby się rozwiązać problem realokacji, można by jednoznacznie stwierdzić, że metoda PROBABILITY jest lepsza niż TOP.

Dosyć dużym błędem odznacza się macierz o rozmiarze 171.