**Sprawozdanie lista1**

**Autorzy:** Tomasz Hołub (261718) & Piotr Piotrowski (261723)

**Wstęp:**

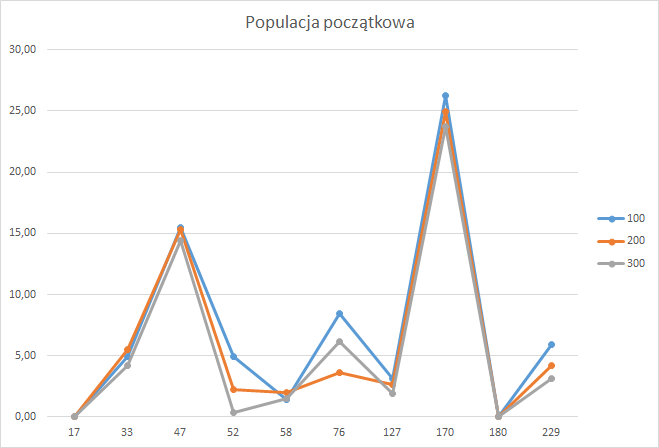
Sprawozdanie zawiera opis i analizę badań prowadzonych nad algorytmem genetycznym, rozwiązującym problem komiwojażera. Rozpatrywanymi czynnikami są:

* Wpływ wielkości instancji na czas wykonywania algorytmów
* Wpływ parametrów na dokładność uzyskanego wyniku.
* Wpływ doboru pomocniczych algorytmów na dokładność wyniku.

Algorytmy zostały napisane w języku Python, pełny kod znajduję się w repozytorium :

<https://github.com/holubek01/algorytm_genetyczny>

1. **Badania**

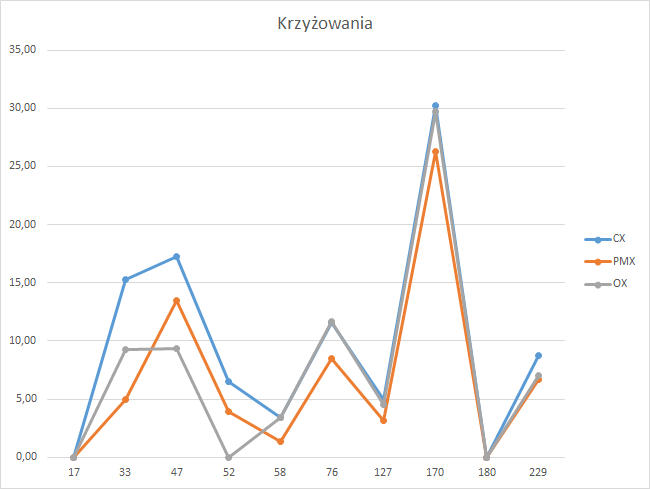


Wykres 1. opisujący procentową wartość odchylenia w zależności od początkowej wielkości populacji.

Pierwszym badanym parametrem jest wielkość populacji początkowej. Na podstawie powyższego wykresu można zauważyć, że im większa liczba osobników w populacji początkowej, tym większa dokładność algorytmu (poza instancją pr76). Dla testu końcowego zostanie więc wybrana wielkość populacji początkowej równa 300. Nie wybieramy większej populacji z tego względu, że wraz ze zwiększaniem się tej liczby rośnie czas wykonywania się algorytmu. Można natomiast zauważyć, że zwiększenie liczby populacji początkowej o 100 dla niektórych instancji, nieznacznie poprawia dokładność wyniku.

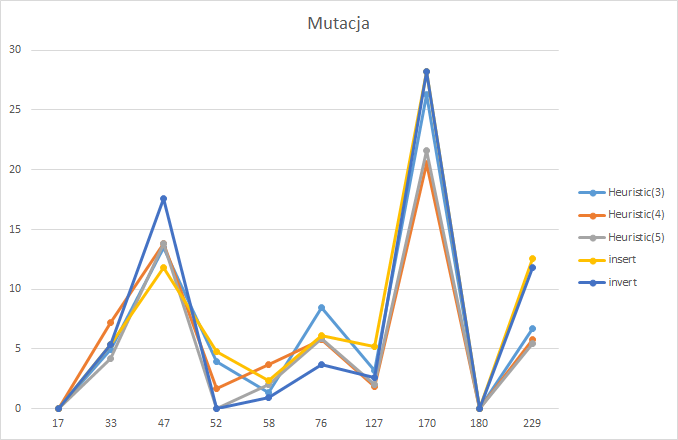
Wartość odchylenia wykorzystywana przy wszystkich wykresach (oś pionowa) liczona jest według poniższego wzoru:

Gdzie:



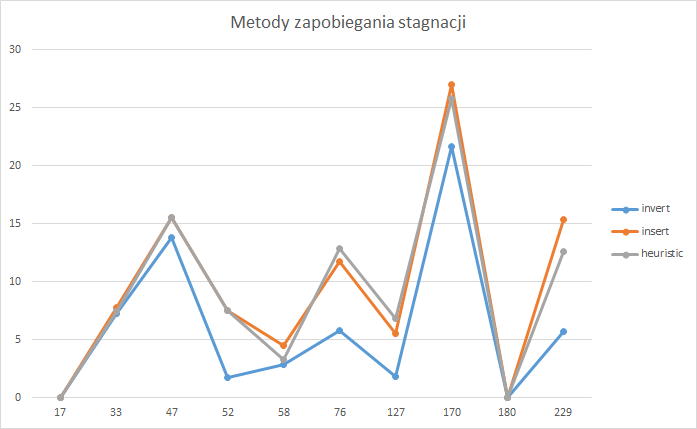
Wykres 2. opisujący procentową wartość odchylenia w zależności od doboru algorytmu krzyżowaniai.

Kolejnym elementem poddanym badaniom był rodzaj krzyżowania użyty w procesie generowania dzieci. PMX był algorytmem, który optymalizował najlepiej 2 instancje (ftv33 i ftv170), dla których Tabu Search znajdował mało optymalne rozwiązania. Algorytm cykliczny CX daje zdecydowanie gorsze rezultaty niż pozostałe. Z tego powodu do ostatecznego testu został wybrany operator krzyżowania PMX.



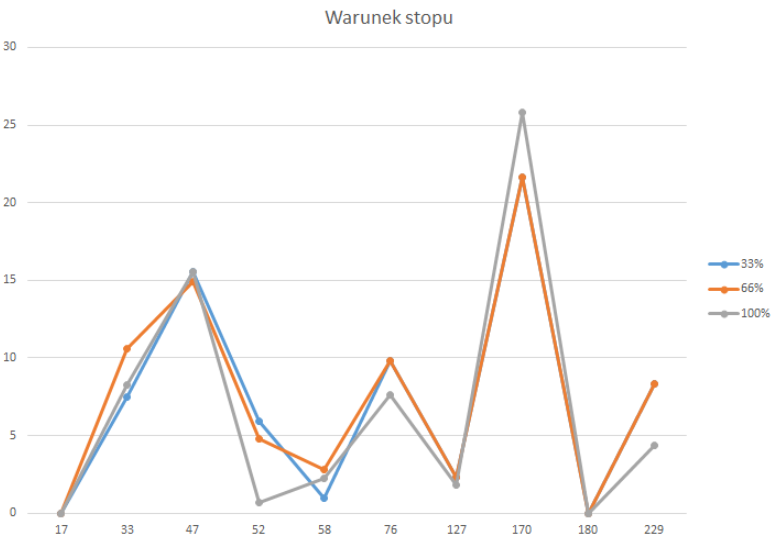
Wykres 3. opisujący procentową wartość odchylenia w zależności od doboru algorymu mutacjii.

Operacja mutacji w głównej mierze polega na zmianie genotypu dziecka po krzyżowaniu. Standardowymi metodami są zastosowanie opracji „insert” lub „invert”. Badaniu został poddany także algorytm mutacji heurystycznej w trzech konfiguracjach. Widoczna jest zależność, że im więcej miast wybranych jest do losowej permutcji w mutacji heurystycznej, tym większa dokładność algorytmu. Ponadto „insert” zdaje się być jednym z gorszych wyborów. Z tego powodu do finalnego testu należałoby wybrać „invert” lub „heuristic(5)”. Można zauważyć, że dla instancji ftv47 oraz 170ftv „invert” daje gorsze wyniki od reszty. Z tego powodu do testu ostatecznego został wybrana metoda „invert”.



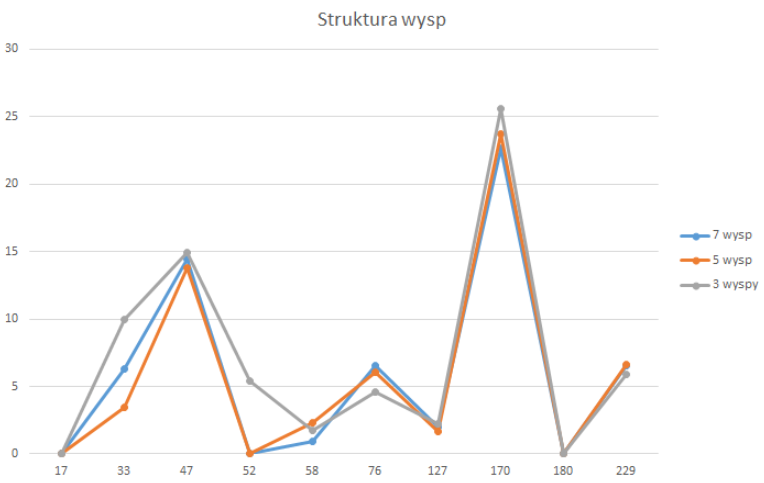
Wykres 4 opisujący procentową wartość odchylenia w zależności od doboru metody zapobiegającej stagnacji..

Podobnie jak w poprzednim przypadku, jako metody służące do zapobiegania stagnacji, zostały przebadane operacje „invert”, „insert” oraz „mutacja heurystyczna”. W tym przypadku, wbrew przewidywaniom najlepsze rezultaty daje zastosowanie inverta, natomiast najgorzej wypada „mutacja heurystyczna”.



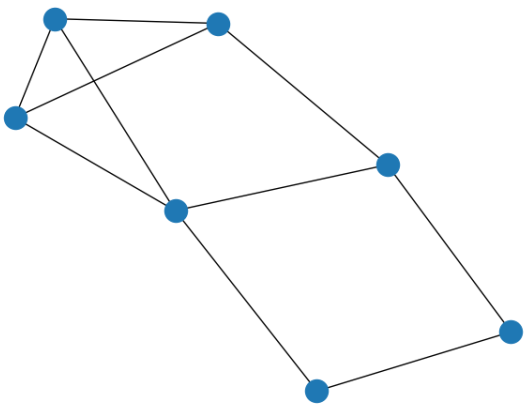
Wykres 5. opisujący procentową wartość odchylenia w zależności od warunku stopu.

Kolejnym badanym czynnikiem jest warunek stopu, po którym algorytm genetyczny kończy swoje działanie. Dokładniej rzecz ujmując kończy swoje działanie gdy liczba iteracji bez poprawy wyniku przekroczy pewną stałą. W przypadku „33%” algorytm zwracał wynik, gdy liczba iteracji bez poprawy przekroczyła liczbę daną wzorem . Analogicznie dla 66% i 100%. Z wykresu można odczytać, że najlepiej sprawdza się opcja „100%”, która jednak wypadała gorzej dla pierwszych 2 instancji oraz dla instacji brazil58 i ftv 170. Duże znaczenie ma czynnik losowy, który czasami zaburza wyniki, które nie zawsze są zgodne z oczekiwaniami. Do testów końcowych wybraliśmy wariant „100%”.

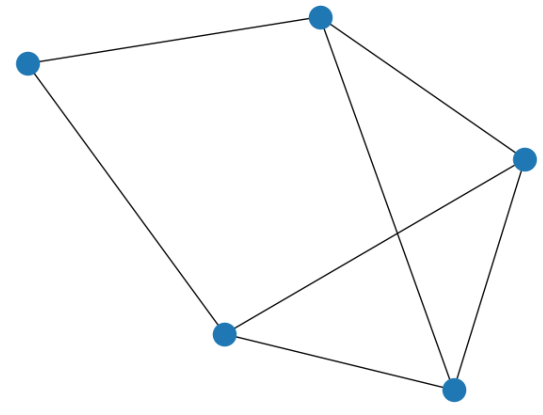


Wykres 6. opisujący procentową wartość odchylenia w zależności od struktury i liczby wysp.

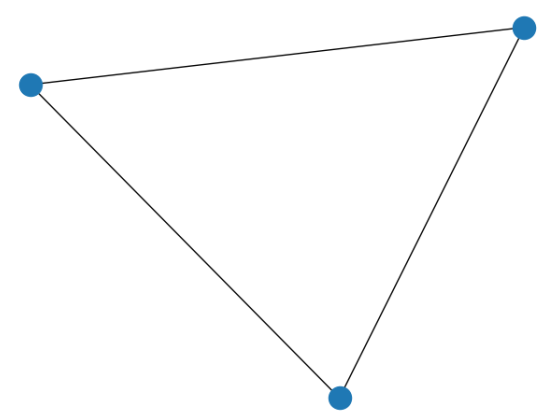
Powyższy wykres przedstawia wpływ struktury i liczby wysp na działanie algorytmu. Podobnie jak w poprzednim wykresie wyniki nie są jednoznaczne. Można stwierdzić, że podobnie zachowują się struktury składające się z 5 lub 7 wysp, ale struktura 3 wysp jest od nich gorsza. Do końcowych testów wykorzystamy strukturę składającą się z 7 wysp. Przykładowe stosowane struktury wysp przedstawiają poniższe diagramy.



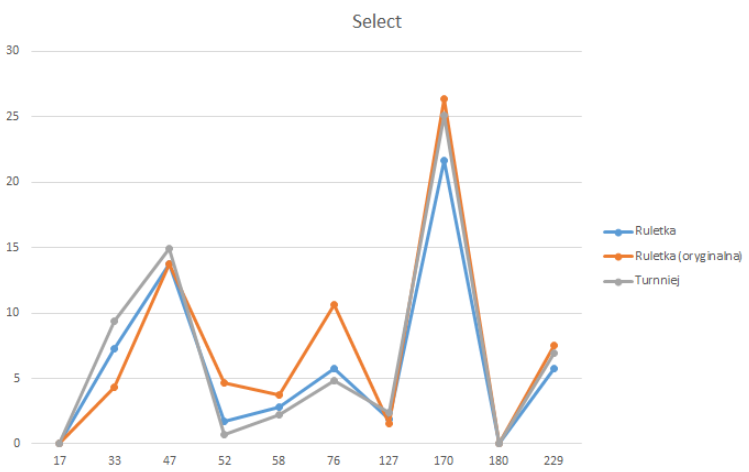
Graf cykliczny z dodatkowymi połączeniami, składający się z 7 wysp



Graf cykliczny z dodatkowymi połączeniami, składający się z 5 wysp

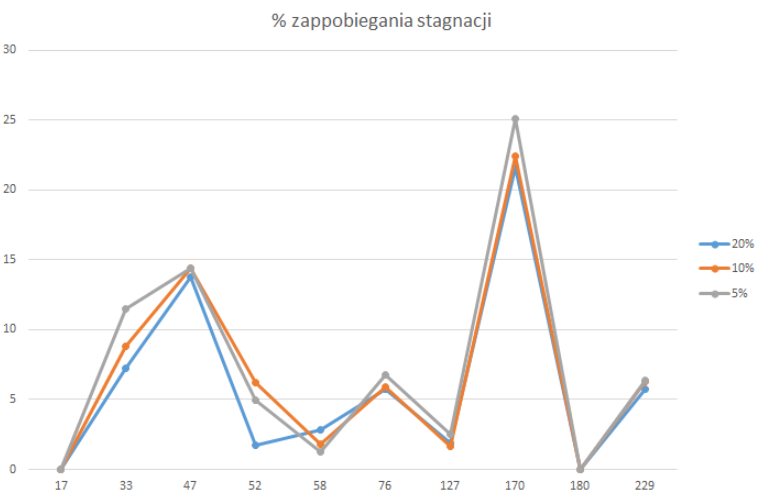


Graf cykliczny, składający się z 3 wysp



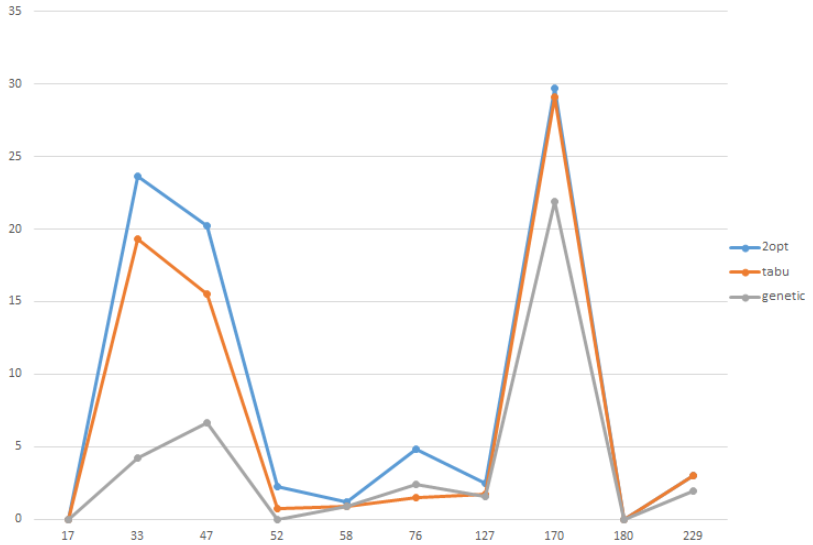
Wykres 7 opisujący procentową wartość odchylenia w zależności od doboru algorytmu wyboru rodziców do krzyżowania

Kolejnym badanym czynnikiem był algorytm dobierający rodziców (osobników) do krzyżowania. Wnioskując z powyższego wykresu, w większości przypadków algorytmem dającym najlepsze rezultaty była ruletka, która jednak przegrywała z ruletką (oryginalną) dla instancji fnv33 i gr229 oraz z turniejem dla instancji berlin52, brazil58 oraz pr76. To testów końcowych wybieramy ruletkę.



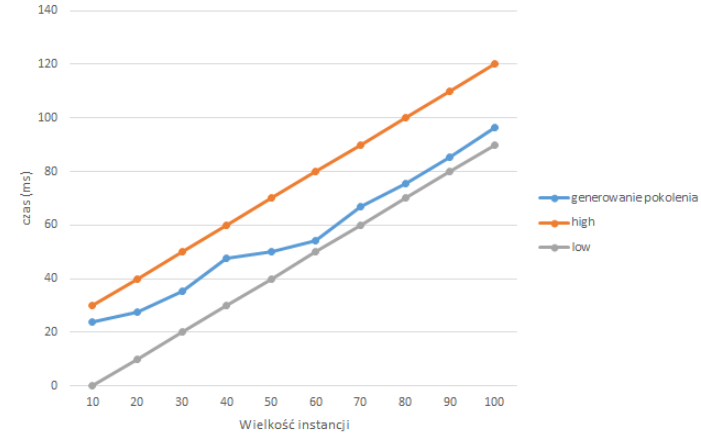
Wykres 8 opisujący procentową wartość odchylenia w zależności od % jednakowych osobników , od którego zaczynamy zapobiegać stagnacji.

Następnym czynnikiem poddanym badaniom jest % jednakowych osobników, od którego zaczynamy zapobiegać stagnacji. Jak widać na wykresie im większy jest ten odsetek tym wynik jest dokładniejszy. Odsetek 20% nie daje jednak najlepszych rezultatów dla instancji brazil58, jednak jest on porównywalny. Zgodnie z oczekiwaniami najgorzej wypdada opcja 5%.



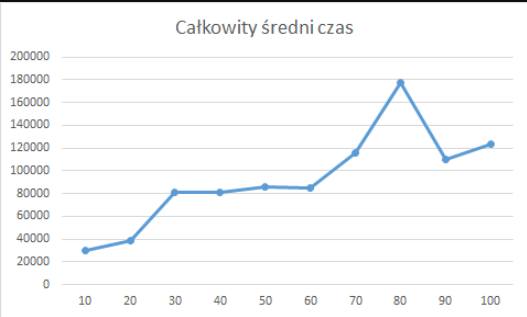
Wykres 9 opisujący procentową wartość odchylenia w zależności od doboru algorytmu

Powyższy wykres przedstawia porównanie algorytmów rozwiązująych probelem komiwojażera. Są to algorytmy „genetyczny”, „2OPT”, oraz „Tabu Search”. Dwa ostatnie implemenowane były na poprzendnich listach. Jak można zauważyć algorytm genetyczny znajduje lepsze wyniki (oprócz instancji pr76 ale tutaj różnica jest niewielka), a w niektórych różnice są znaczące. Jest tak np. dla instancji ftv33, ftv47 oraz ftv170.



Wykres 10 opisujący złożoność czasową algorytmu generującego polecenie.

W celu obliczenia złożoności czasowej funkcję czasu ograniczamy z góry funkcją liniową oraz z dołu funkcją liniową . Wnioskujemy zatem, że złożoność czasowa algorytmu generowania pokolenia jest liniowa O(n).



Wykres 11 opisujący złożoność czasową algorytu generującą