

PROJEKTOWANIE EFEKTYWNYCH ALGORYTMÓW – PROJEKT 3

Temat: Badanie algorytmów dla problemu komiwojażera (TSP).

Autor: Michał Tomaszewicz, 235568

Prowadzący projekt: Dr inż. Tomasz Kapłon

1. Wstęp

Celem trzeciego projektu było zaimplementowanie w języku C++ algorytmu genetycznego dla problemu komiwojażera..

Problem Komiwojażera (ang. Travelling salesman problem) – jest to zagadnienie optymalizacyjne należące do rodziny problemów NP-trudnych. Polega ono na znalezieniu w grafie ważonym cyklu Hamiltona o minimalnej sumie wag krawędzi.

Twórcą algorytmu genetycznego (w skrócie AG) był John Henry Holland. Wzorował się on na zjawisku ewolucji biologicznej/ ewolucji darwinowskiej. Występują tu pojęcia z dziedziny biologii takie jak: populacja, genotyp krzyżowanie, mutacja i selekcja.

Grupa osobników, w której każdy posiada osobny genotyp nazywamy populacją. Jak w teorii ewolucji populacja dąży do przetrwania, czyli do jak najlepszego przystosowania do panujących warunków. W modelu, jakim jest algorytm genetyczny dla TSP, miarą przystosowania jest jakość genotypu, czyli koszt drogi (cyklu Hamiltonowskiego). Im mniejszy koszt, tym większa miara przystosowania i bardziej wartościowy osobnik.

Aby osiągnąć cel jakim jest osiągnięcie jak największego przystosowania, osobniki rozmnażają się, tj. krzyżują się i tworzą nowe pokolenie. Zjawisko to poprzedza proces selekcji – czyli dobieranie osobników z populacji w pary. W biologii selekcja następuje tak, że osobniki dobierają się w pary tak, aby wydać na świat jak najsilniejsze potomstwo. W badanym problemie też ma miejsce takie postępowanie, ale nieco zmodyfikowane, ze względu na zagrożenie ujednolicenia populacji. Takie ujednolicenie powoduje, że w wyniku krzyżowania powstają osobniki, które nieznacznie różnią się od swoich rodziców, przez co ich przystosowanie może być tylko nieznacznie większe od starszego pokolenia.

Na przystosowanie osobników ma wpływ także mutacja. To zjawisko polega na nieznacznej modyfikacji genotypu osobnika. W AG występuje ona w celu zwiększenia różnorodności populacji i zapobieganiu ujednoliceniu.

Instancje z których korzystano podczas wykonywania badań pochodzą za strony: <https://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/>

Link 1. Link do strony TSPLIB

2. Badania

Pomiarów dokonywano za pomocą funkcji QueryPerformanceCounter. Program został napisany w środowisku MS Visual Studio C++ oraz skompilowany do 64 bitowej wersji. Do przeprowadzenia eksperymentu używano komputera PC z procesorem AMD FX – 6300 3,50 GHz i 12 GB pamięci RAM.

Wyświetl podstawowe informacje o tym komputerze

Wersja systemu Windows

Windows 10 Education

© 2017 Microsoft Corporation. Wszelkie prawa zastrzeżone.



System

Procesor:	AMD FX(tm)-6300 Six-Core Processor	3.50 GHz
Zainstalowana pamięć (RAM):	12,0 GB	
Typ systemu:	64-bitowy system operacyjny, procesor x64	

Rys. 1. Specyfikacja platformy testowej

Wykorzystane instancje:

- Burma14
- Gr17
- Fri26
- Gr48
- Gr120
- Gr229
- Gr666

Liczby na końcach nazw oznaczają rozmiar instancji, to znaczy ilość miast. Dane wczytywano z plików tekstowych. Instancje burma14, gr229 oraz gr666 zostały podane w formie

współrzędnych geograficznych 2D. Po przekonwertowaniu ich zgodnie z szablonem znajdującym się w instrukcji na stronie (*Link 1.*) do postaci tablicy odległości wyniki otrzymane w wyniku działania algorytmu różniły się od wyników podanych na stronie. Sprawdzono to jedynie na instancji burma14 przy użyciu algorytmu Brutal Force i Helda Karpa. W tym przypadku różnica wynosiła około 4% (długości trasy). Dla instancji gr229 oraz gr666 podobny błąd też ma duże prawdopodobieństwo wystąpić, co ma wpływ na wielkość błędu otrzymanego dla wyniku po wykonaniu algorytmu symulowanego wyżarzania.

Instancje są przechowywane jako tablica sąsiedztwa. Wagi są nieujemnymi liczbami całkowitymi. Odległości są symetryczne, co oznacza, że koszt podróży z miasta A do miasta B jest taki sam co koszt podróży z miasta B do miasta A. Algorytmy obliczają minimalny koszt, ale też pokazują ścieżkę, czyli miasta, które składają się na cykl Hamiltona.

2.1.Opis zaimplementowanego algorytmu

W algorytmie zaimplementowano 2 różne operatory krzyżowania – jednopunktowe oraz dwupunktowe NWOX(non-wrapping order crossover). 2 operatory mutacji – swap oraz insert. 2 metody selekcji – selekcja turniejowa, z turniejem o wielkości 5, oraz selekcja polegająca na łączeniu w pary osobników o najniższym koszcie z osobnikami o najwyższym koszcie (potocznie mówiąc najlepszych z najgorszymi). Wielkość badanej populacji jest stała, definiowana na początku działania programu. Badania przeprowadzono dla 2 różnych wielkości populacji: $1 \cdot n$ oraz $5 \cdot n$, gdzie n to liczba genów (miast).

Dodatkowo do nowego pokolenia włączany jest fragment starego pokolenia. Na fragment ten składają się (potocznie mówiąc) najlepsze oraz najgorsze osobniki starego pokolenia. Osobniki o najwyższym koszcie są eliminowane i na ich miejsce pojawiają się osobniki z poprzedniego pokolenia. Procedurę tą zastosowano, gdyż miała ona pozytywny wpływ na jakość otrzymanych wyników, to znaczy algorytm bez tej operacji zwracał rozwiązanie o błędzie większym niż algorytm z zaimplementowaną operacją.

Ze względu na dużą ilość parametrów w algorytmie zdefiniowano stałe:

- Prawdopodobieństwo krzyżowania równe 70%
- Prawdopodobieństwo mutacji równe 10%
- Fragment starego pokolenia włączany do nowego pokolenia równy 30%

Algorytm posiada 4 badane parametry parametry o 2 przebadanych wartościach. Daje to $2^4 = 16$ różnych pomiarów do wykonania (dla 1 instancji!). Wartości te wybrano jako dające najniższe błędy.

2.1.1. Opis zaimplementowanego algorytmu do badania przeprowadzonego w podpunkcie 2.2.1.

Jedyną z dwóch zmian, która zaszła w algorytmie, aby umożliwić badanie wpływu prawdopodobieństw krzyżowania i mutacji na wynik jest dodanie 2 kolejnych parametrów opisujących prawdopodobieństwo krzyżowania i prawdopodobieństwo mutacji.

Druga zmiana polega na zmianie strategii tworzenia nowego pokolenia. Nie występuje tu stały fragment starego pokolenia włączany do nowego pokolenia, ale jest on

zależny od prawdopodobieństwa krzyżowania. Gdy z operacji krzyżowania powstaną nowe osobniki a wielkość nowego pokolenia jest mniejsza niż stała wielkość pokolenia to osobniki ze starego pokolenia są umieszczane aż do momentu gdy wielkość nowego pokolenia osiągnie oczekiwany poziom.

2.2.Procedura badawcza

Dla każdej z 7 różnej wielkości instancji przeprowadzono pomiary w 2 seriach, wyniki uśredniono. Wynikiem pomiarów był błąd – różnica między kosztem optymalnym, a otrzymanym w wyniku działania algorytmu przez 5 minut dla każdej wariacji wartości parametrów.

5 minut nie jest przypadkowym czasem - podczas badania algorytmu symulowanego wyżarzania pomiar także trwał 5 minut, więc aby jak najrzetelniej porównać ze sobą te algorytmy należało zastosować te same czasy pomiarów.

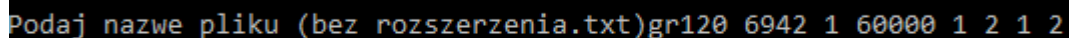
2.2.1 Dodatkowe badanie (prawdopodobieństwo krzyżowania i mutacji)

Dla instancji gr666 przeprowadzono jeszcze jedno badanie – wpływu prawdopodobieństwa krzyżowania oraz prawdopodobieństwa mutacji na wielkość błędu wyniku otrzymanego po działaniu algorytmu genetycznego przez 5 minut. Parametrami stały się:

- Wielkość populacji = $(1 * n) = n$
- Operator krzyżowania – krzyżowanie 2-punktowe NWOX
- Metoda selekcji – turniejowa (z turniejem wielkości 5)
- Operator mutacji – insert

Wybór tych parametrów został podyktowany najniższym błędem w badaniach przedstawionych w tabeli z wynikami (w punkcie 3.1 Tabele z wynikami). Przeprowadzono 2 serie pomiarów, wyniki uśredniono.

3. Legenda do tabel z wynikami pomiarów



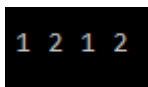
```
Podaj nazwe pliku (bez rozszerzenia.txt)gr120 6942 1 60000 1 2 1 2
```

Rys.2.

Na Rys.2. znajduje się linijka inicjalizująca działanie algorytmu. W tym podpunkcie wyjaśnione będzie jak korzystać z programu oraz wytłumaczone kolejne wartości parametrów.

- gr120 – nazwa pliku, w którym znajdują się dane
- 6942 – optymalny koszt trasy (podany w *Link 1*)
- 1 – ile razy algorytm ma się wykonać
- 60000 – czas podany w milisekundach (1/1000 sekundy)

Kolejne liczby (Rys.3.) są parametrami zawierającymi informację o wielkości pokolenia, operatorze krzyżowania, rodzaju selekcji oraz operatorze mutacji. Pozycje liczb będą numerowane od lewej strony od 1 do 4.



Rys.3.

Liczba na pozycji 1. służy do obliczenia wielkości populacji wyrażonej wzorem:

$$\text{wielkość_populacji} = x * n$$

gdzie „n” oznacza ilość miast instancji, a „x” oznacza liczbę na pozycji 1. W tym przypadku jest to 1, a więc wielkość badanej populacji jest równa:

$$\text{wielkość_populacji} = 1 * 120$$

$$\text{wielkość_populacji} = 120$$

Liczba na pozycji 2. określa operator krzyżowania.

- 1 – krzyżowanie 1-punktowe
- 2 – krzyżowanie 2-punktowe (NWOX)

Na Rys.2. na pozycji 2. wpisano liczbę 2, a więc wybrano krzyżowanie 2-punktowe NWOX.

Liczba na pozycji 3. określa rodzaj selekcji.

- 1 – selekcja turniejowa
- 2 – selekcja „najlepsi z najgorszymi”

Na Rys.2. na pozycji 3. wpisano liczbę 1, a więc wybrano selekcję turniejową

Liczba na pozycji 4. określa operator mutacji.

- 1 – mutacja swap
- 2 – mutacja insert

Na Rys.2. na pozycji 4. wpisano liczbę 2, a więc wybrano mutację insert

W nagłówkach tabel znajdują się analogicznie oznaczone parametry jak na Rys.3. Dodatkowo są one poprzedzone literami:

- p – oznacza wielkość populacji
- k – oznacza operator krzyżowania
- s – oznacza rodzaj selekcji
- m – oznacza operator mutacji

3.0.1 Legenda do tabeli z pomiarami z dodatkowego badania (podpunkt 2.2.1)

Podaj nazwe pliku (bez rozszerzenia.txt)gr666 294358 1 300000 1 2 1 2 0.1 0.9

Rys.3.5

Na Rys.3.5. znajduje się linijka inicjalizująca działanie algorytmu w badaniu z podpunktu 2.2.1. W porównaniu z parametrami inicjalizującymi z Rys.2 pojawiły się tu 2 dodatkowe liczby.

Jako, że 2 dodatkowe liczby następują bezpośrednio po opisanych w punkcie 3. oraz na rysunku Rys.3. liczbach, numeracja ich pozycji będzie kontynuowana. Oznacza to, że 0.1 znajduje się na pozycji piątej, a liczba 0.9 znajduje się na pozycji szóstej.

- Liczba na pozycji piątej oznacza prawdopodobieństwo krzyżowania
- Liczba na pozycji szóstej oznacza prawdopodobieństwo mutacji

W nagłówkach tabeli (podpunkt 3.2.) znajdują się analogicznie oznaczone parametry jak na Rys.3. na pozycjach 5. oraz 6. Dodatkowo są one poprzedzone literami:

- pk – prawdopodobieństwo krzyżowania
- pm – prawdopodobieństwo mutacji

3.1. Tabele z wynikami pomiarów

Dostępne w pliku [tabele.PNG](#)

3.2. Tabele z wynikami pomiarów badania z podpunktu 2.2.1

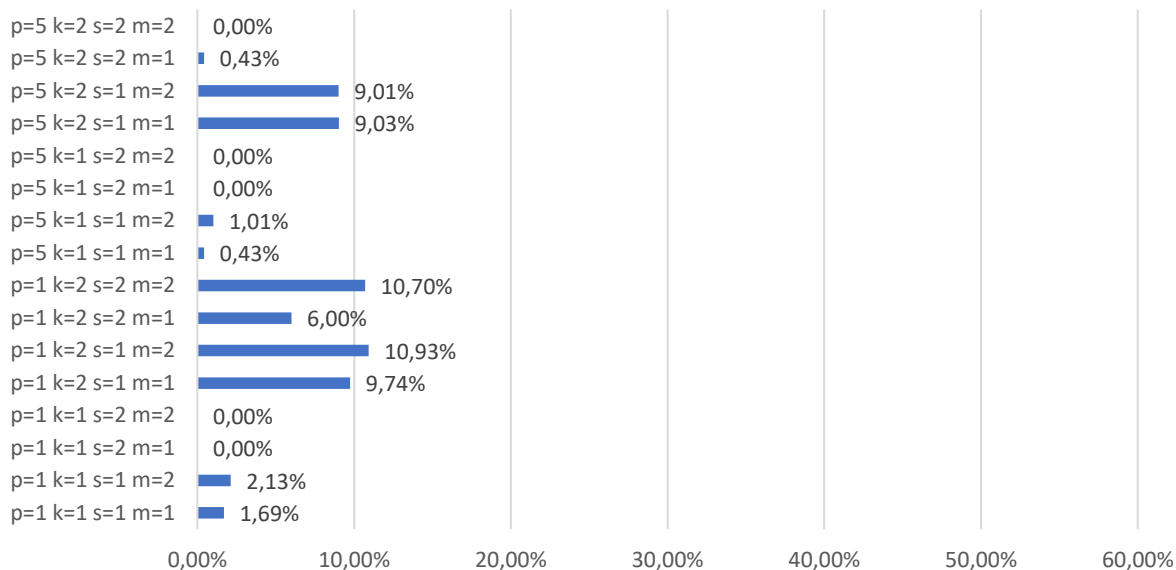
Dostępne w pliku [tabele2.PNG](#)

Komentarz:

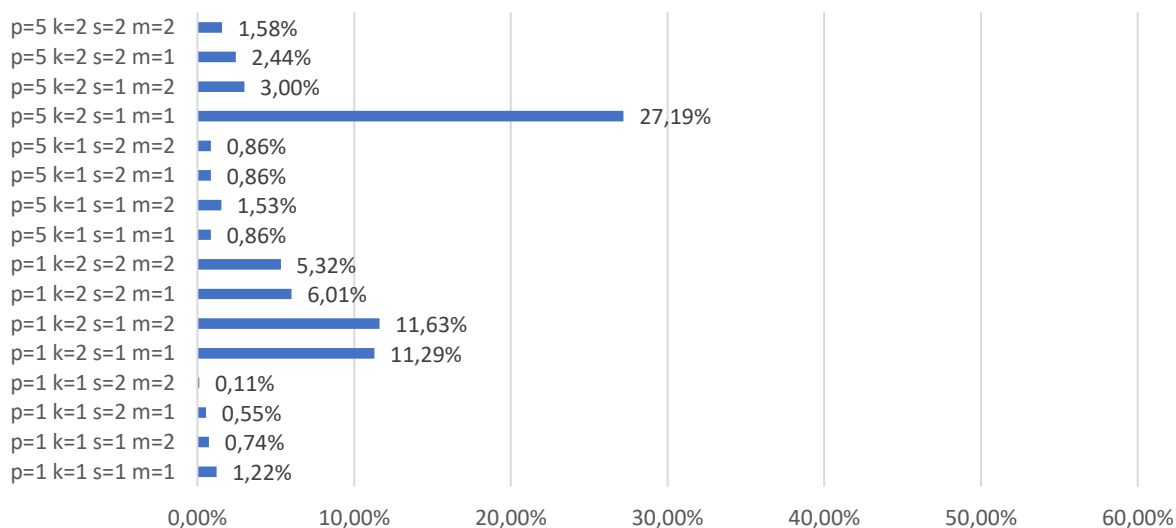
Błąd dla instancji gr229 i gr666 może być zawyżony we względu na format danych wejściowych (problem opisany w paragrafie 2. Badania).

4. Wykresy

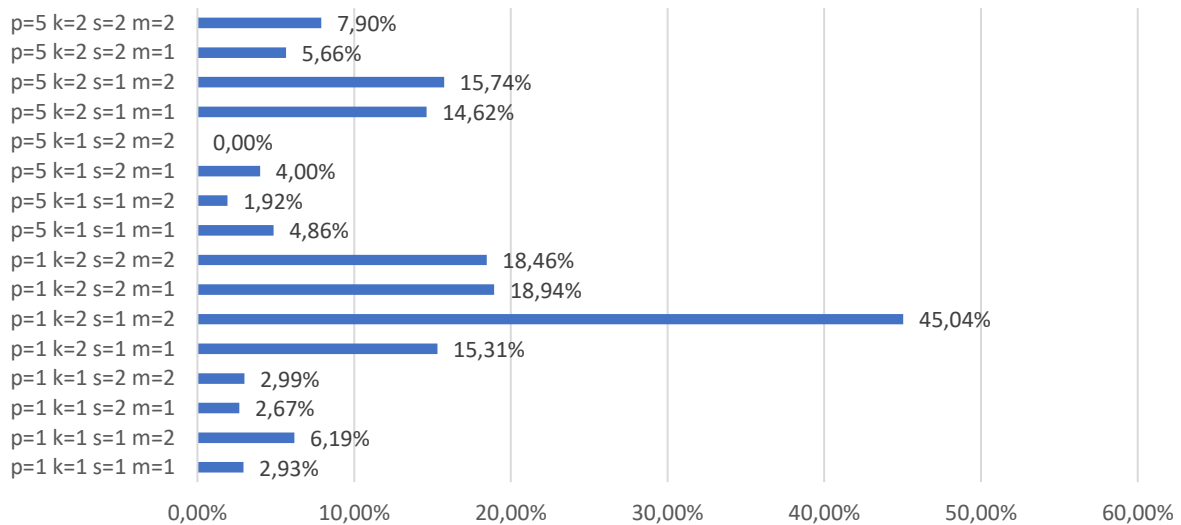
Wykres zależności otrzymanego błędu od wartości parametrów algorytmu genetycznego dla instancji burma14



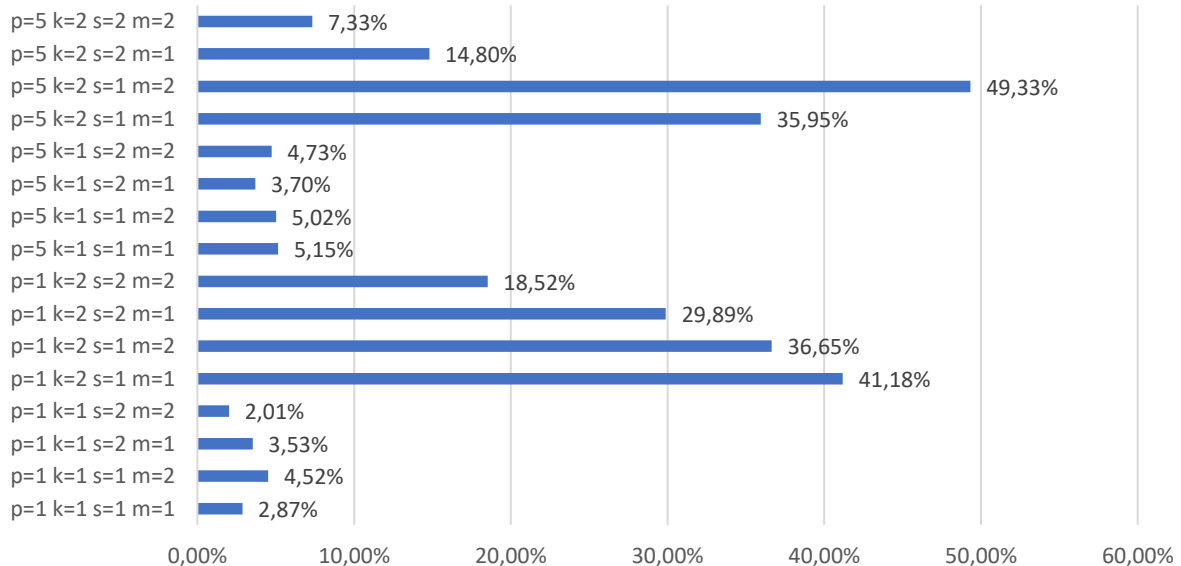
Wykres zależności otrzymanego błędu od wartości parametrów algorytmu genetycznego dla instancji gr17



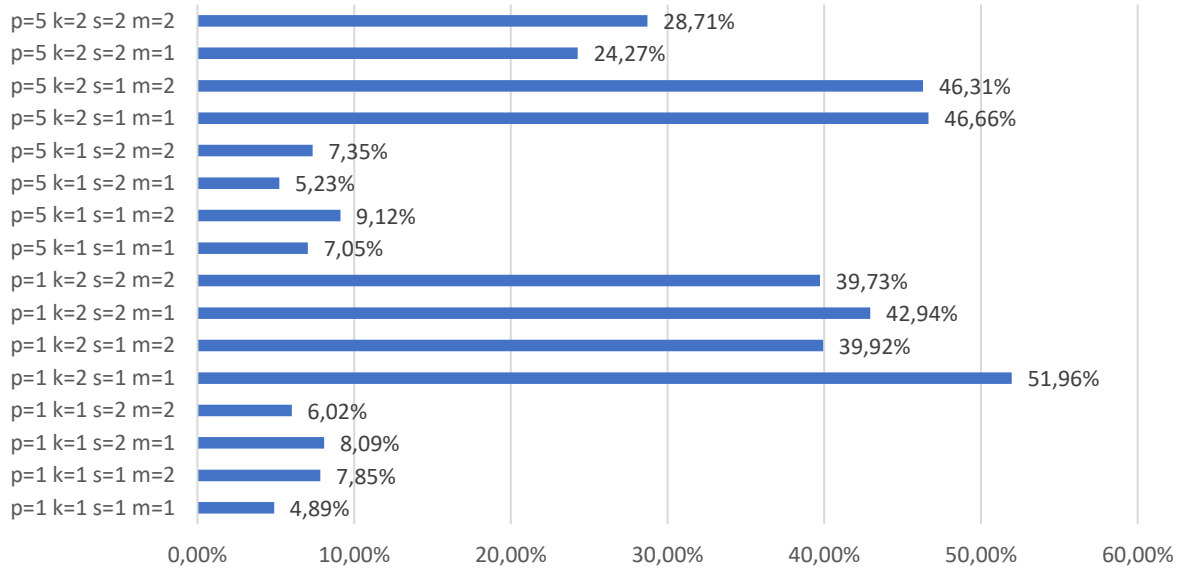
Wykres zależności otrzymanego błędu od wartości parametrów algorytmu genetycznego dla instancji fri26



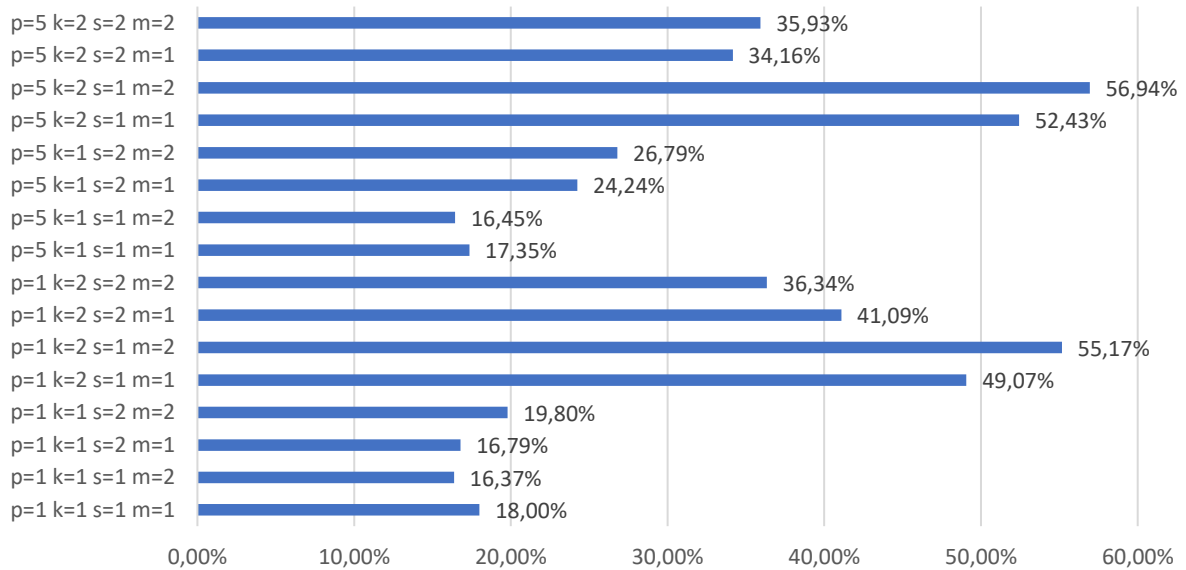
Wykres zależności otrzymanego błędu od wartości parametrów algorytmu genetycznego dla instancji gr48

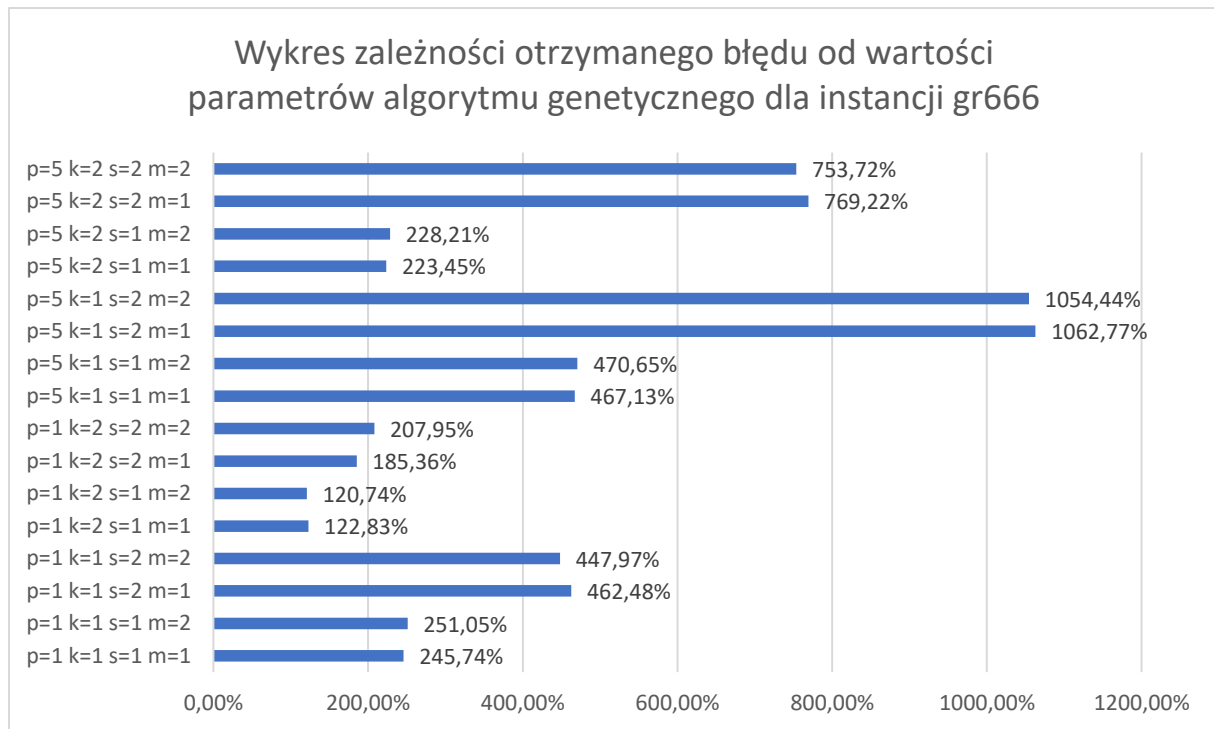


Wykres zależności otrzymanego błędu od wartości parametrów algorytmu genetycznego dla instancji gr120



Wykres zależności otrzymanego błędu od wartości parametrów algorytmu genetycznego dla instancji gr229





5. Wnioski

Podsumowując udało się zaimplementować poprawnie działający algorytm genetyczny. Zaimplementowałem w nim różne sposoby selekcji, mutacji i krzyżowania, które mają widoczny wpływ na jakość otrzymanego rozwiązania.

Wielkość populacji ma wpływ na błąd w otrzymanym rozwiązaniu. Widać w tabelach z wynikami (3.1), że populacja o wielkości n (czyli taka jak ilość miast w instancji) jest wystarczająca by otrzymać rozwiązanie różniące się od optimum zaledwie o kilka procent. Duża populacja ($5 \cdot n$) pozwala na otrzymanie dokładniejszego wyniku, ma wadę – zwiększa złożoność obliczeniową, przez co czas wykonania algorytmu rośnie.

Krzyżowanie 1-punktowe jest lepszym wyborem niż krzyżowanie 2-punktowe NWOX dla małych i średnich instancji (na przykładzie do 229 miast). Dla dużych instancji (na przykładzie 666 miast) sytuacja się przeciwna – to krzyżowanie 2-punktowe daje lepsze rezultaty. Z taką różnicą, że dla dużych populacji różnice między krzyżowaniami są mniejsze niż dla małych populacji.

Selekcja turniejowa dla mniejszych (do 229 miast) populacji jest gorsza niż selekcja „gorszy z lepszym”, natomiast przy większej (około 666) populacji sytuacja zmienia się na przeciwną.

Operator mutacji, czy to swap, czy to insert ma niewielki wpływ na otrzymane rozwiązanie, choć widoczna jest niewielka przewaga insert nad swap.

Analizowanie zależności między wieloma parametrami jest niezwykle skomplikowanym procesem, który wymaga dużo doświadczenia by wnioski brzmiały sensownie.

Spośród badanych parametrów ciężko jest wyłonić zestaw, który byłby uniwersalny i dawał najmniejsze błędy. Dla mniejszych instancji (do 229 miast) dobre jest krzyżowanie 1-punktowe połączone z selekcją „najgorszy z najlepszym”.

Dla instancji większych (około 666 miast) zdecydowanym faworytem jest krzyżowanie 2-punktowe połączone z mutacją insert, selekcja turniejowa oraz mała populacja.

Porównując GA do SA (symulowanego wyżarzania) należy zauważyć, że GA po dobraniu odpowiednich parametrów zwraca wyniki o mniejszych błędach niż najlepsze wyniki SA wyżarzania. Algorytm GA jest znacznie bardziej skomplikowany niż SA. Występuje tu znacznie więcej zmiennych którymi można manewrować w celu otrzymania wyniku jak najbliższemu rzeczywistemu wynikowi.

O porównaniu GA do Brutal Force należy jedynie wspomnieć, że potrzebny czas, podany w latach, na wykonanie się algorytmu BF dla 666 miast, zapisany cyframi w czcionce 12 zająłby pół strony A4.

Najniższy błąd jaki zdołano zanotować dla instancji gr666 to 52% po 22 minutach (Rys.4.) działania algorytmu. Zwiększenie czasu działania algorytmu nie dawało już lepszych wyników (Rys.5.).

5.1. Wnioski do badania z podpunktu 2.2.1

Prawdopodobieństwa krzyżowania i mutacji są parametrami, których zmiana wartości ma wpływ na wynik działania algorytmu. Jak wynika z tabeli (podpunkt 3.2) im większe prawdopodobieństwo krzyżowania tym algorytm zwraca wynik z mniejszym błędem. Przy wyjątkowo dużych prawdopodobieństwach mutacji ($p_m \geq 0,4$) ta zależność przestaje działać.

Podobna sytuacja ma miejsce w przypadku prawdopodobieństwa mutacji – gdy prawdopodobieństwo krzyżowania jest mniejsze od 1 to zależność „im większa wartość prawdopodobieństwa mutacji tym mniejszy błąd” jest prawdziwa. Gdy krzyżowanie jest zdarzeniem pewnym to nie można jednoznacznie określić wpływu prawdopodobieństwa mutacji na wynik.

Należy pamiętać, że w tym projekcie przyjęto specyficzną strategię tworzenia nowych pokoleń i otrzymane wyniki są także on niej zależne.

```
C:\Users\michal_internet\Desktop\PEN_PWR\PEA\Pro_3_genetyczny\PEA\x64\Release\PEA.exe

Podaj nazwe pliku (bez rozszerzenia.txt)gr666 294358 1 1300000 1 2 1 2

Podaj parametry dla algorytmu genetycznego

Oto ilosc miast: 666
oto wielkosc populacji: 666

oto najlepszy447619 po 20489 pokoleniach
    w czasie 1.30001e+06 ms

sredni koszt 447619
sredni czas 1.30001e+06 ms

sredni blad 52.0662 %

Aby wyjsc z programu nacsni ESC
Aby pozostac w programie nacsni jakikolwiek inny przycisk
```

Rys.4.

<pre>C:\Users\michal_internet\Desktop\PEN_PWR\PEA\Pro_3_genetyczny\PEA\x64\Release\PEA.exe Podaj nazwe pliku (bez rozszerzenia.txt)gr666 294358 1 7200000 1 2 1 2 Podaj parametry dla algorytmu genetycznego Oto ilosc miast: 666 oto wielkosc populacji: 666 oto najlepszy459987 po 118796 pokoleniach w czasie 7.20005e+06 ms sredni koszt 459987 sredni czas 7.20005e+06 ms sredni blad 56.2679 % Aby wyjsc z programu nacsni ESC Aby pozostac w programie nacsni jakikolwiek inny przycisk</pre>	<pre>C:\Users\michal_internet\Desktop\PEN_PWR\PEA\Pro_3_genetyczny\PEA\x64\Release\PEA.exe Podaj nazwe pliku (bez rozszerzenia.txt)gr666 294358 1 3600000 1 2 1 2 Podaj parametry dla algorytmu genetycznego Oto ilosc miast: 666 oto wielkosc populacji: 666 oto najlepszy546984 po 58476 pokoleniach w czasie 3.60002e+06 ms sredni koszt 546984 sredni czas 3.60002e+06 ms sredni blad 85.8227 % Aby wyjsc z programu nacsni ESC Aby pozostac w programie nacsni jakikolwiek inny przycisk</pre>
---	--

Rys.5.

```
C:\Users\michal_internet\Desktop\PEN_PWR\PEA\Pro_3_genetyczny\PEA\x64\Release\PEA.exe

Podaj nazwe pliku (bez rozszerzenia.txt)gr666 294358 1 300000 1 2 1 2 0.1 0.9

Podaj parametry dla algorytmu genetycznego

Oto ilosc miast: 666
oto wielkosc populacji: 666

oto najlepszy643349 po 4696 pokoleniach
    w czasie 300009 ms

sredni koszt 643349
sredni czas 300009 ms

sredni blad 118.56 %

Aby wyjsc z programu nacsni ESC
Aby pozostac w programie nacsni jakikolwiek inny przycisk
```

Rys.6.