Projektowanie Efektywnych Algorytmów

Kierunek *Informatyka*	**Termin** *Czwartek 19:05*
Temat *Algorytmy populacyjne*	**Problem** *(A)TSP*
Skład grupy *241284 Jakub Plona*	**Nr grupy** -
Prowadzący *Mgr inż. Radosław Idzikowski*	**data** *28 stycznia 2020*

Spis treści

1 Opis problemu

Jak można przeczytać na Wikipedii,

> problem komiwojażera (ang. travelling salesman problem, TSP) to zagadnienie optymalizacyjne, polegające na znalezieniu minimalnego cyklu Hamiltona w pełnym grafie ważonym.

Problem ten można przedstawić bez wykorzystania terminologii teorii grafów. Problem komiwojażera jest problemem optymalizacyjnym z danymi wejściowymi w postaci zbioru n miast oraz danymi kosztami przejścia między dowolnie wybraną parą miast, za wyjątkiem przejścia z danego miasta do tego samego miasta (przejście to nie istnieje). W problemie tym poszukuje się drogi o najmniejszym koszcie przejścia, przechodzącej przez każde miasto dokładnie raz, wracającej do miasta początkowego. Jakość rozwiązania ocenia się za pomocą tzw. funkcji celu, która jest definiowana jako suma kosztów przejścia między kolejnymi miastami na drodze będącej rozwiązaniem problemu wraz z przejściem łączącym ostatnie odwiedzone (unikalne) miasto z miastem startowym. Rozwiązanie, dla którego funkcja ta przyjmuje wartość minimalną nazywa się rozwiązaniem optymalnym.

W zależności od tego, czy przejście z miasta A do miasta B ma taki sam koszt, co przejście z miasta B do miasta A dla dowolnej pary miast (miasta muszą być różne od siebie) mamy do czynienia z symetrycznym problemem komiwojażera, jeżeli koszty te są równe, zaś w przeciwnym przypadku mówi się o asymetrycznym problemie komiwojażera. Tematem projektu jest asymetryczny problem komiwojażera. Warto dodać, iż każdy symetryczny problem komiwojażera można potraktować tak, jak asymetryczny, uprzednio dublując koszty przejścia pomiędzy każdą parą miast, tak aby przejście było możliwe w obie strony.

2 Metody i algorytmy rozwiązywania problemu

2.1 Algorytm genetyczny

Tematem aktualnego etapu projektu był algorytm genetyczny. Podstawowym pojęciem, którego używa się w tego typu algorytmach jest populacja. Nazywa się tak grupę znalezionych rozwiązań rozważanego problemu (osobników), którą poddaje się różnym operacjom mającym na celu poprawę ogólnej jakości rozwiązań w tej grupie (zwiększenie przystosowania osobników do „środowiska"). Rozwiązania są oceniane na na podstawie kryterium - w algorytmach genetycznych mówi się o ocenie osobników za pomocą funkcji przystosowania (ang. fitness function). Z założenia tej metody rozwiązania „lepsze" pod względem kryterium oceny będą miały większy wpływ na kształt pozostałych rozwiązań w populacji; będą wpływały na poprawę średniej wartości funkcji przystosowania osobników do środowiska.

W dziedzinie algorytmów genetycznych wyróżnia się następujące operacje wykonywane na osobnikach obecnych w populacji:

- selekcja,
- krzyżowanie,
- mutacja.

Są to tzw. operatory genetyczne. Celem selekcji jest wybór osobników „dobrze" przystosowanych do środowiska. Osobniki te zostaną poddane dalszym operacjom, które będą dążyć do poprawy ich wartości funkcji przystosowania do środowiska. Zazwyczaj selekcja posiada również czynnik losowy odpowiedzialny za zróżnicowanie osobników mających wpływ na „rozwój" populacji. W ten sposób algorytmy genetyczne bronią się przed zbyt szybką zbieżnością do pewnego (nie koniecznie najlepszego) rozwiązania. Osobniki wybrane przez operator selekcji tworzą populację macierzystą. Na osobnikach tej populacji wykonuje się krzyżowanie; do tego celu służy operator krzyżowania. Operacja krzyżowania polega na wymianie części informacji pomiędzy osobnikami, co z założenia ma poprawić ich poziom przystosowania do środowiska, które jest określane przez funkcję przystosowania. Wynikiem pojedynczej operacji krzyżowania są dwa osobniki potomne, potencjalnie lepiej przystosowane

- zazwyczaj zastępują one swoich „rodziców”. Operację krzyżowania przeprowadza się na całej populacji macierzystej zazwyczaj z pewnym prawdopodobieństwem jej zajścia na wybranej parze osobników. Następnie osobniki populacji macierzystej zostają (również z pewnym prawdopodobieństwem) operandami operatora mutacji. Jest to operator jednoargumentowy, który sprawia, iż w osobniku zachodzą losowe zmiany; posiadana przez niego informacja jest lekko zniekształcana. Ma to na celu zwiększenie zawartości operacji eksploracji pośród wszystkich operacji zachodzących w algorytmie genetycznym. W tym ujęciu można powiedzieć, iż operator krzyżowania ma cechy operacji eksploatacji. Wynikiem zastosowania opisanych operatorów jest przetworzona populacja macierzysta, która zazwyczaj staje się kolejną populacją poddawaną przebiegowi algorytmu genetycznego w kolejnej jego iteracji.

W zaimplementowanym algorytmie genetycznym populacja początkowa były generowana losowo. Wykorzystanymi metodami selekcji były:

- ruletka,
- turniej.

Zaimplementowano jeden operator krzyżowania; OX. Do przeprowadzania operacji mutacji wykorzystano operatory:

- insertion,
- invertion,
- transposition.

Algorytm posiadał dodatkową mechanikę, którą był elitaryzm. Jego zaimplementowanie polegało na bezpośrednim przenoszeniu najlepszych osobników z populacji rozważanej w danej generacji (iteracji) do populacji następnej generacji. Liczbę osobników wybieranych do przeniesienia można było dostosować. Celem wprowadzenia tej techniki jest utrzymanie osobników dobrze przystosowanych w kolejnych generacjach. W ten sposób ich dane nie ulegają znaczącym modyfikacjom (wynika to z pominięcia zastosowania na nich operatorów genetycznych) i mają znaczący wpływ na poziom średniego przystosowania kolejnych populacji. W zaimplementowanym algorytmie osobniki tak zachowane zastępowały najgorzej przystosowane osobniki z populacji macierzystej po zastosowaniu na niej operatorów genetycznych. Parametrami liczbowymi algorytmu, którymi można było sterować były:

- rozmiar populacji,
- liczba generacji (iteracji),
- prawdopodobieństwo krzyżowania,
- prawdopodobieństwo mutacji,
- liczba osobników w mechanice elitaryzmu,
- liczba osobników w rundzie turniejowej (dla selekcji turniejowej).

3 Eksperymenty obliczeniowe

W ramach projektu przeprowadzono badania wpływu parametrów na jakość rozwiązań dostarczanych przez zaimplementowany algorytm. Ponadto przeprowadzono eksperymenty obliczeniowe, które polegały na pomiarze czasu wykonania oraz jakości zwracanych rozwiązań dostrojonego algorytmu, w zależności od wielkości instancji problemu. Czas był mierzony w milisekundach, zaś do samego jego pomiaru wykorzystano pakiet z biblioteki standardowej C++ - *std::chrono*. Instancje były podawane z pliku - były to wybrane instancje testowe dostarczone razem z zadaniem projektowym. Komputer, na którym przeprowadzono testy posiadał procesor Intel Core i7-6700HQ oraz był wyposażony w 16 GB RAMu.

Podczas badań wpływu parametrów na jakość rozwiązań ustawiano domyślne parametry o wartościach z tabeli 1. Wykorzystane w tym badaniu parametry są zwarte w 2 kolumnie. Oczywiście badany parametr był zmieniany. Podczas badania czasu i jakości dostarczanych przez algorytmy rozwiązań wykorzystano wyniki badań parametrów i empirycznie wyznaczono optymalne parametry pracy zaimplementowanego algorytmu. Są one zawarte w 3 kolumnie tabeli 1.

Parametr	Badanie parametrów	Badanie czasu i jakości rozwiązań
Rozmiar populacji	50	50
Liczba generacji	1000	2000
P-stwo krzyżowania	0,9	1,0
P-stwo mutacji	0,1	0,12
Liczba osobników elitarnych	10	12
Liczba osobników w rundzie turniejowej	5	1
Operator selekcji	ruletka	ruletka
Operator krzyżowania	OX	OX
Operator mutacji	insertion	insertion

Tablica 1: Parametry użyte podczas eksperymentów obliczeniowych

3.1 Analiza wpływu parametrów algorytmu na jakość rozwiązań

Przedstawione w tej sekcji wykresy reprezentują wyniki badań wpływu parametrów na jakość dostarczanych przez algorytm rozwiązań. Każdy wykres przedstawia dane 4-wymiarowe, gdzie:

- oś X przedstawia wartość badanego parametru,
- oś Y przedstawia błąd średni względny, liczony jako

$$E_{mr} = \frac{S_m - S_{opt}}{S_{opt}} * 100\%,$$

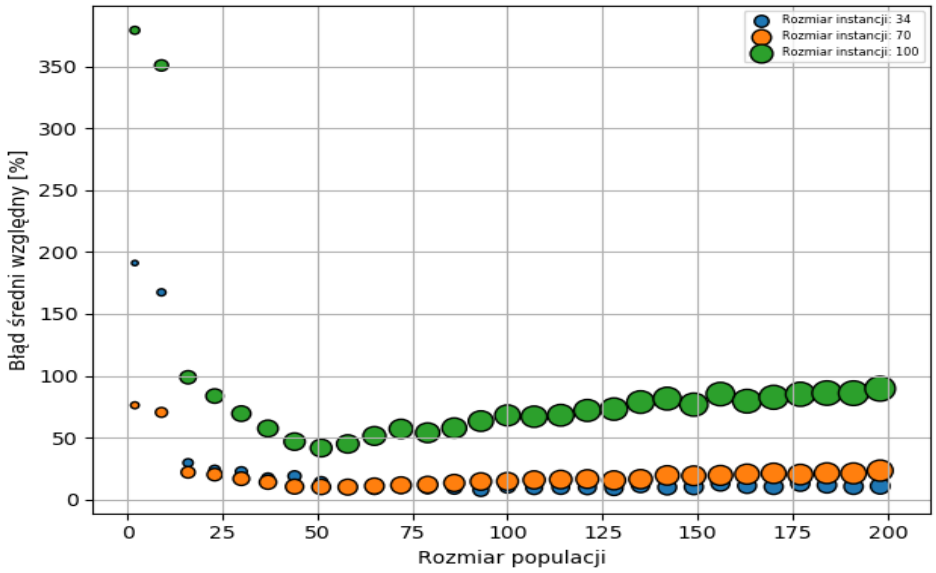
  gdzie S_m oznacza średnią wartość funkcji celu rozwiązania z powtórzonych obliczeń dla tej samej instancji i wartości parametru, zaś S_{opt} oznacza wartość funkcji celu rozwiązania optymalnego,
- kolor punktu przedstawia rozmiar badanej instancji,
- rozmiar punktu obrazuje średni czas trwania badania z powtórzonych obliczeń dla tej samej instancji i wartości parametru.

Przy wyznaczaniu parametrów optymalnej pracy algorytmów kierowano się kryteriami:

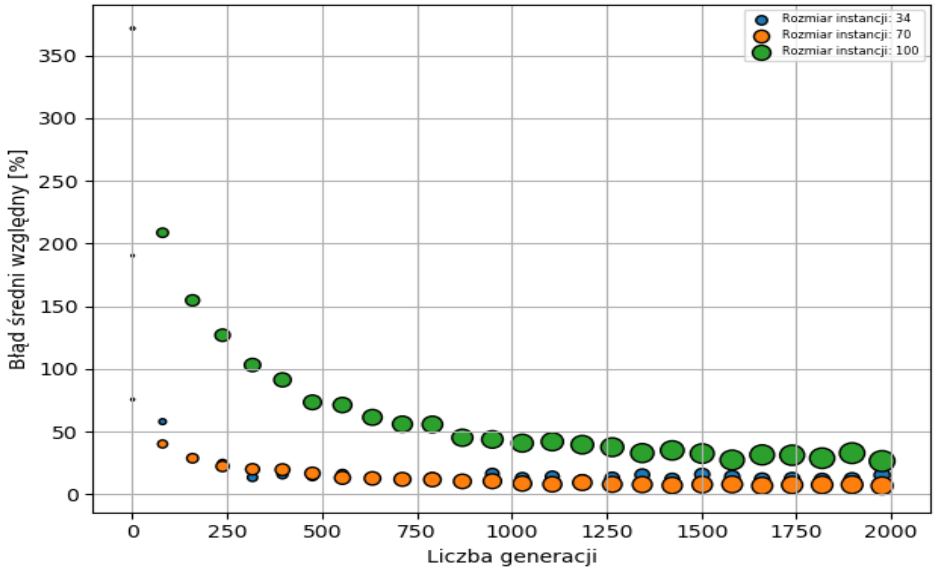
1. możliwie małego błędu średniego względnego,
2. możliwie dużego zagęszczenia punktów pomiarowych,
3. możliwie krótkiego czasu badania.

Kryterium o mniejszym numerze porządkowym było istotniejsze. W przypadkach niejednoznacznych, istotność kryterium nie była brana pod uwagę. Wyznaczone empirycznie optymalne parametry pracy algorytmów są zawarte w 3. kolumnie tabeli 1.

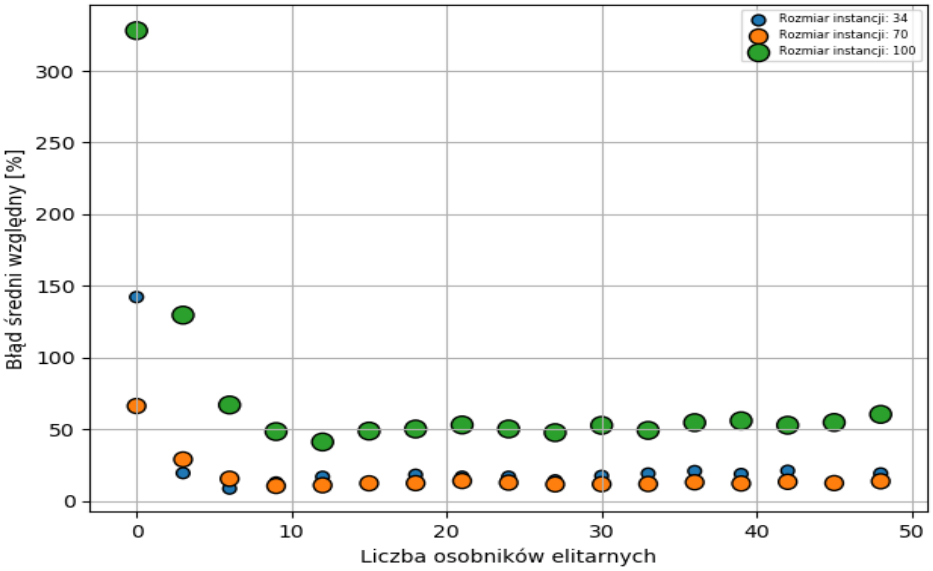
3.1.1 Algorytm genetyczny



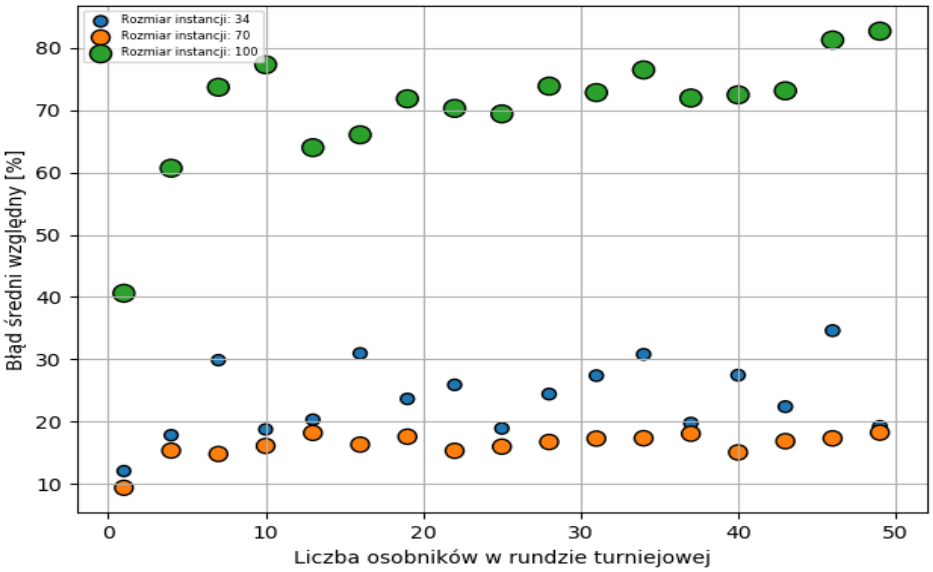
Rysunek 1: Rozmiar populacji



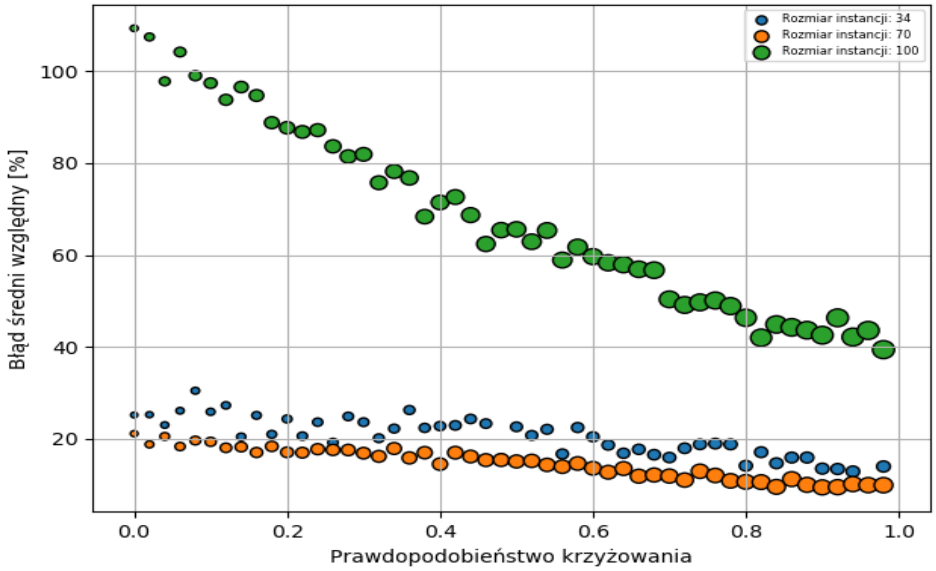
Rysunek 2: Liczba generacji



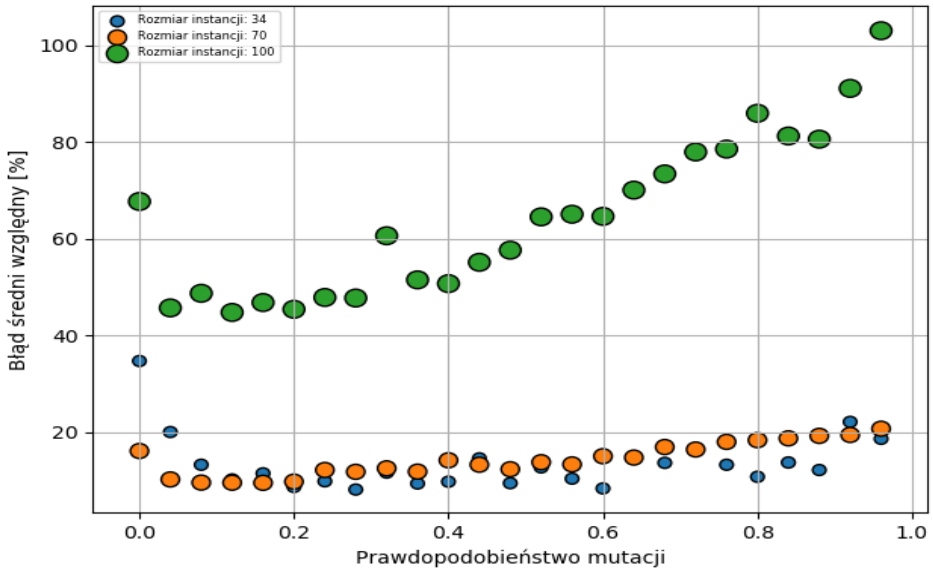
Rysunek 3: Liczba osobników elitarnych



Rysunek 4: Liczba osobników w rundzie turniejowej



Rysunek 5: Prawdopodobieństwo krzyżowania



Rysunek 6: Prawdopodobieństwo mutacji

3.2 Testy czasowe i jakościowe

Za punkt odniesienia, względem którego dokonano analizy czasowej i jakościowej algorytmu genetycznego przyjęto zaimplementowany na poprzednim etapie projektowym algorytm poszukiwania lokalnego z zakazami (wersja z macierzą tabu).

3.2.1 Tabela - zestawienie danych pomiarowych

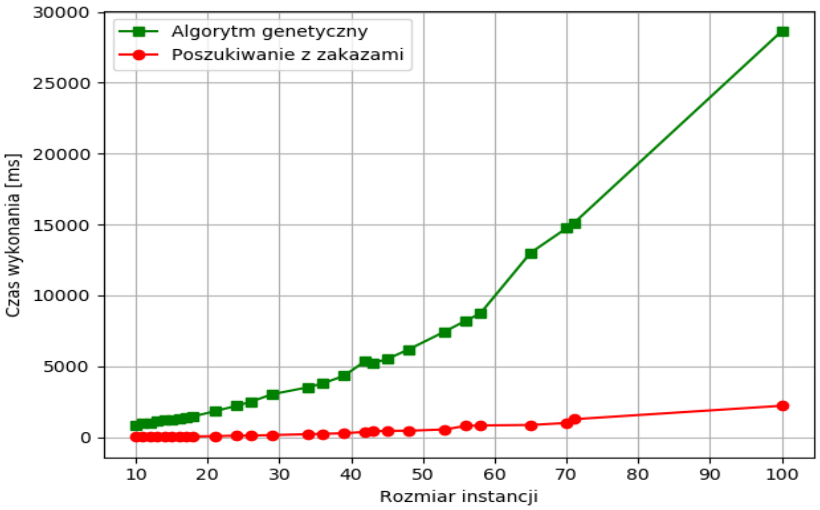
Rozmiar instancji	GA: czas [ms]	TS: czas [ms]	GA: błąd średni względny [%]	TS: błąd średni względny [%]
10	866.45	15.38	1.89	3.30
11	951.91	18.90	4.95	13.37
12	1025.13	22.96	0.00	0.00
13	1145.14	27.62	1.86	0.00
14	1241.56	31.57	18.40	35.20
15	1241.15	36.86	2.06	0.00
16	1268.87	41.94	16.67	33.97
17	1349.85	54.14	0.00	15.38
18	1463.19	53.71	13.90	0.00
21	1833.22	74.19	2.70	10.71
24	2209.38	109.82	5.90	7.39
26	2494.82	116.79	8.22	2.35
29	3030.60	149.36	2.73	8.76
34	3521.84	212.84	10.19	10.50
36	3785.45	236.74	11.95	6.52
39	4327.51	280.14	8.63	8.10
42	5378.27	373.79	10.01	12.16
43	5214.78	409.57	0.23	0.37
45	5511.28	450.52	14.45	14.07
48	6185.84	455.79	7.44	17.04
53	7441.88	545.50	12.53	22.40
56	8225.06	797.50	14.49	12.81
58	8726.63	830.45	7.84	6.62
65	13007.20	862.42	21.59	6.63
70	14737.70	1003.37	6.86	7.41
71	15085.20	1268.66	25.79	9.95
100	28643.30	2217.11	26.65	10.79

Tablica 2: Zestawienie wyników testów czasowych i jakościowych

Legenda:

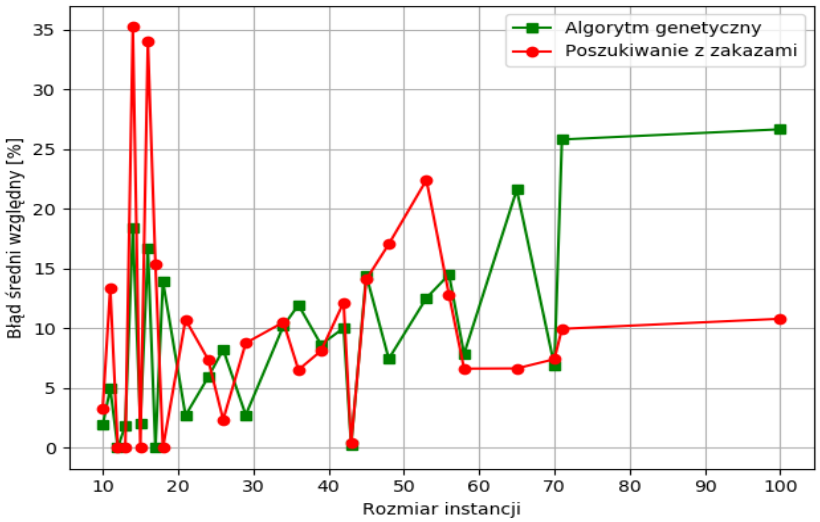
- GA - algorytm genetyczny,
- TS - poszukiwanie z zakazami.

3.2.2 Analiza wyników przeprowadzonych testów



Rysunek 7: Wydajność zaimplementowanego algorytmu

Na rysunku 7 można łatwo zauważyć rozrzut pomiędzy wynikami czasowymi dla algorytmu tabu search oraz algorytmu genetycznego. Jak pokazują badania algorytm genetyczny nie okazał się wydajny. Dla instancji o rozmiarze 100 algorytm genetyczny działa ponad 10 razy dłużej od algorytmu poszukiwania z zakazami. Widoczny na rysunku trend ukazuje, iż różnica ta będzie powiększać się wraz ze wzrostem rozmiaru instancji. Należy zaznaczyć, iż nie można tutaj mówić o jednoznacznym porównaniu, gdyż algorytmy te działały w oparciu o dobraną ręcznie liczbę ich iteracji. Parametr ten był ustawiany tak, aby jakość znalezionych rozwiązań była jak najlepsza w rozsądnie długim czasie pracy.



Rysunek 8: Jakość rozwiązań dostarczanych przez zaimplementowany algorytm

Jak można wywnioskować z rysunku 8 algorytm genetyczny nie okazał się lepszy pod względem jakości zwracanych rozwiązań od algorytmu tabu search dla zbadanych instancji. Wyniki niepokoją tym bardziej, gdyż trend błędu średniego względnego dla algorytmu genetycznego wydaje się być rosnący. Nie można natomiast jednoznacznie określić trendu algorytmu poszukiwania z zakazami. Mimo to wydaje się, iż błąd w tym wypadku jest zależny głównie od typu instancji, nie zaś jej rozmiaru. Tak jak w przypadku testów czasowych, algorytmy te działały w oparciu o eksperymentalnie dobrane parametry, które wydawały się zapewniać ich optymalną pracę.

4 Wnioski

Algorytm genetyczny nie okazał się być lepszy od algorytmu przeszukiwania z zakazami. Wpływ na ten stan rzeczy mogły mieć różne czynniki. Czas działania jest potencjalnym elementem do optymalizacji. Mogłaby ona polegać na zastąpieniu operacji na osobnikach, które w przypadku opisanej implementacji są reprezentowane za pomocą wektora miast oraz wartości funkcji celu przez operację na odniesieniach do tych osobników (np. za pomocą wskaźników). Nie następowałoby w takim wypadku niepotrzebne kopiowanie całych obszarów pamięci, a tylko wskazania na te obszary, co mogłoby uprościć niektóre operacje na osobnikach o złożoności $O(n)$ do $O(1)$. Innym czynnikiem wpływającym na czas jest potencjalnie duża liczba odwołań do operacji generacji liczby losowej. Obszar ten wydaje się być wart przeanalizowania i zastąpienia szeregu operacji losowania pojedynczym losowaniem, gdzie jest to tylko możliwe.

W celu poprawy jakości rozwiązań zwracanych przez algorytm genetyczny należałoby przede wszystkim zaimplementować i sprawdzić działanie innych operatorów krzyżowania, niż wykorzystany w projekcie operator OX. Cechuje się on dużą utrata informacji o stanie problemu, co może prowadzić do niepożądanej, częstej degeneracji średniej wartości funkcji przystosowania osobników populacji.

Mnogość operatorów genetycznych, a także pomysłów na usprawnienie algorytmu genetycznego obrazuje, iż jest to temat warty uwagi. Potencjał metody wyznaczania rozwiązań wykorzystywanej przez algorytmy genetyczne wydaje się spory. Trudnością jest znalezienie metod odpowiednich do rozwiązywanego problemu.