Politechnika Wrocławska

Algorytmy optymalizacji inspirowane naturą
projekt, wtorek 13:15

IMPLEMENTACJA I BADANIE ALGORYTMÓW EWOLUCYJNYCH

Jakub Krupiński, 255356
ITE, IMT
W4N

Prowadzący:
dr hab. inż. Michał Przewoźniczek

18 listopada 2024

Spis treści

1 Wstęp teoretyczny

1.1 Cel projektu

Celem tego projektu jest zapoznanie się z metaheurystyką algorytmu ewolucyjnego przez jego implementację i udowodnienie jej użyteczności w praktycznych zastosowaniach w kontekście problemu komiwojażera w porównaniu z klasycznymi algorytmami deterministycznymi.

1.2 Algorytmy ewolucyjne

Algorytmy ewolucyjne to ogólny termin opisujący oparte na populacji stochastyczne algorytmy przeszukiwania, które starają się naśladować naturalną ewolucję. W ich działaniu kluczowe jest przetwarzanie populacji potencjalnych rozwiązań, które są reprezentowane przez tzw. osobników. Każdy osobnik ma przypisany genotyp oraz fenotyp, gdzie genotyp jest zakodowaną informacją o rozwiązaniu, a fenotyp to konkretne rozwiązanie problemu.[1] Proces algorytmu ewolucyjnego obejmuje kilka kroków: inicjację populacji, ocenę osobników przy użyciu funkcji przystosowania, selekcję najlepszych osobników oraz zastosowanie operacji genetycznych, takich jak krzyżowanie i mutacja

1.3 Problem komiwojażera (TSP)

Problem komiwojażera (TSP) to klasyczny problem optymalizacyjny NP-trudny, który polega na znalezieniu najkrótszej trasy łączącej zestaw miast, przy założeniu, że każdy z nich musi być odwiedzony dokładnie raz i powrót do punktu startowego jest konieczny. Problem ten można sformalizować jako poszukiwanie minimalnego cyklu Hamiltona w pełnym grafie ważonym, gdzie wierzchołki reprezentują miasta, a krawędzie odpowiadają odległościom między nimi[2]

2 Słowniczek

1. Wskaźnik sukcesu/success rate - $S = \frac{\text{wynik osiągnięty}}{\text{najlepszy możliwy wynik}} \times 100\%$.
   Jest to przyjęta przeze mnie miara oceny tego jaki wynik został osiągnięty przez zaproponowany przeze mnie algorytm - było to możliwe dzięki temu, że każda badana przeze mnie instancja miała już podane rozwiązania optymalne globalnie, które zostały przeze mnie użyte.

2. Eksperyment podstawowy - eksperyment, który stanowi bazę dla wszystkich moich obliczeń. Za każdym razem, kiedy w tej pracy wyraźnie nie jest wskazane jaka jest specyfika użytych rozwiązań, oznacza to, że jest używany eksperyment podstawowy.
   Metody użyte w ramach tego eksperymentu to:

   - Inicjalizacja: *losowa*
   - Selekcja: *turniejowa*
   - Krzyżowanie: *Ordered Crossover*
   - Mutacja: *swap*

   Parametry tego eksperymentu to:

   - Czas działania algorytmu: *20 minut*
   - Rozmiar selekcji turniejowej: *2*
   - Szybkość mutacji: *0,07*
   - Wielkość populacji: *800*

3. Eksperyment najlepszy/najbardziej optymalny/najbardziej optymalnej konfiguracji - eksperyment, którego metody oraz parametry zostały dobrane w ramach wykonywania kolejnych eksperymentów i wyciągania z nich wniosków w celu znalezienia rozwiązania, które zapewni na koniec najlepsze rezultaty.
   Metody użyte w ramach tego eksperymentu to:

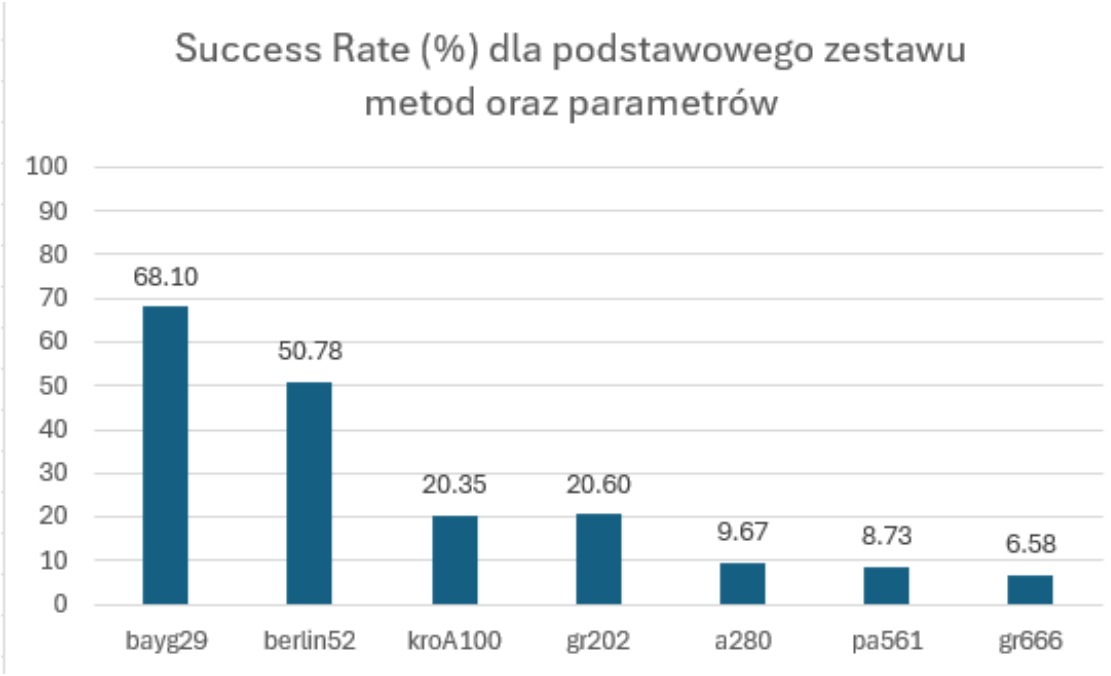
- Inicjalizacja: *zachłanna*
- Selekcja: *turniejowa*
- Krzyżowanie: *Cycle Crossover*
- Mutacja: *inverse*

Parametry tego eksperymentu to:

- Czas działania algorytmu: *20 minut*
- Rozmiar selekcji turniejowej: *2*
- Szybkość mutacji: *0,1*
- Wielkości populacji: *100, 800* oraz *1600*

3 Analiza wyników

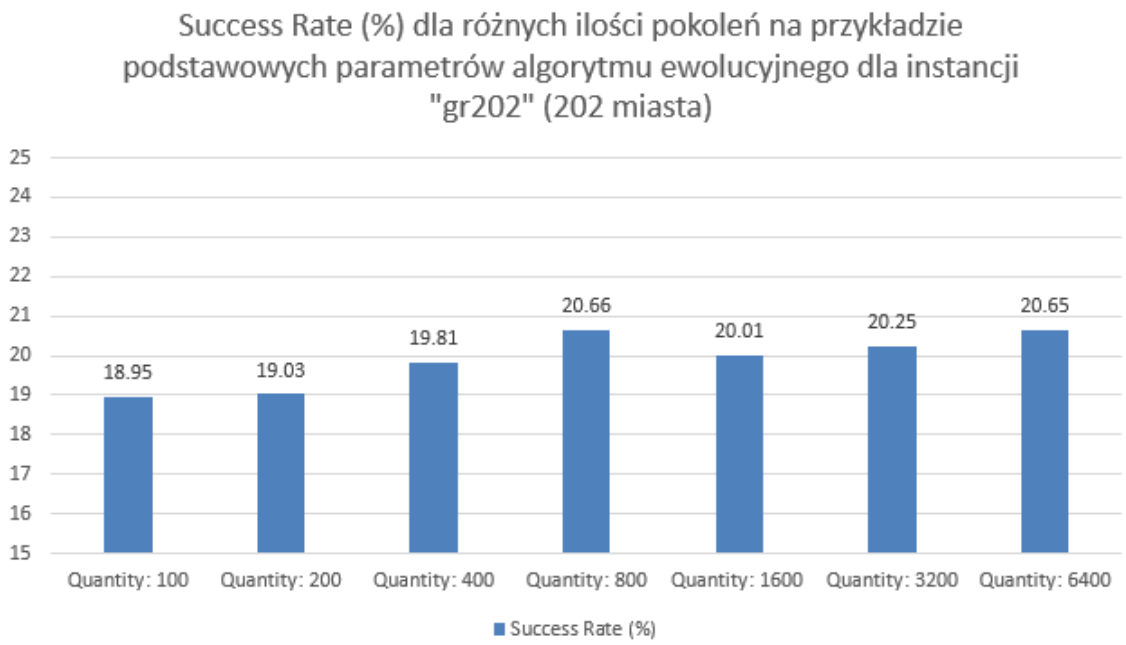
3.1 Eksperyment podstawowy



Rysunek 1: Wyniki eksperymentu podstawowego

Eksperyment podstawowy przyniósł wyniki zdecydowanie gorsze od spodziewanych, co pokazuje tylko, że niezwykle istotne dla algorytmu ewolucyjnego jest to, żeby jego konkretne metody oraz parametry nie były zawsze takie same, tylko że muszą być one świadomie dobrane biorąc pod uwagę specyfikę problemu. W przypadku instancji o niewielkiej ilości miast wyniki prezentują się zadowalająco, jednakże przy ilości miast większej lub równej 100 efektywność algorytmu spada drastycznie - widoczne jest, że nie jest on w stanie przejrzeć odpowiednio dokładnie przestrzeni rozwiązań bez znacznego zwiększenia czasu bądź też zasobów obliczeniowych. Celem kolejnych eksperymentów jest zbadanie tego które operatory algorytmu ewolucyjnego oraz parametry posiadają największy wpływ na jego efektywność, a następnie wykorzystanie zdobytej wiedzy przy próbie dobrania odpowiedniej konfiguracji, a następnie sprawdzenie jak będzie ona w stanie poradzić sobie w porównaniu z algorytmami deterministycznymi.

3.2 Eksperyment badający wpływ liczby pokoleń na wyniki algorytmu ewolucyjnego

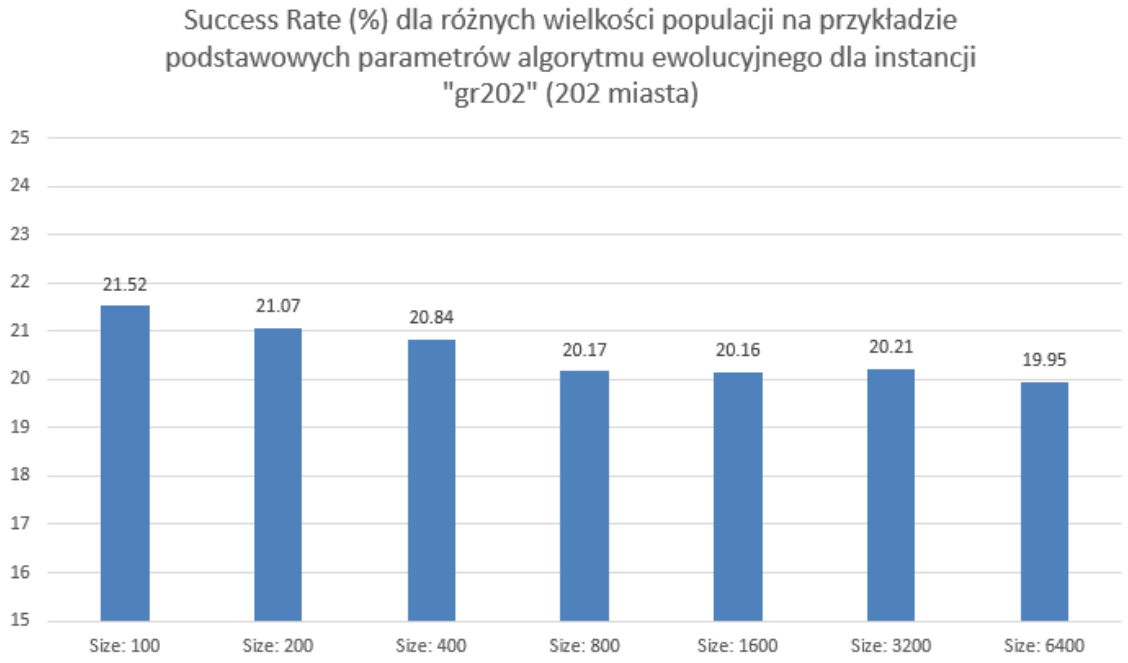


Rysunek 2: Wyniki eksperymentu: różne liczby pokoleń

Eksperyment ten został przeprowadzony dla tej samej konfiguracji, co eksperyment podstawowy, z tą różnicą, warunkiem stopu nie jest tutaj czas działania algorytmu, a liczba pokoleń. Choć podstawowym warunkiem przerwania działania algorytmu ewolucyjnego w ramach moich badań nie jest liczba pokoleń, a czas, który upłynął podczas działania algorytmu, postanowiłem zawrzeć tu również dodatkowe badanie, które sprawdzi jak sama liczba pokoleń wpłynie na wyniki. Korzystanie z czasu działania pod wieloma względami jest bardziej obiektywne pod kątem oceny różnych wersji algorytmu, jako że daje on równą szansę dla wszystkich konfiguracji - oznacza to bowiem, że jeśli algorytm będzie "lekki" w znaczeniu tego, że przeanalizowanie pełnego pokolenia trwa krócej, taka konfiguracja ma wtedy szansę na zaprezentowanie tego przez wykonanie o wiele większej ilości pokoleń niż inna, która będzie te obliczenia wykonywać znacznie wolniej. Należy jednak pamiętać, że nie zawsze jest to do końca optymalne - mogą się bowiem zdarzyć takie sytuacje, w których np. algorytm ewolucyjny osiągnie fitness optimum lokalnego F_1 po 5 minutach działania programu i nie opuści go przez kolejne 15 minut prowadzenia obliczeń. Dzięki przeprowadzaniu badań na eksperymencie podstawowym, w których pod uwagę brany jest nie czas, a liczba pokoleń, uzyskana zostaje odpowiedź na pytanie: "Przy jakiej liczbie pokoleń zaproponowana przeze mnie konfiguracja algorytmu ewolucyjnego "utknie" w optimum lokalnym i przestanie poprawiać swoje wyniki?" Na podstawie przedstawionych przeze mnie wyników nie wygląda jednak na to, aby taka wartość została osiągnięta - pomijając słupek z ilością 800-set pokoleń, którego wyniki są wyjątkowo dobre w porównaniu do jego sąsiadów na wykresie, pomiędzy kolejnymi ustaleniami liczb pokoleń widoczne jest, że wyniki osiągane przez algorytm ewolucyjny stale wzrastają w dość stałym tempie. Ze względu na ograniczenia czasowe wynikające z wybranej przeze mnie technologii (język Python) oraz ograniczonej prędkości obliczeniowej sprzętu, który wykonywał dane eksperymenty nie zostały sprawdzone większe wartości ilości pokoleń, przez co eksperymentu tego nie można nazwać niestety sukcesem. Można jednak przypuszczać, że gdyby proces ten był kontynuowany dla odpowiedniej ilości kolejnych większych wartości liczby pokoleń, to osiągnięty zostałby moment, w którym różnice we wzroście $fitnessu$ byłyby pomijalnie małe bądź zerowe (osiągnięty został punkt, w którym algorytm ewolucyjny utknął w optimum lokalnym) - wiedza taka mogła-

by być więc wykorzystana np. w przypadku przeprowadzania badań przy użyciu wspomnianego wcześniej warunku stopu opartego na czasie działania algorytmu - jeśli osoba przeprowadzająca obliczenia widziałaby, że wspomniana przeze mnie graniczna wartość pokoleń dla danej konfiguracji została już przekroczona a wyniki osiągane przez algorytm nie ulegają już widocznym poprawom, mogłoby pozwolić to na przerwanie działania algorytmu wcześniej, jako że mało prawdopodobne byłoby już wtedy, żeby ten wynik znacząco poprawić, co oznaczałoby znaczną oszczędność czasu oraz zasobów.

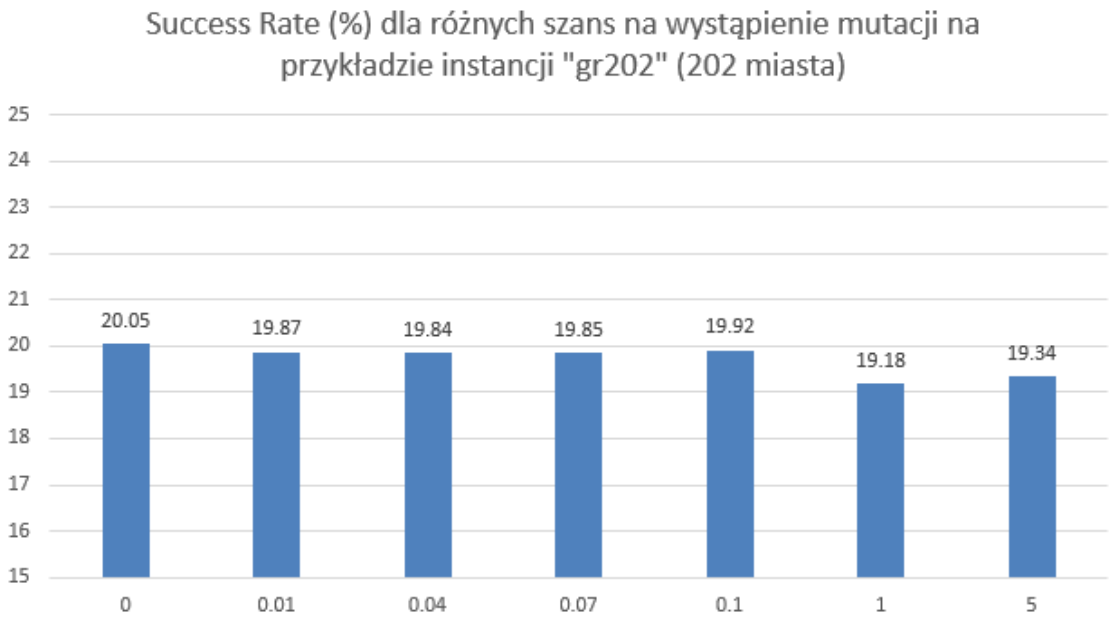
3.3 Eksperyment badający wpływ wielkości populacji na wyniki algorytmu ewolucyjnego



Rysunek 3: Wyniki eksperymentu: różne wielkości populacji

Eksperyment ten został przeprowadzony dla tej samej konfiguracji, co eksperyment podstawowy, z tą różnicą, że parametr wielkości populacji zmienia się. Wyniki tego eksperymentu w kontekście problemu komiwojażera okazały się być dla mnie zaskakujące. Rezultaty których oczekiwałem podczas rozpoczynania tego eksperymentu, to znalezienie optymalnej odpowiedzi wśród wartości "średnich" z tych, które zaproponowałem - okazało się jednak, że algorytm ewolucyjny najlepiej poradził sobie w przypadku najmniejszych populacji, wartości "średnie" poradziły sobie natomiast bardzo słabo. Im mniejsza liczba osobników, tym mniej czasu zajmuje mojemu algorytmowi obliczenie pełnego pokolenia i przejście do następnego, co jest bardzo korzystne, jako że w tym eksperymencie głównym warunkiem wyjścia nie jest liczba pokoleń, a czas. Pozwolenie mniejszej ilości osobników na przeszukiwanie przestrzeni rozwiązań w przeciągu większej liczby pokoleń okazało się bardziej skuteczne, niż duża populacja, która wykonuje te same obliczenia, jako że czas, który zajmuje przeanalizowanie pełnego pokolenia jest wtedy znacznie większy i algorytm nie dostaje wystarczająco dużo czasu, aby za pomocą mutacji opuścić znalezione ekstrema lokalne i istotnie zbliżyć się do ekstrema globalnego.

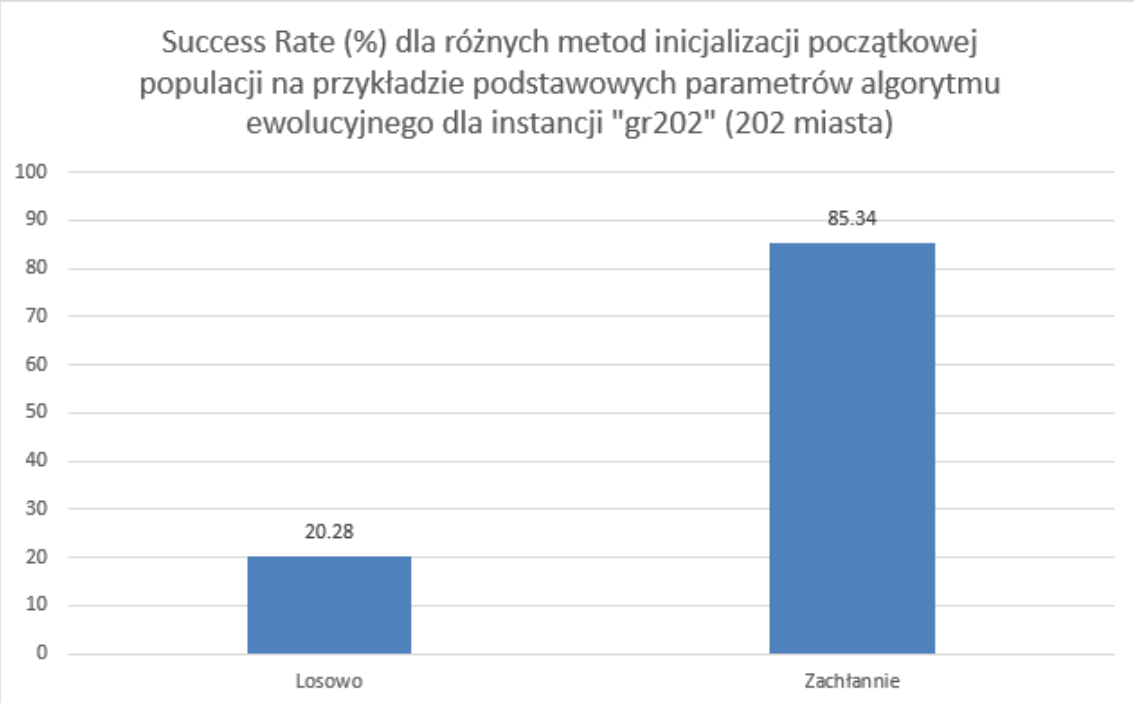
3.4 Eksperyment badający wpływ różnych wartości parametru szybkości mutacji na wyniki algorytmu ewolucyjnego



Rysunek 4: Wyniki eksperymentu: różne wartości szybkości mutacji

Dobranie odpowiedniej szybkości mutacji stanowi nie lada wyzwanie. Ustawienie zbyt niskiej wartości sprawia, że rozwiązania nigdy nie opuszczą znalezionych przez siebie ekstremów lokalnych, zbyt duża wartość może natomiast doprowadzić do tego, że populacja przegląda przestrzeń rozwiązań w sposób zbyt losowy i nieprzewidywalny, zbyt często opuszczając obszary, które normalnie zostałyby uznane za obiecujące pod kątem szukania lepszych rozwiązań, ze względu na losowo występującą mutację. Eksperyment ten został przeprowadzony dla tej samej konfiguracji, co eksperyment podstawowy, z tą różnicą, że parametr szybkości mutacji zmienia się. Pierwszą rzeczą, która rzuca się w oczy podczas oceny wyników danego eksperymentu jest to, że najlepsze wyniki osiągnięte zostały w przypadku, kiedy mutacje nigdy nie występowały - oznacza to, że dla danej konfiguracji algorytmu ewolucyjnego, przy tak niskich wskaźnikach sukcesu, algorytm ewolucyjny radził sobie lepiej, kiedy losowe mutacje nie zaburzały jego procesu przeszukiwania przestrzeni rozwiązań. Choć jest to coś, co na pewno trzeba wziąć pod uwagę, należy pamiętać też o tym, że im wyższy wskaźnik sukcesu został już osiągnięty, tym ciężej algorytmowi opuścić znalezione optima lokalne, podczas gdy przy tak niskich wartościach wskaźnika sukcesu jak w przypadku przeprowadzanego tu eksperymentu, znalezienie lepszego rozwiązania nawet przy "naiwnym" podejściu do tego problemu nie powinno stanowić trudności. Warto zauważyć, że wartości z zalecanego zakresu (*0.01 - 0.1*) poradziły sobie tutaj bardzo dobrze, podczas gdy zaproponowane dla celów tego eksperymentu wartości, które są znacznie wyższe (*1* oraz *5*), osiągnęły zauważalnie gorsze rezultaty.

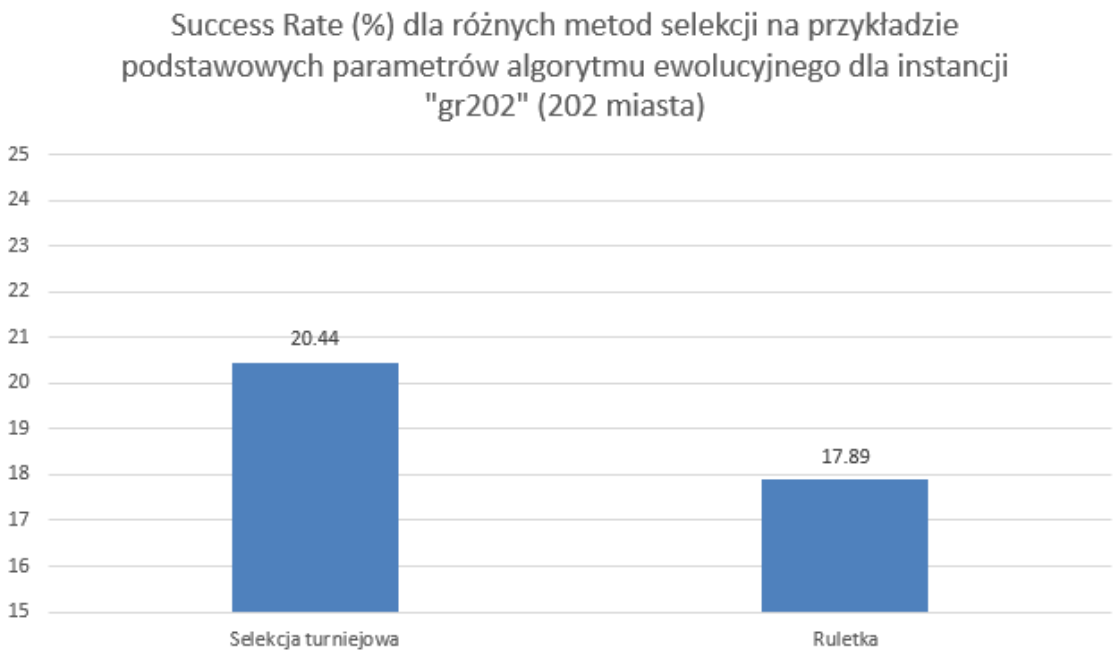
3.5 Eksperyment badający wpływ różnych metod inicjalizacji początkowej populacji na wyniki algorytmu ewolucyjnego



Rysunek 5: Wyniki eksperymentu: różne typy inicjalizacji początkowej populacji

Jest to eksperyment, który doskonale pokazuje jak bardzo wiele można osiągnąć poprzez wspomaganie algorytmu ewolucyjnego przez odpowiednie dobranie używanych w nim metod. Zaproponowano tutaj dwa różne podejścia: losową inicjalizację populacji początkowej, oraz inicjalizację zachłanną, w której po wybraniu losowego osobnika startowego jego kolejne połączenia wybierane były tak, aby były one lokalnie jak najkrótsze. Zapewniło to początkową populację, której startowe wartości funkcji *fitness* były znacznie wyższe niż w przypadku inicjalizacji losowej, dzięki czemu algorytm ewolucyjny mógł skupić się na poprawianiu dobrych rozwiązań na jeszcze lepsze, zamiast na szukaniu w przestrzeni takich, które stanowiłyby dobrą podstawę do dalszych analiz. Algorytm zachłanny wykonuje tutaj bardzo dużą część pracy, dzięki czemu wyniki osiągane przez algorytm ewolucyjny są o wiele bardziej imponujące, mimo że jego konfiguracja wciąż pozostaje podstawowa.

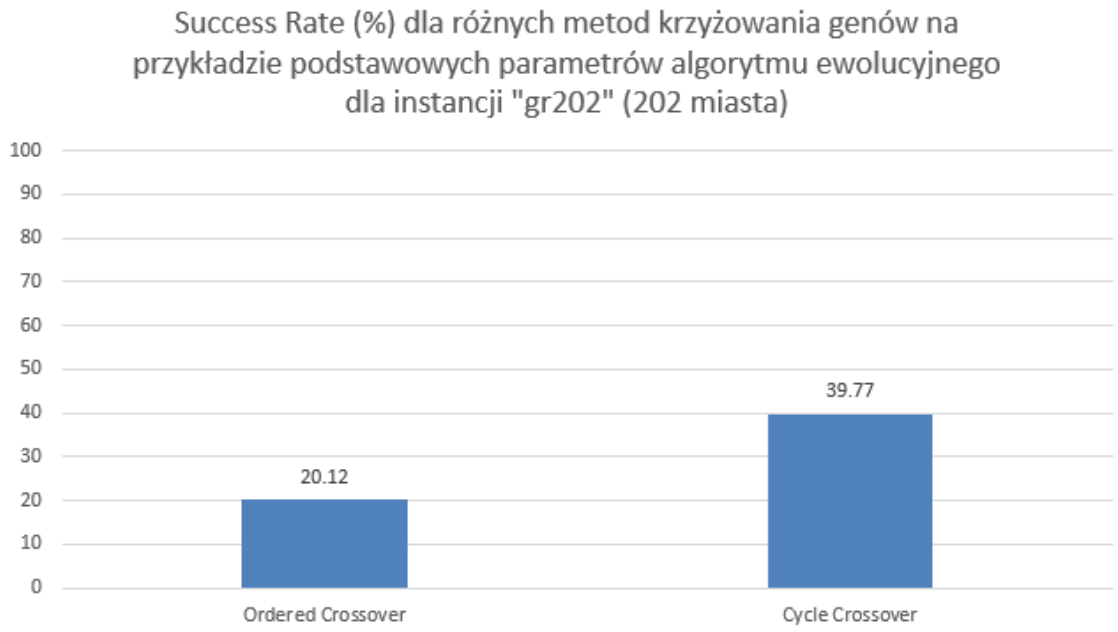
3.6 Eksperyment badający wpływ różnych metod selekcji na wyniki algorytmu ewolucyjnego



Rysunek 6: Wyniki eksperymentu: różne typy metod selekcji

Eksperyment ten został przeprowadzony dla tej samej konfiguracji parametrów, co eksperyment podstawowy, ale z tą różnicą, że badania przeprowadzone zostały dla dwóch różnych metod selekcji - ruletki oraz selekcji turniejowej (rozmiar turnieju: 2). Wyniki są tutaj jednoznaczne - selekcja turniejowa radzi sobie wyraźnie lepiej od ruletki, co było spodziewanym rezultatem. Obie te metody wprowadzają możliwość wylosowania rozwiązań o niskiej wartości funkcji *fitness*, co czasem może być korzystne ze względu na to, że pozwala utrzymać większą różnorodność wśród osobników, ale może być też bardzo problematyczne, ze względu na to, że "niekorzystne" losowania mogą niemal bezpowrotnie pogorszyć jakość osobników znajdujących się w nowej generacji. Selekcja turniejowa nie promuje tej różnorodności tak bardzo jak ruletka, jako że przy wylosowaniu dwóch osobników zawsze wybiera tego, którego wartość funkcji *fitness* jest wyższa, zapewnia jednak przejście bardziej jakościowych osobników do następnej generacji. To która z tych metod jest bardziej korzystna bez wątpienia zależne jest od indywidualnej charakterystyki badanego problemu oraz konkretnej jego instancji, jednakże w przypadku przeprowadzonych przeze mnie badań selekcja turniejowa okazała się być znacznie skuteczniejsza w znajdywaniu optymalnych rozwiązań problemu komiwojażera. Należy pamiętać też, że zaproponowane przeze mnie rozwiązania nie uwzględniają mechanizmy *elitaryzmu*, który pozwala na zachowaniu określonej liczby najlepszych osobników bieżącej populacji w następnej generacji niezależnie od procesu selekcji, co mogłoby być ciekawym rozwiązaniem m.in. w przypadku selekcji ruletkowej, ponieważ postanowiłoby zachować dużą różnorodność osobników wraz z jednoczesnym zapewnieniem, że nie dojdzie do straty tych rozwiązań, które są najbardziej obiecujące w kontekście dalszych poszukiwań optimów.

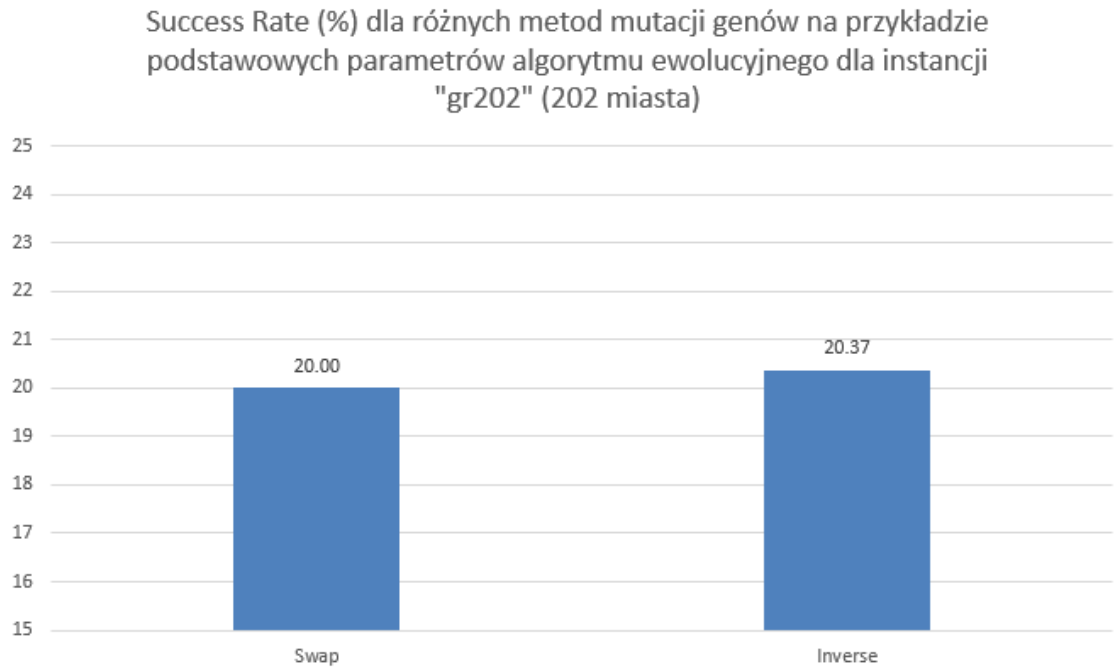
3.7 Eksperyment badający wpływ różnych metod krzyżowania na wyniki algorytmu ewolucyjnego



Rysunek 7: Wyniki eksperymentu: różne typy metod krzyżowania

Eksperyment ten został przeprowadzony dla tej samej konfiguracji parametrów, co eksperyment podstawowy, ale dla dwóch różnych metod krzyżowania osobników - Ordered Crossover (OX) oraz Cycle Crossover (CX). Metodą eksperymentalną przekonałem się tutaj, że dla przykładu problemu komiwojażera dla mojej implementacji algorytmu genetycznego metoda CX jest zdecydowanie bardziej skuteczna, dlatego to ona będzie brana pod uwagę podczas doboru najbardziej optymalnej konfiguracji parametrów oraz metod.

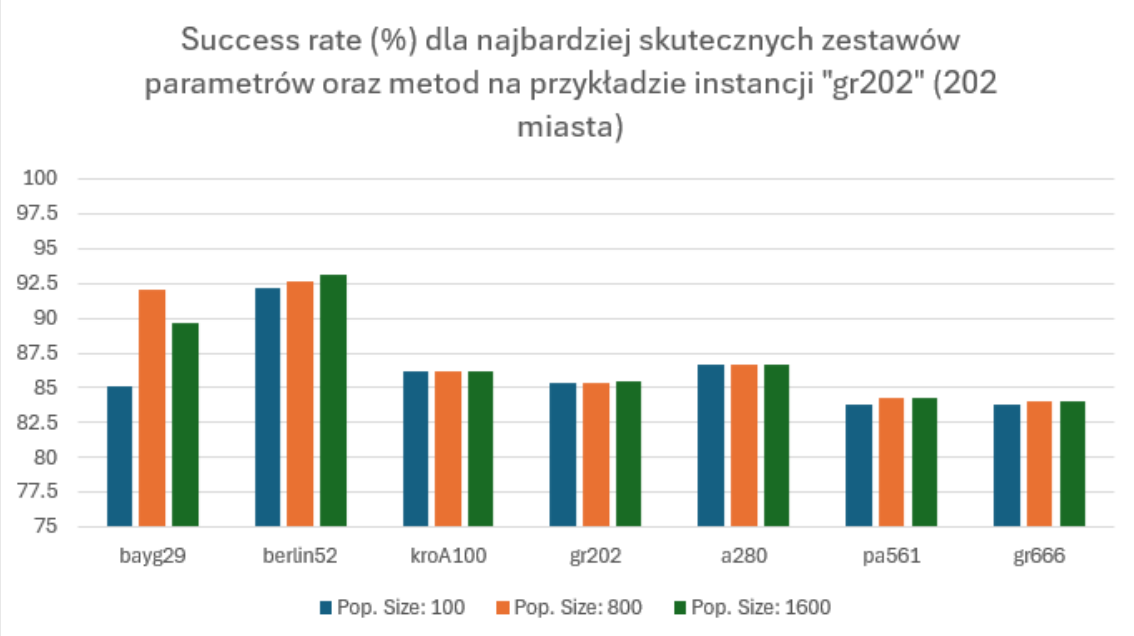
3.8 Eksperyment badający wpływ różnych metod mutacji na wyniki algorytmu ewolucyjnego



Rysunek 8: Wyniki eksperymentu: różne metody mutacji

Eksperyment ten został przeprowadzony dla tej samej konfiguracji parametrów, co eksperyment podstawowy, ale z tą różnicą, że badania przeprowadzone zostały dla dwóch różnych metod mutacji - *swap* oraz *inverse*. Różnice w wynikach pomiędzy tymi dwoma strategiami są niewielkie, mutacja *inverse* poradziła sobie jednak nieco lepiej. Ze względu na charakter obu tychh strategii, podejrzewam że rezultat ten można przypisać temu, że ponieważ mutacja *inverse* zmienia większy fragment chromosomu, to pozwala ona na o wiele lepszą dywersyfikację osobników i skuteczniejszą eksplorację przestrzeni rozwiązań; główną zaletą mutacji *swap* jest bez wątpienia jej prostota, jednakże zamiana jedynie dwóch pozycji w chromosomie jest niedużą zmianą, przez co skuteczność tej metody może maleć w miarę wzrostu skomplikowania problemu (np. w przypadku problemu komiwojażera: wzrostu liczby miast).

3.9 Eksperyment badający wpływ optymalnych metod i parametrów na wyniki algorytmu ewolucyjnego



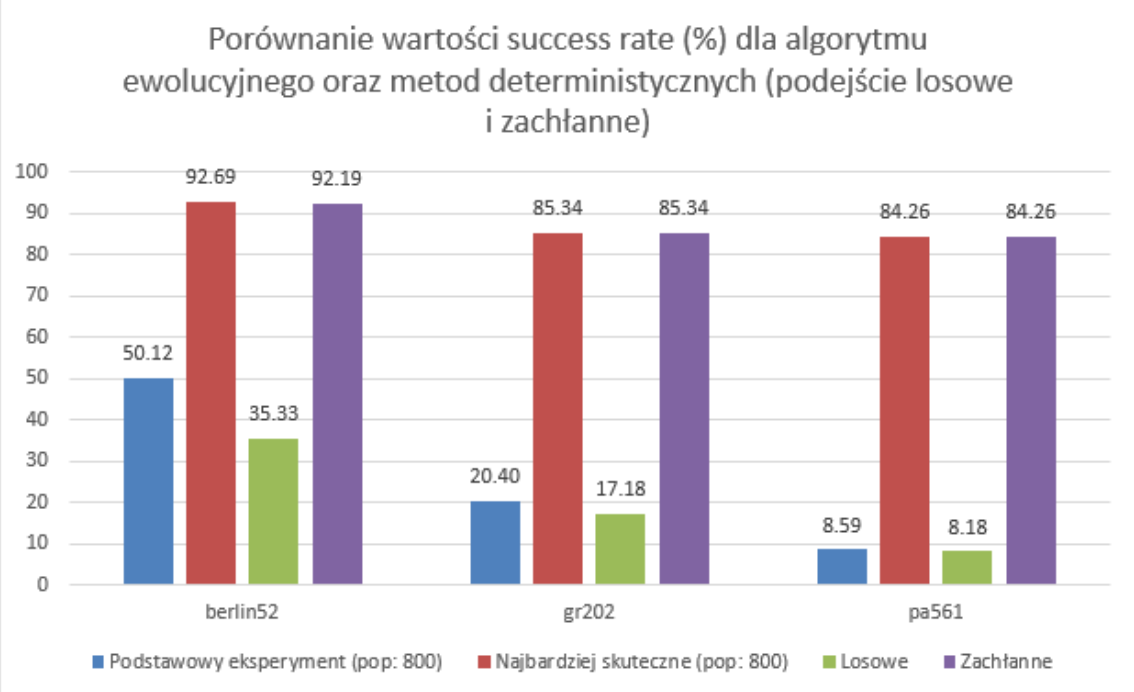
Rysunek 9: Wyniki eksperymentu: najbardziej optymalna konfiguracja

Eksperyment najbardziej optymalnej konfiguracji obejmuje przeprowadzenie powtórnych obliczeń dla każdych z rozważanych w ramach moich badań instancji przy pomocy algorytmu ewolucyjnego, którego metody oraz parametry zostały dobrane na drodze eksperymentalnej w poprzednich częściach tej pracy. Metody te to:

1. Inicjalizacja zachłanna
2. Selekcja turniejowa (rozmiar turnieju: 2)
3. Krzyżowanie cykliczne
4. Mutacja Inverse

Wszystkie eksperymenty były przeprowadzane dla szybkości mutacji *0.1*, dla trzech różnych wielkości populacji: *100*, *800* oraz 1600. Choć na podstawie poprzednich punktów badań można było dojść do wniosku, że to *100* jest optymalną ilością populacji początkowej, eksperyment dowiódł, że nie jest to prawdą, jako że większym wielkościom początkowym udało się osiągnąć zauważalnie wyższe wskaźniki sukcesu. Widoczna jest tutaj duża różnica, która pojawia się w przypadku instancji o ilości miast 100 lub więcej, gdzie znajdywana wartość wskaźnika sukcesu maleje znacznie, co można przypisać rosnącemu skomplikowaniu kombinatorycznemu problemu oraz trudnościom wynikającym z samej instancji.
Na podstawie tego, co udało mi się zbadać w miarę przygotowywania tego projektu jest to zestaw najlepszych rozwiązań, które udało się osiągnąć - aby osiągnąć lepsze rezultaty należałoby więc spojrzeć jeszcze raz na zaproponowane przeze mnie wartości parametrów oraz implementacje metod i spróbować znaleźć inne podejścia, które będą w stanie lepiej radzić sobie w ramach rozwiązywania problemu komiwojażera.

3.10 Eksperyment porównujący wyniki algorytmu ewolucyjnego z algorytmami deterministycznymi



Rysunek 10: Wyniki eksperymentu: porównanie algorytmu ewolucyjnego z algorytmami deterministycznymi

Eksperyment ten obejmuje przeprowadzenie czterokrotnych obliczeń za pomocą różnych rozwiązań problemu dla trzech różnych instancji o różnych ilościach miast (kolejno 52, 202 i 561 miast). Wspomniane rozwiązania problemu to:

1. Eksperyment podstawowy
2. Eksperyment najbardziej optymalnej konfiguracji
3. Algorytm deterministyczny losowe
4. Algorytm deterministyczny zachłanny

Spośród zaproponowanych podejść natychmiast zauważalne jest to, że dwa spośród tych rozwiązań radzą sobie znacznie lepiej, a dwa - znacznie gorzej. Najsłabiej wypadają: *eksperyment podstawowy* oraz *algorytm losowy*, podczas gdy najlepsze wyniki udało się osiągnąć przy pomocy *eksperymentu najbardziej optymalnej konfiguracji* oraz *algorytmu zachłannego*.

Niezadowalające wyniki eksperymentu podstawowego zostały przedstawione już wcześniej, tu pojawia się on natomiast ponownie jedynie w celach porównawczych. Algorytm losowy osiąga - zgodnie z oczekiwaniami - zdecydowanie najgorsze rezultaty, jako że nie kierują nim żadne mechanizmy, które pomagałyby naprowadzać go w kierunku rozwiązań, które zapewniają wyższe wartości funkcji *fitness*. Nawet, gdyby jedna z prób algorytmu losowego przyniosła dobre wyniki, nie byłoby to coś, co jest niezawodne i powtarzalne, zwłaszcza w przypadku instancji o dużych ilościach miast, gdzie problem staje się kombinatorycznie zbyt skomplikowany.

Algorytm zachłanny przynosi dobre rezultaty, a celem zaimplementowanego przeze mnie algorytmu ewolucyjnego było znalezienie sposobu na poprawienie jego wyników. Cel ten nie jest

niestety zawsze osiągalny. Determinizm algorytmu zachłannego powoduje, że zawsze osiąga on takie same rezultaty, algorytm genetyczny posiada natomiast wbudowane elementy losowości, które mają pozwolić mu na bardziej efektywne przeszukiwanie przestrzeni rozwiązań. Wyniki obliczeń wykonywanych na najlepszej konfiguracji algorytmu genetycznego często różnią się od siebie - nawet jeśli różnice są nieznaczne - a różnice te wynikają z tego, czy algorytmowi udało się uciec z ekstremów lokalnych (często z tych samych jak te, które zostały osiągnięte przez algorytm zachłanny) i w przypadku zaprezentowanych tu wyników eksperymentu nie udało się osiągnąć założonego celu. Wyniki są jednak bardzo podobne dla instancji "berlin52" i identyczne w przypadku instancji "gr202" i "pa561", co wskazuje na to, że algorytmowi udaje się osiągać rozwiązania o dość wysokiej jakości; być może nie jest to w pełni niezawodne, jednakże niewykluczonym jest, że w przypadku w którym algorytm ten otrzymuje więcej czasu, udałoby mu się uciec z ekstremów lokalnych i osiągnąć kolejne, jeszcze wyższej jakości rozwiązania. Niewykluczonym jest też, że osiągnięcie lepszych rezultatów byłoby możliwe przez zaproponowanie jeszcze innych metod inicjalizacji, selekcji, krzyżowania lub mutacji, jednakże na potrzeby zaproponowanych w tym zadaniu eksperymentów ja postanowiłem ograniczyć się do jedynie dwóch opcji dla każdego ze wspomnianych tu aspektów.

4 Podsumowanie

Cele projektu, które zostały założone na jego początku nie zostały niestety w pełni spełnione. Po wykonaniu sumiennych badań na podstawie mojej autorskiej implementacji algorytmu ewolucyjnego znalazłem bowiem taką jego konfigurację, która jest w stanie w skończonym, założonym na początku każdego z eksperymentów czasie (*20 minut*) osiągać lepsze wyniki niż klasyczne algorytmy deterministyczne - różnice te są jednak niewielkie i nie udaje się ich osiągać w sposób w pełni powtarzalny - niemniej jednak udało mi się osiągnąć je wielokrotnie, co udowadnia, że algorytmy te potrafią osiągać zadowalające wyniki po ich odpowiednim skonfigurowaniu. Na potrzeby tych badań przeprowadziłem testy par różnych metod inicjalizacji pierwszej generacji, selekcji, krzyżowania i mutacji - należy jednak pamiętać o tym, że istnieje również wiele innych rozwiązań dla każdej z tych operacji, które bez wątpienia również wpłynęłyby na to jakie wyniki osiągał mój algorytm i istnieje bardzo duża szansa, że niektóre z nich wpłynęłyby na osiągane przeze mnie wyniki w sposób pozytywny. Choć nie udało mi się w pełni osiągnąć celu całkowicie powtarzalnego otrzymywania lepszych wyników od algorytmów deterministycznych, to bez wątpienia udało mi się udowodnić, że algorytm ewolucyjny może znajdować praktyczne zastosowanie przy rozwiązywaniu trudnych obliczeniowo problemów takich jak problem komiwojażera - a dalsze badania, które objęłyby nowe zestawy parametrów, a także nowe ulepszenia oraz inne metody implementowania wspomnianych operacji podstawowych (np.: wprowadzenie odpowiednio dobranego *elitaryzmu*, *Inicjalizacji opartej na danych historycznych*, *Ranked Selection*, *Partially Matched Crossover*, *Frameshift Mutation* itp.) bez wątpienia pozwoliłyby osiągać jeszcze lepsze rezultaty, które w ramach założonych przeze mnie ograniczeń czasowych byłyby niemożliwe do osiągnięcia przez algorytmy deterministyczne w sposób powtarzalny.

Literatura

[1] Bartz-Beielstein, T., Branke, J., Mehnen, J., Mersmann, O.: Evolutionary algorithms. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **4**(3), 178–195 (2014). https://doi.org/10.1002/widm.1124, https://doi.org/10.1002/widm.1124

[2] Razali, N., Geraghty, J.: Genetic algorithm performance with different selection strategies in solving tsp. vol. 2 (01 2011)