Maciej Szulc 211630

Piątek 17^{05}

Sztuczna inteligencja i inżynieria wiedzy

Sprawozdanie z ćwiczenia nr 1 – algorytmy genetyczne

Wstęp

Zadanie polegało na zapoznaniu się, zaimplementowaniu oraz analizie wpływu parametrów na metaheurystyki algorytmów genetycznych dla problemu QAP.

Algorytm genetyczny jest ogólną strategią znajdowania rozwiązań optymalnych lub niewiele się od nich różniących, bazującą na ideach selekcji naturalnej i genów. Algorytm genetyczny symuluje przetrwanie najlepszych osobników w kolejnych generacjach w celu rozwiązania zadanego problemu. Każda populacja składa się z określonej ilości osobników, które w tym problemie sį tożsame z pełnymi rozwiązaniami.

Algorytm zaimplementowano w języku Python3. Są 2 wersje algorytmu, jedna ograniczona liczbą iteracji, druga ograniczona czasem wykonania.

Zagadnienia projektowe

Zastosowanie algorytmu genetycznego wymaga rozwiązania kilku zagadnień, które zależą od natury problemu. Są to:

- Reprezentacja osobnika
- Wyznaczanie początkowej populacji
- Obliczanie wartości przystosowania osobników
- Selekcja – wybór puli rozrodczej
- Krzyżowanie (Crossover) – tworzenie potomków danych osobników
- Mutacje
- Rozmiar populacji
- Prawdopodobieństwo mutacji i krzyżowania
- Warunek zakończenia algorytmu

Algorytm

Reprezentacja osobnika

W kontekście problemu QAP osobnik reprezentuje konkretne rozmieszczenie fabryk w lokalizacjach.

Wyznaczanie początkowej populacji

W algorytmie genetycznym początkowa populacja składa się z określonej liczby losowo utworzonych osobników.

Obliczanie wartości przystosowania

Funkcja kosztu przyjmuje postać

$$\sum_{i}^{n}\sum_{j}^{n} d_{ij} * f_{\phi(i)\phi(j)}$$

gdzie d i f to odpowiednio elementy macierzy odległości i przepływów, a ϕ to reprezentacja osobnika.

Selekcja

Zaimplementowano 3 sposoby wyboru puli rozrodczej:

- selekcja turniejowa
- selekcja ruletkowa

- najlepsza połowa osobników

Krzyżowanie

Zaimplementowano krzyżowanie operatorem PMX.

Operator PMX wybiera ciąg genów z jednego rodzica, który bez zmian zostaje przekopiowany do potomka. Pozostałe geny z drugiego rodzica zostają rozproszone po potomku wg. określonego algorytmu. Algorytm PMX można opisać za pomocą następujących punktów:

1. Wybierz losowo ciąg genów z rodzica 1 przekopiuj je bezpośrednio do potomka. Zapamiętaj indeksy początku i końca segmentu.
2. Spośród indeksów należących do segmentu, wybierz wszystkie geny z rodzica 2 które nie zostały jeszcze przekopiowane do potomka.
   a) Dla każdego genu A z tych genów:
   b) Zapamiętaj index genu w rodzicu 2. Znajdź gen V, znajdujący się w rodzicu 1 na tej samej pozycji.
   c) Znajdź ten sam gen V w rodzicu 2.
   d) Jeżeli index genu V w rodzicu 2 jest częścią oryginalnego segmentu, przejdź do punktu (b) używając tego genu.
   e) W przeciwnym wypadku, umieść gen A w potomku na pozycji odpowiadającej indeksowi genu V w rodzicu 2.
3. Przekopiuj pozostałe geny z rodzica 2 do potomka

Mutacje

Mutacje mogą zajść z określonym prawdopodobieństwem dla każdego genu. Zaimplementowano mutacje typu swap (zamiana danego genu z losowym innym genem danego osobnika) oraz insert (zmiana pozycji danego genu w genotypie i odpowiednie przesunięcie pozostałych genów).

Badania wpływu parametrów na działanie algorytmu

Celem eksperymentu było badanie zaprogramowanego algorytmu pod względem dokładności wyników i czasów wykonania.

Wykorzystano skrypt testujący, który dla danych parametrów uruchamia algorytm wiele razy, a następnie uśrednia wyniki i wyświetla je na wykresie. Wyraźne, kolorowe linie są uśrednieniem wszystkich przebiegów testowych. Średnie są wyciągane dla każdej iteracji osobno, jeśli nie wszystkie uruchomienia dotarły do i-tej iteracji, średnia dla tej iteracji nie jest liczona (stąd niektóre wykresy „urywają się”). Średni najlepszy koszt (Best avg cost) jest średnią kosztów najlepszych uzyskanych rozwiązań ze wszystkich uruchomień.

Do testów jest wykorzystywany algorytm w którym kryterium zatrzymania jest czas wykonania.

Badania dla danych z pliku had12.dat

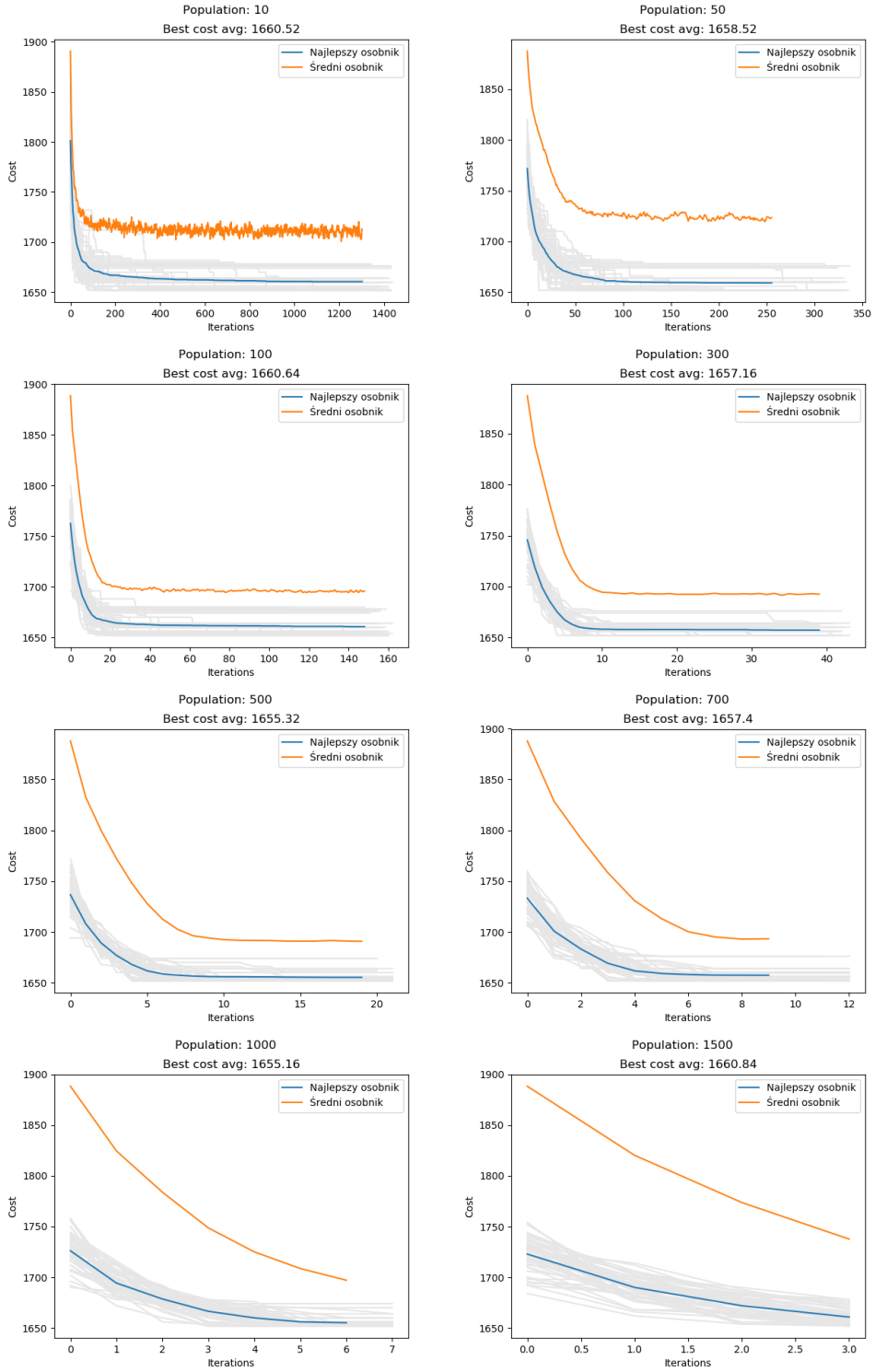
Liczba uruchomień algorytmu dla każdego zestawu parametrów: 50

Wyjściowe parametry (użyte w badaniu, jeśli nie jest podane inaczej przy wykresie):

Limit czasowy: 1 sekunda
Populacja: 500 osobników
Prawdopodobieństwo mutacji: 5%
Prawdopodobieństwo krzyżowania: 100%
Metoda selekcji: turniejowa

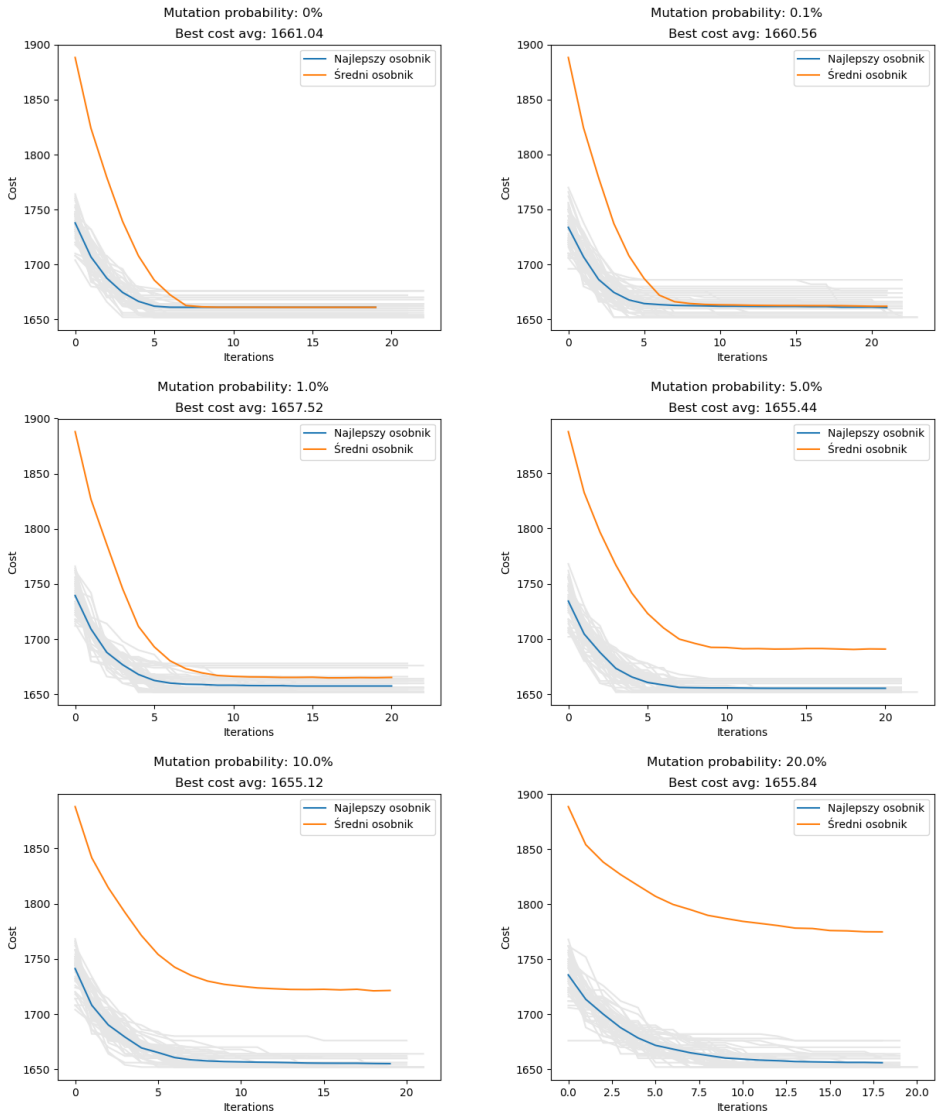
Metoda mutacji: swap
Wielkość turnieju: 5% wielkości populacji

Liczebność populacji



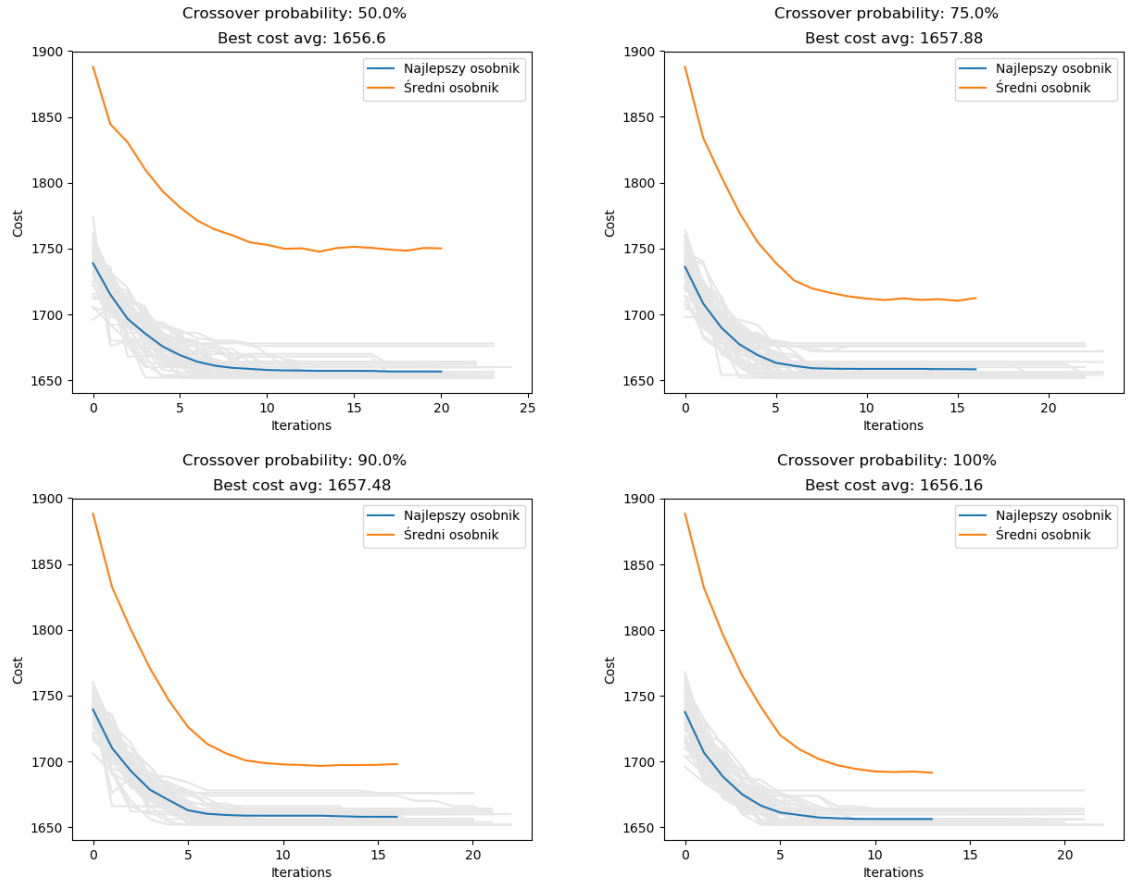
Najlepsze wyniki osiąga liczebność populacji w zakresie 500-1000 osobników. Dla mniejszych gorsze wyniki są spowodowane mniejszą różnorodnością osobników w populacji, a dla większych w tym samym czasie wykonania algorytm wykonuje mniej iteracji, co pogarsza wynik.

Prawdopodobieństwo mutacji



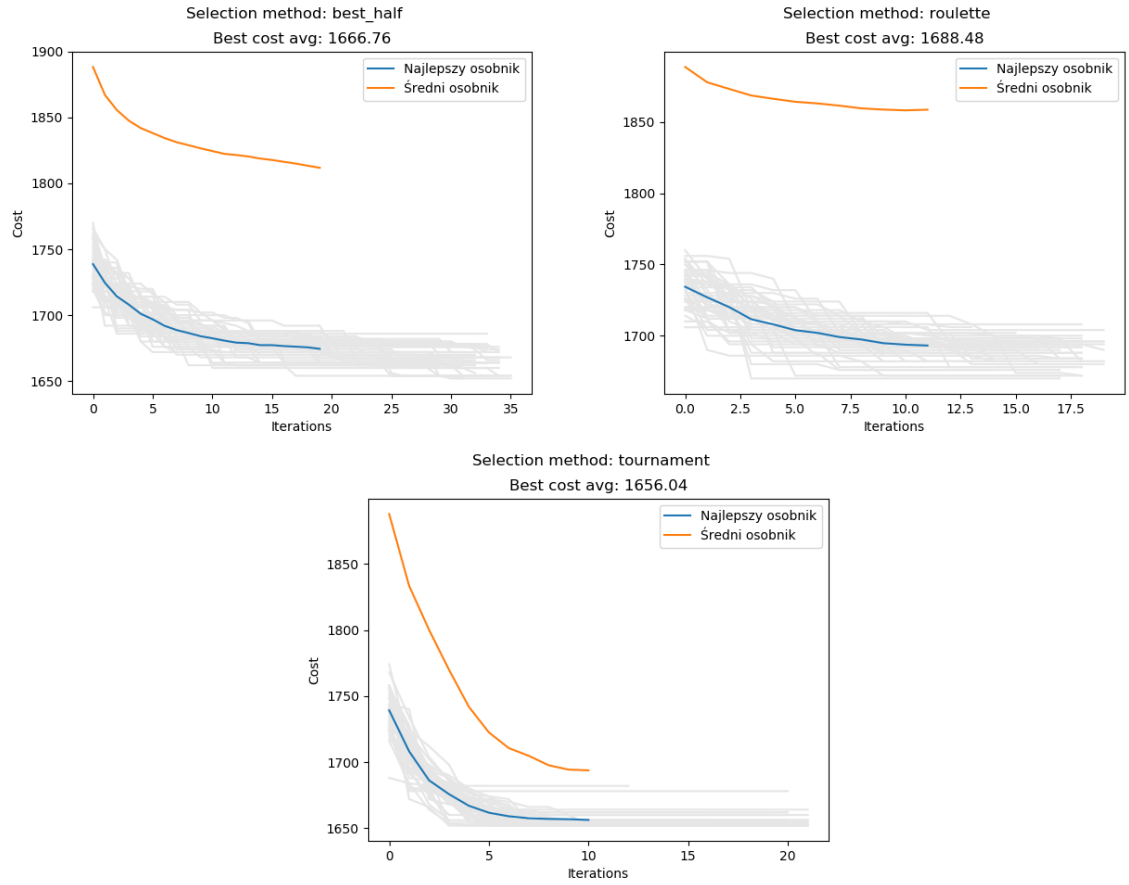
Najlepsze wyniki otrzymano z prawdopodobieństwem mutacji 5-10%.

Prawdopodobieństwo krzyżowania



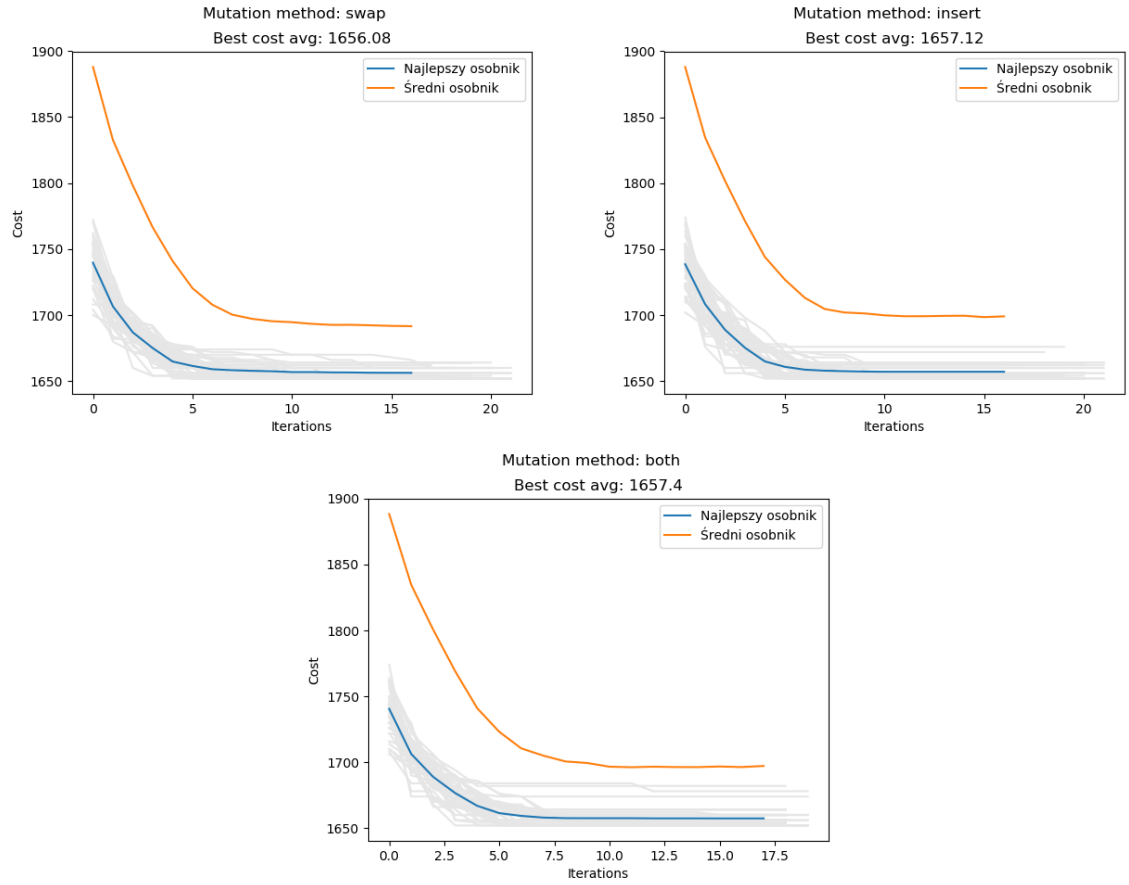
Prawdopodobieństwo krzyżowania ma wpływ na to jak szybko algorytm zbiega do pożądanej wartości – im bliżej 100% tym szybciej.

Metoda selekcji



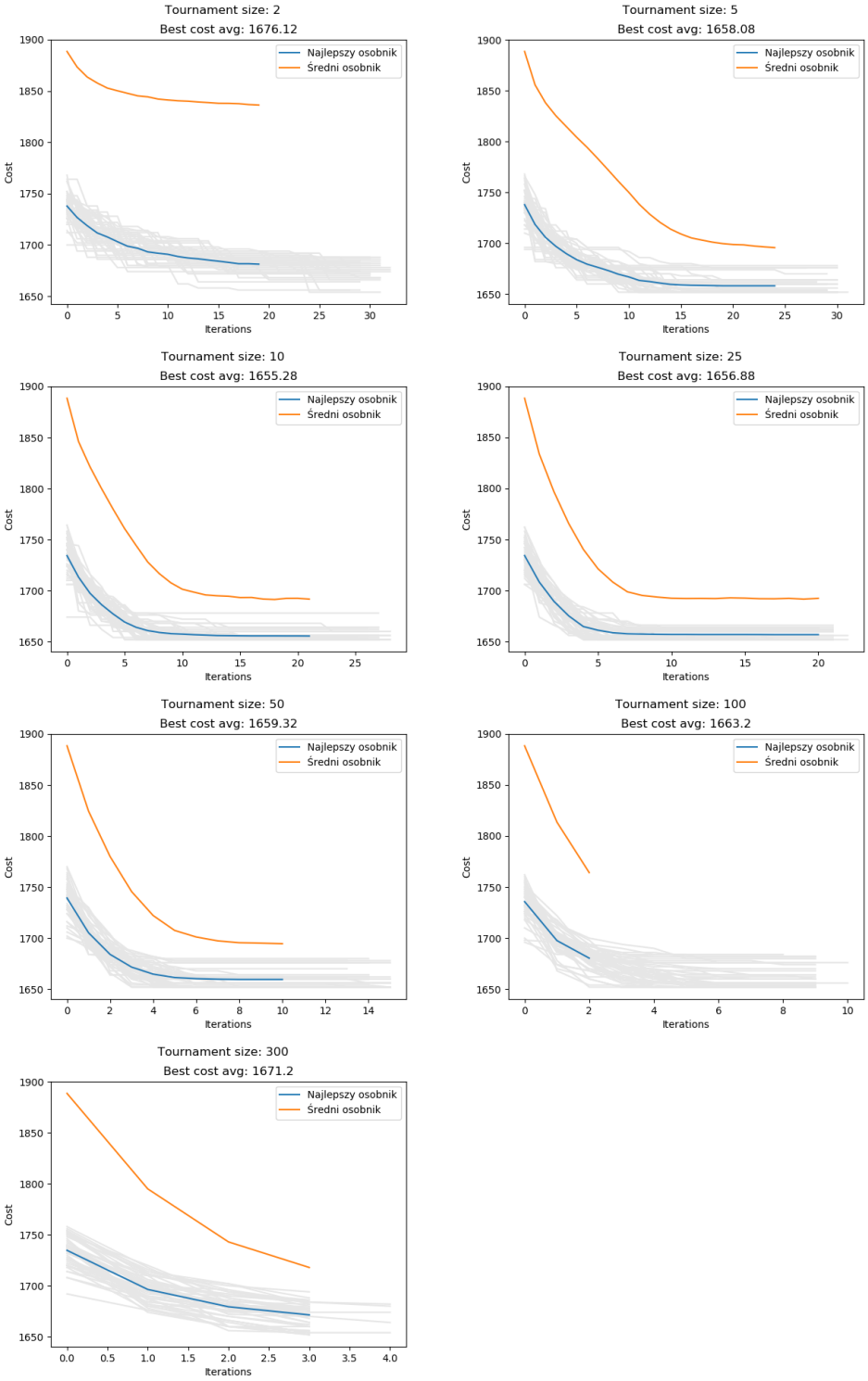
Dla podanych parametrów selekcja turniejowa jest zdecydowanie najskuteczniejsza. Jednak bardzo słaby wynik metody ruletkowej może być spowodowany pozostałymi parametrami, które są zoptymalizowane pod kątem metody turniejowej, np. prawdopodobieństwo mutacji.

Metoda mutacji



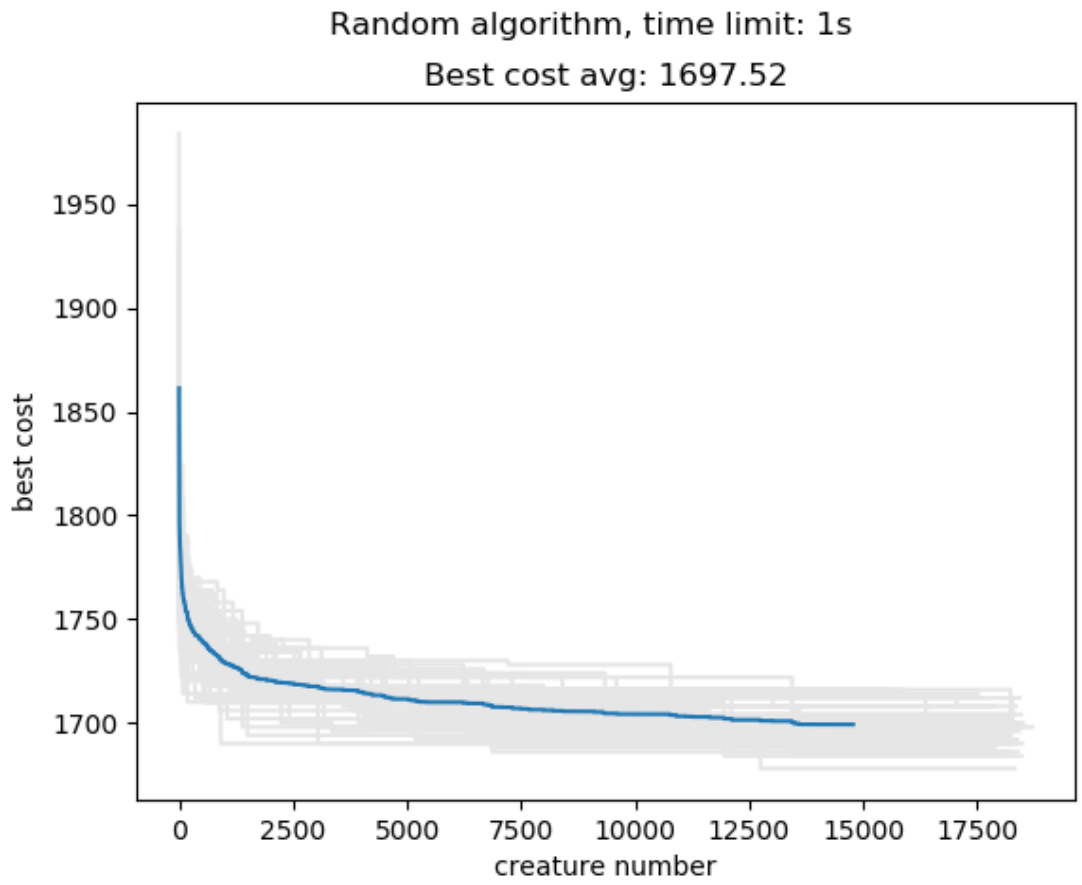
Między metodami mutacji występują niewielkie różnice w osiąganych wynikach. Minimalnie lepsze wyniki daje mutacja typu swap.

Wielkość turnieju



Wielkość turnieju ma istotny wpływ na działanie algorytmu. Jeśli turniej jest mały, wybierane osobniki są różnorodne, ale średnie przystosowanie wybranych osobników nie jest zbyt wysokie. Większe turnieje zwracają lepszych osobników, ale mniej zróżnicowanych. Poza tym duże turnieje są dość „ciężkie” obliczeniowo przez co wykonuje się mniej iteracji algorytmu i wyniki są gorsze.

Porównanie z innymi algorytmami



Średnia wartość osiągnięta w czasie 1s algorytmem losowym: 1697

Wartość osiągnięta algorytmem zachłannym: 7846

Wyniki dla algorytmu genetycznego są znacznie lepsze.

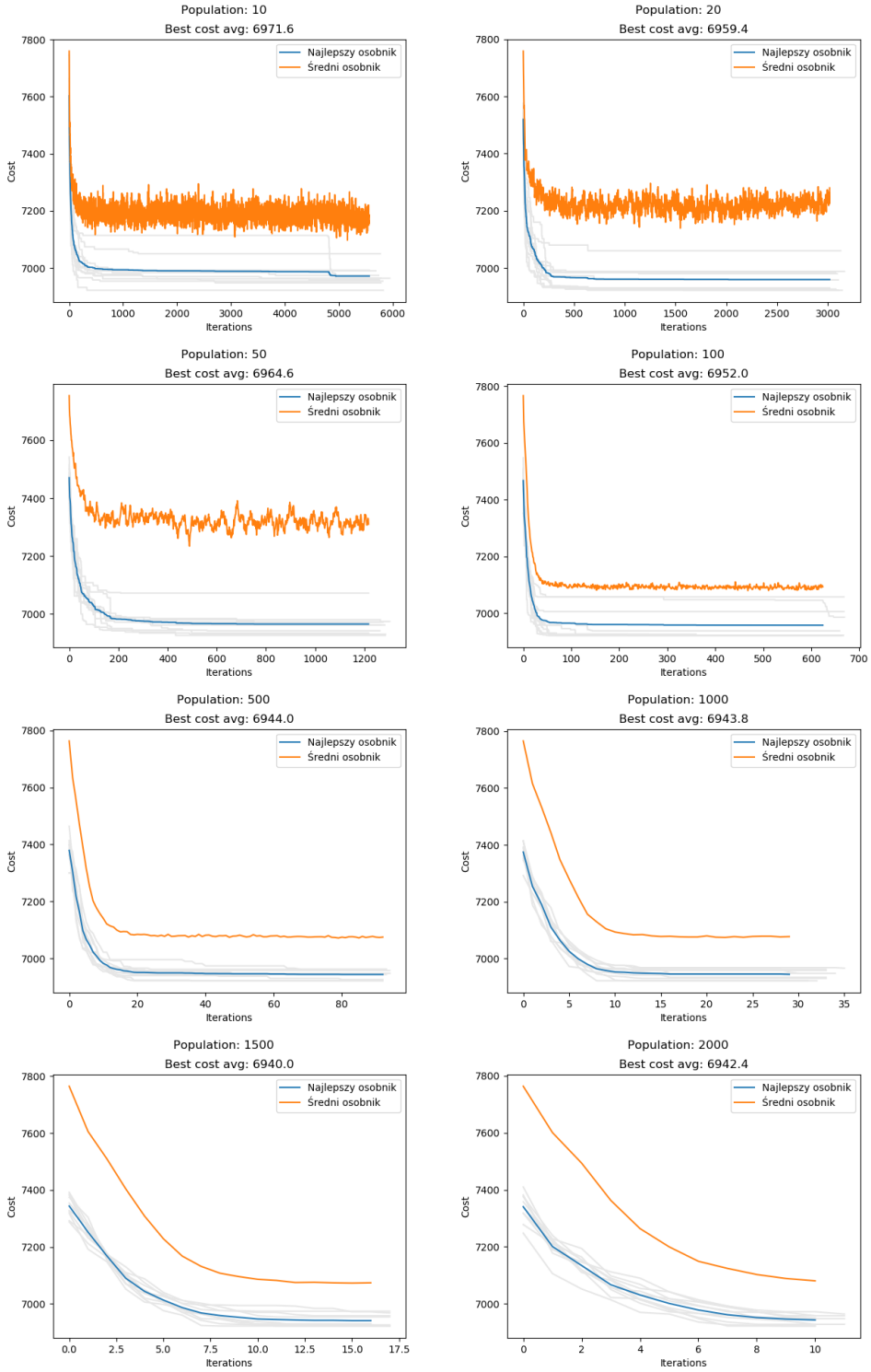
Badania dla danych z pliku had20.dat

Liczba uruchomień algorytmu dla każdego zestawu parametrów: 10

Wyjściowe parametry (użyte w badaniu, jeśli nie jest podane inaczej przy wykresie):

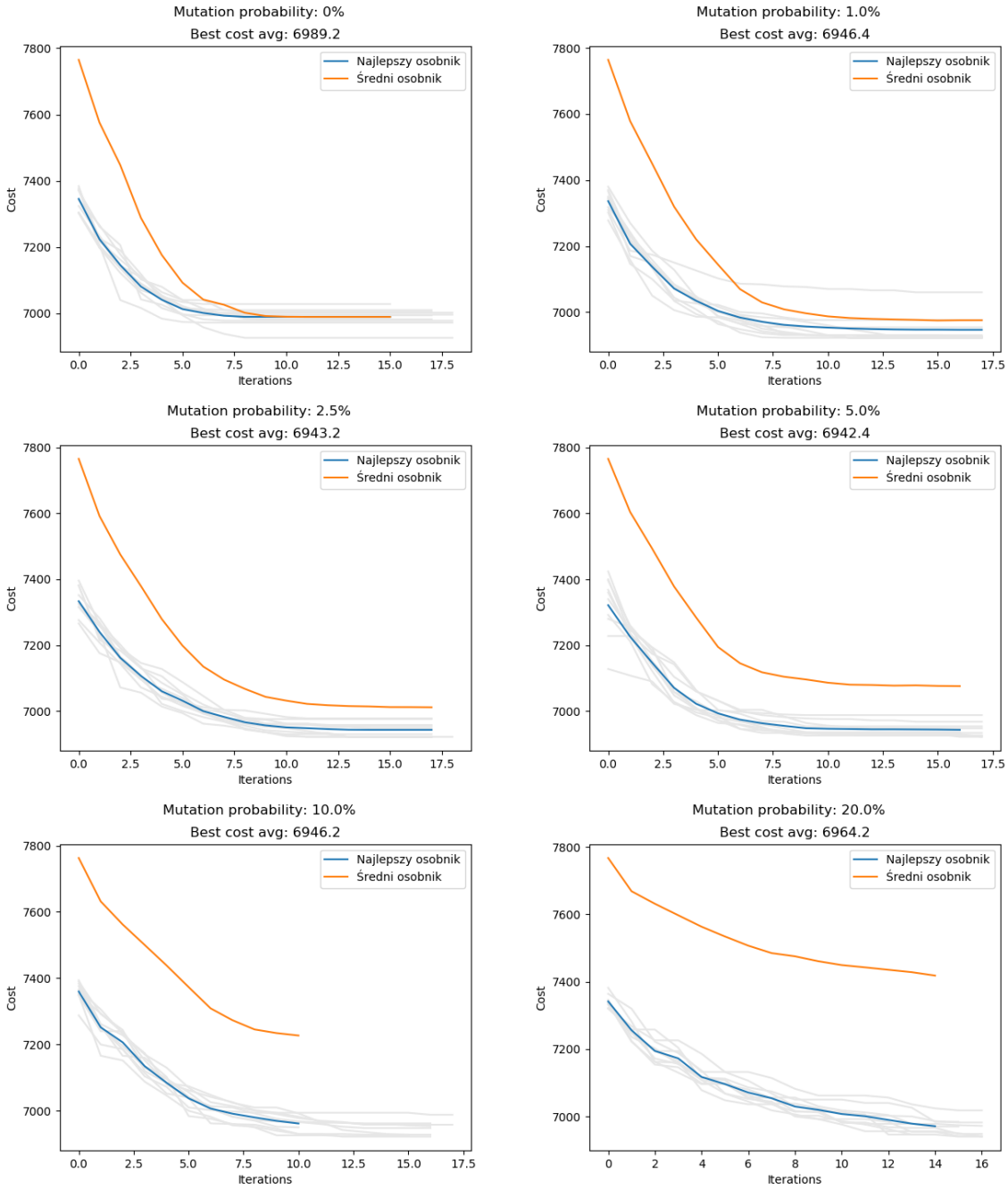
Limit czasowy: 5 sekund
Populacja: 1500 osobników
Prawdopodobieństwo mutacji: 5%
Prawdopodobieństwo krzyżowania: 100%
Metoda selekcji: turniejowa
Liczebność turnieju: 5% liczebności populacji

Liczebność populacji



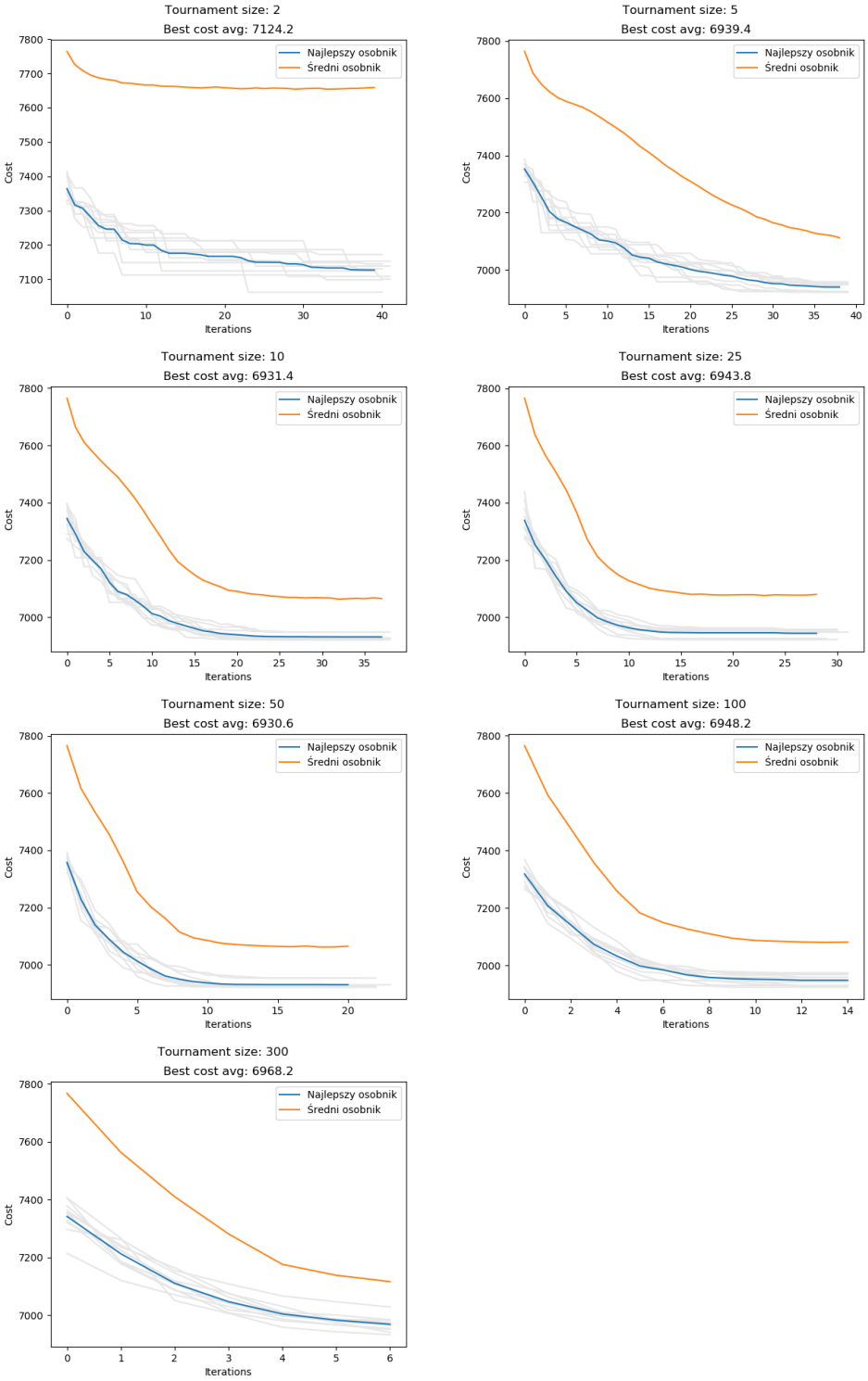
Podobnie jak w przypadku had12.dat, lepiej sprawdzają się duże populacje. Najlepszym wynikiem zakończył się test dla populacji 1500 osobników.

Prawdopodobieństwo mutacji



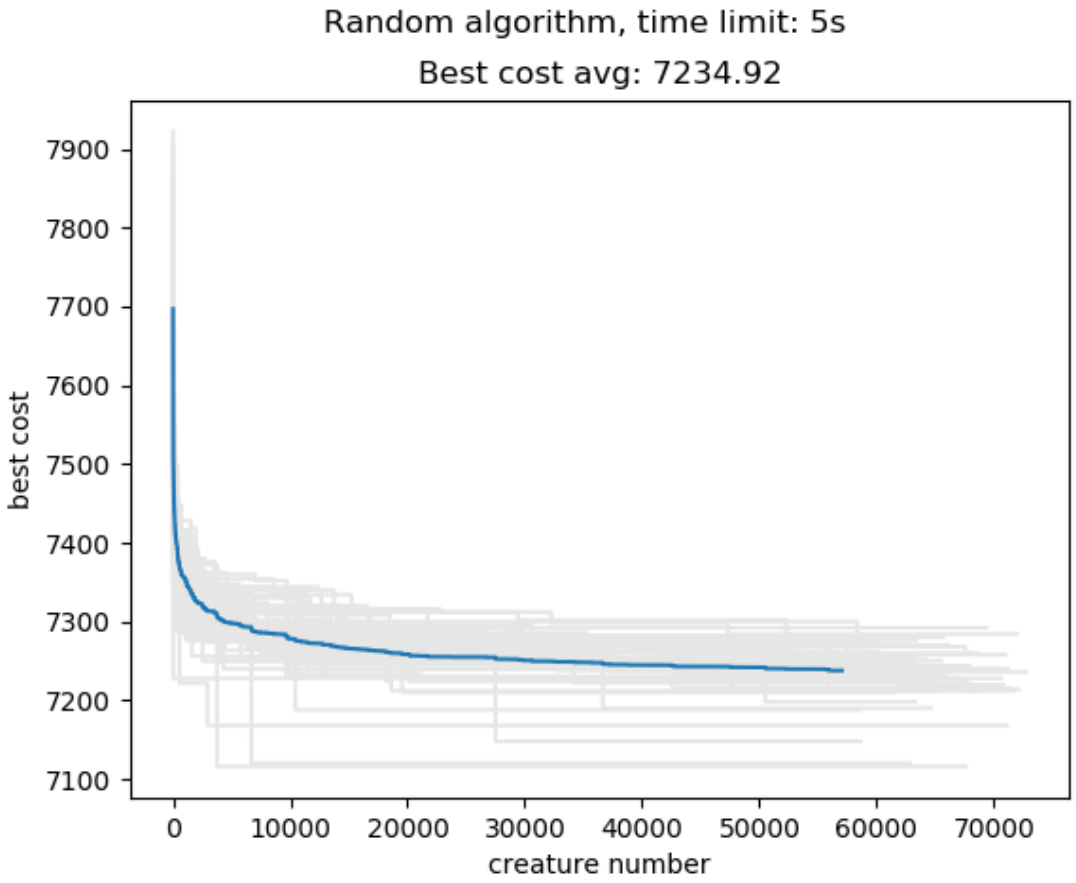
Ponownie najlepszym prawdopodobieństwem mutacji okazało się 5%.

Liczebność turnieju



Najlepsze wyniki uzyskano w zakresie 10-50 osobników, wynik dla 25 osobników jest prawdopodobnie zawyżony (10 uruchomień to dość mała próba).

Porównanie z innymi algorytmami



Średnia wartość osiągnięta w czasie 1s algorytmem losowym dla 50 uruchomień: 7235

Wartość osiągnięta algorytmem zachłannym: 7846

Wyniki dla algorytmu genetycznego są znacznie lepsze.

Wyniki dla pozostałych plików z danymi

TIME_LIMIT = 5 s
POPULATION = 1500
MUTATION_PROBABILITY = 0.05
CROSS_PROBABILITY = 1
SELECTION_METHOD = 'tournament'
MUTATION_METHOD = 'swap'
TOURNAMENT_SIZE = 50

Wartości optymalne dla porównania:

had12: 1652
had14: 2724
had16: 3720
had18: 5358
had20: 6922