Sztuczna inteligencja i inżynieria wiedzy

Algorytm genetyczny

Ewa Skórska, 212151

Poniedziałek 11.15

**1. Implementacja**

**1.1 Inicjalizacja**

Początkowa populacja zostaje zainicjalizowana losowo poprzez przypisanie do zadań dowolnego zasobu o kompetencjach spełniających wymagania zadania oraz ustawienie czasów początku i końca za pomocą algorytmu zachłannego.

**1.2 Przebieg algorytmu**

Po zapełnieniu generacji zerowej następuje wykonanie kolejnych kroków algorytmu:

0) wyłonienie najlepszego osobnika z całej populacji i przypisanie go do nowego pokolenia;

1) selekcja osobników prowadząca do wybrania dwóch najlepszych na podstawie metody turnieju;

2) wykonanie na nich krzyżowania;

3) wykonanie mutacji na uzyskanych z krzyżowania osobnikach

4) przypisanie uzyskanych w wyniku mutacji osobników do nowej populacji.

Krok 0 wykonywany jest tylko raz na początku każdego pokolenia, kolejne kroki (1-4) powtarzane są w pętli tak długo, aż nowe pokolenie będzie liczyć tyle samo osobników, co pokolenie rodziców.

1**.3 Krzyżowanie**

Operacja krzyżowania zachodzi pod warunkiem, że wylosowana na początku operacji liczba będzie mniejsza od zadanej wartości prawdopodobieństwa mutacji, w przeciwnym wypadku do nowego pokolenia zostaną przypisani rodzice w niezmienionej formie. Krzyżowanie w niniejszej implementacji polega na wyborze punktu krzyżowania i zmianie przypisania zasobów leżących dalej niż ten punkt pomiędzy dwoma osobnikami.

AAAAAAAAAA | AAAAA

BBBBBBBBBBB | BBBBB

+

AAAAAAAAAA | BBBBB

BBBBBBBBBBB | AAAAA

**1.4 Selekcja**

Wybrano metodę selekcji za pomocą turnieju. Przed krzyżowaniem następuje wylosowanie z populacji zadanej liczby osobników (nie dbając, by się nie powtarzały) a następnie porównanie ich wartości funkcji przystosowania (czasu trwania harmonogramu) i wyłonienie najlepszego, który zostaje przekazany do krzyżowania. Wielkość turnieju jest określana procentowo, dlatego przy każdej zmianie wielkości populacji obserwujemy wpływ jedynie tego parametru, nie musimy jeszcze uwzględniać wielkości turnieju.

**1.5 Mutacja**

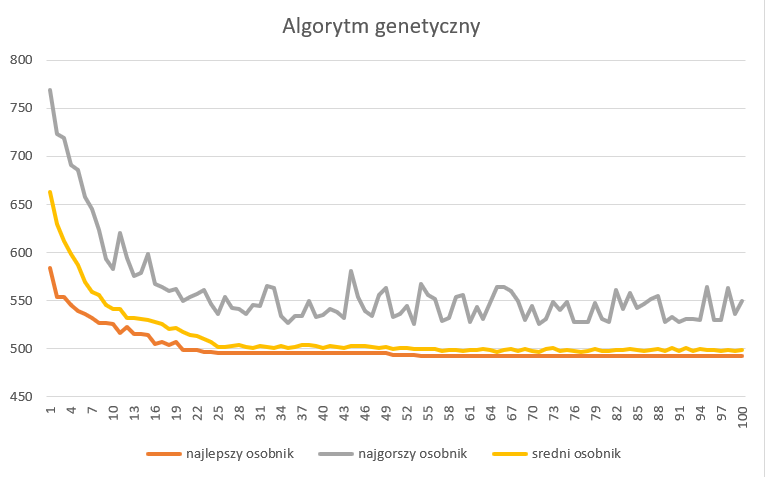
Mutacja przeprowadzana jest, tak jak krzyżowanie, pod warunkiem wylosowania liczby mniejszej od przyjętego prawdopodobieństwa mutacji. Dotyczy każdego zadania (taska) z osobna, czyli każde zadanie ma takie samo prawdopodobieństwo mutacji.

**2. Badania**

**2.1. Badanie na różnych plikach testowych**

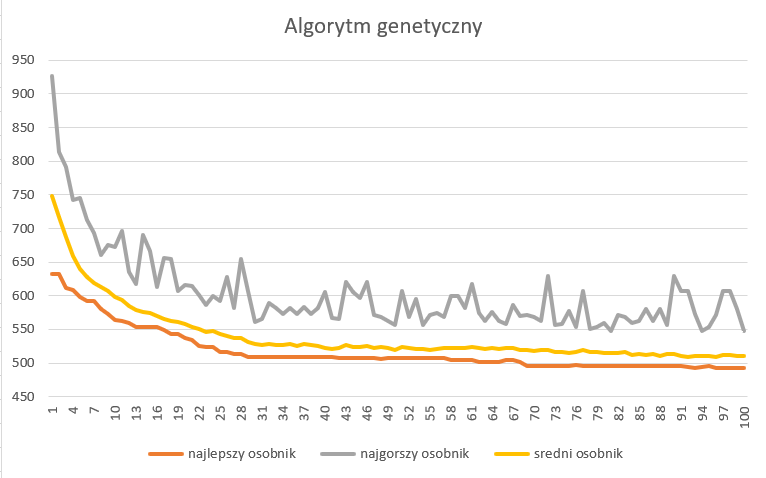
Badanie przeprowadzone zostało dla pięciu różnych plikach testowych przy zachowaniu tych samych parametrów: l. generacji – 100, wielkość populacji – 100, PX = 0.7, Pm = 0.01 oraz turniej wielkości 0.05. Otrzymywane najlepsze wyniki różniły się oczywiście, często o kilkaset jednostek, ponieważ każdy zbiór ma inny potencjał. Bardziej interesujący nas parametr, to różnica pomiędzy uruchomieniami programu dla tego samego zbioru. Największa (nawet 10 jednostek w każdą stronę) była dla zbiorów z większą liczbą tasków, co nie dziwi, ponieważ mamy wówczas więcej możliwości wyników. Należy pamiętać, że algorytm(metaheurystyka) genetyczny niekoniecznie doprowadza nas do optymalnego wyniku i opiera się na pseudolosowości.

1. 100\_5\_22\_15.def



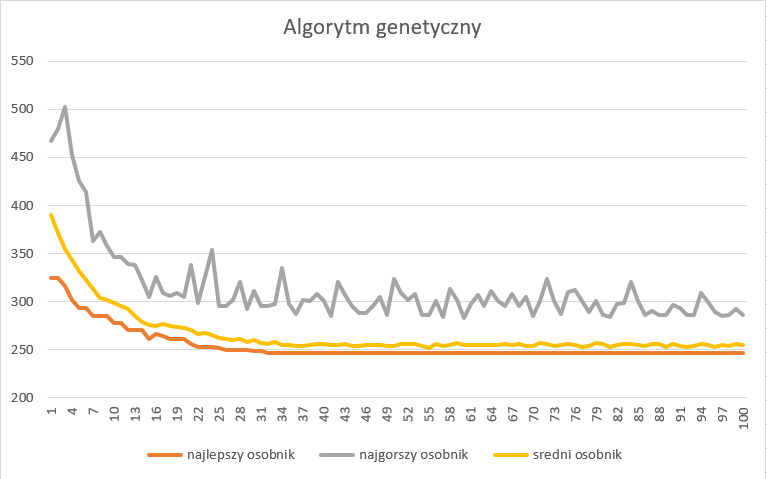
Najlepszy w ewolucji: 492.0

2. 200\_10\_50\_15.def



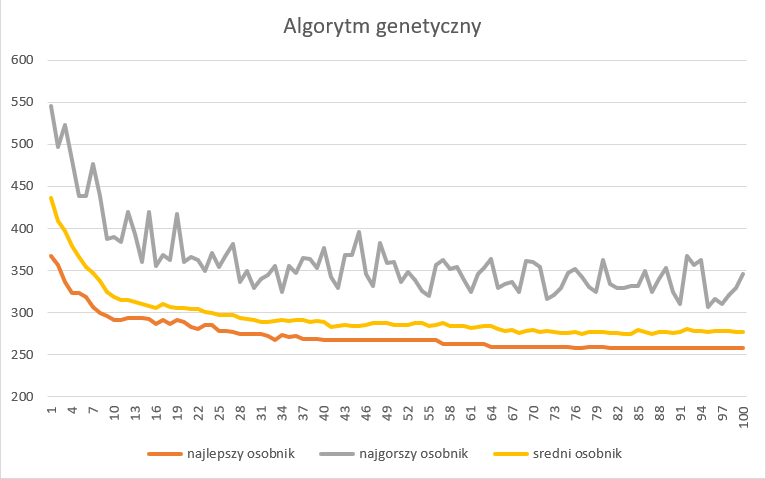
Najlepszy w ewolucji: 493.0

3. 100\_10\_26\_15.def



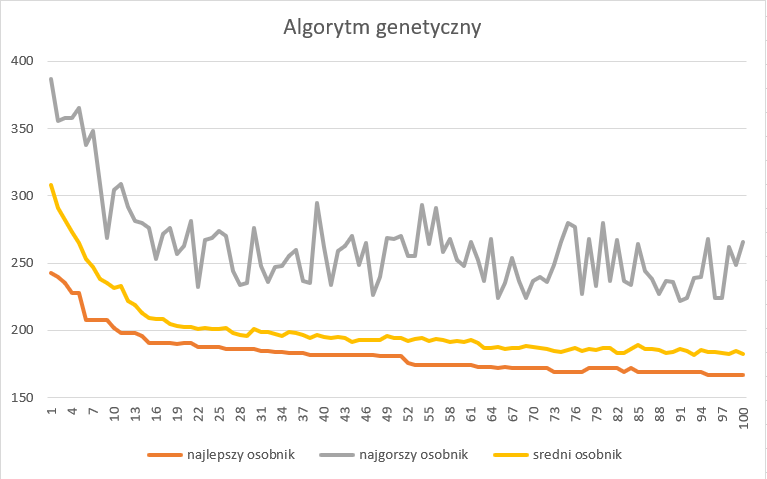
Najlepszy w ewolucji: 247.0

4. 200\_20\_97\_9.def



Najlepszy w ewolucji: 258.0

5. 200\_40\_133\_15.def



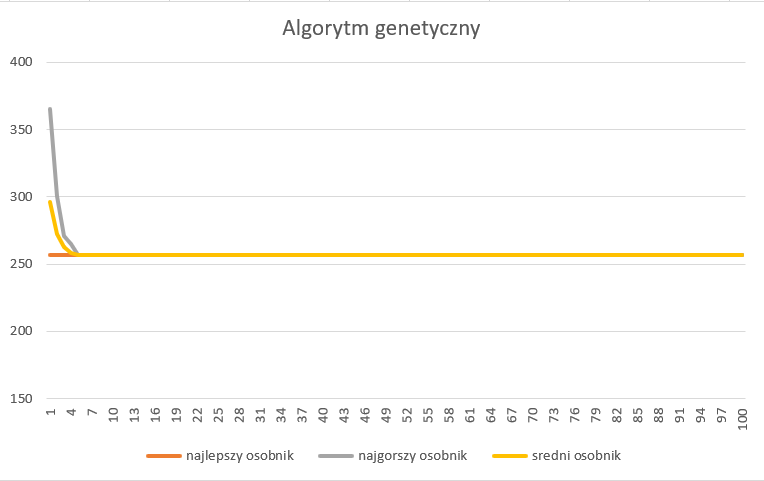
Najlepszy w ewolucji: 167.0

**2.2. Wpływ prawdopodobieństwa krzyżowania i mutacji na otrzymywane wyniki**

Od tego podpunktu badania będę przeprowadzać tylko na jednym zbiorze. Będzie to 200\_40\_133\_15.def, bo jako jeden z największych zbiorów może dawać różnorodne wyniki.

Px = 0.0

Pm = 0.0

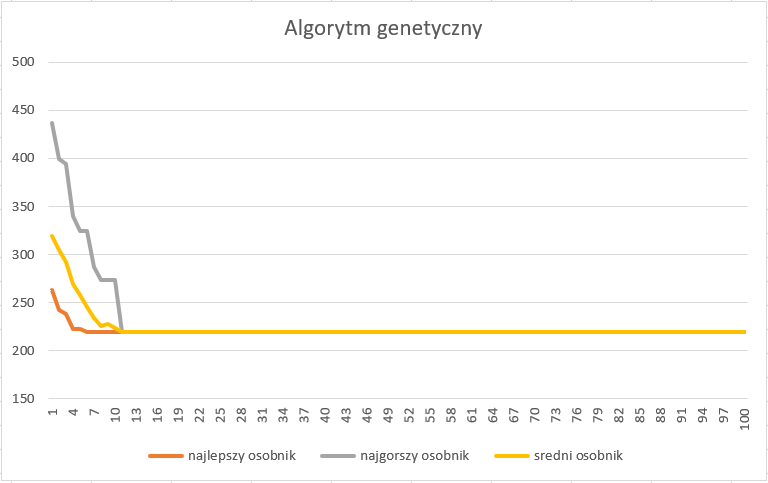


Najlepszy w ewolucji: 257.0

Jak można się było spodziewać brak krzyżowania i mutacji prowadzi do szybkiego ustalenia się relatywnie słabego wyniku, ponieważ poruszamy się jedynie pośród posiadanych od początku rozwiązań i wybieramy najlepsze.

Px = 1

Pm = 0

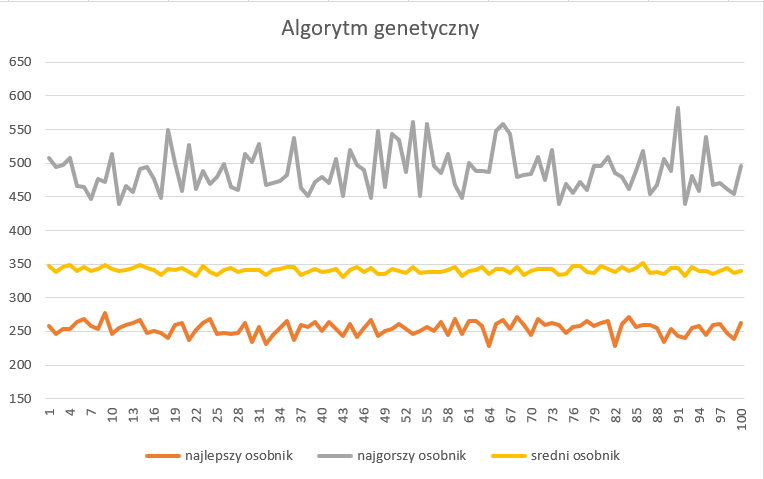


Najlepszy w ewolucji: 219.0

Samo krzyżowanie doprowadza nas do znacząco lepszego wyniku, jednak widzimy, że jego wpływ kończy się bardzo szybko, ponieważ zaczynamy obracać się wokół mieszanek już istniejących rozwiązań.

Px = 0

Pm = 1

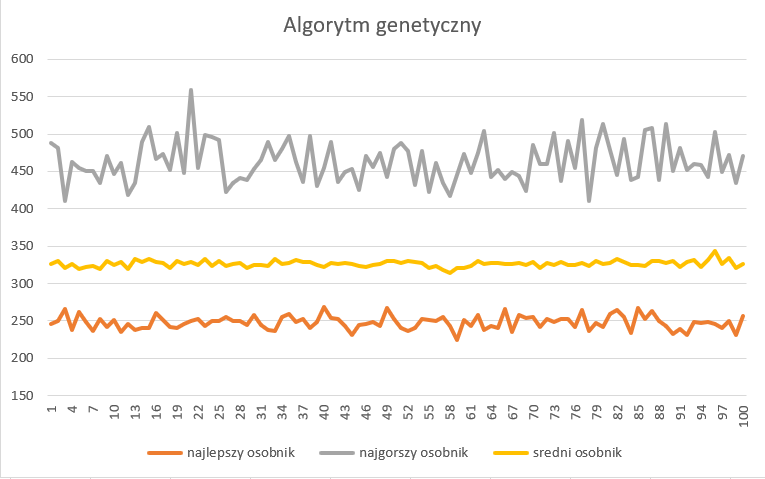


Najlepszy w ewolucji: 228.0

Bardzo interesujące wyniki daje mutacja ustawiona na 0, bez parametru krzyżowania. Widzimy, że brak tu jakiegokolwiek trendu, choć wcale nie oznacza to, że otrzymywane wyniki są gorsze niż w przypadku braku mutacji. W tym wypadku łatwo „wytracamy” raz zdobyte rozwiązanie, jak również łatwo zyskujemy kolejne. Pm = 1 to całkowita losowość w każdym osobniku każdego pokolenia.

Px = 0.5

Pm = 0.5

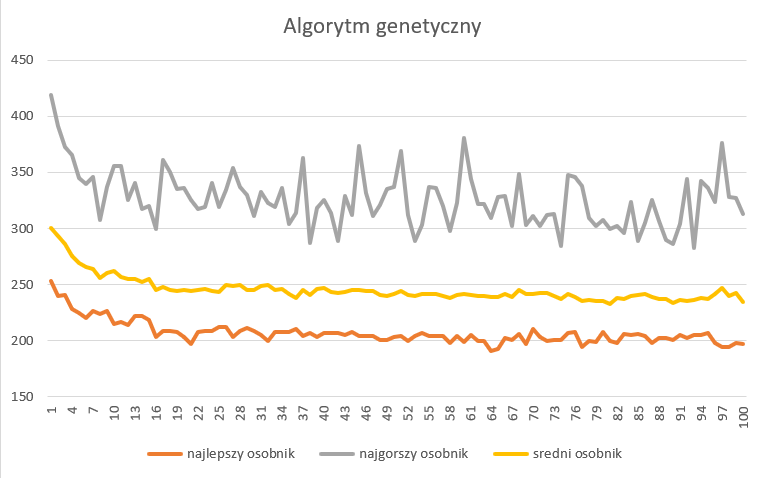


Najlepszy w ewolucji: 225.0

W wypadku Px=0.5 i Pm=0.5 wciąż największy wpływ ma Pm i, jak widać, wciąż jest zbyt duże.

Px = 0.5

Pm = 0.05

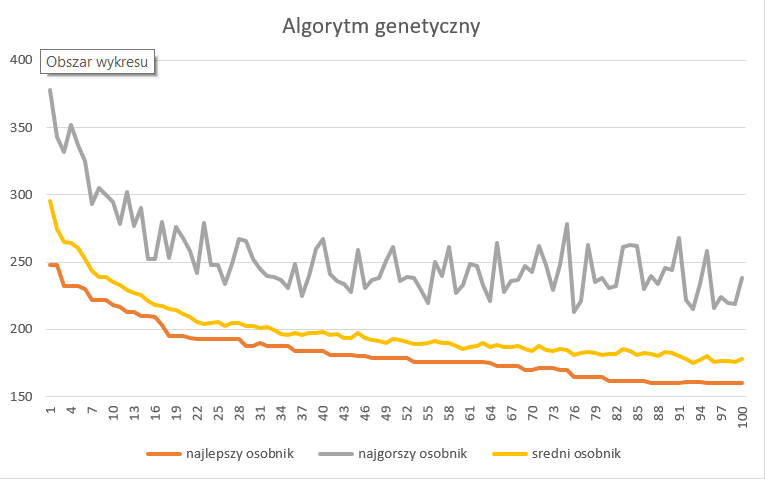


Najlepszy w ewolucji: 191.0

Przy dziesięciokrotnie zmniejszonym Pm zaczynamy obserwować trend spadkowy, znacząco również poprawia nam się wynik najlepszego osobnika.

Px = 0.5

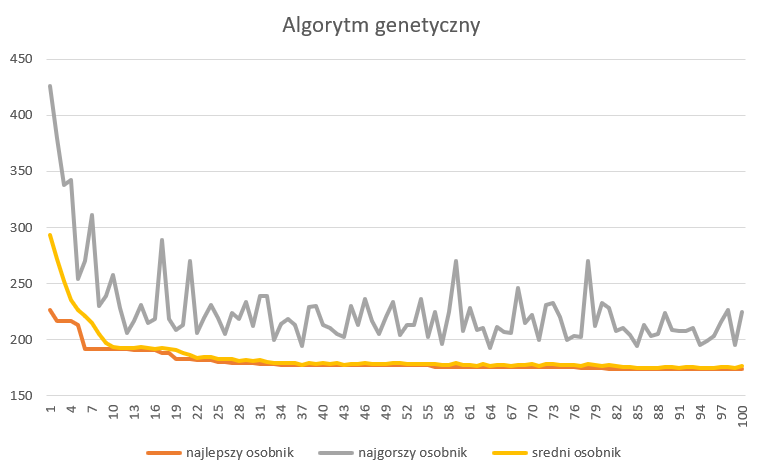
Pm = 0.01



Najlepszy w ewolucji: 160.0

Przy Pm=0.01 trend spadkowy jest już wyraźny, zyskujemy również o wiele lepsze rozwiązania.

Px=0.5 Pm=0.001



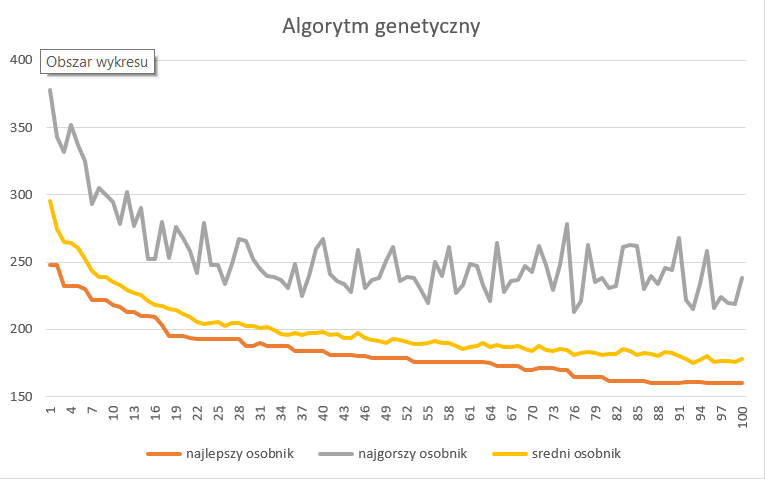
Najlepszy w ewolucji: 174.0

Dziesięciokrotnie mniejsze Pm jest już za małe i nie jesteśmy w stanie osiągnąć tak dobrych wyników jak wcześniej.

**2.2 Badanie wypływu wielkości pokolenia oraz liczby pokoleń**

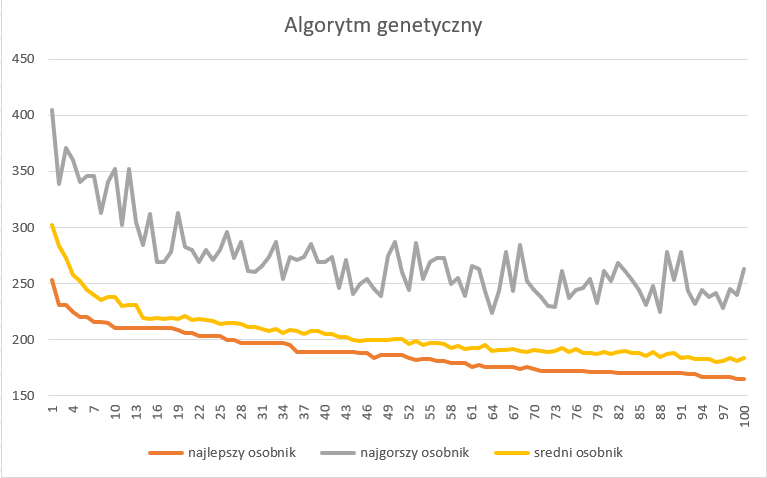
Oba z badanych parametrów powinny wpływać na wyniki i intuicyjnie wiemy, że im większy zbiór tym wyniki powinny być lepsze. Pomijam wielkości rzędu mniejszego niż .

l. pokoleń=100, wielkość populacji=100



Najlepszy w ewolucji: 160.0

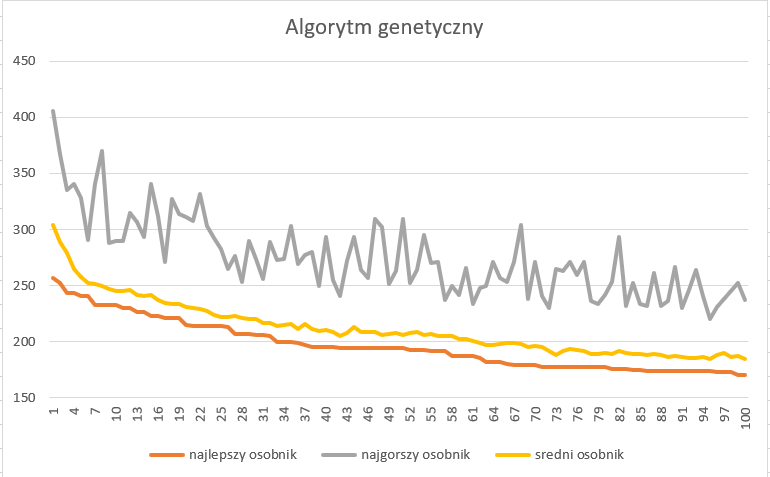
l. pokoleń=1000, wielkość populacji=100



Najlepszy w ewolucji: 146.0

Osiągane wyniki rzeczywiście są o wiele lepsze, jednak wykonanie programu wydłużyło się do ok. 16 sek.

l. pokoleń=10000, wielkość populacji=100

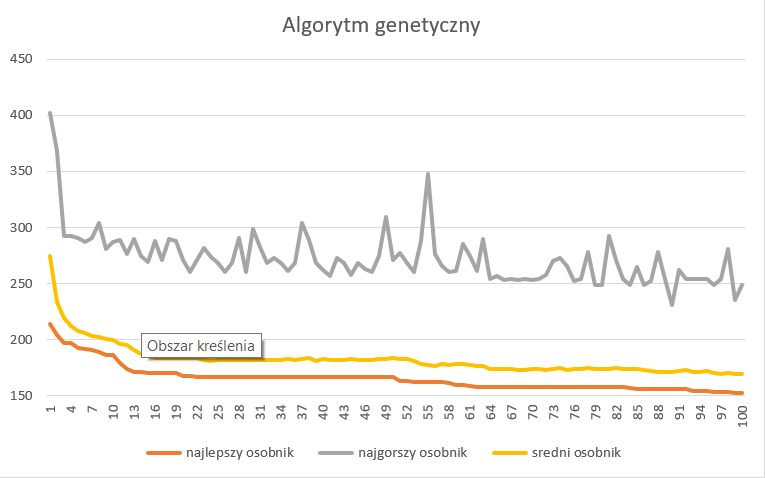


Najlepszy w ewolucji: 144.0

Wykonanie programu trwało około 2.5 minuty dlatego eksperyment powtórzony był tylko raz.

Osiągnięto nieznacznie lepszy wynik przy ogromnym nakładzie czasu, dlatego ten parametr musi pozostać taki jak poprzednio.

l. pokoleń=100, wielkość populacji=1000



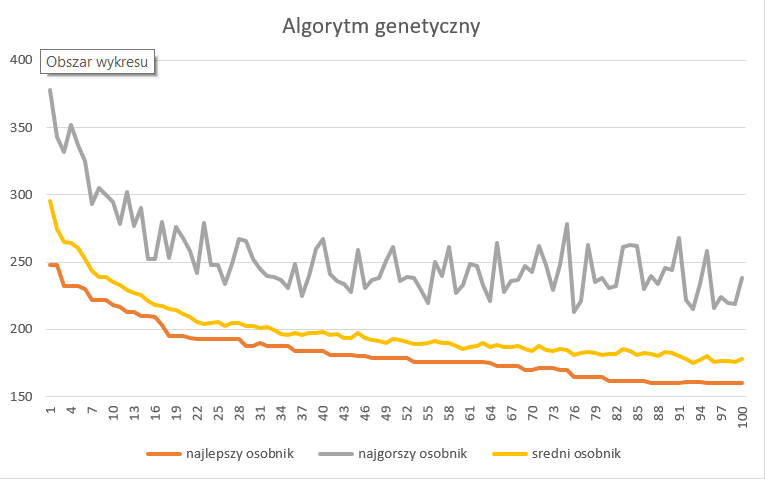
Najlepszy w ewolucji: 152.0

Najlepszy osobnik różni się znacząco od tego przy wielkości populacji=100, jednak jest to zaledwie 8 jednostek, przy prawie 50 razy dłuższym czasie wykonania programu, dlatego uważam, że ten parametr również musi zostać taki jak poprzednio.

**2.3. Badanie wpływu selekcji na skuteczność GA – turniej, ruletka**

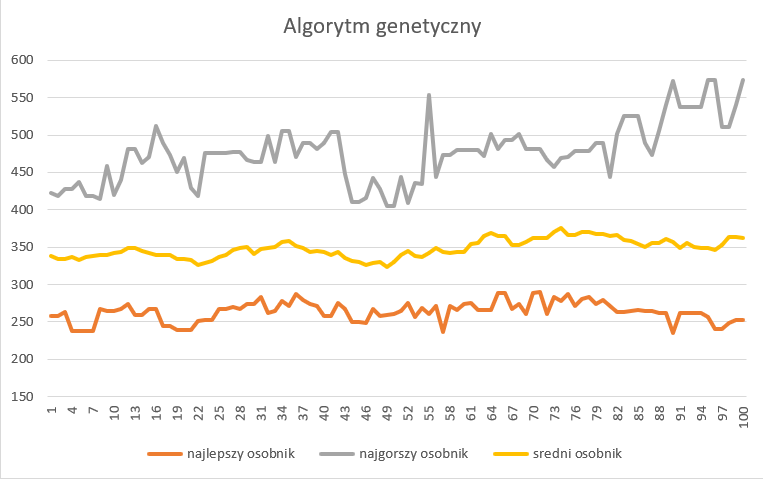
Turniej

Turniej = 0.05 (5%)



Najlepszy w ewolucji: 160.0

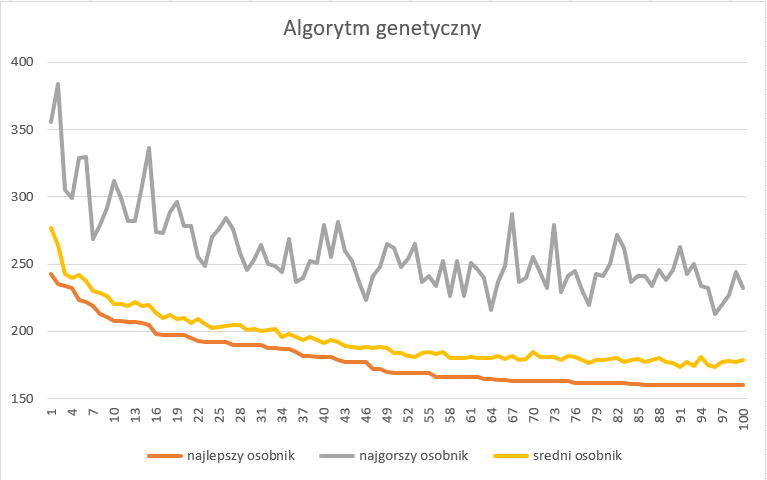
Turniej = 0.01 (1%)



Najlepszy w ewolucji: 235.0

W tym wypadku wybieramy zawsze losowego osobnika, toteż wyniki są zupełnie losowe.

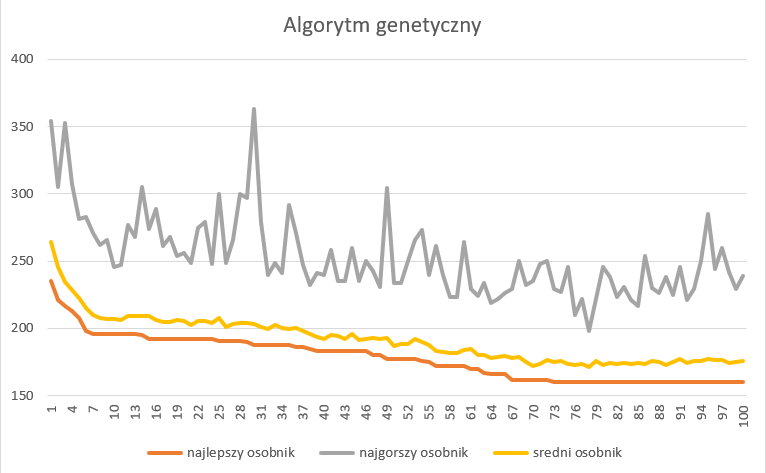
Turniej = 0.5 (50%)



Najlepszy w ewolucji: 160.0

Nie widać znaczącej różnicy między 5% a 50%. Optymalną wartość tego parametru ciężko wyznaczyć.

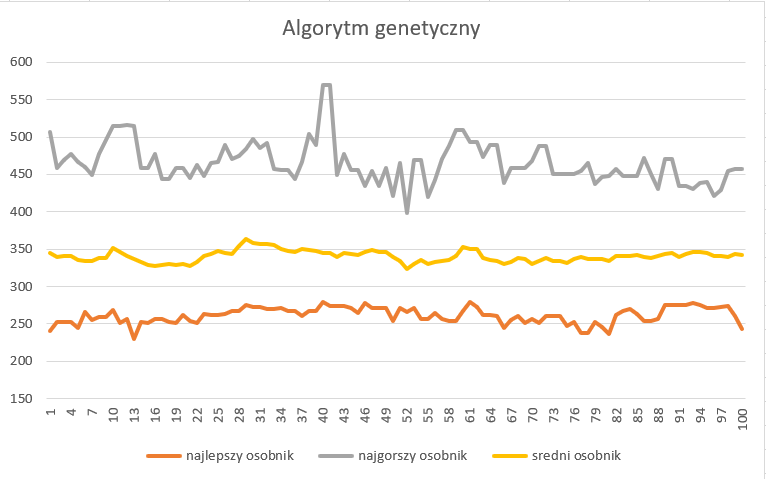
Turniej = 1 (100%)



Najlepszy w ewolucji: 160.0

Ruletka

Ruletka polega na przypisywaniu prawdopodobieństwa wyboru na podstawie funkcji przystosowania. Poniżej przedstawione wyniki zbliżone są do losowych, ponieważ dla tak niewiele różniących się osobników prawdopodobieństwo jest zbliżone i brak ciśnienia selekcyjnego.



Poniższa tabela została utworzona na podstawie 5 wywołań dla każdego zbioru danych.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | GA | | Greedy | |
| Nr zbioru danych | śr. | ϭ | śr. | ϭ |
| 1. 100\_5\_22\_15.def | 488.8 | 2.17 | 580.4 | 13.22 |
| 1. 200\_10\_50\_15.def | 497.0 | 4.36 | 657.8 | 17.41 |
| 1. 100\_10\_26\_15.def | 251.8 | 3.49 | 334.8 | 11.95 |
| 1. 200\_20\_97\_9.def | 256.2 | 3.56 | 382.4 | 9.91 |
| 1. 200\_40\_133\_15.def | 260.4 | 2.41 | 258 | 4.36 |