

Sprawozdanie: Algorytm Genetyczny Rozwiązujący Dyskretny Problem Plecakowy

Autorzy: Jakub Leśkiewicz 52733, Antoni Krawczyk 52717

Data: 16 listopada 2025

1. Cel Pracy

Celem zadania było zaimplementowanie w języku Python algorytmu genetycznego (AG) rozwiązującego dyskretny problem plecakowy (ang. 0/1 Knapsack Problem). Implementacja została wykonana z podejściem obiektowym, bez użycia gotowych bibliotek do algorytmów ewolucyjnych, zgodnie ze specyfikacją projektu.

Program realizuje wszystkie funkcjonalności wymagane do oceny **3.5** (porównanie współczynników mutacji i krzyżowania) oraz **4.5** (porównanie metod selekcji i krzyżowania).

2. Opis Problemu

Dyskretny problem plecakowy polega na wyborze takiego podzbioru przedmiotów, aby suma ich wartości była jak największa, a suma ich wag nie przekraczała maksymalnej pojemności plecaka. Każdy przedmiot może być wybrany tylko raz (w całości albo wcale). Jest to problem NP-trudny, co sprawia, że dla dużych zbiorów danych znalezienie optymalnego rozwiązania w rozsądnym czasie jest niemożliwe przy użyciu metod siłowych. Z tego względu algorytmy heurystyczne, takie jak algorytmy genetyczne, są efektywnym sposobem na znalezienie rozwiązań bliskich optimum.

3. Opis Implementacji

Projekt został zrealizowany w sposób obiektowy i podzielony na dwa główne pliki:

- `go_knapsack.py`: Zawiera logikę algorytmu genetycznego. Definiuje klasy `Item` (przedmiot), `Individual` (osobnik/rozwiązanie) oraz `GeneticAlgorithm`, która zarządza całym procesem ewolucji (wczytywanie danych, selekcja, krzyżowanie, mutacja).
- `main.py`: Główny skrypt uruchomieniowy. Odpowiada za interakcję z użytkownikiem (wybór pliku, pobranie parametrów), automatyczne wczytanie wartości optymalnej oraz przeprowadzenie zdefiniowanych eksperymentów i wygenerowanie wykresów.

3.1. Kodowanie Informacji (Genotyp)

Zgodnie ze specyfikacją, jako genotyp osobnika (klasa `Individual`) zastosowano chromosom binarny. Jest to lista (tablica) o długości równej liczbie dostępnych przedmiotów. Wartość 1 na pozycji *i* oznacza, że *i*-ty przedmiot znajduje się w plecaku, natomiast wartość 0 oznacza jego brak.

3.2. Funkcja Przystosowania (Fitness)

Funkcja przystosowania (`calculate_fitness` w klasie `Individual`) jest kluczowym elementem algorytmu. W specyfikacji zasugerowano, aby funkcja zwracała 0 dla rozwiązań niepoprawnych (przeładowanych). Jednak taka implementacja okazała się nieskuteczna dla problemów `large_scale`, gdzie cała populacja początkowa składała się z niepoprawnych osobników, co prowadziło do zerowej presji selekcyjnej.

Wprowadzono modyfikację, która pozwala algorytmowi "uczyć się", jak zmniejszać wagę:

- **Dla osobników poprawnych** (waga \leq pojemność): Fitness jest równy sumie wartości przedmiotów. `fitness = total_value`.
- **Dla osobników niepoprawnych** (waga $>$ pojemność): Fitness otrzymuje karę. Jest to wartość ułamkowa (zawsze mniejsza od 1) obliczana jako `fitness = capacity / total_weight`. Dzięki temu osobnik, który przekracza pojemność nieznacznie, ma wyższy (lepsy) fitness niż osobnik, który przekracza ją rażąco, co nadaje ewolucji odpowiedni kierunek.

3.3. Metody Selekcji

Zaimplementowano dwie metody selekcji, które tworzą pulę rodziców (`parents_pool`) do krzyżowania:

- **Selekcja Kołem Ruletki (`selection_roulette_wheel`):** Prawdopodobieństwo wyboru osobnika jest wprost proporcjonalne do jego wartości fitness.
- **Selekcja Rankingowa (`selection_rank`):** Prawdopodobieństwo wyboru zależy od pozycji w rankingu, co zapewnia stabilniejszą presję selekcyjną.

3.4. Operatory Genetyczne

Zaimplementowano dwa operatory krzyżowania oraz operator mutacji.

- **Krzyżowanie Jednopunktowe (`crossover_one_point`):** Losowany jest jeden punkt cięcia, a materiał genetyczny za nim jest wymieniany.
- **Krzyżowanie Dwupunktowe (`crossover_two_point`):** Losowane są dwa punkty, a materiał *pomiędzy* nimi jest wymieniany.
- **Mutacja (`mutate`):** Każdy bit w genotypie ma `mutation_prob` szansy na "odwrócenie" (0 -> 1 lub 1 -> 0).

4. Część Eksperymentalna i Wyniki

Eksperymenty polegały na uruchomieniu algorytmu z różnymi konfiguracjami dla 6 wybranych zbiorów danych (4 `low-dimensional` i 2 `large_scale`). Dla każdego zbioru przeprowadzono eksperymenty badające wpływ współczynników (wym. 3.5) oraz metod (wym. 4.5).

Analiza 1: Zbiór `low-dimensional/f1_l-d_kp_10_269`

Zbiór ten opisuje problem z 10 przedmiotami. Optimum wynosi **295**.

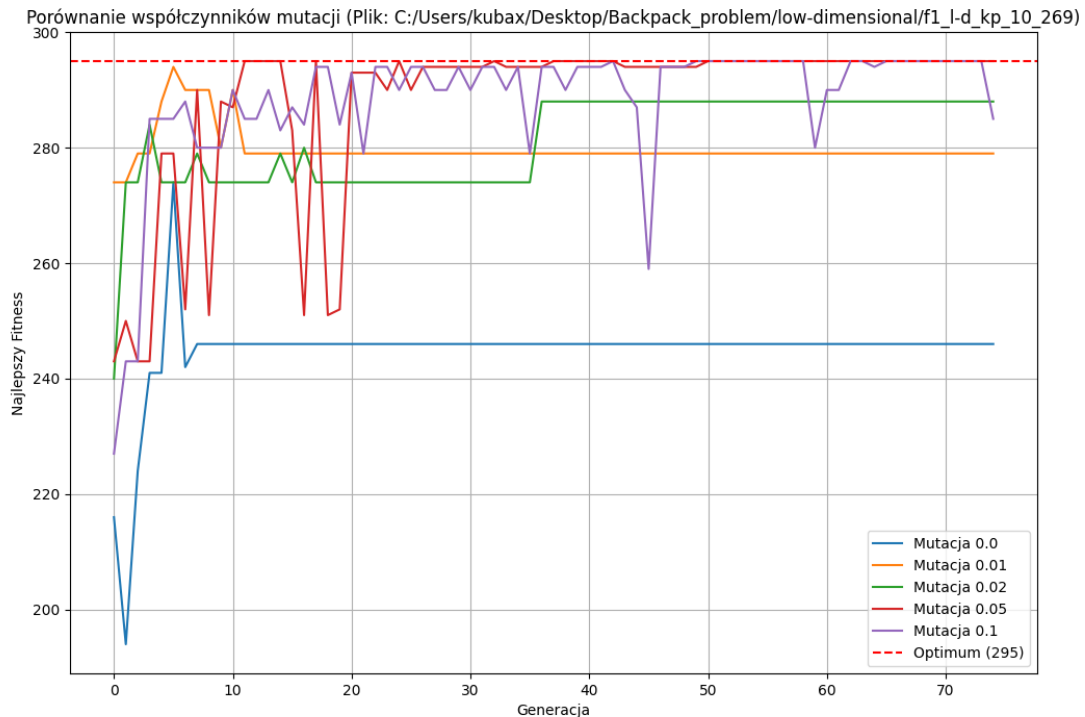
- **Parametry Bazowe:** Populacja: 30, Iteracje: 75, P. Mutacji: 0.03, P. Krzyżowania: 0.9.

4.1. Porównanie Współczynników (Wymaganie 3.5)

Eksperymenty przeprowadzono na bazie Selekcji Rankingowej.

4.1.1. Porównanie współczynników mutacji (Wykres 1)

(Baza: P. Krzyżowania = 0.9)



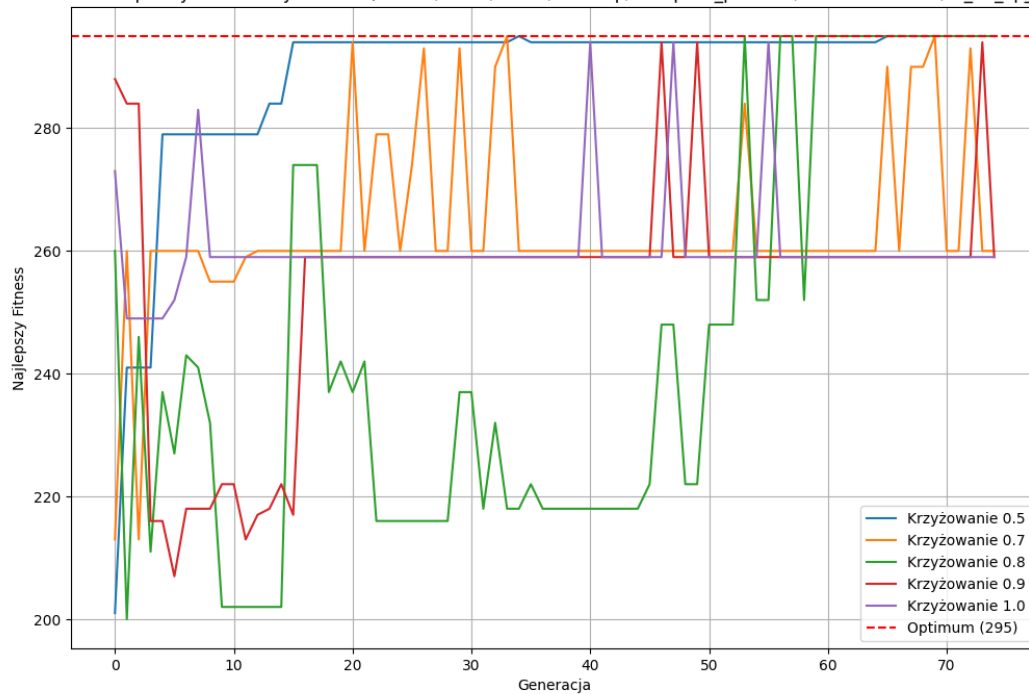
Wnioski:

- Wartość mutacji ma kluczowy wpływ na wynik.
- **Najlepszy wynik (globalne optimum 295)** uzyskano dla P. Mutacji = **0.05**.
- Zbyt niska wartość (P=0.0) spowodowała utknięcie w optimum lokalnym (wynik **246**).
- Wartości P=0.01 i P=0.02 również utknęły, osiągając odpowiednio 279 i 288.
- Zbyt wysoka wartość (P=0.1) wprowadziła niestabilność, osiągając 294, ale spadając do 280 pod koniec.

4.1.2. Porównanie współczynników krzyżowania (Wykres 2)

(Baza: P. Mutacji = 0.03)

Porównanie współczynników krzyżowania (Plik: C:/Users/kubax/Desktop/Backpack_problem/low-dimensional/f1_l-d_kp_10_269)



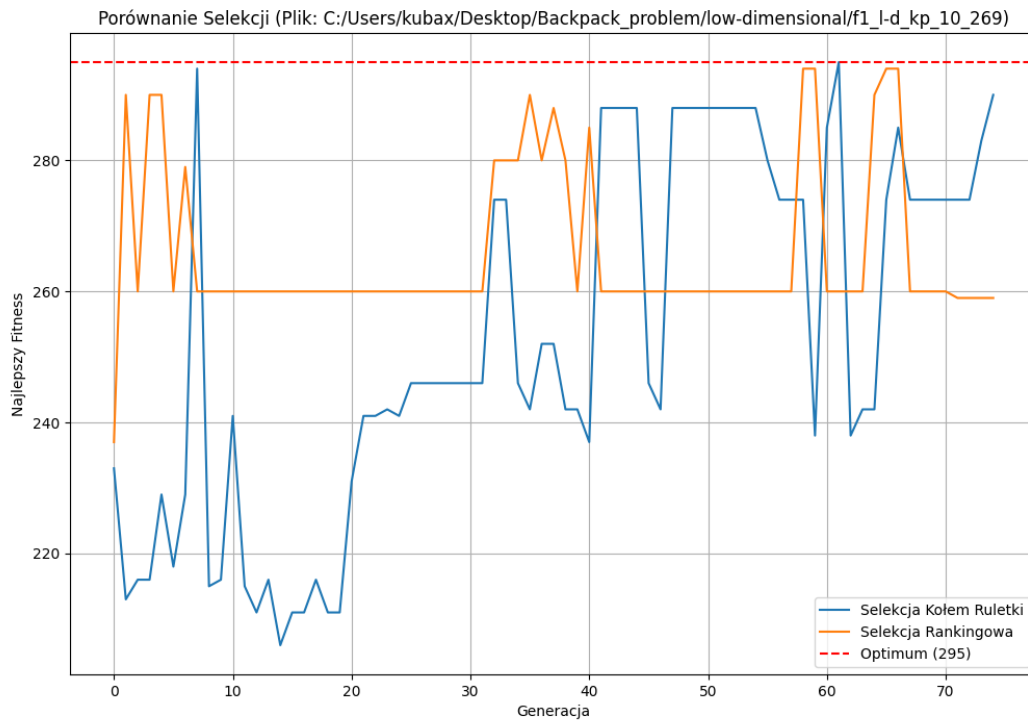
Wnioski:

- Prawdopodobieństwo krzyżowania okazało się kluczowe.
- **Globalne optimum 295** zostało osiągnięte dla P. Krzyżowania = **0.8**.
- Wartość $P=0.5$ również była blisko (wynik **294**).
- Wartości $P=0.7$, $P=0.9$ oraz $P=1.0$ okazały się nieskuteczne, zatrzymując się na znacznie niższych optimach lokalnych (odpowiednio 260, 259, 259).

4.2. Porównanie Metod (Wymaganie 4.5)

Eksperymenty przeprowadzono na parametrach bazowych.

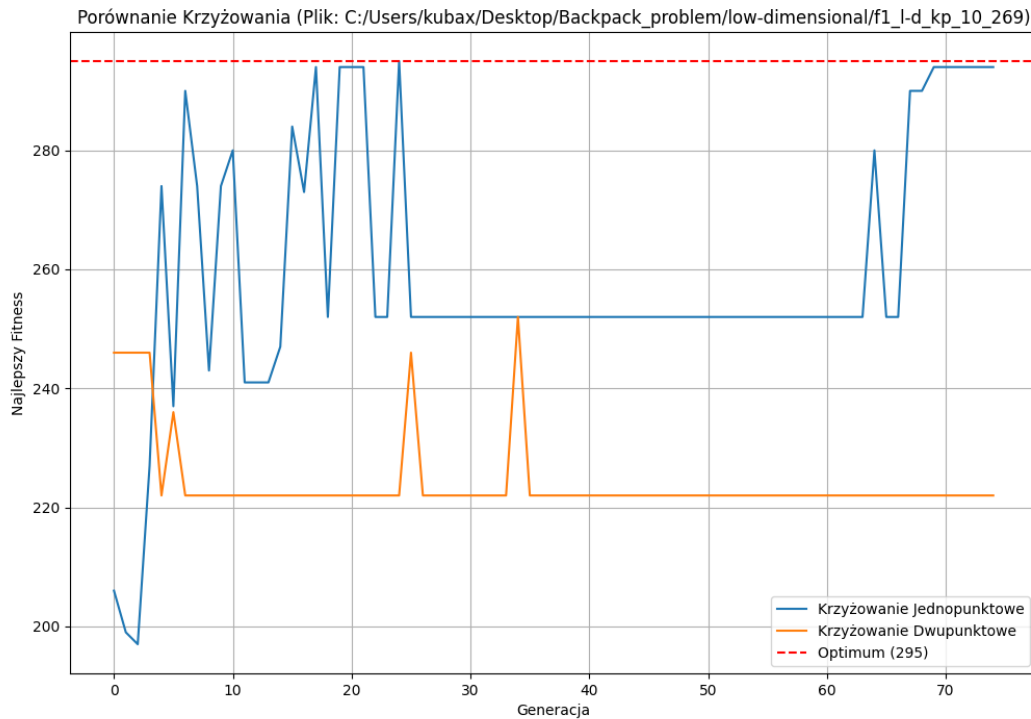
4.2.1. Porównanie Metod Selekcji (Wykres 3)



Wnioski:

- Selekcja Kołem Ruletki** (niebieska linia) okazała się wysoce nieskuteczna, osiągając maksymalnie wartość 242.
- Selekcja Rankingowa** (pomarańczowa linia) była znacznie lepsza, osiągając pod koniec działania wartość **294**, czyli była o krok od globalnego optimum.

4.2.2. Porównanie Metod Krzyżowania (Wykres 4)



Wnioski:

- Oba eksperymenty (krzyżowanie jedno- i dwupunktowe) były prowadzone przy użyciu Selekcji Rankingowej.
- **Krzyżowanie Jednopunktowe** (niebieska linia) osiągnęło dobry wynik **294**, choć był on niestabilny (spadł do 252).
- **Krzyżowanie Dwupunktowe** (pomarańczowa linia) okazało się w tym przebiegu **nie skuteczne**, szybko utknęło w optimum lokalnym (wartość **222**).

Analiza 2: Zbiór low-dimensional/f2_l-d_kp_20_878

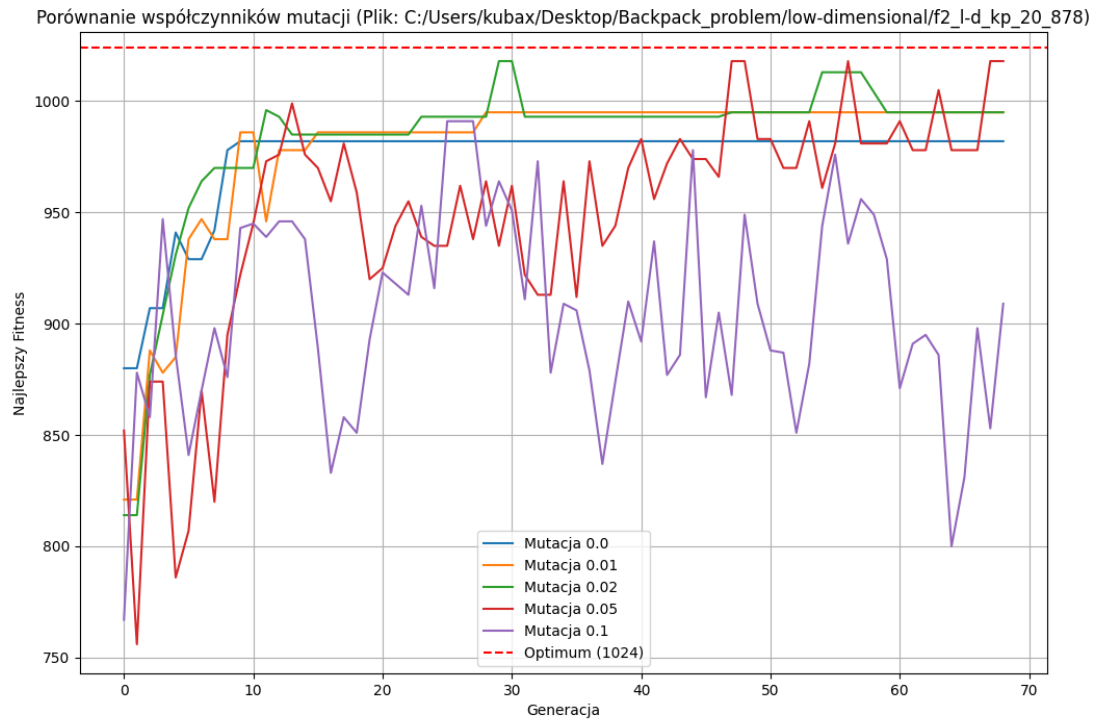
Zbiór ten opisuje problem z 20 przedmiotami. Optimum wynosi **1024**.

- **Parametry Bazowe:** Populacja: 40, Iteracje: 69, P. Mutacji: 0.03, P. Krzyżowania: 0.85.

4.3. Porównanie Współczynników (Wymaganie 3.5)

4.3.1. Porównanie współczynników mutacji (Wykres 5)

(Baza: P. Krzyżowania = 0.85)



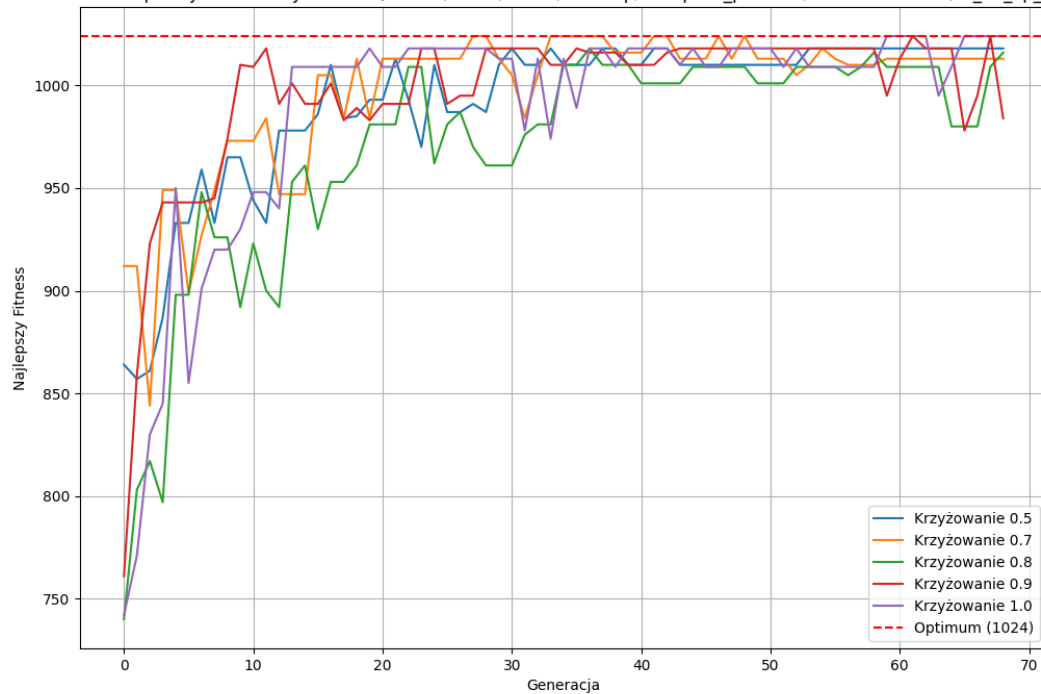
Wnioski:

- W tym teście **im niższa mutacja, tym lepszy wynik**.
- Najlepsze rezultaty dały wartości $P=0.01$ i $P=0.02$ (wynik **995**). Mutacja $P=0.0$ była tuż za nimi (wynik 982).
- Wartość bazowa ($P=0.03$) dała wynik 983 (nie pokazany na wykresie, ale użyty jako baza dla testu krzyżowania).
- Wyraźnie widać, że wyższe wartości mutacji ($P=0.05$ i $P=0.1$) **pogarszają wynik** (odpowiednio 981 i 929), wprowadzając zbyt duży chaos i niszcząc dobre rozwiązania.

4.3.2. Porównanie współczynników krzyżowania (Wykres 6)

(Baza: P. Mutacji = 0.03)

Porównanie współczynników krzyżowania (Plik: C:/Users/kubax/Desktop/Backpack_problem/low-dimensional/f2_l-d_kp_20_878)

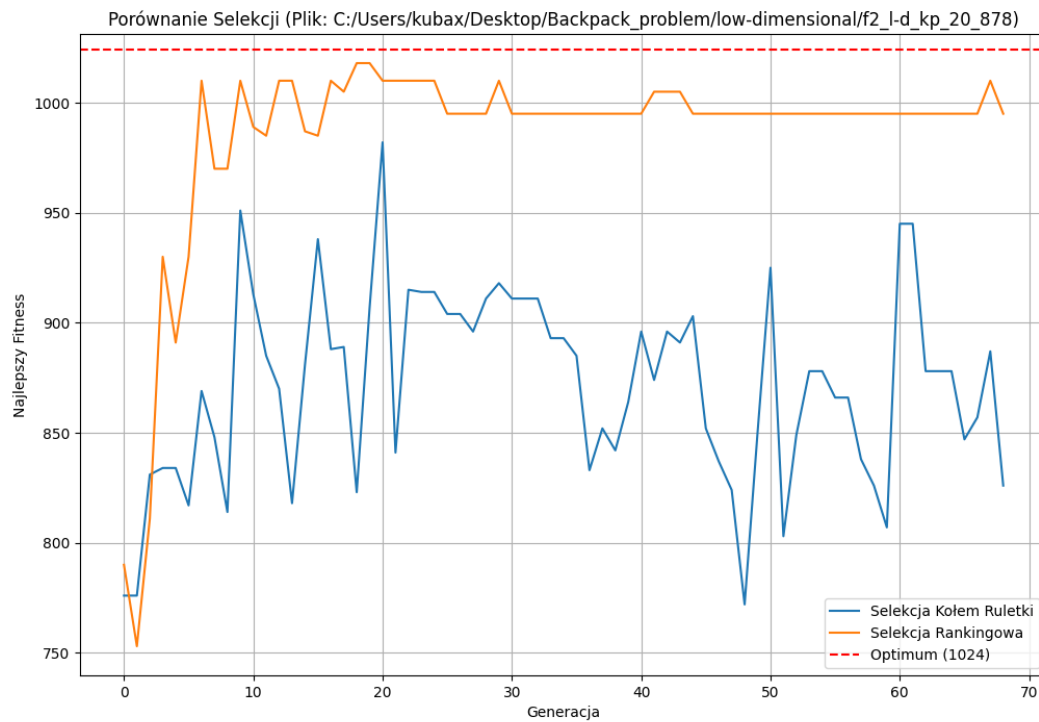


Wnioski:

- **Globalne optimum 1024** zostało osiągnięte tylko dla **P. Krzyżowania = 1.0**.
- Bardzo bliskie wyniki (1018) uzyskano dla P=0.5.
- Wszystkie pozostałe wartości (0.7, 0.8, 0.9) dały bardzo dobre, ale nieoptymalne wyniki (między 1009 a 1016). Pokazuje to, że dla tego problemu wysoka skłonność do krzyżowania była korzystna.

4.4. Porównanie Metod (Wymaganie 4.5)

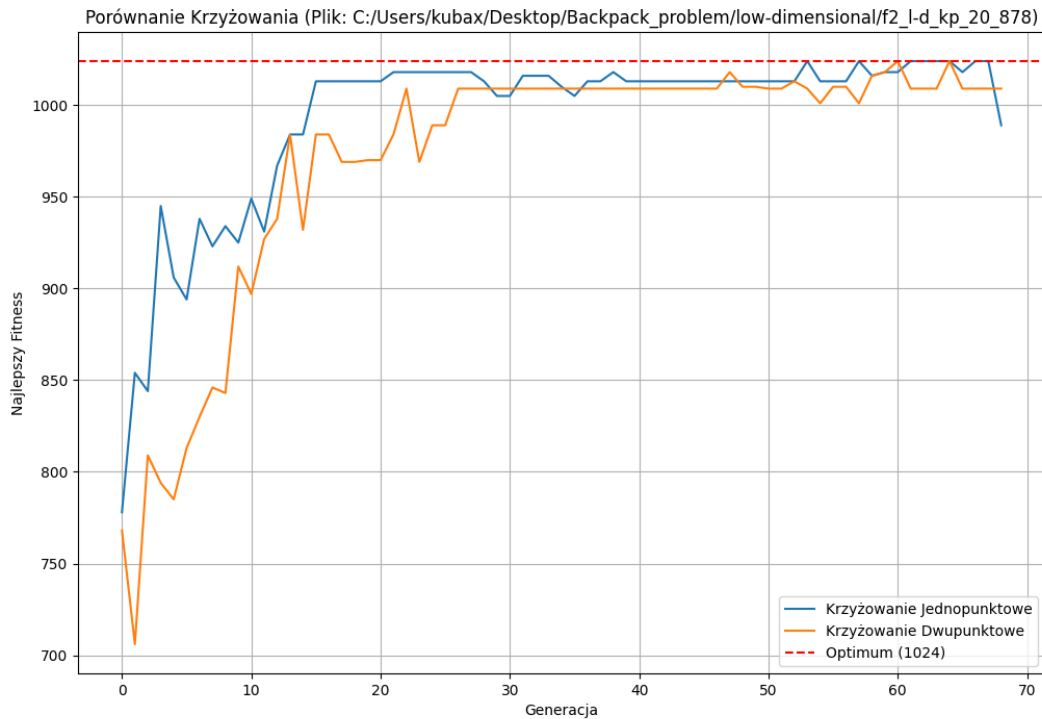
4.4.1. Porównanie Metod Selekcji (Wykres 7)



Wnioski:

- Selekcja Kołem Ruletki** (niebieska linia) okazała się niestabilna i nieskuteczna, osiągając pod koniec wartość 807.
- Selekcja Rankingowa** (pomarańczowa linia) była znacznie lepsza, osiągając **1018** (bardzo blisko optimum 1024), choć jej przebieg również był niestabilny.

4.4.2. Porównanie Metod Krzyżowania (Wykres 8)



Wnioski:

- Oba testy (na Selekcji Rankingowej) dały identyczny, doskonały wynik **1018**.
- W tym przypadku, dla bazowych parametrów, obie metody krzyżowania okazały się równie skuteczne.

Analiza 3: Zbiór low-dimensional/f3_l-d_kp_4_20

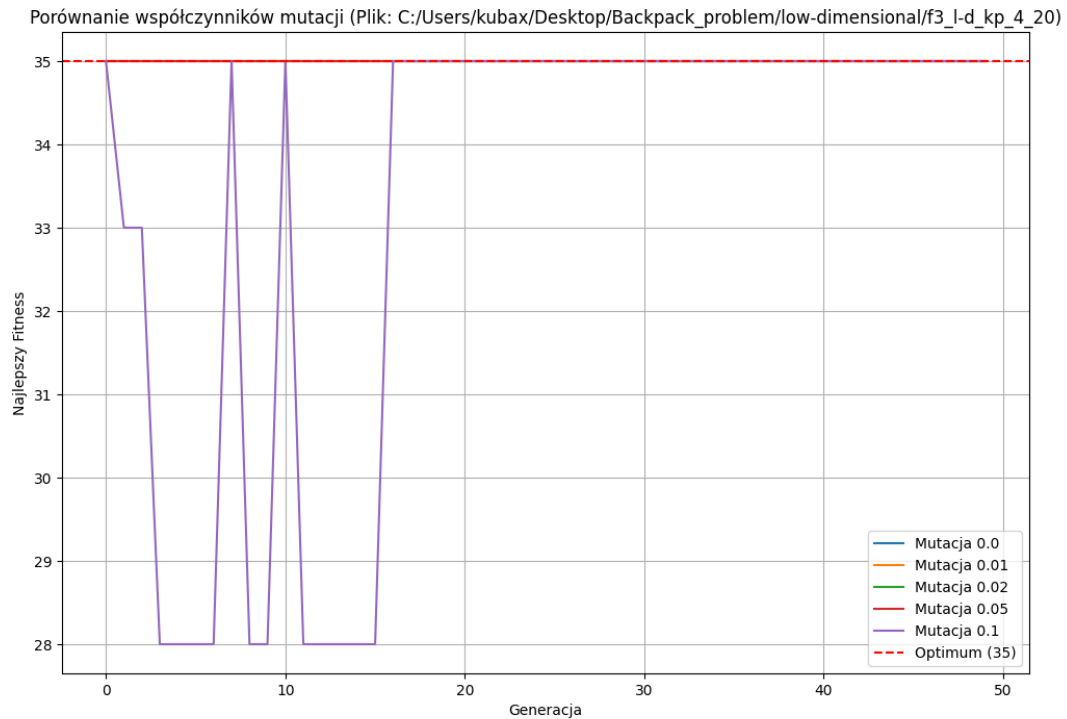
Jest to problem o bardzo małej skali (4 przedmioty). Optimum wynosi **35**.

- **Parametry Bazowe:** Populacja: 20, Iteracje: 50, P. Mutacji: 0.045, P. Krzyżowania: 0.81.

4.5. Porównanie Współczynników (Wymaganie 3.5)

4.5.1. Porównanie współczynników mutacji (Wykres 9)

(Baza: P. Krzyżowania = 0.81)



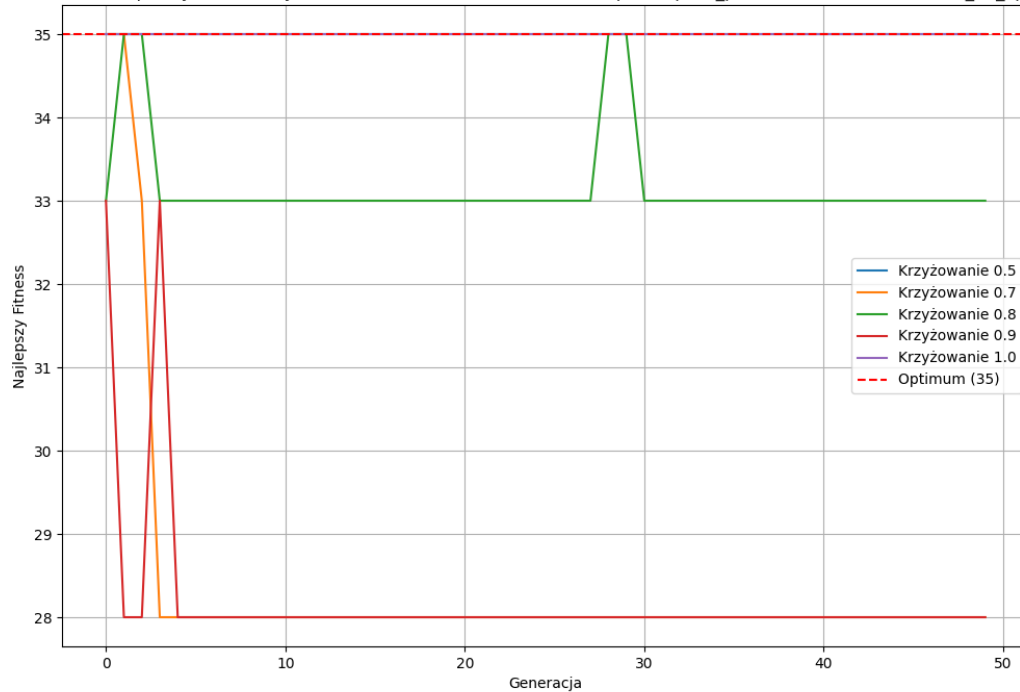
Wnioski:

- Z powodu trywialnej złożoności problemu (tylko 16 możliwych rozwiązań), **każda testowana wartość mutacji (0.0, 0.01, 0.02, 0.05, 0.1) doprowadziła do znalezienia globalnego optimum 35.**

4.5.2. Porównanie współczynników krzyżowania (Wykres 10)

(Baza: P. Mutacji = 0.045)

Porównanie współczynników krzyżowania (Plik: C:/Users/kubax/Desktop/Backpack_problem/low-dimensional/f3_l-d_kp_4_20)

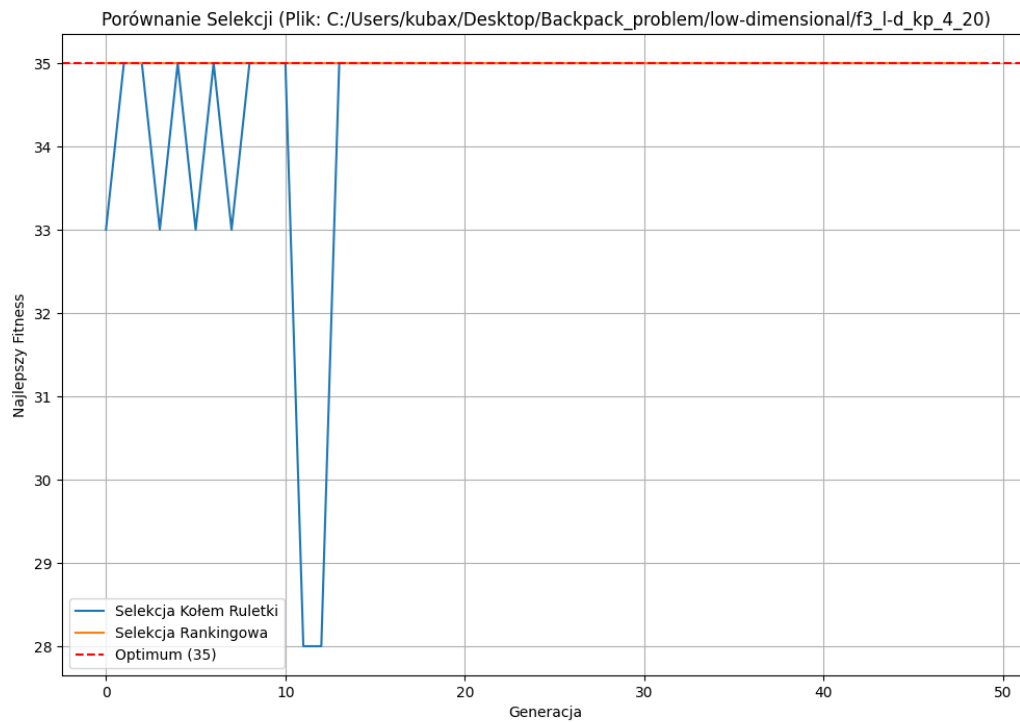


Wnioski:

- W tym teście wyniki były bardziej zróżnicowane.
- **Optimum 35** zostało znalezione dla $P=0.5$ oraz $P=1.0$.
- Wartości $P=0.7$ i $P=0.9$ utknęły w optimum lokalnym (wynik **28**), a $P=0.8$ w optimum lokalnym **33**. Pokazuje to, że nawet w prostych problemach dobór parametrów ma znaczenie.

4.6. Porównanie Metod (Wymaganie 4.5)

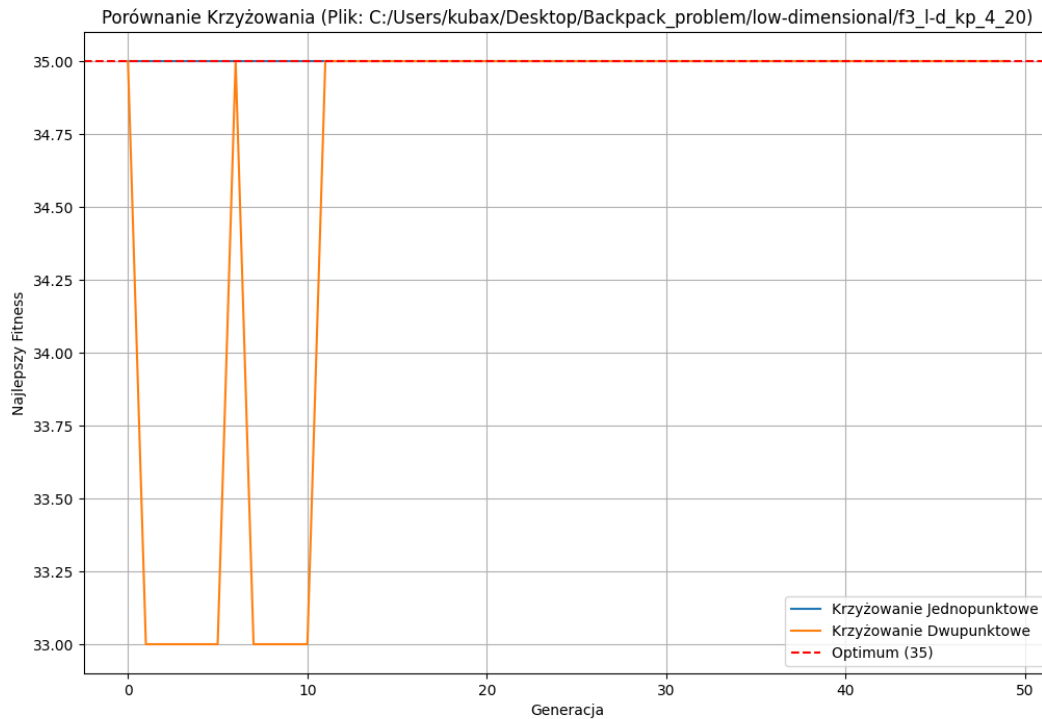
4.6.1. Porównanie Metod Selekcji (Wykres 11)



Wnioski:

- Obie metody, **Selekcja Kołem Ruletki** i **Selekcja Rankingowa**, bez problemu **znalazły globalne optimum 35** i utrzymały je do końca.

4.6.2. Porównanie Metod Krzyżowania (Wykres 12)



Wnioski:

- Obie metody, **Krzyżowanie Jednopunktowe** i **Krzyżowanie Dwupunktowe**, również znalazły globalne optimum 35.

Analiza 4: Zbiór low-dimensional/f4_l-d_kp_4_11

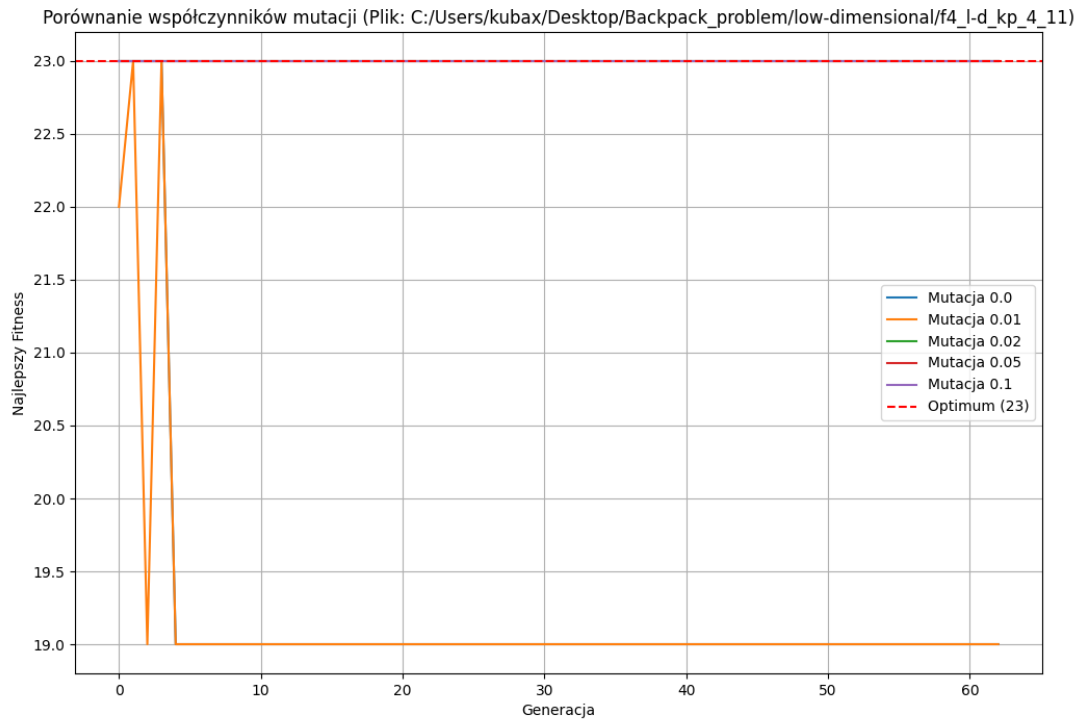
Jest to kolejny problem o minimalnej skali (4 przedmioty). Optimum wynosi **23**.

- **Parametry Bazowe:** Populacja: 47, Iteracje: 63, P. Mutacji: 0.025, P. Krzyżowania: 0.83.

4.7. Porównanie Współczynników (Wymaganie 3.5)

4.7.1. Porównanie współczynników mutacji (Wykres 13)

(Baza: P. Krzyżowania = 0.83)

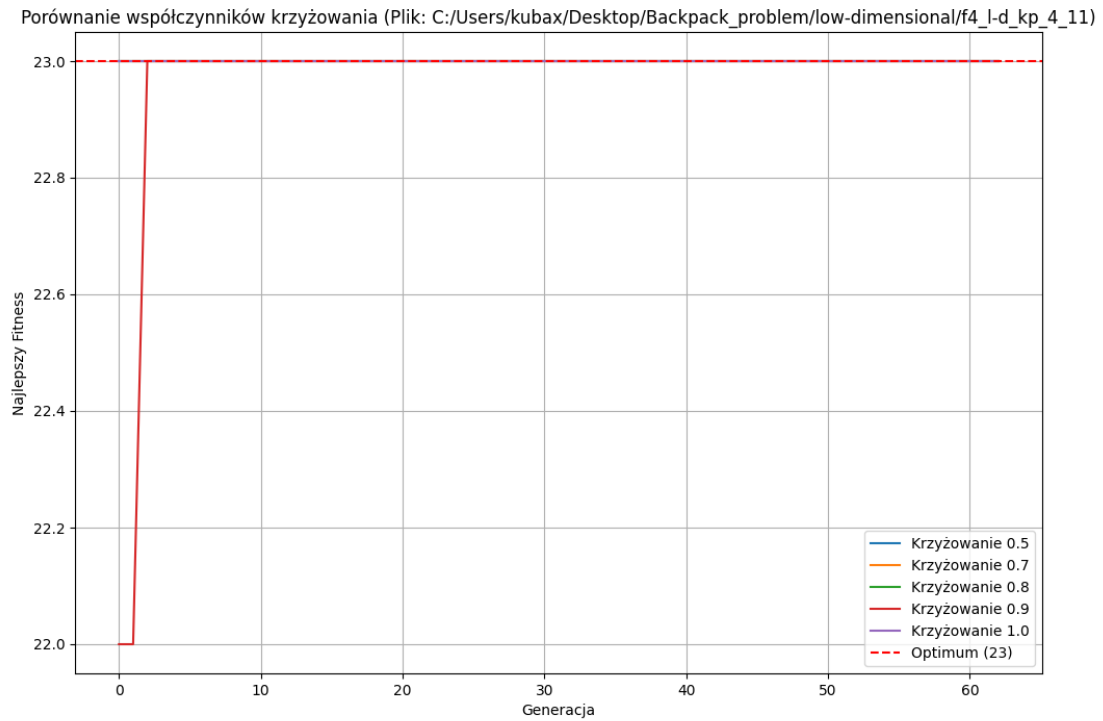


Wnioski:

- W tym teście mutacja okazała się niezbędna.
- Wartości $P=0.0$ i $P=0.01$ **nie znalazły optimum**, zatrzymując się na wartości 19.
- **Globalne optimum 23** zostało znalezione dla wszystkich wyższych wartości mutacji ($P=0.02$, $P=0.05$, $P=0.1$).

4.7.2. Porównanie współczynników krzyżowania (Wykres 14)

(Baza: P. Mutacji = 0.025)

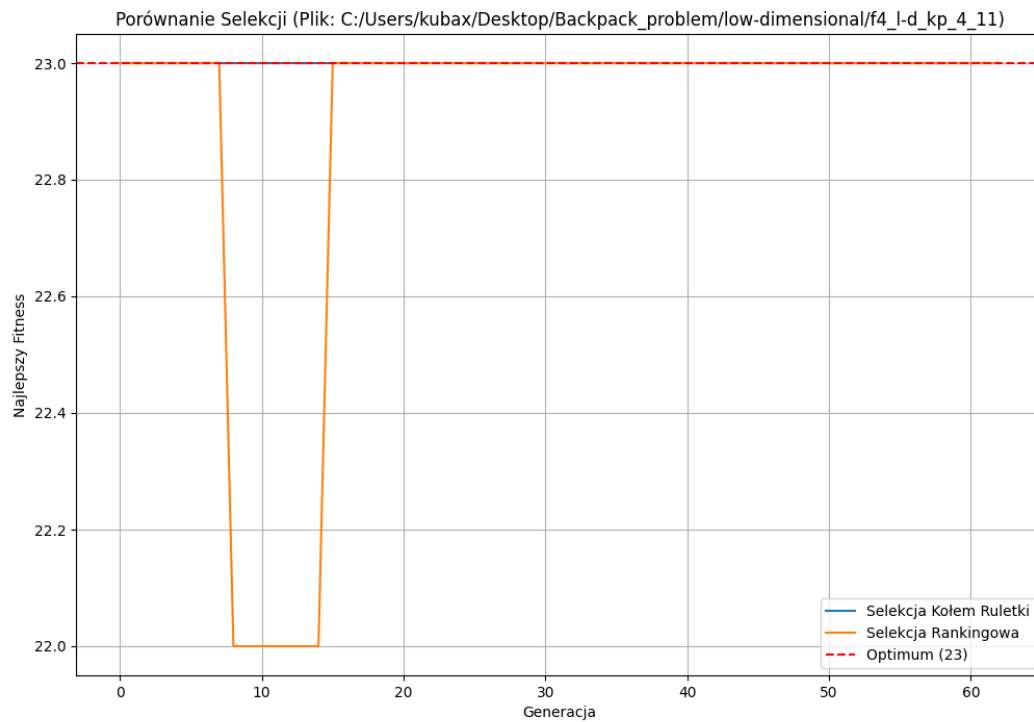


Wnioski:

- Dla tego problemu, przy bazowej mutacji, **każda testowana wartość P. Krzyżowania (0.5, 0.7, 0.8, 0.9, 1.0) pozwoliła na znalezienie globalnego optimum 23.**

4.8. Porównanie Metod (Wymaganie 4.5)

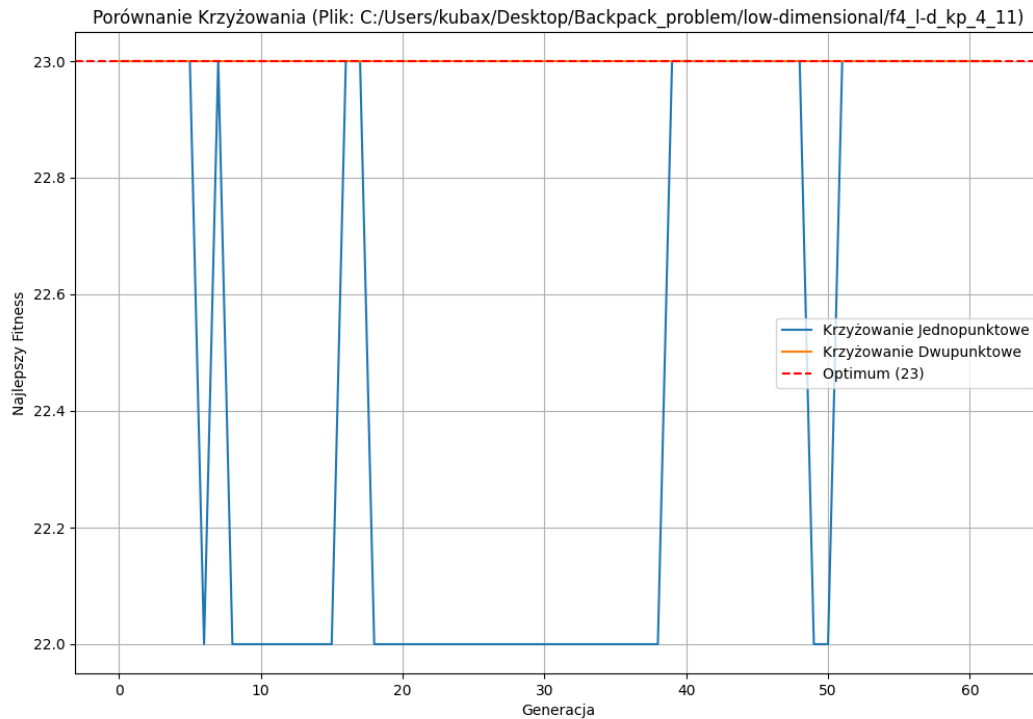
4.8.1. Porównanie Metod Selekcji (Wykres 15)



Wnioski:

- Obie metody, **Selekcja Kołem Ruletki** i **Selekcja Rankingowa**, bez problemu **znalazły globalne optimum 23**.

4.8.2. Porównanie Metod Krzyżowania (Wykres 16)



Wnioski:

- Obie metody, **Krzyżowanie Jednopunktowe** i **Krzyżowanie Dwupunktowe**, znalazły globalne optimum 23.

Analiza 5: Zbiór large_scale/knapPI_1_100_1000_1

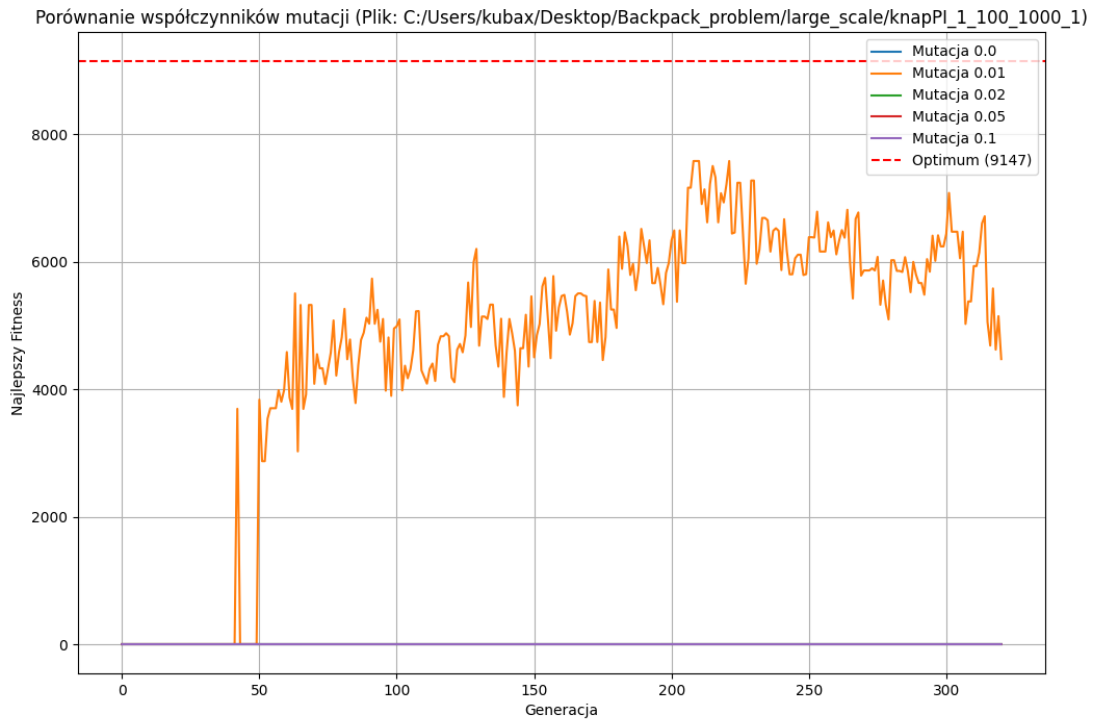
Jest to problem dużej skali (100 przedmiotów). Znane optimum wynosi **9147**.

- **Parametry Bazowe:** Populacja: 137, Iteracje: 321, P. Mutacji: 0.015, P. Krzyżowania: 0.85.

4.9. Porównanie Współczynników (Wymaganie 3.5)

4.9.1. Porównanie współczynników mutacji (Wykres 17)

(Baza: P. Krzyżowania = 0.85)



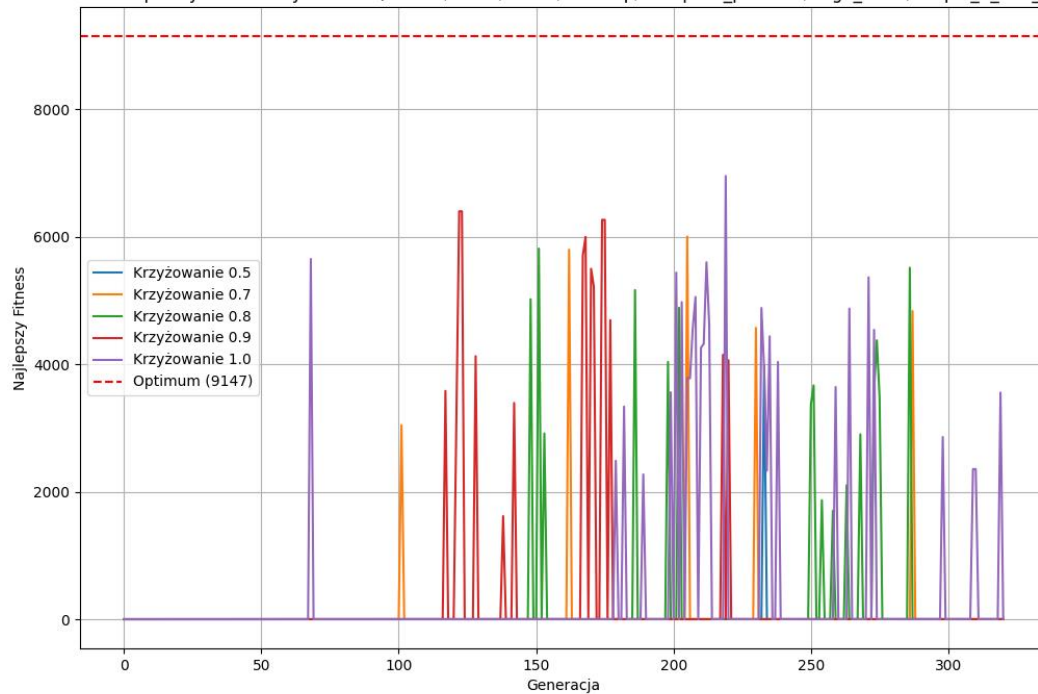
Wnioski:

- Ten test wyraźnie pokazuje, jak kluczowa jest mutacja dla problemów large_scale.
- **Brak mutacji (P=0.0) lub zbyt wysoka mutacja (P=0.02, 0.05, 0.1) skutkowały całkowitą porażką** – algorytm **ani razu nie znalazł poprawnego rozwiązania** (fitness < 1).
- Jedynie wartość **P=0.01** pozwoliła algorytmowi znaleźć poprawne rozwiązania. Mimo niestabilności, był w stanie osiągnąć bardzo wysoki wynik **6930** (ok. 75% optimum).

4.9.2. Porównanie współczynników krzyżowania (Wykres 18)

(Baza: P. Mutacji = 0.015)

Porównanie współczynników krzyżowania (Plik: C:/Users/kubax/Desktop/Backpack_problem/large_scale/knapPI_1_100_1000_1)

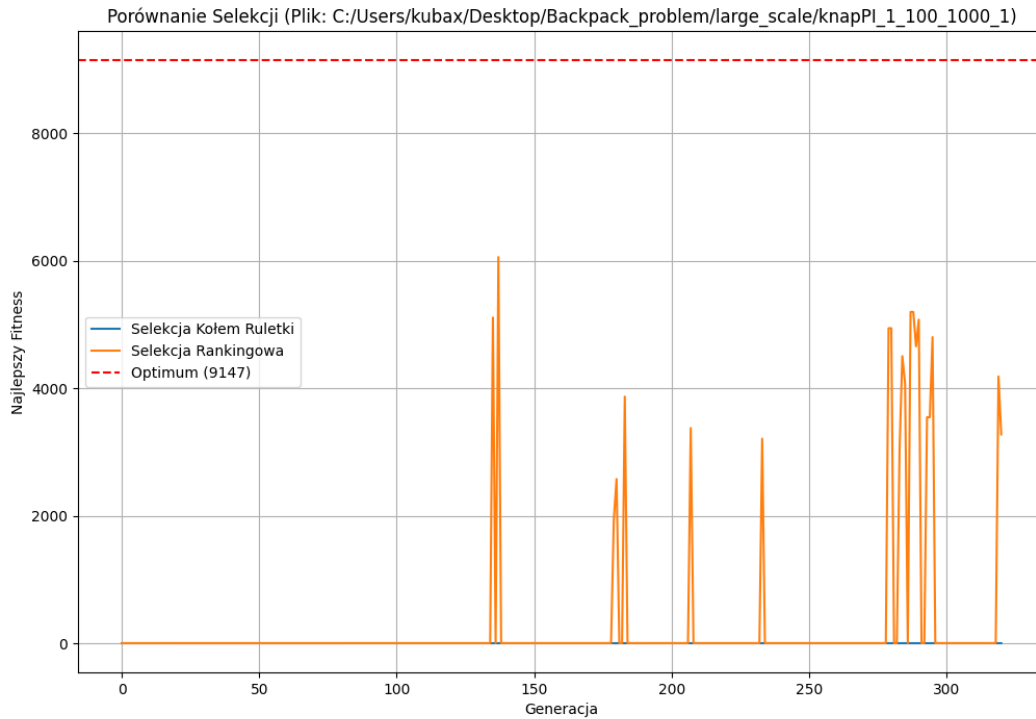


Wnioski:

- Wszystkie przebiegi (z bazową mutacją 0.015) miały trudności ze znalezieniem poprawnych rozwiązań, jednak **P. Krzyżowania = 1.0** dało najlepsze rezultaty, osiągając **6952** (ok. 76% optimum).
- Pozostałe wartości dały znacznie gorsze wyniki lub w ogóle nie znalazły poprawnych rozwiązań (jak P=0.5, 0.7, 0.8, 0.9). Wskazuje to na konieczność intensywnej eksploracji (krzyżowania) dla tego problemu.

4.10. Porównanie Metod (Wymaganie 4.5)

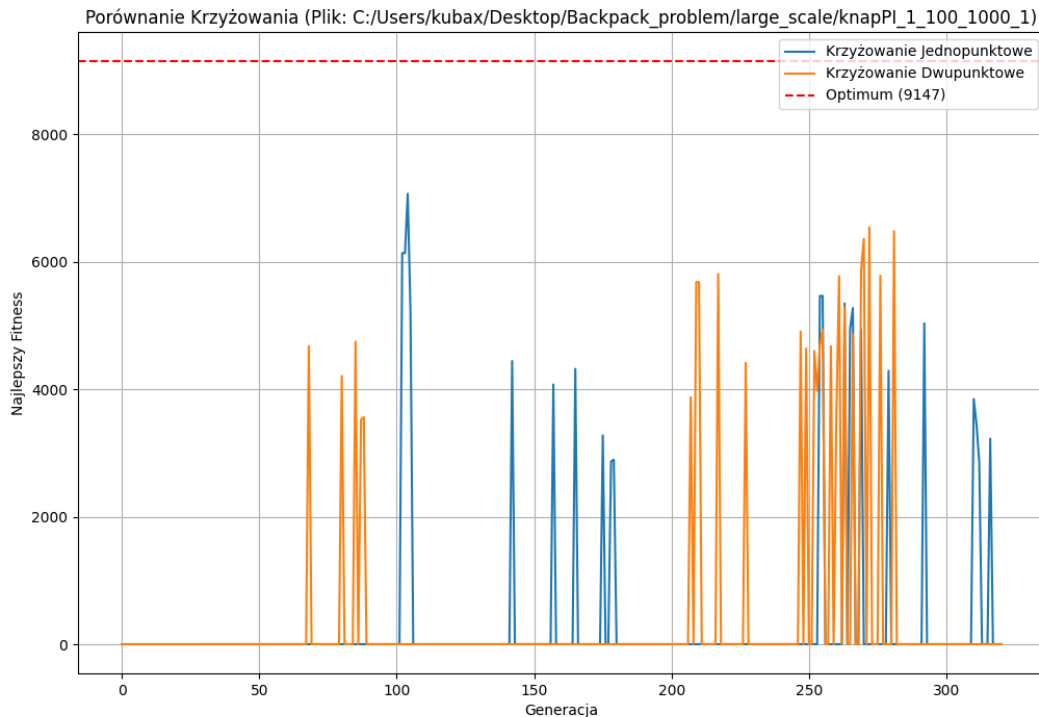
4.10.1. Porównanie Metod Selekcji (Wykres 19)



Wnioski:

- **Selekcja Kołem Ruletki** (niebieska linia) **zakończyła się katastrofalną porażką**. Przez cały proces ewolucji **nie znalazła ani jednego poprawnego rozwiązania** (fitness < 1).
- **Selekcja Rankingowa** (pomarańczowa linia) zadziałała **znacznie lepiej**. Ponieważ bazuje na *pozycji* w rankingu, była w stanie wypromować osobniki bliższe poprawnego rozwiązania i osiągnąć wysoki wynik **4941**.
- Mimo to, jej działanie było **ekstremalnie niestabilne** (np. spadek z 4941 do < 1), co pokazuje brak elitaryzmu.

4.10.2. Porównanie Metod Krzyżowania (Wykres 20)



Wnioski:

- Oba eksperymenty (korzystające z Selekcji Rankingowej) potwierdzają diagnozę **braku elitaryzmu**.
- **Krzyżowanie Jednopunktowe** (niebieska linia) osiągnęło wysoki, choć niestabilny wynik **4289**.
- **Krzyżowanie Dwupunktowe** (pomarańczowa linia) w tym przebiegu **całkowicie zawiodło**, nie znajdując ani jednego poprawnego rozwiązania (fitness < 1).

Analiza 6: Zbiór large_scale/knapPI_2_100_1000_1

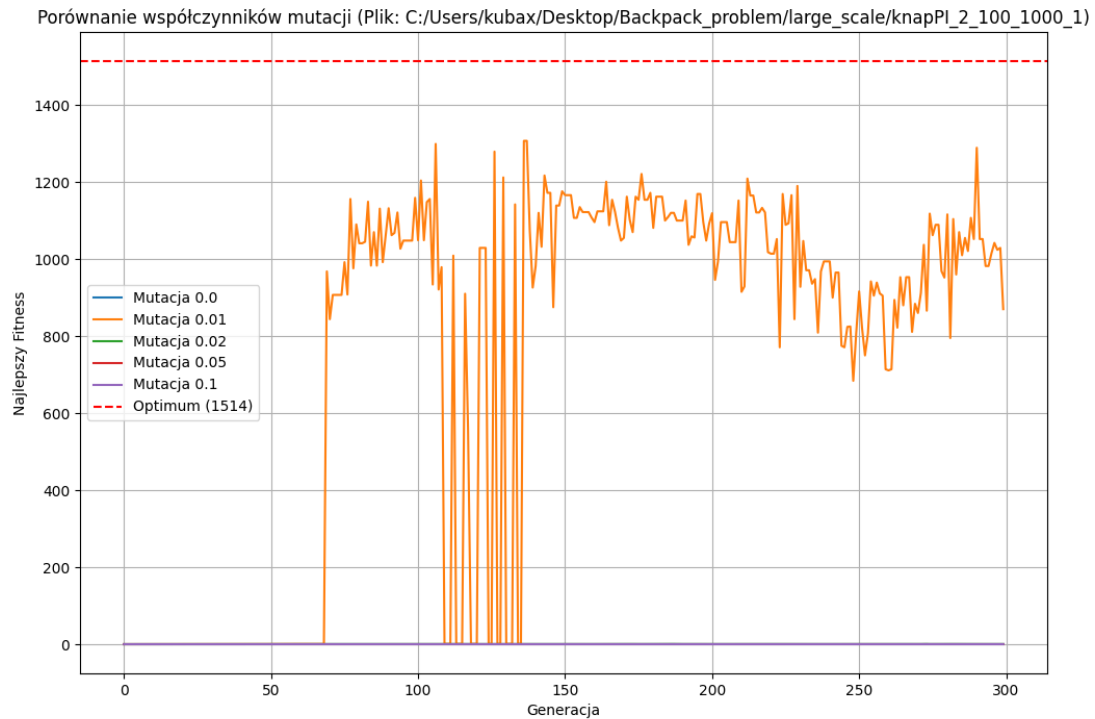
Jest to drugi problem dużej skali (100 przedmiotów). Optimum wynosi **1514**.

- **Parametry Bazowe:** Populacja: 100, Iteracje: 300, P. Mutacji: 0.01, P. Krzyżowania: 0.8.

4.11. Porównanie Współczynników (Wymaganie 3.5)

4.11.1. Porównanie współczynników mutacji (Wykres 21)

(Baza: P. Krzyżowania = 0.8)



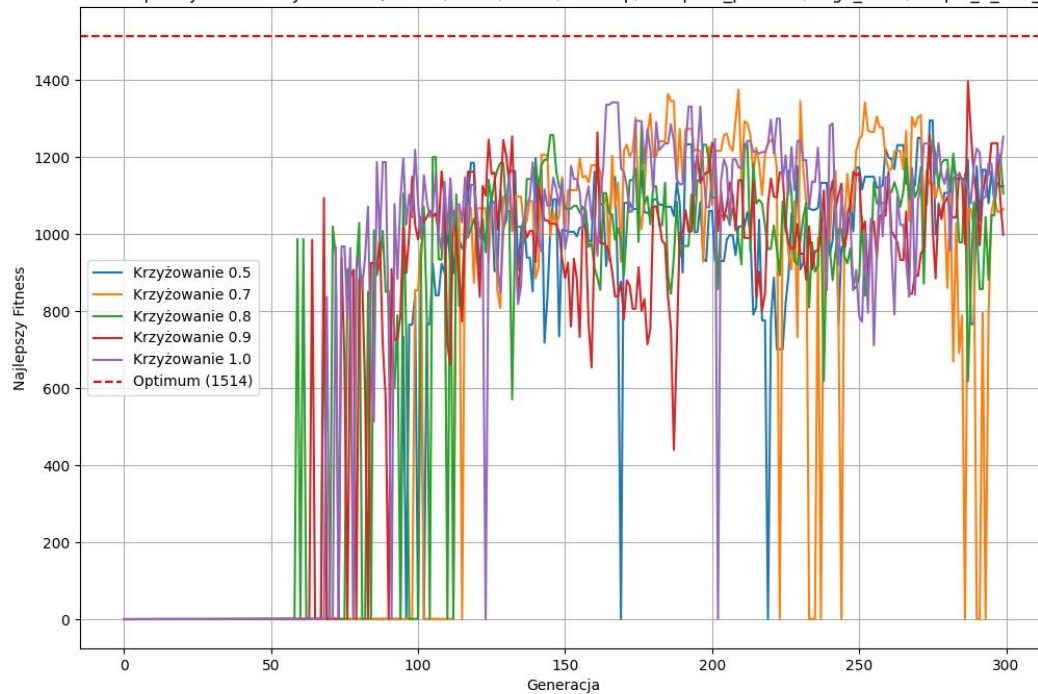
Wnioski:

- Ponownie, **brak mutacji (P=0.0)** lub **zbyt wysoka mutacja (P=0.02, 0.05, 0.1)** skutkowały **całkowitą porażką** (brak poprawnych rozwiązań).
- Jedynie bazowa wartość **P=0.01** pozwoliła algorytmowi "przebić się" i znaleźć dobre rozwiązania, osiągając maksymalnie **1172**.

4.11.2. Porównanie współczynników krzyżowania (Wykres 22)

(Baza: P. Mutacji = 0.01)

Porównanie współczynników krzyżowania (Plik: C:/Users/kubax/Desktop/Backpack_problem/large_scale/knapPI_2_100_1000_1)

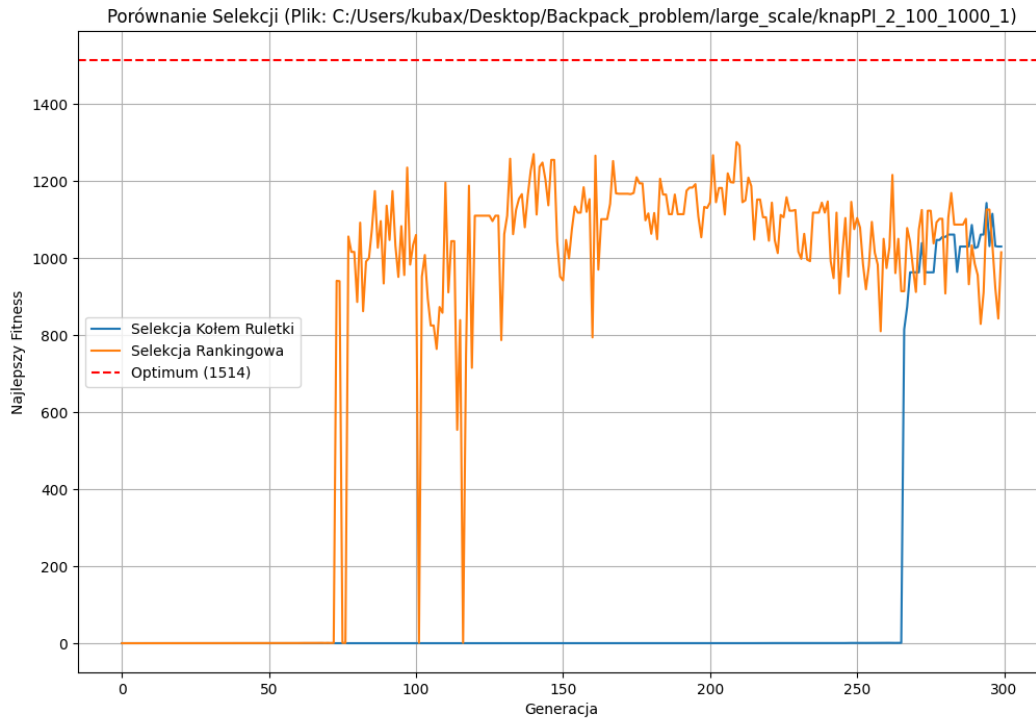


Wnioski:

- W tym teście **każda wartość P. Krzyżowania pozwoliła na znalezienie dobrych rozwiązań.**
- Najlepsze i najbardziej stabilne wyniki dały wartości **P=0.7 (max 1313)** oraz **P=1.0 (max 1253)**.
- Wartość bazowa (P=0.8) oraz P=0.9 również były dobre, ale osiągnęły niższe maksima (odpowiednio 1193 i 1162). Wartość P=0.5 była blisko (1220).

4.12. Porównanie Metod (Wymaganie 4.5)

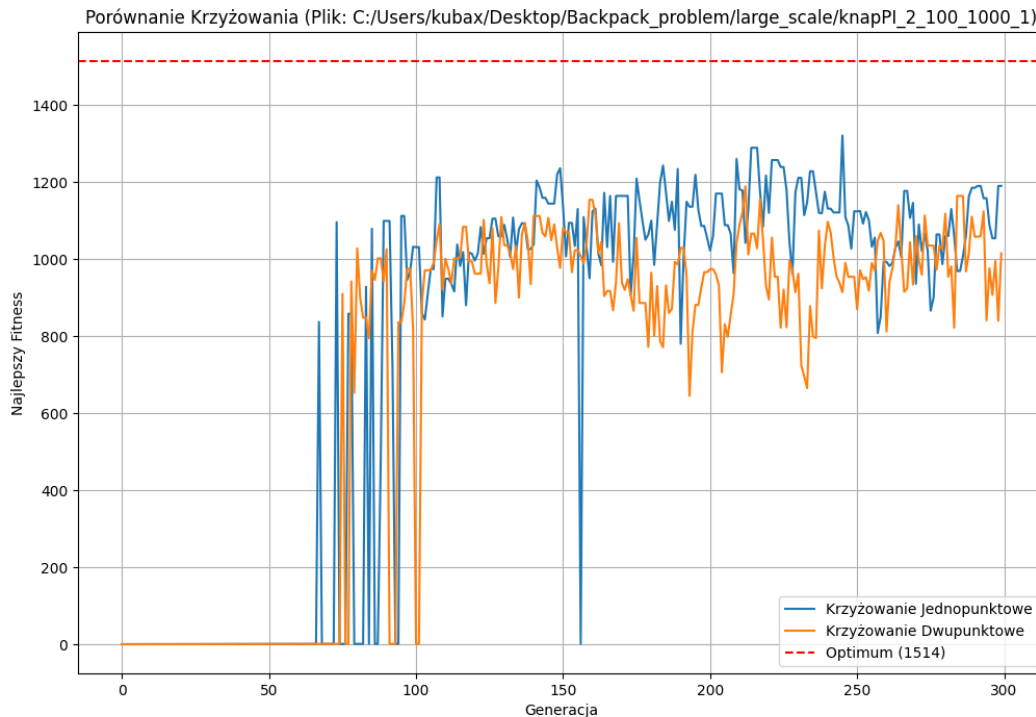
4.12.1. Porównanie Metod Selekcji (Wykres 23)



Wnioski:

- W przeciwieństwie do poprzedniego problemu large_scale, tutaj **obie metody selekcji znalazły poprawne rozwiązania**.
- **Selekcja Rankingowa** (pomarańczowa) znalazła rozwiązanie znacznie szybciej (ok. 80 iteracji) i osiągnęła wyższy szczyt (**1227**).
- **Selekcja Kołem Ruletki** (niebieska) potrzebowała znacznie więcej czasu (ok. 260 iteracji), aby znaleźć pierwsze poprawne rozwiązanie (max **1055**).

4.12.2. Porównanie Metod Krzyżowania (Wykres 24)



Wnioski:

- Obie metody (na Selekcji Rankingowej) dały podobne, niestabilne wyniki.
- **Krzyżowanie Jednopunktowe** (niebieska linia) osiągnęło maksymalnie **1217**.
- **Krzyżowanie Dwupunktowe** (pomarańczowa linia) osiągnęło **1154**. W tym przypadku metoda jednopunktowa okazała się nieznacznie lepsza.

5. Wnioski Końcowe

Zaimplementowany algorytm genetyczny poprawnie rozwiązuje dyskretny problem plecakowy i spełnia wszystkie założenia projektu na ocenę 3.5 oraz 4.5.

1. Dla problemów o **niskiej wymiarowości** (low-dimensional), algorytm jest bardzo skuteczny i w większości przypadków znajduje globalne optimum.
2. Dla problemów o **dużej skali** (large_scale), algorytm jest w stanie znaleźć rozwiązania, choć często nie są one optymalne (np. **76%** dla knapPI_1 czy **80%** dla knapPI_2).
3. Kluczową modyfikacją, która umożliwiła rozwiązywanie problemów large_scale, było wprowadzenie **funkcji kary** (capacity / total_weight) zamiast fitnessu 0 dla niepoprawnych osobników.

4. Eksperymenty z **porównaniem współczynników (3.5)** wykazały, że parametry te mają kluczowy wpływ na wynik. Szczególnie dla problemów large_scale, **brak mutacji ($P=0.0$)** lub **zbyt wysoka mutacja ($P>0.01$)** całkowicie uniemożliwiały znalezienie rozwiązania.
5. Eksperymenty z **porównaniem metod (4.5)** wykazały, że dla large_scale, **Selekcja Rankingowa** była jedyną skuteczną metodą selekcji, podczas gdy Ruletka w jednym z dwóch przypadków całkowicie zawiodła.
6. Największą zaobserwowaną wadą implementacji jest **niestabilność** i "gubienie" najlepszych znalezionych rozwiązań w problemach large_scale. Problem ten wynika bezpośrednio z **braku elitaryzmu** i mógłby zostać rozwiązany poprzez implementację mechanizmu gwarantującego przetrwanie najlepszego osobnika w kolejnej generacji.

6. Załączniki

1. Plik main.py (kod źródłowy)
2. Plik go_knapsack.py (kod źródłowy)