Obliczenia ewolucyjne

Implementacja algorytmów MSPSO, MPSO oraz BOA

Jędrzej Szor 239716

May 17, 2021

1 Opis zaproponowanej wersji algorytmu BOA

Zaimplementowana przeze mnie wersja algorytmu BOA bazuje na dokumencie zamieszczonym na wikampie. W na jedną iterację składa się obliczenie zapachu dla każdego motyla w roju, wybranie najlepszego osobnika oraz obliczenie nowych koordynatów dla każdego motyla. Zapach oblicza się na podstawie poniższych wzorów:

$$f = cI^a \tag{1}$$

$$I = \frac{1}{F} \tag{2}$$

gdzie F jest wartością funkcji dla aktualnych współrzędnych motyla.
Zastosowany sposób obliczania zapachu podstawowego I pozwala osiągnąć minimum globalne będące ≥ 0. W przypadku globalnego minimum ≤ 0 pojawiają się liczby zespolone, w które się nie zagłębiałem.

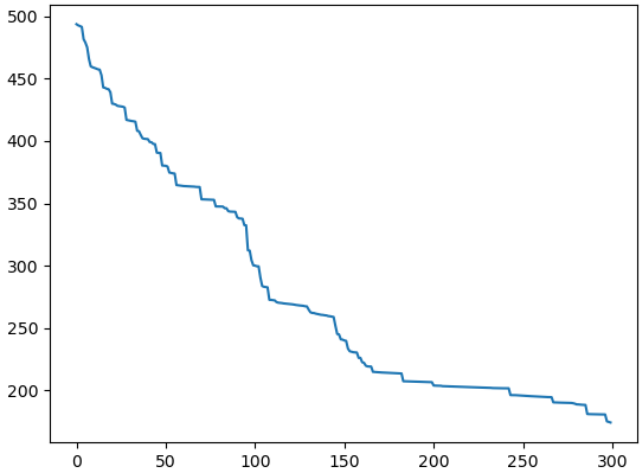
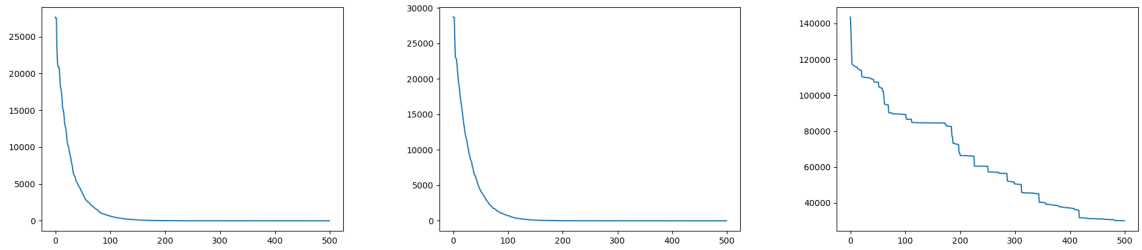


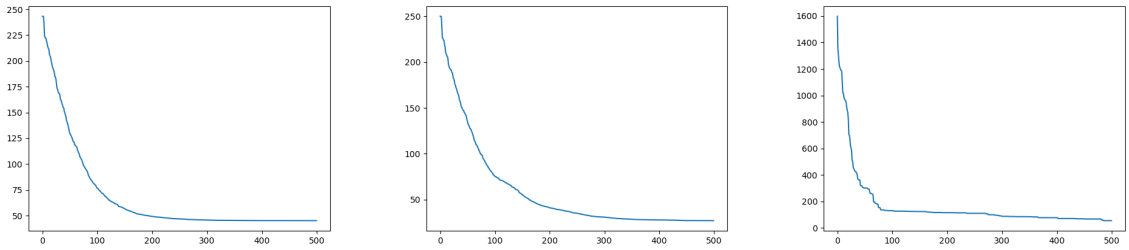
Figure 1: Wykres najlepszych wartości dla funkcji sfery podniesionej o 10.

Poniżej zaprezentowane są wyniki przebiegu poszczególnych algorytmów z wybranymi wartościami parametrów. Wykresy zostały sporządzone na podstawie 30 przebiegów algorytmów po 500 iteracji dla dziedzin przewidzianych w dokumencie z opisami funkcji.



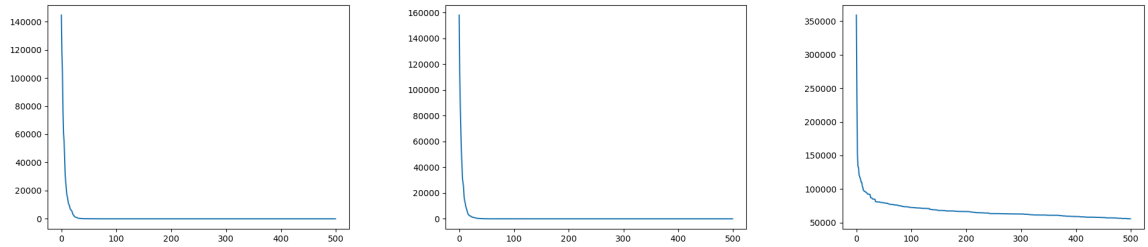
(a) MSPSO (b) MPSO (c) BOA

Figure 2: Wyniki dla funkcji sfery



(a) MSPSO (b) MPSO (c) BOA

Figure 3: Wyniki dla funkcji Rastrigina



(a) MSPSO (b) MPSO (c) BOA

Figure 4: Wyniki dla funkcji Corany

Jak widać na wykresach, algorytmy MSPSO oraz MPSO są o wiele bardziej niezawodne. Algorytm BOA jest jednak wielokrotnie szybszy. Średni czas przebiegu dla BOA wynosił nieco powyżej minuty, podczas gdy czas wykonania pozostałych algorytmów wahał się pomiędzy 30 a 40 minut.

2 Analiza wpływu parametrów na charakterystykę przebiegu algorytmów

Wszystkie eksperymenty w niniejszej sekcji wykonane zostały na funkcji sfery dla 20 wymiarów, 300 iteracji oraz prezentują wyniki uśrednione z 30 przebiegów.

2.1 MSPSO

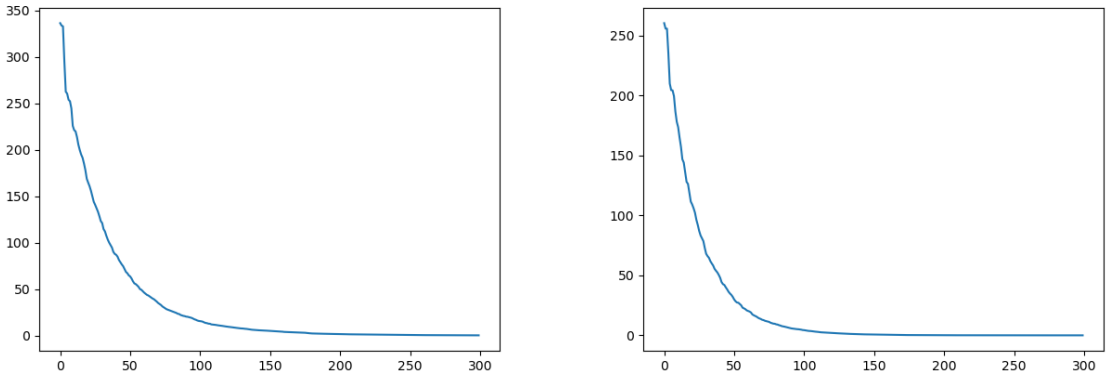
W algorytmach MSPSO oraz MPSO zastosowałem zmienne współczynniki przyspieszenia opisane szczegółowo w poprzednim raporcie.

$$wt = wt_{min} + (wt_{max} - wt_{min})\frac{(t_{max} - t)}{t_{max}}, \tag{3}$$

$$c_1 = (c_{1f} - c_{1i})\frac{t}{t_{max}} + c_{1i}, \tag{4}$$

$$c_2 = (c_{2f} - c_{2i})\frac{t}{t_{max}} + c_{2i} \tag{5}$$

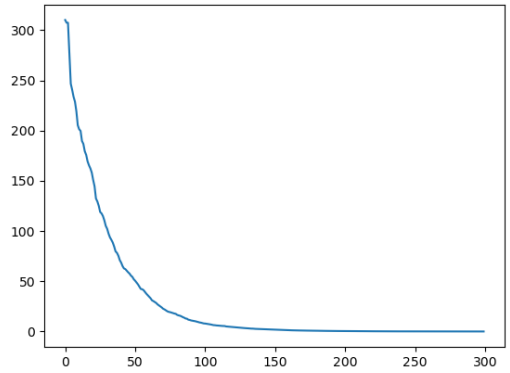
- $w_{init} = 0.9,\ w_{final} = 0.4$
- $c_{1init} = 2,\ c_{1final} = 1$
- $c_{2init} = 1,\ c_{2final} = 2$



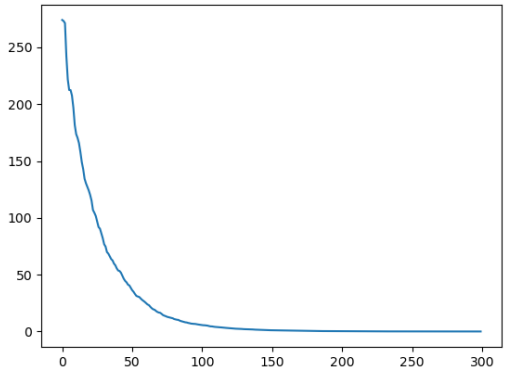
(a) Populacja = 10, min = $9.00 \cdot 10^{-3}$, czas = 1m 43s

(b) Populacja = 30, min = $2.04 \cdot 10^{-4}$, czas = 13m 36s

Figure 5: Wyniki dla zmiennej liczby populacji. Liczba rojów = 5



(a) Liczba rojów = 3, min = $1.30 \cdot 10^{-3}$, czas = 3m 52s

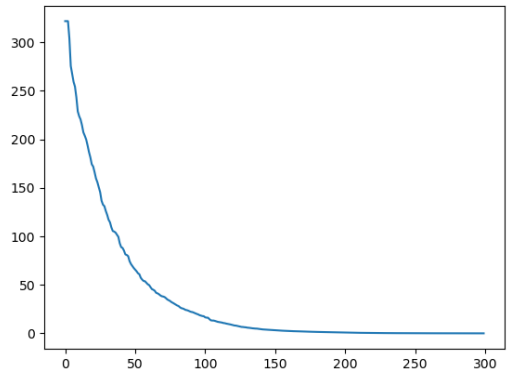


(b) Liczba rojów = 8, min = $2.04 \cdot 10^{-3}$, czas = 10m 8s

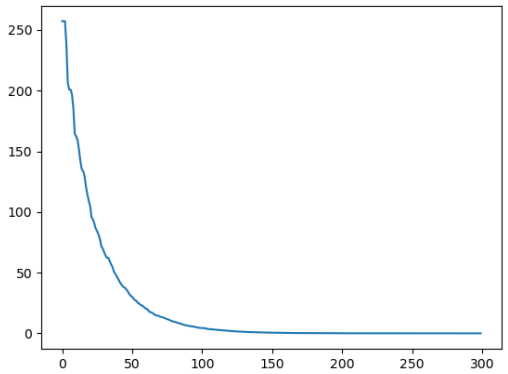
Figure 6: Wyniki dla zmiennej liczby rojów. Populacja = 20

Jak widać na wykresach liczba populacji i rojów poprawia czas zbiegania do minimum, jednak ogromnym kosztem czasowym. Przy zmianach obu parametrów różnice w charakterystyce są znikome, podczas gdy czas wykonania jest kilkukrotnie dłuższy.

2.2 MPSO

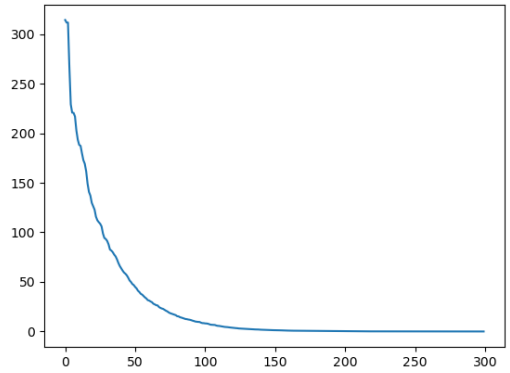


(a) Populacja = 10, min = $7.08 \cdot 10^{-3}$, czas = 1m 42s

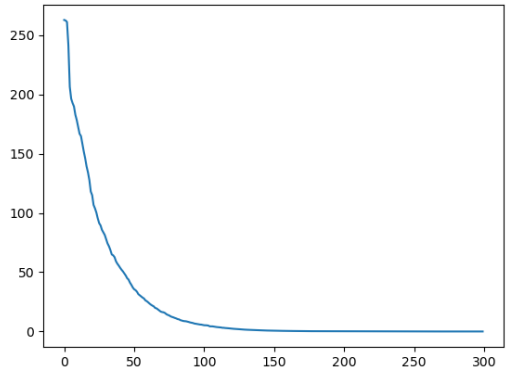


(b) Populacja = 30, min = $3.72 \cdot 10^{-4}$, czas = 13m 42s

Figure 7: Wyniki dla zmiennej liczby populacji. Liczba rojów = 5, tasowanie cząstek co 100 iteracji

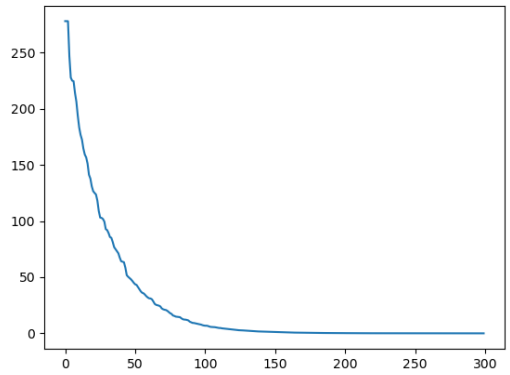


(a) Liczba rojów = 3, min = $1.01 \cdot 10^{-3}$, czas = 3m 50s

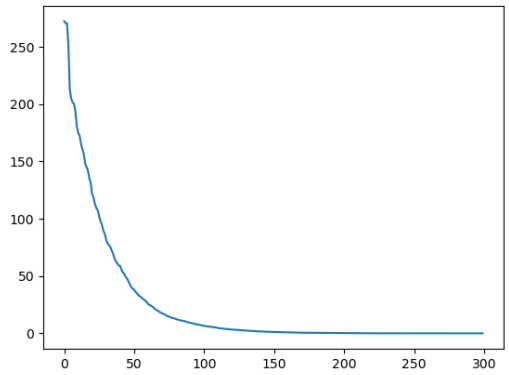


(b) Liczba rojów = 8, min = $4.84 \cdot 10^{-4}$, czas = 9m 57s

Figure 8: Wyniki dla zmiennej liczby rojów. Populacja = 20, tasowanie cząstek co 100 iteracji



(a) Tasowanie cząstek co 20 iteracji, min = $1.56 \cdot 10^{-3}$, czas = 6m 17s



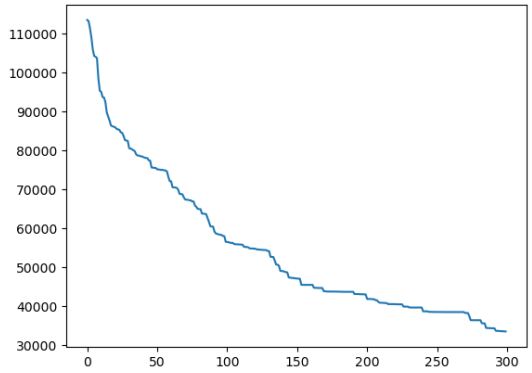
(b) Tasowanie cząstek co 150 iteracji, min = $1.40 \cdot 10^{-3}$, czas = 6m 9s

Figure 9: Wyniki dla zmiennej częstotliwości tasowania cząstek. Populacja = 20, liczba rojów = 5

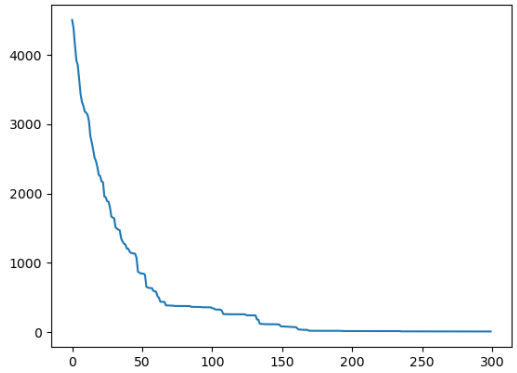
Podobnie jak przy MSPSO, zmiany w charakterystykach podczas zmian ilości cząstek oraz ilości rojów są znikome, podczas gdy znacząco wydłuża się czas wykonania. Interesującym spostrzeżeniem jest to, że lepsze wyniki zostały uzyskane przy rzadszym tasowaniu cząstek.

2.3 BOA

Próbowałem różnych wersji zmiennego współczynnika a: Liniowo malejący od 1 do 0, liniowo malejący od 1 do 0.5, nieliniowo zmienny, jednak najlepsze efekty uzyskałem używając stałej wartości.

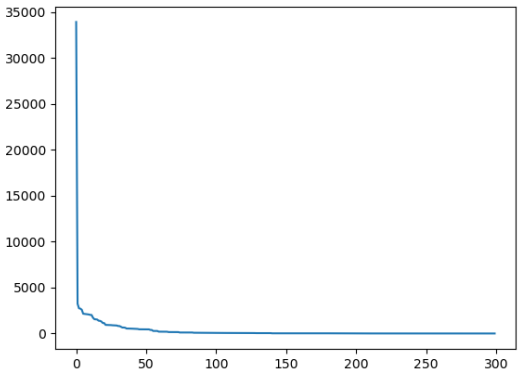


(a) Populacja = 20, min = $3.26 \cdot 10^{-9}$, czas = 4s

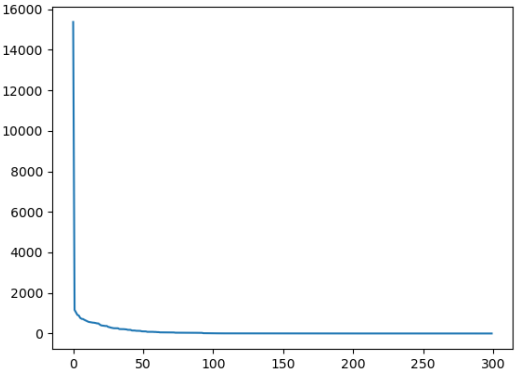


(b) Populacja = 50, min = $1.38 \cdot 10^{-4}$, czas = 11s

Figure 10: Wyniki dla zmiennej liczby populacji. a = 0.5, c = 0.5, p = 0.5

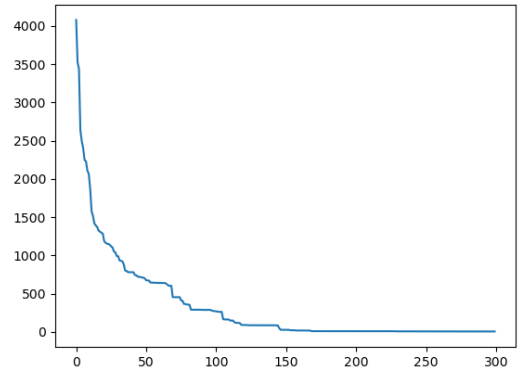


(a) p = 0.7, min = $6.37 \cdot 10^{-5}$, czas = 7s

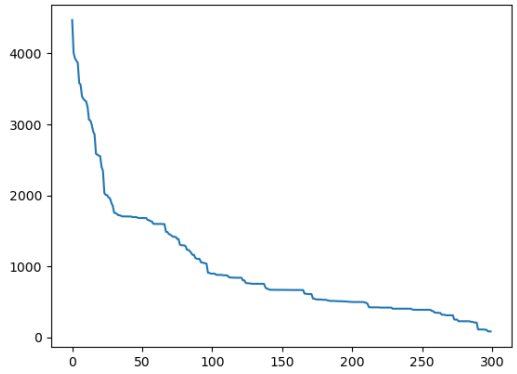


(b) p = 0.3, min = $3.26 \cdot 10^{-5}$, czas = 8s

Figure 11: Wyniki dla zmiennego współczynnika p. a = 0.5, c = 0.5, populacja = 35

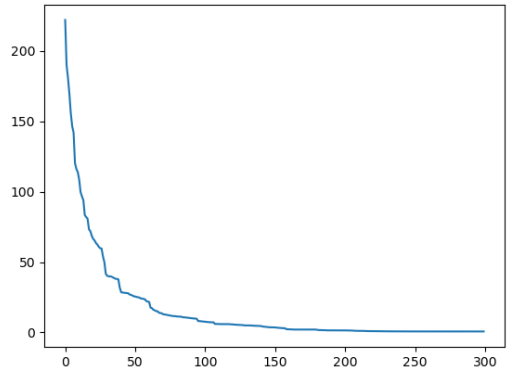


(a) c = 0.7, min = $5.01 \cdot 10^{-5}$, czas = 7s

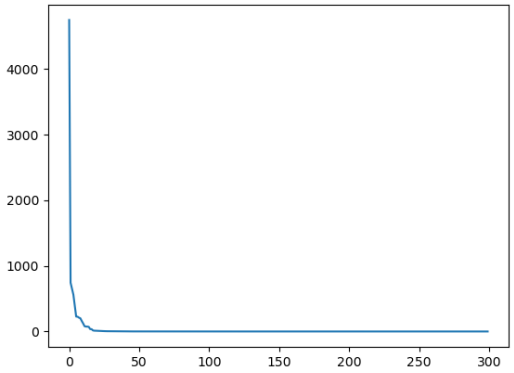


(b) c = 0.3, min = $4.73 \cdot 10^{-6}$, czas = 6s

Figure 12: Wyniki dla zmiennego współczynnika c. a = 0.5, p = 0.5, populacja = 35



(a) a = 0.7, min = $8.76 \cdot 10^{-4}$, czas = 7s



(b) a = 0.3, min = $8.03 \cdot 10^{-9}$, czas = 7s

Figure 13: Wyniki dla zmiennego współczynnika a. c = 0.5, p = 0.5, populacja = 35

W oczywisty sposób, większa liczebność populacji daje lepsze i bardziej wiarygodne wyniki. Niższa wartość parametru p, a zatem niższe prawdopodobieństwo ruchu niezależnie od zapachu, uzyskała bardziej korzystną charakterystykę. Lepszym wyborem okazała się być również wyższa wartość parametru c, czyli parametru liniowo zmieniającego moc zapachu. Dodatkowo, niższa wartość współczynnika a, czyli wykładnika potęgi zapachu, uzyskała lepsze wyniki.

3 Wnioski

Algorytmy MSPSO oraz MPSO są niezwykle podobne zarówno w sposobie działania jak i charakterystyce przbiegu. Okazały się być niezwykle niezawodne i niezależnie od wartości parametrów uzyskiwać bardzo dobre wyniki. Algorytm BOA jest mniej niezawodny. Potrafi uzyskać znakomite wyniki jednak niektóre przetestowane przypadki okazały się być dla niego wyzwaniem. Mimo to, jego szybkość bije na głowę algorytmy MSPSO oraz MPSO, podczas gdy wyniki są niemalże porównywalne. Warto zwrócić uwagę, że wszystkie eksperymenty przeprowadzane były dla ustalonej ilości iteracji, jednak w czasie, w którym te bardziej złożone algorytmy wykonają 300 iteracji, BOA jest w stanie wykonać tysiące, a zatem osiągnąć większą precyzję.