Sprawozdanie - Algorytmy ewolucyjne

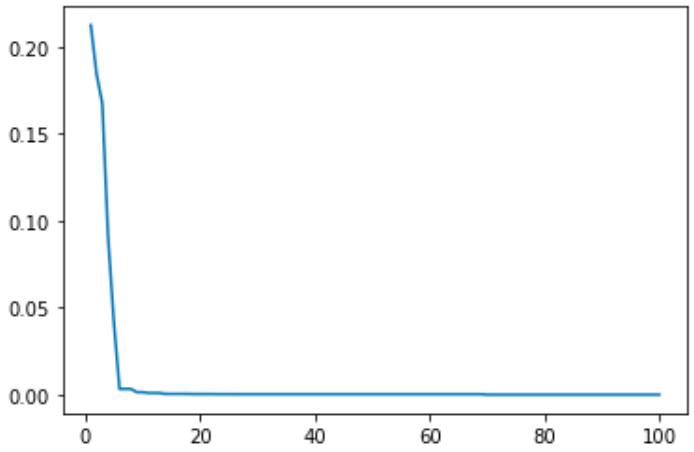
Jakub Lis
305744

Streszczenie

Laboratorium dotyczyło implementacji algorytmu ewolucyjnego. Początkowo powstał algorytm genetyczny do optymalizacji funkcji kwadratowej w R^3. Następne dwa tygodnie dotyczyły cutting stock problem, gdzie trzeba było zaimplementować algorytm maksymalizujący sumę wartości umieszanych prostokątów w kole. Ostatni tydzień dotyczył optymalizacji wag w sieci MLP z użyciem algorytmu genetycznego.

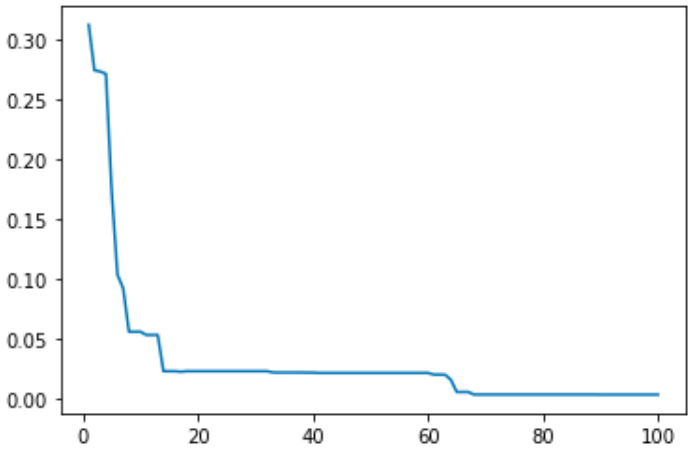
1 Optymalizacja funkcji kwadratowej w R^3

Algorytm na początku inizjalizuje ustaloną liczbę $n \in \mathbb{N}$ początkowych punktów (x, y, z). Są to wstępne rozwiązania, czyli osobniki. Rozwiązania te łączymy w losowe pary i z ustalonym prawdopodobieństwem ze sobą krzyżujemy, biorąc początkową liczbę bitów z jednego rozwiązania i pozostałą liczbę bitów z drugiego. Następnie dla każdego rozwiązania, także tych przed krzyżowaniem, przeprowadzamy mutację polegającą na dodaniu do każdej ze współrzędnych rozwiązania wylosowanej wartości z rozkładu normalnego $N(0, 1)$. Selekcja to losowy wybór n rozwiązań wykorzystywanych w następnej iteracji, ale losowy w taki sposób, aby faworyzować rozwiązania z lepszymi wynikami. Przeprowadziliśmy test na funkcji $f(x, y, z) = x^2 + y^2 + 2z^2$. Przebieg minimalizacji dla populacji $n = 100$ i liczby iteracji równej 100 przedstawia poniższy wykres.



Rysunek 1: Działanie algorytmu genetycznego przy optymalizacji funkcji

Dla każdej iteracji policzona jest uzyskiwana najmniejsza wartość funkcji f przez najlepszy punkt, tzn. najlepszego osobnika. Widzimy, że algorytm zbiegł bardzo szybko. Końcowy uzyskany jest rzędu $2.07 \cdot 10^{-9}$ dla punktu $(3.07 \cdot 10^{-5}, 3.36 \cdot 10^{-5}, -2.64 \cdot 10^{-82})$, czyli punktu bardzo bliskiego $(0, 0, 0)$, który jest analitycznym rozwiązaniem tej optymalizacji. Dla tej samej funkcji, sprawdzimy jeszcze, jak zbiegnie algorytm przy mniejszej populacji, niech $n = 50$.



Rysunek 2: Działanie algorytmu genetycznego przy optymalizacji funkcji

Widzimy, że przy mniejszej populacji zbiegnięcie na podobny poziom jest w późniejszej iteracji, ale algorytm i tak uzyskuje w 100 iteracji satysfakcjonujące nas rozwiązanie. Warto podkreślić, że czas wykonywania tych obliczeń nie przekraczał jednej sekundy, a więc populację w razie potrzeby jeszcze dokładniejszej minimalizacji możnaby było zwiększyć.

2 Wypełnianie koła prostokątami

W kolejnym tygodniu należało przygotować bardziej zaawansowaną implementację algorytmu genetycznego. Miał on rozwiązywać problem maksymalizacji sumy wcześniej ustalonych wartości prostokątów umieszczonych w kole o zadanym promieniu tak, aby prostokąty na siebie nie nachodziły. Zacznijmy od opisania przygotowanej przeze mnie implementacji.

2.1 Inicjalizacja

Tworzymy zbiór rozwiązań o ustalonej wcześniej mocy. Zaczynamy od utworzenia rozwiązań, w których występuje tylko jeden rodzaj prostokąta. Losujemy z prawdopodobieństwem $\frac{1}{2}$, czy prostokąt ma być obrócony, a następnie losujemy współrzędne (x, y) dolnego, lewego wierzchołka i próbujemy umieścić prostokąt w kole tak, aby spełniał wcześniej założone warunki. Jeżeli nie uda się umieścić prostokąta to losujemy ponownie, to czy ma być obrócony, jak i współrzędne. Powtarzamy takie losowanie ustaloną liczbę razy, w zastosowanym przeze mnie algorytmie było to 1000 razy. Jeżeli udało się wcześniej wstawić prostokąt, to oczywiście próbujemy wstawiać kolejny. Jeżeli przez 1000 iteracji się nie udaje, to stopujemy i zwrazamy uzyskane rozwiązanie jako osobnika. Gdy już mamy

rozwiązania zawierające jeden typ prostokąta (czyli takich rozwiązań będzie tyle ile wynosi moc zbioru dozwolonych prostokątów), to tworzymy pozostałe rozwiązania, gdzie prostokąty ze sobą mieszamy. Warto podkreślić, że losowanie prostokąta odbywa się raz, a następnie przez 1000 iteracji losujemy jedynie współrzędnie i to, czy ma być obrócony. Jeżeli uda się wstawić prostokąt, to dopiero losujemy kolejny ze zbioru dozwolonych prostokątów.

2.2 Krzyżowanie

Krzyżowanie w użytej implementacji polega na wzięciu dwóch osobników (rozwiązań) i po kolei losowaniu na przemian po jednym prostokącie z każdego z nich. Jeżeli dany prostokąt da się wstawić do koła, to to robimy, a jeżeli nie to go odrzucamy. Jeżeli jedno z rozwiązań ma więcej prostokątów, to po tym jak skończą się prostokąty do losowania w drugim bierzemy już tylko te, które zostały. W ten sposób mieszamy dwa różne rozwiązania, najprawdopodobniej zmniejszając wynik tego lepszego, ale z nadzieją, że po mutacji możemy uzyskać coś więcej.

2.3 Mutacja

Mutacja przebiega dla każdego rozwiązania z ustalonym prawdopodobieństwem. Jeżeli wylosujemy, że akurat przeprowadzamy mutację to przebiega ona w następny sposób:

- przesunięcie losowego prostokąta maksymalnie na lewo i maksymalnie w dół (aż się nie zbliży do granic koła lub do innych prostokątów)
- obrót losowego prostokąta, o ile jest możliwy
- dodanie prostokąta losowo wybranego z puli - dla wylosowanego prostokąta losujemy 10 tysięcy położeń i sprawdzamy, czy jest możliwośc dodania go; jeżeli prostokąta się nie uda dodać, to ten krok mutacji jest pomijany

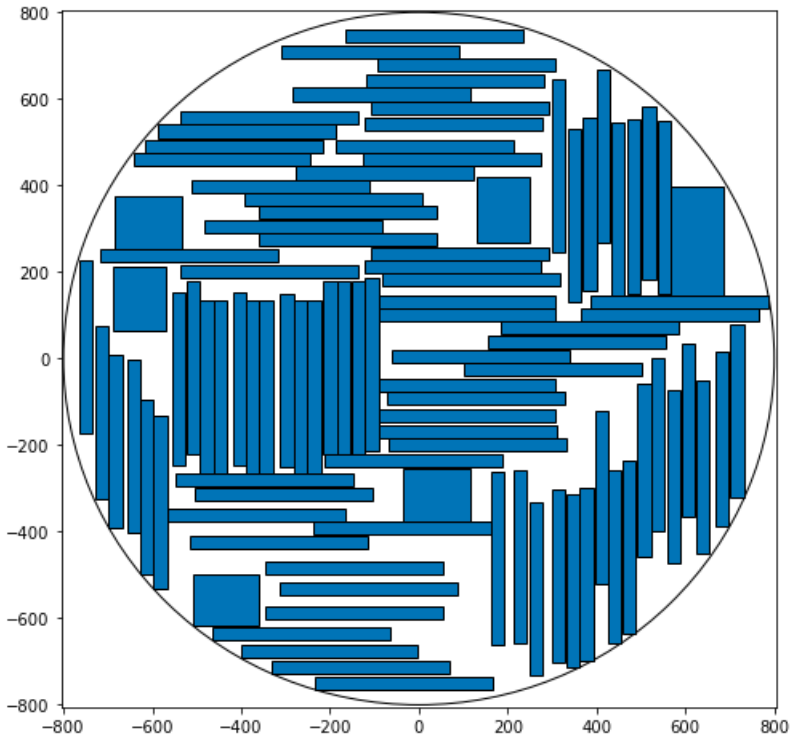
2.4 Selekcja

Selekcja to wybranie z osobników takich, którzy przejdą do kolejnej iteracji algorytmu. Do wyboru mamy osobników z poprzedniej iteracji po zostasowanych ewentualnych mutacjach (zwróćmy uwagę, że mutacja nie może pogorszyć uzyskiwanego wyniku) oraz osobników, którzy powstali podczas krzyżowania i także przeszli mutację. Selekcja jest losowa, ale nie całkowicie losowa. Faworyzowane są rozwiązania, które dają najlepszy wynik. W szczególności najlepsze rozwiązanie przechodzi do kolejnej iteracji z prawdopodobieństwem równym 1 po to, aby nie utracić najlepszego rozwiązania wraz z kolejnymi iteracjami algorytmu. Najgorsze rozwiązania mają szanse przejść dalej, ale z mniejszym prawdopodobienstwem.

2.5 Uzyskiwane wyniki

2.5.1 Zbiór R800

Algorytm został uruchomiony dla populacji liczności 50 i dla 30-tu epok. Najlepsze uzyskane rozwiązanie wynosiło wówczas 37900, a wizualizacja tego rozwiązania znajduje się poniżej.



Rysunek 3: Najlepsze uzyskane rozwiązanie na zbiorze R800

Ponadto dla testów uruchomiony został algorytm dla różnych liczności i różnych liczb epok. Uzyskiwane wtedy najlepsze wyniki przedstawia poniższa tabela.

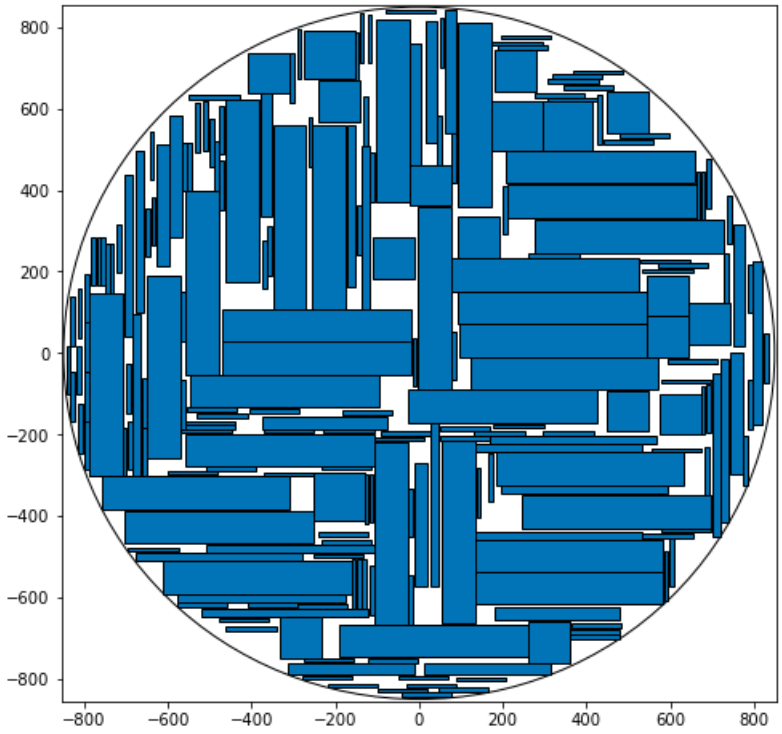
Populacja	Liczba epok	Uzyskany wynik
50	30	37900
100	30	39180
50	50	38140
100	50	39780
40	70	39740

Tablica 1: Uzyskiwane wyniki przy różnych parametrach

W tabeli nie ma wyniku bardzo odbiegającego od reszty. Uzyskiwane wyniki są głównie spowodowane istnieniem jednego prostokąta, który najbardziej opłacało się umieszczać w kole (miał małe pole, a dużą wartość) - jest to prostokąt, który dominuje na wizualizacji najlepszego rozwiązania wyżej.

2.5.2 Zbiór R850

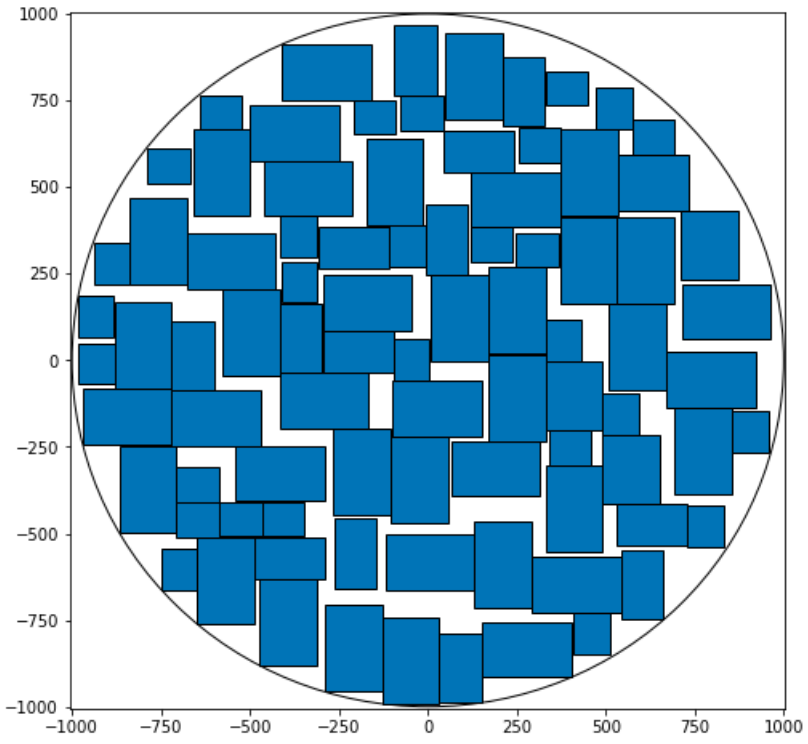
Zbiór R850 był zbiorem bez minimalnego wymogu do uzyskania punktów, dlatego prezentuję jedynie najlepsze rowzwiązanie przy ustawionej liczbie populacji 30 i liczbie epok 50. Suma wartości prostokątów z rysunku poniżej wynosiła 413660.



Rysunek 4: Najlepsze uzyskane rozwiązanie na zbiorze R850

2.5.3 Zbiór R1000

Rozwiązanie prezentowane podczas zajęć uzyskałem przy liczności populacji równej 30 i przy liczbie epok równej 100. Osiągnity wówczas wynik wynosił 25120, a wizualizacja rozwiązania znajduje się poniżej.



Rysunek 5: Najlepsze uzyskane rozwiązanie na zbiorze R1000

Ponadto, przeprowadziłem testy na różnych licznościach populacji oraz różnej liczbie epok, wyniki przedstawia poniższa tabela.

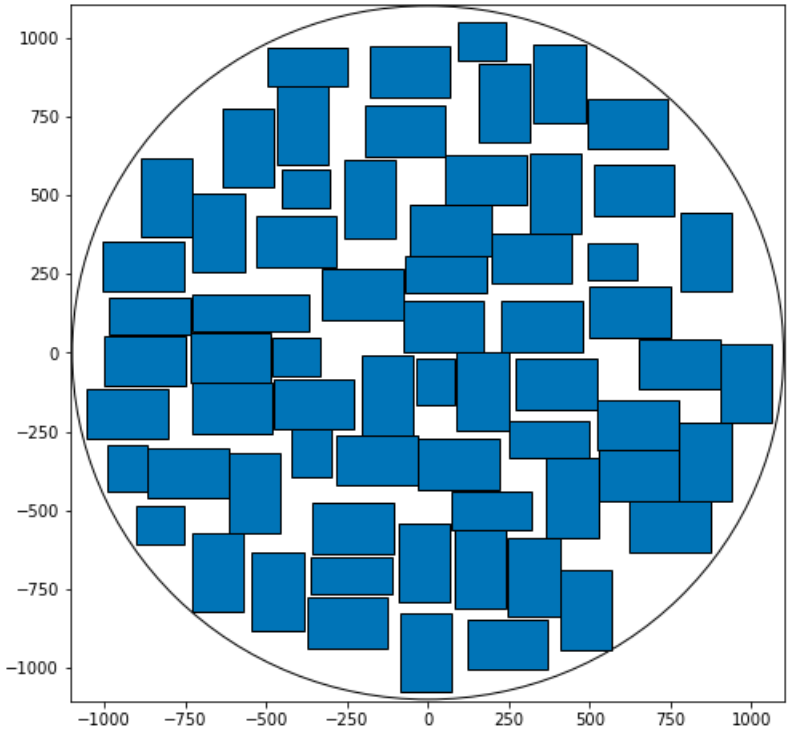
Populacja	Liczba epok	Uzyskany wynik
30	100	25120
30	50	24480
50	50	24760
70	50	24960

Tablica 2: Uzyskiwane wyniki przy różnych parametrach

Ponownie widzimy, że niezależnie od ustawianych parametrów wyniki niewiele się różnią. Na pewno zwiększenie liczby epok poprawia wynik, ponieważ prostokąty podczas mutacji bardziej się ściskają, ale wówczas znacząco wydłuża się czas pracy algorytmu. Aby dobrze dobrać parametry należy ocenić, jaki wynik będzie dla nas satysfakcjonujący. W tym przypadku należało przekroczyć próg 25000, więc potrzebowałem stu iteracji.

2.5.4 Zbiór R1100

Kolejnym zbiorem do testowania działania było koło o promieniu 1100. Prezentowany wynik uzyskałem dla populacji 30 i 25 iteracji. Suma wartości prostokątów w tym kole wynosiła 31820.



Rysunek 6: Najlepsze uzyskane rozwiązanie na zbiorze R1100

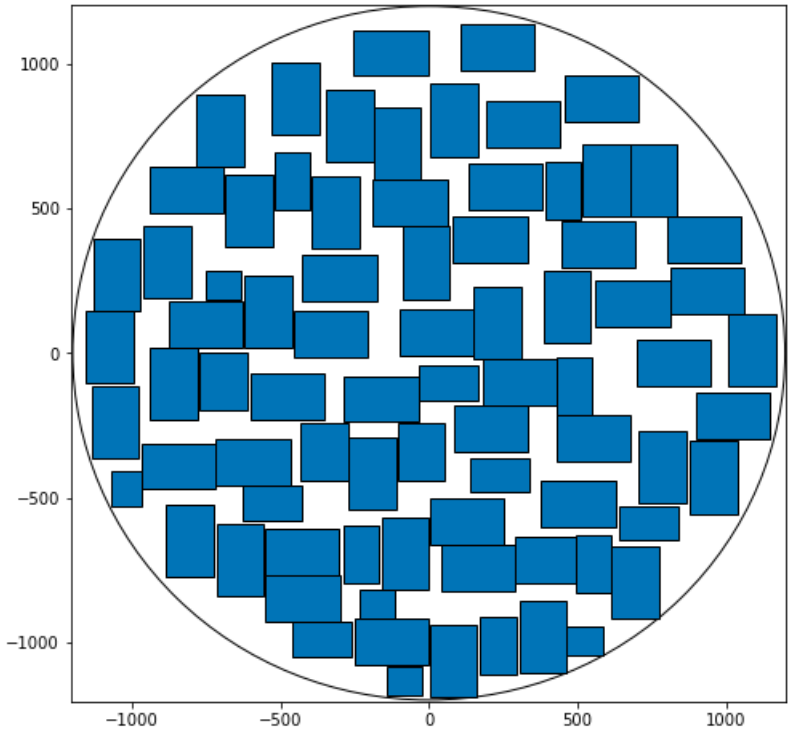
Poniższa tabela przedtsawia wyniki przy użyciu innych wartości dla liczności populacji i dla liczby iteracji algorytmu.

Populacja	Liczba epok	Uzyskany wynik
30	25	31820
30	50	32900
60	25	32820
60	50	33560
20	20	28980

Tablica 3: Uzyskiwane wyniki przy różnych parametrach

2.5.5 Zbiór R1200

Ostatnim zbiorem używanym do testów było koło o promieniu 1200. Najlepszy uzyskany wynik przy liczbie polulacji 30 i 20-tu epokach wynosił 33100. Poniżej wizualizacja uzyskanych wyników.



Rysunek 7: Najlepsze uzyskane rozwiązanie na zbiorze R1200

Ponownie, przeprowadziłem testy jak wpłynie zmiana parametrów na osiągane wyniki.

Populacja	Liczba epok	Uzyskany wynik
30	20	33100
30	50	33980
60	20	33380
60	50	35020
50	100	35560

Tablica 4: Uzyskiwane wyniki przy różnych parametrach

3 Podsumowanie i wnioski

Algorytmy genetyczne w zastosowanych rozwiązaniach bardzo dobrze się sprawdziły i dawały dobre wyniki w zadanych zadaniach. W szczególności zastosowanie ich miało sens dla zadania umieszczania prostokątów w kole, gdzie nie da się rozwiązać tego zadania analitycznie (albo jeszcze takie rozwiązanie nie istnieje...). Niestety, nie zaimplementowałem optymalizacji wag sieci MLP z użyciem algorytmu genetycznego - nie wystarczyło na to zadanie czasu.