WSI – ćwiczenie 2

Algorytmy ewolucyjne i genetyczne

(algorytm genetyczny Hollanda dla wektorów binarnych)

grupa 101

Radosław Pietkun

293469

semestr 21L

Spis treści

1. Funkcja celu i przyjęte założenia

Funkcja celu i jej dziedzina:

$$f(x) = -\frac{\sum_{i=1}^{6}\left(x_i^4 - 16x_i^2 + 5x_i\right)}{2}$$

$$x_i \in \{-4, -3, -2, -1, 0, 1, 2, 3\}$$

Przyjąłem kodowanie w kodzie U2, ponieważ dobrze ono pasuje do określonej dziedziny i pozwala optymalnie wykorzystać 3 bity po zakodowania każdego x_i.

Podczas operacji krzyżowania operowałem na całych wektorach o długości 6x3b = 18b.

W programie populacja jest reprezentowana w postaci tablicy 3D o wymiarach:
liczba_osobników x wymiar_zadania x liczba_bitów_na_liczbę.

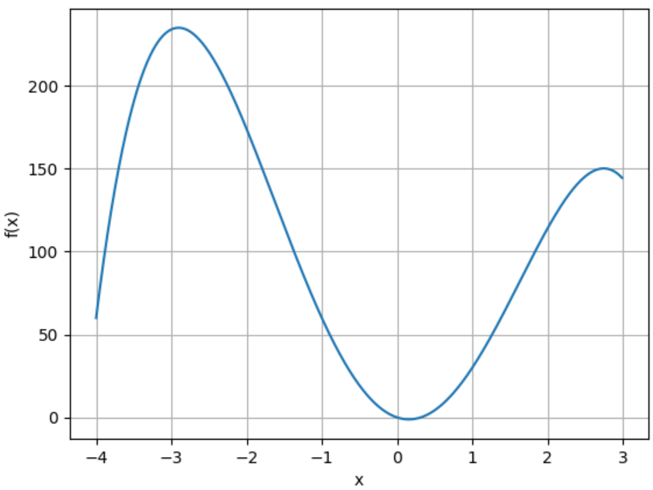
Populację początkową generuję w sposób losowy dla zadanej liczby osobników.

2. Początkowe wartości parametrów algorytmu

Na początku przyjąłem bardzo małe prawdopodobieństwo mutacji (0,01) oraz duże prawdopodobieństwo krzyżowania (0,7). Ustaliłem rozmiar populacji na 100 oraz maksymalną liczbę iteracji na 1000. Wszystkie parametry początkowe są zebrane w poniższej tabeli:

$\lvert P_0 \rvert$	100
μ	100
p_m	0,01
p_c	0,7
t_{max}	1000

Tak dobrane parametry zapewniły satysfakcjonujące działanie algorytmu. Metoda zwróciła w wyniku wektor [[1 0 1] [1 0 1] [1 0 1] [1 0 1] [1 0 1] [1 0 1]], co odpowiada punktowi x_0=[-3, -3, -3, -3, -3, -3] i wartości funkcji f(x)=234. Jest to największa możliwa do osiągnięcia wartość funkcji. Warto zwrócić uwagę, że wszystkie wartości x_i ustabilizowały się na tej samej wartości. Sprawdziłem, jak wyglądałaby funkcja f(x), gdyby była ona jednowymiarowa, czyli przy założeniu $\mathrm{x}_0 = \mathrm{x}_1 = \ldots = \mathrm{x}_5$ (rys. 1). I rzeczywiście w okolicy $x_i = -3$ taka funkcja osiąga maksimum globalne.



Rys. 1. Wykres 2D funkcji f(x) przy założeniu $\mathrm{x}_0 = \mathrm{x}_1 = \ldots = \mathrm{x}_5$ i dla $\mathrm{x}_i \in \mathrm{R}$

3. Badanie wpływu parametru p_m na działanie algorytmu

Ze względu na to, że przyjęte wartości parametrów okazały się skuteczne i nie wymagały dostrajania, to w dalszej części badań skupiłem się tylko na badaniu wpływu parametru p_m. Stopniowo zwiększałem wartość p_m i sprawdzałem, jakie rozwiązania zwracał algorytm genetyczny.

p_m	x	f(x)
0,01	[-3, -3, -3, -3, -3, -3]	234
0,1	[-3, -3, -3, -3, -3, -3]	234
0,2	[-3, -3, -3, -3, -3, -3]	234
0,3	[-3, -3, -3, -3, **-2**, -3]	**224**
0,4	[-3, -3, -3, -3, -3, -3]	234
0,5	[-3, -3, -3, **-2**, -3, -3]	**224**
0,6	[-3, -3, -3, -3, -3, -3]	234
0,7	[-3, -3, -3, -3, -3, -3]	234
0,8	[-3, -3, -3, -3, -3, -3]	234
0,9	[-3, -3, -3, -3, -3, -3]	234
1	[-3, -3, -3, **2**, -3, -3]	**214**

Okazało się, że nawet przy bardzo dużych wartościach p_m, algorytm był w stanie znaleźć najlepsze lub prawie najlepsze rozwiązanie. Jedynie w trzech przypadkach pojedyncze zmienne x_i miały inne wartości niż -3, co przełożyło się na mniejsze wartości funkcji celu.

W następnym kroku zmniejszyłem zarówno rozmiar populacji początkowej, jak i docelowej: $|P_0| = \mu = 10$. I ponownie zacząłem manipulować parametrem p_m.

p_m	x	f(x)
0,01	[-3, -3, -3, **-2**, **-2**, -3]	**214**
0,1	[**-2**, -3, -3, -3, -3, -3]	**224**
0,2	[-3, -3, -3, -3, -3, **-2**]	**224**
0,3	[**3**, -3, -3, -3, -3, -3]	**219**
0,4	[-3, -3, -3, **-2**, **-2**, -3]	**214**
0,5	[**-2**, -3, **-2**, -3, -3, -3]	**214**
0,6	[-3, **2**, -3, -3, -3, -3]	**214**
0,7	[-3, -3, -3, **-2**, **3**, -3]	**209**
0,8	[-3, -3, -3, -3, **-2**, -3]	**224**
0,9	[**3**, -3, -3, -3, -3, -3]	**219**
1	[-3, **-4**, -3, **-1**, **-4**, -3]	**147**

Tym razem w żadnym przypadku wyznaczone wartości funkcji nie są równe możliwej wartości maksymalnej wynoszącej 234. Jednak wyniki uzyskane dla różnych wartości p_m są do siebie dosyć zbliżone – najczęściej tylko jedna lub dwie wartości x_i nie osiągały najlepszej wartości -3.

W celu sformułowania ogólniejszych wniosków przeprowadziłem też po kilka testów dla każdej wartości parametru p_m i sprawdziłem, w ilu przypadkach udało się wyznaczyć najlepszą wartość funkcji celu równą 234. Dalej pozostałem przy wartości: $|P_0| = \mu = 10$. Wyniki są zebrane w poniższej tabeli.

p_m	Liczba przeprowadzonych testów (ogółem)	Liczba testów, dla których udało się wyznaczyć maksimum funkcji f(x)=234
0	5	0
0,001	5	0
0,005	5	0
0,01	5	3
0,025	5	5
0,05	5	5
0,075	5	2
0,1	5	1
0,2	5	0
0,3	5	0

Okazuje się, że dobranie odpowiedniej wartości parametru p_m jest bardzo ważne. Dla zbyt małych wartości ($p_m < 0{,}01$) mutacje występowały bardzo rzadko, tak że jedyną siłą napędową algorytmu i powodującą różnicowanie uzyskiwanych osobników było krzyżowanie. W takich przypadkach algorytm nie był w stanie znaleźć najlepszego rozwiązania. Z kolei zbyt duże wartości ($p_m > 0{,}075$) powodowały, że mutacje występowały już zbyt często i jakość znajdowanych rozwiązań ponownie zaczęła się pogarszać. Najlepsze wyniki i bardzo skuteczne działanie algorytmu uzyskałem dla wartości $p_m \in \{0{,}025;\ 0{,}05\}$.

4. Podsumowanie

Mutacja jest istotnym elementem algorytmu genetycznego. Jej wpływ nie może być zbyt duży, ponieważ wyznaczane osobniki będą wtedy za bardzo różnić się od swoich rodziców i jakość wyznaczanych rozwiązań może nie być satysfakcjonująca. Jednak nie można też całkowicie pominąć wpływu mutacji, ponieważ jej wpływ jest potrzebny do wprowadzania losowych zmian w osobnikach i może to przyspieszyć proces znajdywania najlepszego rozwiązania.