dr inż. Mateusz Gorczyca

Algorytmy genetyczne i sztuczne sieci neuronowe

Projekt 1

Projekt polega na napisaniu w języku Python algorytmu genetycznego do rozwiązania problemu plecakowego [1]. Projekt można wykonać samodzielnie lub w grupie dwuosobowej.

Dane wejściowe problemu to:

n – liczba przedmiotów,

$p_1, p_2, \ldots, p_n$ - wartość przedmiotów,

$w_1, w_2, \ldots, w_n$ - waga przedmiotów,

C – pojemność plecaka.

Zadanie 1. Korzystając z reprezentacji binarnej napisz algorytm genetyczny dla problemu plecakowego z opisaną poniżej funkcją celu. Można w tym celu wykorzystać fragmenty algorytmu z poprzedniej listy zadań.

Funkcja celu – należy wykorzystać wartość przedmiotów w plecaku, od której odejmowana jest kara za przekroczenie pojemności plecaka. Oznaczając:

$\mathbf{x}$ - chromosom reprezentujący rozwiązanie (podzbiór przedmiotów w plecaku),

$f(\mathbf{x})$ - wartość przedmiotów w plecaku,

$Pen(\mathbf{x})$ - kara za przekroczenie pojemności plecaka,

za **funkcję celu** obieramy wartość:

$$f(\mathbf{x}) - Pen(\mathbf{x}),$$

którą należy zmaksymalizować.

Przykład:

Liczba przedmiotów = 5,

wartości przedmiotów: $p_1, p_2, \ldots, p_5$ = 5, 3, 1, 4, 2,

wagi przedmiotów: $w_1, w_2, \ldots, w_n$ = 1, 2, 3, 4, 5,

pojemność plecaka C = 10.

Chromosom: $\mathbf{x} =< 0, 1, 0, 1, 1 >$,

wartość plecaka: $f(\mathbf{x}) = 0 \cdot 5 + 1 \cdot 3 + 0 \cdot 1 + 1 \cdot 4 + 1 \cdot 2 = 3 + 4 + 2 = 9$,

waga przedmiotów w plecaku $0 \cdot 1 + 1 \cdot 2 + 0 \cdot 3 + 1 \cdot 4 + 1 \cdot 5 = 2 + 4 + 5 = 11 > C$.

Kara za przekroczenie pojemności plecaka: $Pen(\boldsymbol{x}) = \rho V(\boldsymbol{x})$, gdzie

$V(\boldsymbol{x}) = max\left\{0, \sum_{i=1}^{n} x_i w_i - C\right\}$ oraz $\rho = max_{i=1,\ldots,n}\left\{\frac{p_i}{w_i}\right\}$.

W naszym przykładzie:

$\rho = max_{i=1,\ldots,5}\left\{\frac{5}{1}, \frac{3}{2}, \frac{1}{3}, \frac{4}{4}, \frac{2}{5}\right\} = 5$,

$V(\mathbf{x}) = \max\{0, 11 - 10\} = 1$,

$Pen(\mathbf{x}) = 5 \cdot 1 = 5$,

zatem funkcja celu ma wartość $f(\mathbf{x}) - Pen(\mathbf{x})$ = 9 – 5 = 4.

Uwaga: tak określona funkcja celu może przyjmować wartości ujemne.

Po zaimplementowaniu algorytmu przetestuj go na wybranych danych wejściowych dla następujących funkcji kar:

$Pen(\boldsymbol{x}) = \log_2(1 + \rho V(\boldsymbol{x}))$,

$Pen(\boldsymbol{x}) = \rho V(\boldsymbol{x})$,

$Pen(\boldsymbol{x}) = (\rho V(\boldsymbol{x}))^2$.

Można wykorzystać sposób generowania danych wejściowych z Zadania 4 lub wykonać testy w ramach Zadania 5.

Zadanie 2. Zrealizuj algorytm genetyczny, w którym jako funkcja celu występuje wartość przedmiotów w plecaku, ale dla (niektórych) rozwiązań przekraczających ograniczenie na pojemność plecaka stosowana jest procedura naprawy. Dla rozwiązania takiego procedura polega na usuwaniu z plecaka przedmiotów do momentu, w którym waga przedmiotów pozostałych w plecaku nie przestanie przekraczać ograniczenia. Należy zweryfikować 2 możliwe kolejności usuwania przedmiotów z plecaka:

- zgodnie z rosnącym stosunkiem wartości do wagi $\frac{p_i}{w_i}$,
- losową.

Procedurę naprawy rozwiązania niedopuszczalnego należy stosować z prawdopodobieństwem $p_r = 5\%$.

Po zaimplementowaniu algorytmu przetestuj go na wybranych danych wejściowych dla obu kolejności usuwania przedmiotów z plecaka. Można wykorzystać sposób generowania danych z Zadania 4 lub wykonać testy w ramach zadania 5.

Zadanie 3 (*). Zrealizuj algorytm genetyczny (ewolucyjny) z opisanym poniżej kodowaniem za pomocą permutacji, które nie dopuszcza rozwiązań nieprawidłowych (przekraczających pojemność plecaka). Przy chromosomach opisanych permutacją konieczne jest napisanie funkcji:

- tworzących losowe permutacje do populacji początkowej,
- krzyżujących permutacje (np. krzyżowanie PMX),
- mutujących permutację (np. przez zamianę dwóch losowo wybranych pozycji),
- dekodujących permutację zgodnie z poniższym opisem.

Permutacja reprezentuje kolejność wkładania przedmiotów do plecaka z listy przedmiotów. Algorytm dekoduje permutację dodając po kolei do plecaka przedmioty z permutacji w taki sposób, że element na *i*-tej pozycji permutacji jest indeksem przedmiotu na liście przedmiotów. Dany przedmiot jest jednak dodawany jedynie wtedy, kiedy suma jego wagi i wagi przedmiotów już włożonych do plecaka nie przekraczają pojemności plecaka. Przykład zamieszczono poniżej.

Przykład kodowania rozwiązania za pomocą permutacji. Dla wag przedmiotów = 2,5,2,3,5, wartości przedmiotów = 5,4,7,1,5, pojemności plecaka $C = 13$, listy przedmiotów L = a,b,c,d,e oraz rozwiązania (permutacji) $\boldsymbol{x}$ = 2,3,4,5,1, przedmioty umieszczane w plecaku to kolejno:

- *b* ($x_1 = 2$, drugi na liście L jest przedmiot *b*, jego waga to 5),
- *c* ($x_2 = 3$, trzeci na liście L jest przedmiot *c*, suma jego wagi i wagi przedmiotów w plecaku to 7),
- *d* ($x_3 = 4$, czwarty na liście L jest przedmiot *d*, suma jego wagi i wagi przedmiotów w plecaku to 10),
- do plecaka nie mieści się już przedmiot *e* (10+5 > $C = 13$),
- *a* ($x_5 = 1$, pierwszy na liście L jest przedmiot a, suma jego wagi i wagi przedmiotów w plecaku to 12).

Wartością plecaka jest całkowita wartość przedmiotów *a*,*b*,*c*,*d* czyli 17.

Dla realizacji z kodowaniem za pomocą permutacji należy zastosować listę przedmiotów L, na której przedmioty są umieszczone w kolejności:

- malejącego stosunku wartości do wagi $\frac{p_i}{w_i}$,
- losowej.

Po zaimplementowaniu algorytmu przetestuj go na wybranych danych wejściowych dla obu kolejności przedmiotów na liście L. Można wykorzystać sposób generowania danych z Zadania 4 lub wykonać testy w ramach zadania 5.

Zadanie 4. (*) Wygenerowanie danych wejściowych do Zadania 5.

Wartość i wagę przedmiotów należy wygenerować losowo na 3 sposoby:

1. Wagi i wartości nieskorelowane:
   - w_i losowane z przedziału [1, v] z rozkładem jednostajnym,
   - p_i losowane z przedziału [1, v] z rozkładem jednostajnym,
2. Wagi i wartości słabo skorelowane:
   - w_i losowane z przedziału [1, v] z rozkładem jednostajnym,
   - $p_i = w_i$ + wartość losowana z przedziału [-r, r] z rozkładem jednostajnym (losowana dopóki p_i nie będzie dodatnie),
3. Wagi i wartości skorelowane
   - w_i losowane z przedziału [1, v] z rozkładem jednostajnym,
   - $p_i = w_i + r$

Należy przyjąć v = 10 oraz r = 5. Liczba przedmiotów do uwzględnienia: n = 100, 250 oraz 500. Dodatkowo należy także uwzględnić dwie wartości ograniczenia plecaka:

- $C_1 = 2v$ (ograniczona pojemność)
- $C_2 = 0{,}5 \sum_{i=1}^{n} w_i$ (średnia pojemność).

Dla każdej z 18 kombinacji parametrów danych wejściowych (3 wartości n × dwa sposoby generowania C × 3 sposoby generowania wag i wartości przedmiotów) należy wygenerować 25 zestawów losowych danych wejściowych.

Zadanie 5. (*) Dla każdej z 18 kombinacji parametrów danych wejściowych z Zadania 4 należy wygenerować 25 losowych zbiorów danych. Dla każdego z tych zbiorów należy przetestować każdy z 7 wyżej wymienionych realizacji algorytmu genetycznego:

- 3 dla różnych funkcji kar,
- 2 dla różnych kolejności usuwania przedmiotów w procedurze naprawy
- 2 dla różnej kolejności przedmiotów na liście w kodowaniu za pomocą permutacji.

Zadanie dodatkowe (nieobowiązkowe): sprawdzić, jak skuteczny jest algorytm z procedurą naprawy dla wartości p_r innych niż 5%.

Sprawozdanie z wszystkich rozwiązanych zadań. Należy napisać sprawozdanie z projektu opisujące:

- sposób realizacji algorytmów,

- wyniki eksperymentów (dla Zadania 5 - średnie wartości dla każdego z 18 zbiorów danych i 7 algorytmów genetycznych, o ile oczywiście algorytm potrafił znaleźć jakiekolwiek rozwiązanie dopuszczalne dla danego zbioru danych),
- wnioski.

Format sprawozdania: ipynb lub pdf.

Literatura:

[1] Z. Michalewicz, Algorytmy genetyczne + struktury danych = programy ewolucyjne, Wydawnictwa Naukowo-Techniczne, 2003.

[2] L. Scrucca, GA: A Package for Genetic Algorithms in R, Journal of Statistical Software, Vol. 53 (1), 2013.

[3] http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsplib.html