Igor Kraszewski

310164

Wprowadzenie do sztucznej inteligencji – ćwiczenie 1, spotkanie 1

Zadanie na pierwsze spotkanie (za 3 punkty)

Mamy problem plecakowy jak na wykładzie:

```
w = np.array([8, 3, 5, 2]) #waga przedmiotów
W = 9 #maksymalna waga plecaka
p = np.array([16, 8, 9, 6]) #wartość przedmiotów
```

Zadania:

1. Znaleźć rozwiązanie optymalne przez przegląd wyczerpujący.
2. Rozwiązać problem przy użyciu heurystyki: do plecaka pakujemy przedmioty według kolejności wynikającej ze stosunku p/w. Uwaga: heurystyka to nie funkcja heurystyczna. Nie używamy tu jeszcze funkcji heurystycznej i algorytmu A*.

Pytania (odpowiedzi proszę umieścić w pliku tekstowym):

- Jakie rozwiązania i jaką wartość funkcji oceny uzyskano? Czy uzyskano takie same rozwiązania?
- Jak dużą instancję problemu (liczba przedmiotów) da się rozwiązać w około minutę metodą przeglądu wyczerpującego?
- Jak dużą instancję problemu (liczba przedmiotów) da się rozwiązać w około minutę metodą zachłanną (używając heurystyki)?
- Jak bardzo wydłuży obliczenia dodanie jeszcze jednego przedmiotu?
- Jakie wnioski można wyciągnąć na podstawie wyników tego ćwiczenia?

Implementacja zadania pierwszego znajduje się w pliku „zad1.py”, zadania drugiego w pliku „zad2.py”, natomiast ich wykonanie w pliku „main.py”. Dodatkowo dane do zadanie umieściłem w oddzielnym pliku „env.py”. Test czasu wykonania został zaimplementowany w pliku „time_test.py”.

1. Znaleźć rozwiązanie optymalne przez przegląd wyczerpujący.

W celu wykonania zadania stworzyłem klasę **KnapsackProblemExeu**, która oprócz konstruktora zawiera 3 metody: *get_combinations, exaustive_approach* oraz *__str__*. Pierwsza z wymieniowych metod służy do stworzenia wszystkich możliwych kombinacji,

o różnej ilości elementów, par (wartość, waga), w celu późniejszego sprawdzenia każdej z nich podczas poszukiwania optymalnego rozwiązania. Funkcja ta wykorzystuje metodą *combinations* z biblioteki *itertools*. Metoda *exaustive_approach* jako argument przyjmuje wyliczone kombinacje i przechodząc po każdej z nich, wylicza dla niej sumaryczną wartość i wagę przedmiotów. W każdej iteracji może nastąpić aktualizacja rozwiązania, jeżeli wartość była wyższa od poprzedniej, a waga nie przekroczyła limitu lub przejście do kolejnej kombinacji, jeżeli wartość okazałą się mniejsza lub została przekroczona maksymalna waga przedmiotów. Ostatnia metoda służy do przedstawienia tekstowego obiektu poprzez podanie otrzymanego rozwiązania optymalnego oraz wartości i wagi sumarycznej przedmiotów.

2. Rozwiązać problem przy użyciu heurystyki: do plecaka pakujemy przedmioty według kolejności wynikającej ze stosunku p/w. Uwaga: heurystyka to nie funkcja heurystyczna. Nie używamy tu jeszcze funkcji heurystycznej i algorytmu A*.

W celu rozwiązania problemu metodą z zadania drugiego, zaimplementowałem klasę **KnapsacProblemHeuristic,** która również implementuje 3 metody: *sort_by_coeff, heuristic_approach* oraz *__str__*. Pierwsza z wymienionych metod sortuje przedmioty malejąco względem współczynnika jakim jest stosunek wartości przedmiotu do jego wagi, po czym zwraca listę podającą indeksy z oryginalnej listy przedmiotów według wyliczonej kolejności. Metoda *heuristic_approach* wybiera kolejne elementy z posortowanej listy indeksów, zlicza ich wartość i masę aż nie przekroczy limitu wagowego. Ostania metoda, analogicznie jak w omawianej wcześniej klasie, zwraca słowny opis otrzymanego rozwiązania. W pliku „main.py” tworzone są obiekty opisanych klas i wywołane odpowiednie metody w celu ukazania wyników w terminalu.

- Jakie rozwiązania i jaką wartość funkcji oceny uzyskano? Czy uzyskano takie same rozwiązania?

```
Exercise 1
Optimal solution are these (value, weight) pairs: (8, 3) (9, 5) with value: 17 and weight: 8

Exercise 2
Solution are these (value, weight) pairs: (6, 2) (8, 3) with value: 14 and weight: 5
```

Dla przeszukiwania wyczerpującego otrzymana wartość to 17, co jest rozwiązaniem optymalnym. W przypadku podejścia przy użyciu heurystyki otrzymujemy rozwiązanie z wartością 14, nie jest to rozwiązanie optymalne, ale za to uzyskane szybciej.

- Jak dużą instancję problemu (liczba przedmiotów) da się rozwiązać w około minutę metodą przeglądu wyczerpującego?

W celu sprawdzenia utworzyłem funkcje w pliku *'time_test.py'* – *test_one_minute,* która sprawdza czas wykonania przeszukiwania wyczerpującego na określonym przedziale między minimalną, a maksymalną podaną ilością przedmiotów, dla których wagi, wartości i limit wagowy plecaka są dobierane losowo. Funkcja przerywa się, gdy czas wykonania dla danej ilości przedmiotów był dłuższy niż 50 sekund. Ze względu na to, że rozwiązanie używa liczb pseudolosowych, funkcja powtarzana jest 25 razy, a następnie wyliczam średnią z wyników. W ten sposób otrzymałem, że w około minutę (średnio 52,27 sekundy) da się rozwiązać problem z 25 przedmiotami w plecaku.

```
52.26973085403442 25.0
```

- Jak dużą instancję problemu (liczba przedmiotów) da się rozwiązać w około minutę metodą zachłanną (używając heurystyki)?

W celu sprawdzenia wielkości instancji problemu, stworzyłem analogiczną funkcję jak dla przeszukiwania wyczerpującego. W tym przypadku otrzymane wyniki są dużo mniej dokładne, ponieważ dla 33500000 przedmiotów otrzymujemy średnio 64,08 sekundy czasu wykonania. Dalszą próba przybliżania się do minuty uważam za zbędną, ponieważ już ten wynik pokazuje jak dużo szybsza jest ta metoda od metody przeszukiwania wyczerpującego.

```
64.077906s 33500000.0
```

- Jak bardzo wydłuży obliczenia dodanie jeszcze jednego przedmiotu?

Jako że złożoność obliczeniowa dla całego algorytmu jest stała, zależna od ilości przedmiotów w plecaku, sprawdzenia dokonałem na mniejszych ilościach przedmiotów, aby otrzymać wynik szybciej. Funkcja *test_ratio* podobnie jak opisana wyżej funkcja liczy czas wykonania zadania, jednak nie przerywa się po wykonaniu trwającym przynajmniej 50 sekund oraz wywołałem ją na szerszym przedziale. Dodatkowo liczy ona na podstawie otrzymanych czasów w każdej iteracji dotyczącej ilości przedmiotów stosunek czasów wykonania przy jednym dodatkowym przedmiocie. Po wykonaniu 25 razy (ze względu na liczby pseudolosowe) i wyciągnięciu średniej, otrzymałem, że dodanie jednego dodatkowego przedmiotu w przypadku metody wyczerpującej wydłuża czas wykonania zadania o około 2,1 razy.

```
2.1190118159825304
```

Wyliczenia dla heurystyki pominąłem patrząc na ogromność ilości przedmiotów względem czasu podczas ustalania instancji problemu dla minuty wykonywania. Różnica byłaby na tyle mała, że nawet uśredniając wyniki z 25 wykonań, otrzymano by różne wyniki.

- Jakie wnioski można wyciągnąć na podstawie wyników tego ćwiczenia?

Na podstawie wyników tego ćwiczenia można stwierdzić, że za pomocą przeszukiwania wyczerpującego zawsze jesteśmy w stanie otrzymać rozwiązanie optymalne, a w przypadku użycia heurystyki rozwiązanie nie zawsze musi być optymalnym. Jednak czas wykonywania problemu plecakowego metodą przeszukiwania wyczerpującego rośnie ponad dwukrotnie za każdy dodatkowy przedmiot, a rozwiązanie wykorzystujące heurystykę jest bardzo szybkie.
Kiedy nasz problem jest duży mając do wyboru tylko te dwa rozwiązania, powinniśmy skorzystać z heurystyki, ponieważ mimo, że otrzymanie rozwiązania optymalnego nie jest pewne, wynik otrzymamy w zdecydowanie szybszym czasie.