Opracowanie problemu plecakowego

Szymon Kaczmarek, nr indeksu 148056

1 Wprowadzenie

Niniejsze opracowanie ma na celu przedstawienie trzech sposobów rozwiązania binarnego problemu plecakowego: siłowy, dynamiczny i rekurencyjny. Wszystkie algorytmy i programy zostały napisane w Pythonie (repozytorium GitHub).

2 Problem plecakowy

Problem plecakowy polega na znalezieniu wypełnienia plecaka, które daje największą wartość spakowanych przedmiotów mając podaną listę przedmiotów z nadaną wagą i wartością. Wersja binarna tego problemu zakłada, że jest dostępny tylko jeden egzemplarz danej rzeczy więc istnieją tylko dwie możliwosci podczas podejmowania decyzji: spakować albo nie spakować. W opisach algorytmów n oznacza liczbę przedmiotów, a m pojemność plecaka.

2.1 Algorytm siłowy

Ze względu na binarną cechę decyzji odnośnie zawierania danego przedmiotu w odpowiedzi rozwiązanie takie można przedstawić za pomocą liczby binarnej gdzie cyfra na n-tej pozycji odpowiada na pytanie czy n-ty przedmiot jest częścią wyniku. Największą taką liczbą byłaby wtedy $2^n - 1$ odpowiadająca spakowaniu wszystkich przedmiotów, a najmniejszą byłoby 0. Algorytm siłowy polega na wygenerowaniu wszystkich możliwych odpowiedzi i wybraniu tej optymalnej. Wystarczy więc przeiterować po wszystkich wartościach od 0 do $2^n - 1$ i sprawdzić czy odpowiadające danej liczbie rozwiązanie jest tym poszukiwanym. Z tego powodu algorytm ten posiada złożoność $O(2^n * n)$. Warto zauwarzyć, że złożoność ta nie zależy od zadanej pojemności plecaka.

2.2 Algorytm rekurencyjny

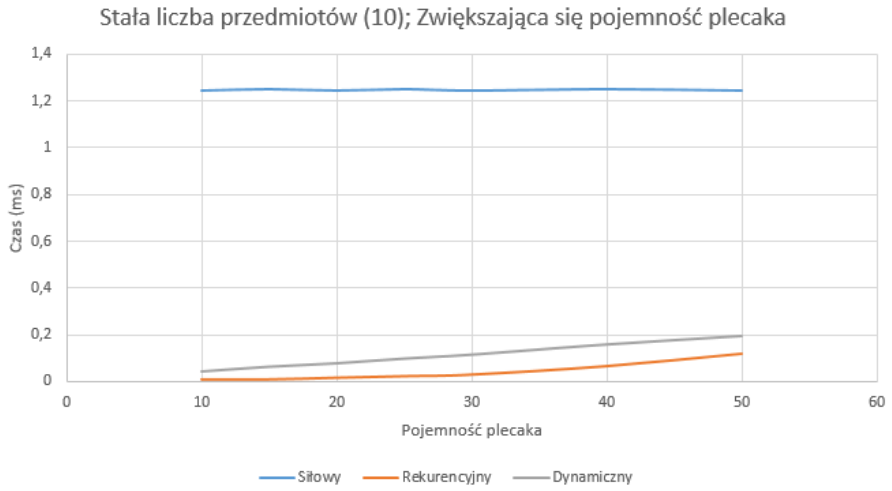
Ideą tego algorytmu jest fakt, że decyzja na dodanie danego elementu do rozwiązania sprowadza się do sprawdzenia czy po jego dodaniu otrzymamy lepsze rozwiązanie niżeli byśmy tego nie robili. Można dlatego rekurencyjnie zadawać takie pytanie dla każdego kolejnego przedmiotu, który chcemy dodać. Przez fakt rozdzielania się na liczenie dwóch rozwiązań dla kolejnych przedmiotów algorytm ten posiada złożoność $O(2^n)$, prawie niezależną od pojemności plecaka.

Algorytm będzie działał szybciej dla mniejszych pojemności plecaka, ponieważ niektóre gałęzie rekurencji zostaną odcięte przez fakt niemieszczenia się przedmiotów w plecaku albo szybkim wypełnieniu go małą liczbą przedmiotów. To będzie można zauważyć na wykresach czasowych, które, z uwagi na okropną szybkość algorytmu siłowego, zostały przeprowadzone dla mniejszych danych. Piętą achillesową tego algorytmu jest fakt powielanego liczenia funkcji rekurencyjnej dla niektórych wartości.

2.3 Algorytm dynamiczny

Algorytm ten polega na stworzeniu tablicy $n \times m$ i wypełnieniu jej optymalnymi rozwiązaniami. Komórka $Tab_{i,j}$ zawiera wartość spakowanych rzeczy w optymalnym rozwiązaniu dla i dostępnych przedmiotów oraz j pojemności plecaka. Wypełnianie tablicy polega na zadaniu pytania: czy po dodaniu danego elementu otrzymamy rozwiązanie lepsze niż aktualne. Zapytanie jest praktycznie takie same jak w algorytmie rekurencyjnym, lecz tu odwołujemy się do już wypełnionych komórek w tablicy. Złożoność algorytmu to $O(n * m)$ ponieważ polega on jedynie na wypełnieniu tablicy.

3 Czasy działania



Jak widać, dla stałej liczby przedmiotów, złożoność algorytmu siłowego jest stała, a dynamicznego liniowa. Tak jak wyżej jest powiedziane czas algorytmu rekurencyjnego jest rosnący, lecz dla większych pojemności plecaka czasy są stałe (z testów wynika, że dla tych danych ten czas wynosi $0.46ms$).

Na powyższych wykresach widać różnicę między złożonościami algorytmów. Jak można zauważyć algorytm siłowy wystrzeliwuje już przy 25 przedmiotach, gdzie rekurencyjny dopiero zaczyna. Spowodowane jest to tym, że algorytm siłowy dla każdego rozwiązania musi policzyć jego wartość i wagę w czasie $O(n)$, a rekurencyjny nie. Algorytm dynamiczny dla stałego jednego z parametrów znowu przyjmuje złożoność liniową.

4 Wnioski

Na podstawie powyższych danych można powiedzieć, że algorytm siłowy jest nie efektywny dla większości przypadków i należy go unikać. Jedyna sytuacja, która mogłaby usprawiedliwić jego wykorzystanie nad algorytmem dynamicz-

nym jest taka, gdy posiadamy małą ilość przedmiotów oraz pojemność plecaka jest bardzo duża. Jednakże, algorytm rekurencyjny lepiej by sobie poradził od rozwiązania siłowego we wszystkich przypadkach. Rekurencja jest również szybsza od algorytmu decyzyjnego w przypadkach gdy plecak jest szybko zapełniany małą ilością przedmiotów oraz gdy sama ilość przedmiotów jest bardzo mała. W pozostałych przypadkach najlepiej jest wykorzystywać algorytm dynamiczny. Można jescze wspomnieć o algorytmie rekurencyjnym z memoizacją, który teoretycznie powinien być szybszy od dynamicznego ponieważ taki algorytm by obliczał tylko te wartości w tablicy, które doprowadzałyby do rozwiązania. Różnica w podjęciu decyzji o wykorzystaniu takiego algorytmu sprowadzałaby się do potrzeby późniejszego wykorzystania tak wygenerowanej tablicy.

5 Zakończenie

W tym opracowaniu został przedstawiony binarny problem plecakowy i trzy sposoby jego rozwiązania. Zostało przygotowane zestawienie i wnioski z niego wynikające.

Spis treści