Informatyczny kącik olimpijski (84): Sklep ze słodyczami

Tym razem zadanie *Candies* z Bałtyckiej Olimpiady Informatycznej z 2010 roku. W sklepie znajduje się n paczek z cukierkami, i-ta z nich zawiera b_i cukierków. Sklepikarz może sprzedawać jedynie całe paczki, więc jeśli w sklepie są cztery paczki zawierające 1, 3, 4 i 4 cukierki, to może on obsłużyć następnego klienta tylko wtedy, gdy ten złoży zamówienie jednego z 9 rozmiarów (konkretnie gdy poprosi o 1, 3, 4, 5, 7, 8, 9, 11 lub 12 cukierków). Sklepikarz zastanawia się, jak bardzo mógłby zwiększyć liczbę możliwości, gdyby zamienił liczbę cukierków w jednej z paczek. Przykładowo, po wyrzuceniu jednej paczki z 4 cukierkami i dodaniu paczki z 9 cukierkami, liczba możliwych do zrealizowania zamówień zwiększy się do 13 (będą to 1, 3, 4, 5, 7, 8, 9, 10, 12, 13, 14, 16, 17). Jeśli istnieje więcej niż jedno optymalne rozwiązanie, w którym sklepikarz zamienia paczkę z p cukierkami na paczkę z q cukierkami, należy podać to, w którym para (p, q) jest minimalna leksykograficznie.

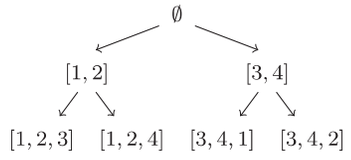
Załóżmy na początek, że wyrzuciliśmy którąś z paczek, a zestaw pozostałych $n-1$ paczek umożliwia zrealizowanie zamówień o wielkościach $x_0, x_1, \ldots, x_k$ (zakładamy przy tym, że zamówienie $x_0 = 0$ również jest poprawne). Jeśli dodamy do tego zestawu paczkę q cukierków, to będziemy mogli dodatkowo zrealizować zamówienia $q, x_1 + q, \ldots, x_k + q$, a zatem jeśli q nie jest równe żadnej różnicy $x_i - x_j$, to liczba zamówień wzrośnie dwukrotnie. Możemy to sobie zagwarantować, dodając wystarczająco dużą paczkę, np. rozmiaru $q = b_1 + \ldots + b_n + 1$. Rozwiązanie zadania można zatem podzielić na dwie części: znalezienie paczki p, której wyrzucenie jak najmniej zmniejszy liczbę możliwych zamówień, oraz znalezienie jak najmniejszej paczki q, która spowoduje dwukrotne zwiększenie liczby możliwych zamówień.

Zadanie to jest pewną wariacją na temat problemu plecakowego (o którym pisaliśmy m.in. w numerze 6/2013). W szczególności wyznaczenie liczby zamówień możliwych do zrealizowania przez początkowy zestaw paczek to zastosowanie tego algorytmu do n przedmiotów rozmiarów $b_1, \ldots, b_n$ i plecaka o udźwigu $m = b_1 + \ldots + b_n$. Pseudokod algorytmu znajduje się na marginesie. Przypomnijmy istotne dla nas kwestie: czas działania algorytmu wynosi $O(nm)$, analizuje on przedmioty pojedynczo (w czasie $O(m)$ każdy), aktualizując tablicę $d[0..m]$ (w wynikowej tablicy mamy $d[j] = 1$, jeśli zamówienie rozmiaru j może być zrealizowane), oraz dla ostatecznego wyniku nie ma znaczenia kolejność, w której przedmioty są analizowane. Możemy abstrahować od szczegółów implementacyjnych i zamknąć go w strukturze danych, która w czasie $O(m)$ będzie umożliwiała wykonywanie trzech operacji: zrobienie kopii struktury, dodanie nowego przedmiotu oraz wyznaczenie liczby zamówień realizowanych przez dodane przedmioty.

```
d[0] := 1;
for i := 1 to n do
  for j := m downto b_i do
    if d[j − b_i] = 1 then d[j] := 1;
```

Aby znaleźć paczkę p, potrzebujemy rozwiązać problem plecakowy dla każdego podzbioru $n-1$ przedmiotów. Jeśli rozpatrzymy każdy z tych podzbiorów niezależnie, dostaniemy algorytm o złożoności czasowej $O(n^2 m)$.

Możemy postąpić nieco sprytniej. Będziemy utrzymywać listę struktur danych (na początek zaczynamy z jedną pustą strukturą danych). Teraz, dopóki struktury na liście zawierają mniej niż $n-1$ przedmiotów, to dla każdej struktury B dzielimy zbiór przedmiotów S *niewystępujących* w tej strukturze na dwa w miarę równe podzbiory S_1 i S_2, a strukturę B zastępujemy jej kopiami B_1 i B_2, do których dodajemy przedmioty odpowiednio z podzbiorów S_1 i S_2 (patrz rysunek). Jeśli dla uproszczenia założymy, że n jest potęgą dwójki, to w i-tej fazie tego algorytmu wygenerujemy 2^i struktur, do każdej z nich dodając $n/2^i$ przedmiotów. Zatem złożoność czasowa każdej z $\log_2 n$ faz wyniesie $O(nm)$, czyli rozwiązanie problemu plecakowego dla wszystkich podzbiorów $n-1$ przedmiotów zajmie czas $O(nm \log n)$.



Ilustracja algorytmu generującego struktury danych opisujące podzbiory o $n-1$ elementach dla $n = 4$ przedmiotów.

Pozostaje teraz znalezienie minimalnej liczby q, która nie będzie równa żadnej różnicy $x_i - x_j$. Ponieważ x_i są zamówieniami generowanymi przez wybrany podzbiór $b'_1, \ldots, b'_{n-1}$ paczek, to wszystkie różnice będą generowane przez zbiór $b'_1, -b'_1, \ldots, b'_{n-1}, -b'_{n-1}$. Wystarczy zatem rozwiązać problem plecakowy dla takich przedmiotów (w naszej wersji tego problemu przedmioty o ujemnym rozmiarze nie stanowią problemu, należy jednak zwrócić uwagę na rozmiar tablicy i poprawną kolejność pętli) i zwrócić jako q minimalną dodatnią liczbę, dla której $d[j] = 0$. Czas działania tej części rozwiązania to $O(nm)$.

```
d[0] := 1;
for i := 1 to n − 1 do
  for j := m downto −m + b'_i do
    if d[j − b'_i] = 1 then d[j] := 1;
  for j := −m to m − b'_i do
    if d[j + b'_i] = 1 then d[j] := 1;
```

Tomasz IDZIASZEK