Informatyczny kącik olimpijski (63): Plecak

Jednym z klasycznych problemów algorytmicznych jest tzw. problem plecakowy przedstawiany m.in. w następującej wersji: mamy n przedmiotów, ponumerowanych liczbami od 1 do n. Każdy przedmiot ma określoną masę – i-ty przedmiot waży $m[i]$ kilogramów. Mamy do dyspozycji plecak, do którego możemy zapakować przedmioty o łącznej masie co najwyżej M kilogramów. Chcemy dowiedzieć się, jaka jest największa masa przedmiotów, które możemy zapakować do plecaka, tak by nie przekroczyć jego udźwigu.

Rozwiązanie tego problemu korzysta z metody programowania dynamicznego. Mamy tablicę $d[0\,..\,M]$, na początku zainicjowaną wartościami $-\infty$. Kolejne przedmioty rozpatrujemy w kolejnych fazach algorytmu. Będziemy utrzymywali niezmiennik, że po i-tej fazie algorytmu $d[j] \neq -\infty$, jeśli spośród przedmiotów ze zbioru $\{1, \ldots, i\}$ możemy wybrać podzbiór P o łącznej masie j. Ponadto $d[j]$ będzie największym numerem przedmiotu, który można umieścić w pewnym P.

Rozwiązanie opiera się na spostrzeżeniu, że jeśli $i \in P$, to $P \setminus \{i\}$ musi być podzbiorem przedmiotów ze zbioru $\{1, \ldots, i-1\}$ o masie $j - m[i]$, czyli $d[j - m[i]] \neq -\infty$ po $(i-1)$-szej fazie algorytmu.

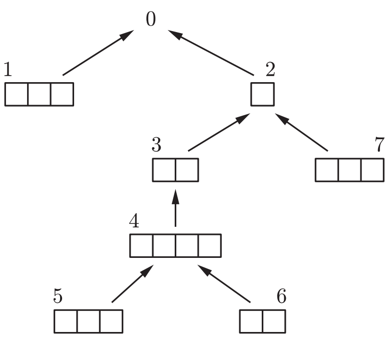
$d[0] := 0;$
for $i := 1$ **to** n **do**
  for $j := M$ **downto** $m[i]$ **do**
    if $d[j - m[i]] \neq -\infty$ **then** $d[j] := i;$

Warto zwrócić uwagę na malejącą kolejność, w jakiej iterujemy zmienną j w wewnętrznej pętli – iterowanie w kolejności rosnącej daje rozwiązanie innej wersji problemu plecakowego, w której mamy do dyspozycji dowolną liczbę przedmiotów każdego z rodzajów.

Odpowiedzią jest, oczywiście, największe j, takie że $d[j] \neq -\infty$. Nasze rozwiązanie działa w czasie $O(nM)$ i pamięci $O(n + M)$.

W praktyce jednak, wybierając się na wycieczkę, pakujemy do plecaka rzeczy, biorąc pod uwagę ich przydatność, a nie tylko to, jak imponująco będzie wyglądał nasz plecak. W szczególności niektóre z przedmiotów są bezużyteczne, jeśli w plecaku nie znajdą się z innymi, np. wzięcie na wakacje statywu nie jest zbyt mądre, jeśli nie weźmiemy aparatu fotograficznego. I właśnie z tą ciut praktyczniejszą wersją problemu plecakowego musieli zmierzyć się uczestnicy finału Potyczek Algorytmicznych 2012. Dla każdego przedmiotu i został określony inny przedmiot $p[i]$, bez którego przedmiot i będzie bezużyteczny (albo $p[i] = 0$, jeśli i jest przedmiotem przydatnym samym w sobie). Pytamy znów, jak ciężki plecak możemy zapakować, nie przekraczając jego udźwigu i nie zabierając żadnego bezużytecznego przedmiotu.

Zależności między przedmiotami możemy przedstawić jako drzewo: dla każdego przedmiotu i tworzymy krawędź skierowaną z węzła i do węzła $p[i]$ (patrz rysunek). Dodajemy także sztuczny przedmiot numer 0 o wadze $m[0] = 0$. Widzimy, że aby można było zapakować do plecaka przedmiot i, muszą tam znaleźć się wszystkie przedmioty na ścieżce od węzła i w górę drzewa.



Mamy $n = 7$ przedmiotów i plecak o udźwigu $M = 11$. W powyższym drzewie masa przedmiotu oznaczona jest liczbą kwadracików pod numerem przedmiotu, np. $m[1] = 3$, $p[1] = 0$, zaś $m[6] = 2$, $p[6] = 4$.

Wzięcie przedmiotów 3, 4, 5 i 6 całkowicie wypełnia plecak, ale jest niepoprawne, gdyż przedmiot 3 jest bezużyteczny bez przedmiotu 2. Optymalnym rozwiązaniem jest wzięcie przedmiotów 2, 3 i 4 oraz jednego z przedmiotów ze zbioru $\{1, 5, 7\}$, co daje sumaryczną masę 10 kilogramów.

Okazuje się, że problem można rozwiązać całkiem prosto, ale wymaga to pewnej dozy pomysłowości i na zawodach udało się to tylko dwóm z 20 zawodników. Będziemy przeglądać przedmioty w kolejności występowania ich w drzewie w porządku *preorder* (czyli w kolejności przechodzenia drzewa w głąb – dla ułatwienia przenumerujmy te przedmioty, jeśli ich kolejność jest inna). Rozważamy przedmiot i i pytamy się o warunek istnienia upakowania P o masie j przedmiotami ze zbioru $\{1, \ldots, i\}$, jeśli bierzemy przedmiot i, ale nie bierzemy żadnego bezużytecznego przedmiotu. Załóżmy, że istnieje poprawne upakowanie plecaka przedmiotami $P' \subseteq \{1, \ldots, i-1\}$ o masie $j - m[i]$ i że k jest największym możliwym numerem przedmiotu w P'. Zauważmy, że jeśli $k = p[i]$, to w oczywisty sposób $P = P' \cup \{i\}$ jest poprawnym upakowaniem. Podobnie jeśli $p[i] < k < i$ (czyli węzeł k znajduje się w poddrzewie zaczepionym w lewym bracie węzła i), to ponieważ P' zawiera wszystkie przedmioty na ścieżce od k do korzenia, więc zawiera również przedmiot $p[i]$, zatem znowu bierzemy $P = P' \cup \{i\}$. Jeśli natomiast $k < p[i]$, to oznacza, że żadne upakowanie o masie $j - m[i]$ nie może zawierać przedmiotu $p[i]$, czyli upakowanie P nie może istnieć.

Zaimplementowanie nowego rozwiązania wymaga jedynie kosmetycznej zmiany w poprzednim programie:

$d[0] := 0;$
for $i := 1$ **to** n **do**
  for $j := M$ **downto** $m[i]$ **do**
    if $d[j - m[i]] \geqslant p[i]$ **then** $d[j] := i;$

Tomasz IDZIASZEK