Problém zaplatenia nákupu (Dynamické programovanie)

Matej Perejda

Úloha: V peňaženke máme celkom N platidiel (mincí resp. bankoviek) s hodnotami $P[0]$, $P[1]$, ..., $P[N-1]$. Predpokladajme, že hodnoty platidiel sú celé čísla. Cena nákupu je C (opäť celé číslo).

Otázka: Vieme platidlami v peňaženke zaplatiť za nákup tak, aby nám nebol vrátený žiaden výdavok, t.j. vieme z platidiel v peňaženke presne vyskladať sumu C ? V prípade kladnej odpovede vypíšte, aké platidlá treba použiť.

Definícia: Dané je pole P, ktoré obsahuje N kladných celých čísel $P[0, ..., N-1]$ a cieľová hodnota C. Úlohou je zistiť, či existuje nejaké podpole poľa P, ktorého súčet prvkov je rovný vstupnej cieľovej hodnote C. V prípade, ak takéto podpole existuje, vypíšeme prvky nájdeného podpoľa.

> Príklad:
> Nech $P = \{3, 5, 2, 4, 8\}$ a cieľová hodnota $C = 13$. Existuje také podpole R poľa P, kde $R = \{5, 8\}$, ktorého súčet prvkov je rovný cieľovej hodnote C. Ak by sme chceli nájsť podpole, ktorého sučet prvkov je $C = 32$, tak takéto podpole neexistuje.

Riešenie: Túto úlohu je tiež možné riešiť pomocou rekurzie, ktorá má exponenciálnu zložitosť a zároveň nie je veľmi účinná. Pre efektívnosť je lepšie využiť metódu založenú na dynamickom programovaní.

Nech $D[i, j]$ je definovaná ako TRUE práve vtedy, ak existuje také podpole prvkov $P[0, ..., N-1]$, ktorých súčet je rovný i. Potom riešenie nášho problému je $D[C, N]$. Inak povedané:

$$D[C, N] = \textbf{TRUE} \Leftrightarrow \textbf{"sumu } C \textbf{ vieme vyplatiť pomocou } N \textbf{ platidiel"}$$

Nech $D[\,][\,]$ je dvojrozmerné pole (matica) údajového typu boolean o veľkosti $(C+1) \times (N+1)$. Hodnota TRUE na pozícii $D[i][j]$ znázorňuje, že existuje podpole poľa $P[0, ..., N-1]$ so súčtom i. Naším cieľom je nájsť hodnotu $D[C][N]$, ktorá ak bude rovná hodnote TRUE, sumu C je možné vyplatiť bezozvyšku. Aby sme sa ale k výsledku dopracovali musíme najprv vypočítať každú hodnotu poľa $D[\,][\,]$.

Inicializujeme nasledujúce vstupy ako triviálne prípady, ktoré budú slúžiť na postupné vyplnenie celého poľa $D[\,][\,]$.

$D[0][j] =$ TRUE $\mid 0 \leq j \leq N$, každé pole obsahuje podpole so súčtom prvkov $C = 0$

$D[i][0] =$ FALSE $\mid 0 \leq i \leq C$, nie je možné nájsť žiadne vyplatenie sumy $C > 0$ v prázdnom poli

Vo všeobecnosti plati, že $D[i][j] =$ TRUE práve vtedy a len vtedy, keď je aspoň jedna z nasledujúcich podmienok splnená:

1. D[i][j-1] = TRUE

Podmienka hovorí o tom, že sumu i možno vyskladať pomocou niektorých platidiel na indexoch $P[0, ..., j-1]$. Z toho vyplýva, že sumu i je možné vyskladať aj pomocou platidiel na indexoch $P[0, ..., j-1, j]$. Preto sú si obe podmienky ekvivalentné. $D[i][j] \Leftrightarrow D[i][j-1]$

2. D[i - P[j]][j-1] = TRUE

Znamená to, že sumu $i - P[j]$ je možné vyskladať pomocou niektorých platidiel na indexoch $P[0, ..., j-1]$.

Tieto dve dôležité podmienky zabezpečia výpočet všetkých hodnôt v poli D. Relácia vyzerá nasledovne:

$$\mathbf{D[i][j] \Leftrightarrow D[i][j\text{-}1] \vee D[i - P[j]][j\text{-}1]}$$

```java
 * Nájde vyplatenie.
public static List<Integer> daSaZaplatit(int suma, int[] platidla) {
    boolean[][] D = new boolean[suma + 1][platidla.length + 1];
    D[0][0] = true;

    for (int pocetPlatidiel = 1; pocetPlatidiel <= platidla.length; pocetPlatidiel++) {
        int poslednePlatidlo = platidla[pocetPlatidiel - 1];

        for (int s = 0; s <= suma; s++) {
            D[s][pocetPlatidiel] = D[s][pocetPlatidiel - 1]
                    || ((s >= poslednePlatidlo) && D[s - poslednePlatidlo][pocetPlatidiel - 1]);
        }
    }

    if (!D[suma][platidla.length])
        return null;

    List<Integer> voVyplateni = new ArrayList<Integer>();
    for (int pocetPlatidiel = platidla.length; pocetPlatidiel > 0; pocetPlatidiel--) {
        // Vieme, že D[suma][pocetPlatidiel] = true
        if (D[suma][pocetPlatidiel - 1] == false) {
            // Musíme použiť platidlo platidla[pocetPlatidiel-1]
            voVyplateni.add(platidla[pocetPlatidiel - 1]);
            suma = suma - platidla[pocetPlatidiel - 1];
        }
    }
    return voVyplateni;
}

 * Main.
public static void main(String[] args) {
    int[] P = { 3, 5, 2, 4, 8 };
    int C = 13;
    System.out.println(daSaZaplatit(C, P));
}
```

Obr. 1: Java kód