Közelítő algoritmusok, dinamikus programozás

Algoritmuselmélet

2019. tavasz

11. gyakorlat

Tétel. Az alábbi problémák NP-teljesek.

HÁTIZSÁK

Bemenet: $s_1, \ldots, s_m; v_1, \ldots, v_m; b; k$ pozitív egészek.

Kérdés: van-e olyan $I \subseteq \{1, \ldots, m\}$, melyre

$$\sum_{i \in I} s_i \leq b \quad \text{és} \quad \sum_{i \in I} v_i \geq k?$$

LÁDAPAKOLÁS

Bemenet: $s_1, \ldots, s_m \in [0, 1]$ racionális súlyok, k pozitív egész szám.

Kérdés: el lehet-e helyezni a súlyokat k darab 1 súlykapacitású ládába?

FirstFit algoritmus; állítás

A LÁDAPAKOLÁS protblémára az FF algoritmus (miszerint a soron következő súlyt a legkisebb sorszámú olyan ládába helyezzük, amelybe még belefér) egy 2-közelítő algoritmus.

FirstFitDecreasing algoritmus

A súlyokat először nemnövekvő sorrendbe rendezzük, majd alkalmazzuk rájuk az FF algoritmust.

1. Hétfőre több elvégzendő feladatunk is van (házi feladatok beadása, zh-ra készülés, stb., mind különböző tantárgyakhoz kapcsolódik). Tegyük fel, hogy tudjuk, hogy az i-edik feladathoz t_i órára van szükség. Ha ennyit rászánunk, akkor biztosan megkapjuk a tárgyhoz tartozó k_i kreditet, de ha kevesebb idő jut rá, akkor biztosan nem kapjuk meg ezt a k_i kreditet. Összesen még h óra van hátra, ami nem feltétlenül elég mindenre. Úgy akarjuk kiválasztani, hogy ez alatt mely feladatokat teljesítsük, hogy összesen minél több kreditet kapjunk. Fogalmazzuk meg a megfelelő eldöntési problémát (nyelvet)! P-beli vagy NP-teljes az így kapott probléma?

2. A LÁDAPAKOLÁS feladatban legyenek a súlyok $s_1 = 0{,}4$; $s_2 = 0{,}7$; $s_3 = 0{,}1$; $s_4 = 0{,}6$. Hajtsuk végre ezen az FF algoritmust, illetve az FFD algoritmust! Hány ládát használ az optimális pakolás?

3. A LÁDAPAKOLÁS feladatban legyen két súly 0,34 és négy súly 0,33 értékű. Hajtsuk végre ezen az FFD algoritmust! Hány ládát használ az optimális pakolás?

4. A LÁDAPAKOLÁS problémában adottak az $s_1, \ldots, s_n$ méretű tárgyak, melyeket minél kevesebb 1 méretű ládába szeretnénk elpakolni úgy, hogy minden ládában a tárgyak összmérete legfeljebb 1. Adjunk polinomidejű algoritmust, ami meghatározza a szükséges ládák minimális számát, ha csak 1/2 és 1/4 méretű tárgyak vannak!

5. P-beli vagy NP-teljes a LÁDAPAKOLÁS problémának az a változata, amikor minden súly 1/4 vagy 4/5?

6. A ládapakolás feladatra egy lehetséges eljárás a BestFit, amikor sorban vesszük a tárgyakat és az s_i méretű tárgy ($i = 1, 2, \ldots$) az egyik olyan ládába kerül, ahol a legkisebb (de persze legalább s_i) az üres hely mérete. Igazoljuk, hogy ez az eljárás is egy 2-közelítő algoritmus!

7. Egy $n \times n$ méretű táblázat mezőin lépkedünk a bal alsó sarokból a jobb felső sarokba úgy, hogy egy lépésben a táblázatban vagy felfelé vagy jobbra egyet lépünk, de van néhány „tiltott” mező, ahova nem léphetünk. Adjunk egy dinamikus programozást használó eljárást, ami meghatározza, hogy hányféleképpen érhetünk célba! Mi az algoritmus lépésszáma?

8. Legyen $s_1 s_2 \dots s_n$ és $t_1 t_2 \dots t_m$ egy n és egy m hosszú karaktersorozat. Azt szeretnénk, hogy az $n \times m$ méretű A mátrix $A[i,j]$ eleme tartalmazza azt a legnagyobb k számot, melyre az $s_1 s_2 \dots s_i$ és a $t_1 t_2 \dots t_j$ sorozatok utolsó k karaktere megegyezik. Adjunk eljárást, ami az A tömböt $O(nm)$ lépésben kitölti.

9. Legyen $S \in \{\texttt{0}, \texttt{1}\}^*$ egy n hosszú szó. Egy $S[j]S[j+1]\dots S[i]$ részszó súlya jelentse azt, hogy mennyivel több `1` van benne, mint `0` (a súly lehet negatív is, ha több a `0`). Az S szöveghez definiáljuk azt a T tömböt, melyben a $T[i]$ értéke az $S[i]$-vel végződő, nemüres részszavak súlya közül a legnagyobb. Adjunk $O(n)$ lépésszámú algoritmust a T tömb kitöltésére! Hogyan és hány lépésben lehet a kitöltött T tömb alapján meghatározni az S-beli részszavak súlyai közül a maximálisat?

10. Adott az $a_1, a_2, \dots, a_n$ (nem feltétlenül csak pozitív) egész számokból álló sorozat. Ebben olyan szigorúan monoton csökkenő részsorozatot keresünk, hogy a részsorozatban előforduló számok összege minimális legyen. (Az üres sorozat összege 0.) Adjunk $O(n^2)$ lépésben működő dinamikus programozást használó algoritmust a legkisebb ilyen összeg meghatározására!

11. A $T[0..n, 0..m]$ táblázat elemei egész számok. A $T[0,0]$ elemből úgy akarunk eljutni a $T[n,m]$ elemhez, hogy minden lépésben vagy csak az első vagy csak a második indexet növeljük eggyel. Egy ilyen útvonal értéke legyen az érintett $T[i,j]$ számok közül a pozitívoknak a szorzata. Adjunk algoritmust, amely $O(nm)$ időben meghatározza, hogy mi a legnagyobb érték, ami egy, a feltételnek megfelelő útvonalon elérhető!

12. Egy f fokú létrán bizonyos fokok annyira rozogák, hogy ha rálépünk, leszakadnak. Szerencsére tudjuk, hogy melyik fokok ilyenek, hova nem szabad lépnünk. Egy lépéssel legfeljebb 3 fokot tudunk lépni. Adjunk $O(n)$ lépésszámú algoritmust, ami meghatározza, hogy a létra aljától hányféleképpen tudunk feljutni a létra legfelső fokára! (Feltehető, hogy a legfelső fokra rá szabad lépni.)

13. Adott egy $x_1 x_2 \dots x_n$ szó a $\Sigma = \{a, b, c, \dots, z\}$ abc felett. Adjunk olyan $O(n^2)$ futásidejű algoritmust, ami meghatározza a szóban található leghosszabb olyan részszó hosszát, ami palindroma.

14. Zárthelyit szervezünk, amin összesen H hallgató fog részt venni, és ehhez T darab terem áll rendelkezésünkre. Tudjuk, hogy az i-edik terembe h_i hallgató fér be. de lehet benne kevesebb is; $\sum h_i \geq H$. A terem geometriája, oszlopok, stb. miatt ha az i-edik terembe kerül zh-t író hallgató, akkor itt f_i fő zh-felügyelőre van szükségünk. (Tegyük fel, hogy f_i független a terembe ténylegesen kerülő hallgatók számától, feltéve, hogy ez legalább egy fő.) Adjunk dinamikus programozást használó algoritmust, amivel $O(T \cdot H)$ lépésben meghatározható, mely termeket használjuk a T közül ahhoz, hogy elég legyen a hely a hallgatóknak, de az összesen szükséges teremfelügyelők száma minimális legyen!