

Algoritmusalgebra vizsgázárthelyi

A rendelkezésre álló munkaidő 100 perc.

2012. január 05.

Kérjük, minden résztvevő **nevét**, **NEPTUN kódját**, a dolgozat *minden* lapjának jobb felső sarkában *olvashatóan* és *helyesen* tüntesse fel. Ezen kívül a legfelső lapra írja rá **gyakorlatvezetője nevét** is (akihez a NEPTUN szerint jár).

Minden egyes feladat helyes megoldása 10 pontot ér. A dolgozatok értékelése: 0-31 pont: 1, 32-43 pont: 2, 44-55 pont: 3, 56-67 pont: 4, 68-80 pont: 5. A puszta (indoklás nélküli) eredményközlést nem értékeljük. A megindokolt részeredményért arányos pontszám jár. Az évvégi jegy kiszámításakor a (legalább elégséges) zh pontszámát vesszük figyelembe.

Írószeren és papírokon kívül semmilyen segédeszköz használata sem megengedett, így tilos az írott vagy nyomtatott jegyzet, a számoló- és számítógép ill. mobiltelefon használata, továbbá a dolgozatírás közbeni együttműködés.

Az eredményeket hétfő délig igyekezünk közzétenni a honlapon.

Megtekintés, szóbeli: 2012. január 9. hétfő, 14:00-15:00, valahol a tanszék környékén

1. Definiáld az $O(f)$ és $\Omega(f)$ jelölések jelentését!

Egy lehetséges megoldás:

$O(f)$ jelölés definíciója: Ha $f(x)$ és $g(x)$ az $\mathbb{R}^+$ egy részhalmazán értelmezett, valós értéket felvevő függvények, akkor $g = O(f)$ jelöli azt a tényt, hogy $\exists c > 0$ és $x_0 > 0$ állandók, hogy $|g(x)| \leq c \cdot |f(x)|$, ha $x \geq x_0$.

$\Omega(f)$ jelölés definíciója: Ha $f(x)$ és $g(x)$ az $\mathbb{R}^+$ egy részhalmazán értelmezett, valós értéket felvevő függvények, akkor $g = \Omega(f)$ jelöli azt a tényt, hogy $\exists c > 0$ és $x_0 > 0$ állandók, hogy $|g(x)| \geq c \cdot |f(x)|$, ha $x \geq x_0$.

2. Írd le a radix rendezés algoritmusát és bizonyítsd be, hogy az algoritmus mindig helyes eredményt ad! Mennyi az algoritmus lépésszáma? (A lépésszámot nem kell indokolni.)

Egy lehetséges megoldás:

A radix rendezés egy olyan lexikografikus rendezés, ami először k . koordináta szerint ládarendez, majd $(k-1)$., $(k-2)$., ..., 1. szerint, ebben a sorrendben. Mindig az egész listát rendezzük.

Ekkor azt állítjuk, hogy a lista rendezve van.

Az állítás bizonyítása:

Két tetszőleges elem legyen $b = (b_1, b_2, \dots, b_k)$ és $c = (c_1, c_2, \dots, c_k)$, ahol $b < c$. Az állítás szerint a rendezés befejeztével b előrébb van, mint c .

$b_i = c_i \forall i$ esetén, amire $1 \leq i < j$, és legyen $b_j < c_j$, ahol $1 \leq j \leq k$.

A radix rendezés előbb a k ., majd a $(k-1)$., ..., $(j+1)$. koordináta szerint ládarendez. A j . koordináta szerinti ládarendezés során a b a c elé kerül. A továbbiakban a $(j-1)$., ..., 1. koordináta szerinti rendezés nem változtat a kettőjük sorrendjén, tehát a rendezés befejeztével b a c előtt lesz, így az állítást igazoltuk. $\square$

Lépésszám: n elem rendezésének lépésszáma, melyek egyenként k darab koordinátával rendelkeznek $O(n + |A_1|) + O(n + |A_2|) + \dots + O(n + |A_k|) = O(k \cdot n + \sum_{i=1}^n |A_i|)$, ahol az A_i véges, rendezett halmazból az i . koordináta került ki, így $|A_i|$ jelöli az i . koordináta szerinti rendezéshez szükséges ládák számát.

3. Ismertesd az irányított gráf erősen összefüggő komponenseinek meghatározására tanult algoritmust! (Az algoritmus helyességét nem kell bizonyítani.) Mennyi az algoritmus lépésszáma? (Indoklás ehhez sem kell.)

Egy lehetséges megoldás:

Az algoritmus lépései:

- I. DFS futtatása egy választott x csúcsból.
- II. A gráf éleinek irányításának megfordítása után DFS, csökkenő sorrendben indítva az előző befejezési számok szerint.
- III. Ami mindkét irányban egy komponensben van, az egy erősen összefüggő komponensben van.

Lépésszám: $O(n + e)$, ahol n a csúcsok, e pedig az élek száma.

-
4. A Dinamikai Köztársaságban n -féle címletű pénzt használnak: $d_1 \leq d_2 \leq \dots \leq d_n$ dinárosokat. Adj $O(nC)$ lépésszámú algoritmust, ami meghatározza, hogy legkevesebb hány darab pénzzel lehet kifizetni C dinárt! (Feltesszük, hogy C dinárt valahogy ki lehet fizetni a fenti címletekkel.)
-

Egy lehetséges megoldás:

A feladatot dinamikus programozás segítségével oldhatjuk meg.

A megoldás elején egy kis egyszerűsítést tehetünk a dinárosokkal kapcsolatban: mivel a feladatban meg van engedve az egyenlőség az egyes dinárosok között, ezért az egyforma értéket képviselőket kezeljük azonosnak. Ezáltal $n' \leq n$ féle pénz áll rendelkezésünkre.

Kis C értékekre könnyen meghatározható, hogy legkevesebb hány pénzzel lehet az adott értéket kifizetni. (A feladat megadta, hogy a C értéket valahogy ki lehet rakni a dinárosokból.) Ez $C = 0$ esetén 0 érmét jelent.

Ekkor jelölje $\mathcal{T}[i]$ az i összeg előállításához szükséges pénzek minimális számát. Ekkor legyen:

$$\mathcal{T}[i] = \begin{cases} 0 & \text{ha } i = 0 \\ \min_{\forall j} \{\mathcal{T}[i - d_j]\} + 1 & \text{ha } i > 0 \end{cases}$$

ahol a rekurziós képletben a '+1' jelöli az új összeg kirakásához szükséges felhasznált pénzt.

Az algoritmus végrehajtása során sorban elkezdjük kiszámolni $\mathcal{T}[1], \mathcal{T}[2], \dots$ értékeket

Leállási feltétel: ha $i = C$, akkor a megállunk.

Megoldás: $\mathcal{T}[C]$

Az eredmény helyes, mivel minden kiszámolt $\mathcal{T}[i]$ értékhez létezik egy érmekombináció, melyeket mind figyelembe vesszük, így mindenképp ezek minimuma kerül kiválasztásra.

Lépésszám: minden szóhajövő $0, \dots, C$ összeg esetén n' elem minimumát vizsgáljuk, ahol $n' \leq n$, így a lépésszám $O(nC)$.

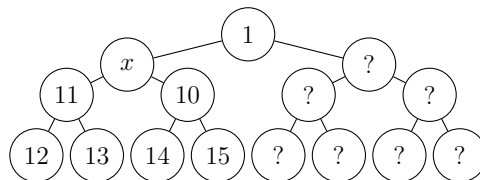
-
5. Egy 15 elemet tartalmazó kupacban az $1, 2, \dots, 15$ számok vannak valamilyen elrendezésben. A kupac tömb reprezentációjának második eleme milyen számot tartalmazhat?
-

Egy lehetséges megoldás:

A kupac tömb reprezentációjáról tudjuk, hogy ha a kupacot reprezentáló teljes bináris fa csúcsait egy $A[1 : n]$ tömbben tároljuk, akkor igaz, hogy $\text{balfiú}(A[i]) = A[2i]$ és $\text{jobbfiú}(A[i]) = A[2i + 1]$. Ezek alapján tehát a feladat meghatározni, hogy mi lehet a balfiú(*gyökér*) eredménye?

Megállapíthatjuk, hogy a gyökérben az 1 fog szerepelni, hiszen ez az elem a legkisebb, és a kupac tulajdonságai miatt a legkisebb elem szerepel a gyökérben. A 15 elemet tartalmazó kupac egy 4 szintű, teljes bináris fával reprezentálható. Ebből következik, hogy a 6 legnagyobb elem (10, 11, ..., 15) az alsó 2 szint valamelyikén helyezkedik el. Ha nem így lenne, akkor sérülne a kupac tulajdonság.

Vegyük az alábbi elrendezést:



Ebben az esetben x helyén állhat $2, 3, \dots, 9$ attól függően, hogy mi volt a beszúrás sorrendje. A fennmaradó 7 számból pedig egy kupacot kell építeni a gyökér jobb részféjeként. Ezek az elemek az ábrán a ?-vel jelölt helyekre kerülnek. Mivel beláttuk, hogy a maradék számok közül egyik sem szerepelhet x helyén, így a megoldás az, hogy a $2, 3, \dots, 9$ számok közül bármelyik állhat az $A[2]$ -ben.

6. Egy kezdetben üres $3n$ méretű hashtáblába nyitott címmel beszúrunk n különböző egész számot a $h(x) = x \pmod{3n}$ hashfüggvény alkalmazásával, kvadratikus próbával. Legfeljebb hány ütközés történhet?

Egy lehetséges megoldás:

Megjegyzés: Mivel a feladat hibásan lett kiírva, ezért lineáris próbálással oldjuk meg (kvadratikus esetben nem lehetne garantálni ezt a felső korlátot). A feladat ilyen megoldására 10 pont járt annak ellenére is, hogy hibás volt.

Mivel a tábla kezdetben üres, emiatt az ütközések elvi maximuma $\sum_{i=0}^{n-1} i = \frac{(n-1) \cdot n}{2}$, hiszen minden beszúrando elem maximum annyi elemmel ütközhet, ahány elem van a táblában. Jó megoldás lehet ilyenkor a feladatra, ha mutatunk egy olyan esetet, ahol az ütközések száma pont a fent nevezett érték, hiszen ennél több ütközés semmiképp nem lehet.

Ha csak olyan különböző x_i elemeket szúrunk be, amire $\forall i$ esetén $x_i \equiv k \pmod{3n}$, ahol k állandó, akkor minden beszúrásnál az ütközések száma maximális lesz. Így beláttuk, hogy az ütközések száma maximum $\frac{(n-1) \cdot n}{2}$ lehet.

7. Éllistával adott egy G gráf, melynek élei súlyozottak (lehetnek negatív súlyok is). Szeretnénk a gráf pontjait két csoportra felosztani – egy X és egy Y ponthalmazra – úgy, hogy a legisebb súlyú olyan él, aminek egyik végpontja X -beli, másik pedig Y -beli, a lehető legnagyobb súlyú legyen. Adj $O(e \log e)$ lépésszámú algoritmust egy ilyen felosztás megtalálására!

Egy lehetséges megoldás:

Először nézzük azt az esetet, amikor a G gráf összefüggő. Ebben az esetben keressünk benne minimális súlyú feszítőfát, válasszuk ehhez a Prim algoritmus éllistas implementációját, ennek a lépésszáma $O(e \log e)$. A csúcsokat az F minimális feszítőfa egyik legnagyobb súlyú éle (melynek súlyát jelöljük s -el) bontsa két részre: az él egyik oldalán lévő pontok alkossák az X , a másik oldalon lévők az Y halmazt.

Megmutatjuk, hogy ez a felosztás megfelelő. Tegyük fel indirekt, hogy egy másik, X', Y' felosztás esetén minden X' és Y' között futó él nagyobb, mint s . Mivel F feszítőfa, biztosan lesz olyan éle, ami X' és Y' között fut. Viszont F minden élének súlya legfeljebb s , így ellentmondásra jutottunk.

Ha a G gráf nem összefüggő, akkor a fenti lépéseket komponensenként kell végrehajtani, majd a minimális feszítőfákból álló feszítőerdőből kell kiválasztani a legnagyobb súlyú élet.

-
8. Igazold, hogy a következő eldöntési probléma P-ben van, vagy azt, hogy NP-teljes:

Input: G gráf, k egész szám

Kérdés: Van-e G -ben olyan feszítőfa, melynek maximális fokszáma pontosan k ?

Egy lehetséges megoldás:

A feladat megoldását kezdjük az NP-beliség belátásával. Jelen esetben egy jó tanú erre az n csúcsú gráfban egy olyan feszítőfa mutatása, aminek a maximális fokszáma k , hiszen ez polinom időben ellenőrizhető: az adott fára megnézzük, hogy minden csúcsát tartalmazza-e a gráfnak ($O(n)$ lépés), illetve a fa minden csúcsára $\deg(v) \leq k$, és van olyan csúcs amire $\deg(v) = k$ (szintén $O(n)$ lépés). Ezzel beláttuk, hogy az adott eldöntési probléma NP-beli.

Jelöljük a feladat eldöntési problémáját L -lel, és L' -vel azt a kicsivel egyszerűbb problémát, hogy ha a kérdést csak legalább 3 pontú gráfokra kell eldönteni. Elég belátni, hogy L' NP-nehéz, hiszen ebből nyilván következik, hogy az általánosabb L is az.

Belátjuk, hogy $Hút \prec L'$. A Karp-redukció a G gráfhoz rendelje a $(G, 2)$ párt, vagyis L' bemente legyen $G' = G$ és $k = 2$.

Az így megadott függvény polinom időben számolható. Továbbá pontosan akkor tartalmaz G Hamilton-utat, ha G' -ben található olyan feszítőfa, aminek a maximális fokszáma $k = 2$. Ez a kettő azért lesz minden esetben ekvivalens, mert az a feszítőfa, aminek a maximális fokszáma 2 az egy út, mely minden G -beli csúcsot tartalmaz.

Tehát L' és ezért L is egy NP-nehéz probléma. Mivel NP-beli is, ezért NP-teljes.
