# Algoritmuselmélet vizsgazárthelyi

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 igyekszünk 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 $k$ darab koordinátából álló sorozatokat rendez. Először a $k$. koordináta szerint futtat egy ládarendezést, majd $(k-1).,(k-2).,\ldots,1.$ szerint. Az $i$. koordináta szerinti ládarendezés inputja az $(i+1)$. koordináta szerinti rendezés outputja, az ott kapott sorrendben adva be a rendezendő sorozatokat. Azt állítjuk, hogy a végén a sorozatok listája rendezett.

Az állítás bizonyítása:
Legyen két tetszőleges elem $b = (b_1, b_2, \ldots, b_k)$ és $c = (c_1, c_2, \ldots, c_k)$, ahol $b < c$. azt kell belátnunk, hogy a rendezés befejeztével $b$ előrébb van, mint $c$.
Mivel $b < c$, ezért létezik olyan $1 \leq j \leq k$, hogy $b_i = c_i$ ha $1 \leq i < j$ és $b_j < c_j$ (azaz a $j$. koordináta az első pozíció, ahol a két szám eltér).
A radix rendezés előbb a $k$., majd a $(k-1).,\ldots,(j+1).$ koordináta szerint ládarendez. A $j$. koordináta szerinti ládarendezés során a $b$ a $c$ elé kerül. A továbbiakban a $(j-1).,\ldots,1.$ koordináta szerinti rendezés nem változtat a kettőjük sorrendjén, mert $b$-t mindig előbb olvassuk be, mint $c$-t az aktuális ládarendezésnél és innentől kezdve mindig ugyanabba a ládába kerülnek ugyanebben a sorrendben, tehát a rendezés befejeztével $b$ a $c$ előtt lesz, így az állítást igazoltuk. □

Lépésszám: $n$ elem rendezésének lépésszáma, melyek egyenként $k$ darab koordinátával rendelkeznek $O(n + |A_1|) + O(n + |A_2|) + \cdots + O(n + |A_k|) = O(k \cdot n + \sum_{i=1}^{n} |A_i|)$, 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 élei 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: $d_1 \leq d_2 \leq \cdots \leq d_n$ 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.

Jelölje $\mathcal{T}[i]$ az $i$ összeg előállításához szükséges pénzek minimális számát, ezt fogjuk kiszámolni minden $0 \leq i \leq C$ esetre. $T[0] = 0$ nyilvánvalóan, hiszen 0 dinárt 0 darab érmével elő tudunk állítani.

Azt állítjuk, hogy a többi érték a következő rekurziós összefüggéssel kapható:

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

A rekurziós összefüggés magyarázata: ha $i$ dinárt kell összeraknunk, akkor azt úgy tehetjük, hogy kiválasztunk egy $d_j$ dináros érmét (innen jön a $+1$-es tag), a maradék $i-d_j$ dinárt pedig próbáljuk a lehető legkevesebb érmével összerakni valahogyan. Azt, hogy ezt hogyan tehetjük meg, a $T[i-d_j]$ érték adja meg. Innen látható, hogy a rekurzió helyes, hiszen így a $\mathcal{T}[i]$ érték kiszámolásakor az összes esetet végignéztük és ezekből választottuk a legkisebbet.

Az algoritmus végrehajtása során sorban elkezdjük kiszámolni $\mathcal{T}[1]$, $\mathcal{T}[2]$, ... értékeket.
Megoldás: $\mathcal{T}[C]$
Lépésszám: minden szóbajövő $0, \ldots, C$ összeg esetén $n'$ elem minimumát vizsgáljuk, ahol $n' \leq n$, így a lépésszám $O(nC)$.

---

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

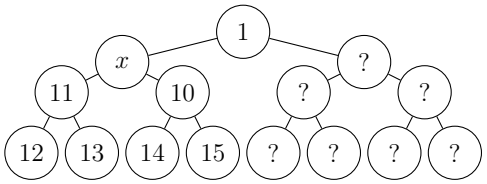
---

*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 balfiú$(A[i]) = A[2i]$ és 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. A hat legnagyobb elem egyike sem lehet a gyökér alatti szinten, hiszen egy itt található csúcs alatt hat nálánál nagyobb elem áll még. Az eddigiek alapján tehát a keresett elem biztosan nem lehet az 1, $10, 11, \ldots, 15$. Belátjuk, hogy a többi szám bármelyike viszont kerülhet a balfiú(*gyökér*) helyre.
Vegyük az alábbi elrendezést:




Ebben az esetben $x$ helyén állhat $2, 3, \ldots, 9$, a fennmaradó 7 számból pedig lehetséges egy kupacot építeni a gyökér jobb részfájaként.

---

6. Egy kezdetben üres $3n$ méretű hashtáblába nyitott címzéssel beszúrunk $n$ különböző egész számot a $h(x) = x \pmod{3n}$ hashfügvény 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 beszúrandó elem maximum annyi elemmel ütközhet, ahány elem van a táblában. Mutatunk egy olyan esetet, ahol az ütközések száma pont a fent nevezett érték, hiszen ennél több ütközés semmiképp nem lehet.
Ha például csupa olyan különböző $x_i$ elemeket szúrunk be, amikre $x_i \equiv 1 \pmod{3n}$ fennáll, akkor minden elem ugyanoda kerülne, az újonnan beszúrt elemek az összes korábbi elemmmel ütközni fognak. Így beláttuk, hogy az ütközések száma maximum $\frac{(n-1)\cdot n}{2}$ és megmutattuk, hogy ez az eset elő is állhat.

---

7. Éllistával adott egy $G$ összefüggő gráf, melynek élei súlyozottak (lehetnek negatív súlyok is). Szeretnénk a gráf pontjait két csoportra felosztani – egy $X$ és egy $Y$ ponthalmazra – úgy, hogy a legkisebb 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:*

Keressünk a gráfban minimális súlyú feszítőfát, válasszuk ehhez a Prim algoritmus éllistás 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. (Ezek csúcsai az $s$ súlyú él elhagyásával keletkező két komponens bejárásával megkaphatóak.) Az így kapott felosztásban

az $X$ és $Y$ halmaz között futó élek közül valóban az $s$ súlyú él lesz a minimális, hiszen ellenkező esetben $s$ lecserélhető lenne a nálánál kisebb, $X$ és $Y$ között futó élre, így előállítva egy kisebb súlyú feszítőfát, ami nem lehetséges.

Megmutatjuk, hogy az így kapott felosztás megfelelő, vagyis ebben lesz a két rész között menő minimális él a lehető legnagyobb. Tegyük fel indirekt, hogy egy másik, $X', Y'$ felosztás esetén a minimális $X'$ és $Y'$ közötti él nagyobb $s$-nél. Ezt úgy is mondhatjuk, hogy ekkor 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.

---

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:*

Azt fogjuk belátni, hogy a probléma NP-teljes. A feladat megoldását kezdjük az NP-beliség belátásával. Jelen esetben egy jó tanú erre az $n$ csúcsú gráfban egy olyan feszítőfa, 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).

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 következik, hogy az általánosabb $L$ is az.

Belátjuk, hogy Hút $\prec L'$. A Karp-redukció a $G$ gráfhoz rendelje a $(G, 2)$ párt, vagyis $L'$ bemenete 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.

---