Algel V. gyakorlat

Egy kupac rendezés

2009. március 10/12.

1. **Legalább hány összehasonlítás kell ahhoz, hogy egy n elemű tömbből egy olyan tagot találjunk meg, ami a tömb 10 legkisebb eleme közé tartozik?**
   $n - 10$, és ennyi elég is. Ennél kevesebb esetén az összehasonlítottsági gráf legalább 11 komponensből állna, az egyes komponensek viszonáról pedig nincs információnk. Ha kidobunk 9 elemet, közöttük minimumot keresve biztos jók vagyunk.

3. **A (növekvően) rendezett $A[1:n]$ tömb k elemét valaki megváltoztatta. A változtatások helyeit nem ismerjük. Javasoljunk $O(n + k \log k)$ költségű algoritmust az így módosított tömb rendezésére!**
   Menjünk végig a tömbön, és a hibás párokat (amik rosszul állnak) szedjük külön. Ez azért kell, mert a rossz szomszédok közül az legalább az egyiket biztos megváltoztattuk (hiba csak változtatásnál lehet), viszont nem tudjuk, melyiket. Így legfeljebb $2k$ elemet vettünk ki, ezeket $O(2k \log 2k = O(k \log k)$ időben tudjuk rendezni, utána $O(n+k)$ időben összefésülés. A kezdeti végigolvasás $O(n)$, vagyis összességében tényleg $O(n + k \log k)$ lépésben végzünk.

7. **Igazoljuk, hogy egy n elemből álló kupac felépítése $\Omega(n)$ összehasonlítást igényel!**
   Mivel a gyökérben a legkisebb elem van, ezért nem lehet hamarabb felépíteni, mint a legkisebb elemet megtalálni, ami pedig $\Omega(n)$.

8. **Adjunk hatékony algoritmust egy kupac tizedik legkisebb elemének meghatározására!**
   Konstans, hiszen az i-edik legkisebb elem a gyökértől legfeljebb i távolságra van, így 2^{10} elem közül kell a tizedik legkisebbet megtalálni, ami konstans.

9. **Adott egy n elemet tartalmazó kupac és egy k kulcs. Keressük meg a kupac k-nál kisebb elemeit! Ha m ilyen elem van, akkor az algoritmus $O(m)$ elemi lépést használhat.**
   Lényegében egy bejárás, elindulunk a gyökérből, és amíg k-nál kisebb elemet találunk, addig kiírjuk, ha $\geq k$-t, akkor az adott ágat hanyagoljuk, hiszen a kupac-tulajdonság miatt arrafele csak még nagyobbak lehetnek. Legfeljebb a nekünk jó elemek mindkét gyerekét vizsgáljuk meg feleslegesen, vagyis $O(2m) = O(m)$ vizsgálatot végzünk.

11. **Adottak a sík egész koordinátájú $P_1 = (x_1, y_1), \ldots, P_n = (x_n, y_n)$ koordinátájú pontjai. Javasoljunk $O(n)$ uniform költségű módszert olyan $P_i \neq P_j$ pontok kiválasztására, amelyeken átmenő egyenes által meghatározott félsíkok közül az egyik tartalmazza az összes pontot!**
   Minimális y kooridántájú pont lesz az egyik, ami meghatározza, a másik pedig az y_{min}-es pontból abszolútértékben legnagyobb meredekségű. Ha ezeken kívül esne egy pont, akkor annak a meredeksége abszolútértékben nagyobb lenne. Meredekséget számolni két pont ismeretében konstans, így a lépésszám két minimumkeresés, azaz $O(n + n) = O(n)$.

14. **[Vizsga: 2007. június 19.] Adott a síkon n pont, melyek koordinátái $(a_1, b_1) \ldots (a_n, b_n)$. Olyan $P = (x, y)$ pontot keresünk a síkon, amire az alábbi összeg minimális.**

$$\sum_{i=1}^{n} (|a_i - x| + |b_i - y|)$$

   Adjon algoritmust, ami $O(n \log n)$ lépésben meghatároz egy ilyen P pontot.
   Észrevesszük, hogy x és y koordináta független. Egy dimenzióban: ha csak két pont van, akkor két pont között bárhol lehet, rajtuk kívül viszont rosszabb. Tehát a két szélső pont között

kell lennie, de rajtuk kívül a két szélső között is stb, tehát páratlan pont esetén a középsőn, páros esetén a középső kettő közül valamelyiken vagy közöttük (indoklás: ha nem így lenne, akkor biztos, hogy tudnánk jobbat csinálni). Tehát koordinátánként egy rendezés, és középső választása, tehát $O(n \log n)$-ben bent vagyunk.

15. **[Vizsga: 2004. június 10.] Az n méretű (nem feltétlenül rendezett) A tömb elemei különböző pozitív egész számok. Adjon algoritmust, amely meghatároz egy $1 \leq k \leq n$ számot és kiválaszt k különböző elemet az A tömbből úgy, hogy a kiválasztott elemek összege nem több, mint k^3. Ha nincs ilyen k, akkor az algoritmus jelezze ezt a tényt! Az algoritmus lépésszáma legyen $O(n \log n)$! (Két szám összehasonlítása, összadása vagy szorzása egy lépésnek számít.)**
Észrevétel: ha van ilyen k, akkor ezeket kicserélhetjük a k legkisebbre, hiszen az összegük így csak csökkenhet, ami továbbra is jó. Tehát elég mindig a k legkisebbel foglalkozni. Rendezzük a tömböt, és induljunk el az elejétől. Folyamatosan összegezzük, és az első k-ra ahol igaz, kész is vagyunk, ha pedig végigértünk, és nem volt ilyen, akkor biztos nincs. Ez így rendezés és egy végigolvasás, azaz $O(n \log n + n) = O(n \log n)$.

16. **[ZH: 2004. április 8.] Az $A[1 : n]$ tömbben levő elemekről tudjuk, hogy $A[1] \neq A[n]$. Adjon $O(\log n)$ összehasonlítást használó algoritmust, amely talál egy olyan i indexet, hogy $A[i] \neq A[i + 1]$!**
Bináris kereséssel. Első és utolsó különbözik, tehát van köztük váltás. Utána középen ha elsőtől különbözik, akkor köztük is van váltás, egyébként hátrafele kell különböznie, tehát köztük van váltás. Így $O(\log n)$.