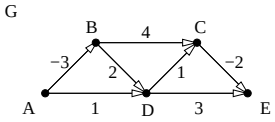
Algoritmuselmélet
2011. február 28.

Schlotter Ildi
ildi@cs.bme.hu

4. gyakorlat
Kupac, keresések



1. Határozza meg az alábbi G gráfban az A-ból induló legrövidebb utakat!

2. a) Építsen kupacot az órán tanult lineáris idejű módszerrel az alábbi tömbből: $31, 6, 50, 7, 2, 51$.
   b) Szúrja be az így kapott tömbbe az 1, majd ezután az 5 számot!
   c) Hajtson végre két egymást követő MINTÖR-t az így kapott kupacon!

3. Adjunk hatékony algoritmust egy kupac tizedik legkisebb elemének a megtalálására!

4. Adott egy kupac és egy k kulcs. Adjunk algoritmust a kupac k-nál kisebb kulcsú elemeinek megkeresésére! Ha m ilyen elem van, akkor az algoritmus $O(m)$ lépést használhat.

5. Ha adott n szám, akkor hívjuk közülük középső elemnek a rendezés szerinti $\lceil n/2 \rceil$-ediket. Kezdetben adottak az $a_1, a_2, \ldots, a_n$ egész számok, amikről tudjuk, hogy az a_1 a középső elem, egyébként a számok rendezetlenek. Ezekből építsen fel egy adatszerkezetet, amiben két művelet van:
   BESZÚR: egy új elemet illeszt az adatszerkezetbe,
   KÖZÉPTÖR: az aktuális középső elemet törli.
   Mindkét művelet megvalósítása $O(\log k)$ összehasonlítást használjon, amikor k tárolt elem van, az adatszerkezet kezdeti felépítése legyen $O(n)$ összehasonlítás!

6. Adjunk olyan algoritmust, amely $\lceil 1.5n \rceil - 2$ összehasonlítással megtalálja egy n elemű tömb legnagyobb és legkisebb elemét!

7. Bizonyítsuk be, hogy ahhoz, hogy egy halmazból a két legnagyobb elemet kiválasszuk, $n + \lceil \log n \rceil - 2$ összehasonlítás szükséges és elégséges!

8. Adott az $A[1 : n]$ csupa különböző egész számot növekvő sorrendben tartalmazó tömb. (A tömbben negatív számok is lehetnek!) Adjunk hatékony algoritmust egy olyan i index meghatározására, melyre $A[i] = i$ (feltéve, hogy van ilyen i): igyekezzünk minél kevesebb elem megvizsgálásával megoldani a feladatot!

Gyakorlás:

1. Egy kupacban definiálhatjuk a NÖVEL műveletet, amelyek az adott helyen levő kulcsot nagyobbra cseréli, és a változtatás után helyreállítja a kupactulajdonságot.
   a) Adjon $O(\log n)$ lépésszámú algoritmust a NÖVEL műveletre! (A változtatandó kulcs helyét tudjuk.)
   b) Módosítsa az eljárást bináris kupacról d-kupacra!

2. A kezdetben üres kupacba egyenként szúrunk be n elemet. Igazolja, hogy előfordulhat, hogy a beszúrások során végzett összehasonlítások száma $\Omega(n \log n)$.

3. Egy orvosi rendelőben 2 orvos rendel, A és B. Bizonyos betegeket csak az egyik orvos láthat el (minden ilyen betegre adott, hogy ez az orvos A vagy B), más betegeket mindkettőjük elláthat. Emellett minden beteg kap egy prioritást, mely az eset súlyosságát jelzi. Adjunk olyan adatstruktúrát, amely a következő műveleteket támogatja:
   BEHÍV(X): a legkisebb prioritású beteget adja vissza az X orvos által elláthatóak közül ($X \in \{A, B\}$).
   TÖRÖL: töröl egy beteget az adatszerkezetből.
   BESZÚR: egy új beteget szúr be az adatszerkezetbe.
   A BEHÍV(X) művelet lépésszáma legyen konstans, a másik kettőé pedig $O(\log k)$, ha k beteg van!

4. Az egész elemeket tartalmazó $A[1 : n]$ tömböt konvexnek nevezzük, ha minden i-re $(1 < i < n)$ teljesül, hogy $A[i] \leq 1/2(A[i-1] + A[i+1])$. Javasoljunk olyan algoritmust, mely minél kevesebb összehasonlítással megtalálja egy konvex tömb minimális elemét!

5. A (növekvően) rendezett $A[1 : n]$ tömb k darab elemét valaki megváltoztatta. A változtatások helyeit nem ismerjük. Javasoljunk $O(n + k \log k)$ költségű algoritmust az így módosított tömb rendezésére!