Algoritmuselmélet
2010. ősz

Kazi Sándor
KAZI(AT)CS.BME.HU

5. gyakorlat
Rendezések

-1. Egy egyszerű operációs rendszerben szeretnénk nem preemptív (egy elkezdett taszk nem szakítható meg) ütemezőt csinálni. Minden taszknak van egy pozitív prioritása, a legkisebb prioritási érték a legfontosabb feladat. Hogyan érdemes tárolnunk a taszkok várakozási sorát, ha az új taszkokat gyorsan akarjuk kiválasztani?

0. Az orosz cár sürgősen levelet szeretne küldeni a kínai császárnak. Ehhez a már jól bejáratott, Szentpétervárról Pekingbe vivő útvonalat szeretné használni, amin n város helyezkedik el sorban egymás után: $v_1, v_2, \dots v_n$. A levelet postagalambok viszik egyik városból a másikba, a célba juttatásához persze több galambra is szükség lehet egymás után. A felesleges kitérők elkerülése érdekében a galambok mindig olyan v_i városból viszik a levelet v_j-be, amire $i < j$ teljesül. Segítsünk megtervezni a cárnak a leggyorsabb útvonalat, ha minden galambról tudjuk, hogy honnan hova tud repülni, és mennyi idő alatt teljesíti a távot! Elemezzük a megadott algoritmus hatékonyságát is!

---

1. Adott egy kupac, mely n darab számot tartalmaz. Egy új kupacot szeretnénk építeni az eredeti kupac elemeinek (-1)-szereseiből. (Ehhez, ha akarjuk, használhatjuk az eredeti kupacot.) Mutassuk meg, hogy az új kupac elkészítéséhez használt összehasonlítások száma $\Omega(n)$!

2. Hogyan működik a beszúrásos, az összefésüléses, a buborék- és a gyorsrendezés eljárás? Milyen lépésszámmal végezhető el egy n elemű lista rendezése ezekkel az algoritmusokkal?

3. Rendezzük a következő listát összefésüléses rendezés segítségével: $4, 11, 9, 10, 5, 6, 8, 1, 2, 16$.

4. Egy csupa különböző egészekből álló sorozat *bitonikus*, ha először nő, utána pedig fogy, vagy fordítva: először fogy, utána nő. Például az $(1, 3, 7, 21, 12, 9, 5)$, $(9, 7, 5, 4, 6, 8)$ és $(1, 2, 3, 4, 5)$ sorozatok bitonikusak. Adjunk $O(n)$ összehasonlítást használó rendező algoritmust n elemű bitonikus sorozatok rendezésére!

5. Az $A[1 : n]$ tömbben egy rendezett univerzum n különböző eleme volt, nagyság szerint növekvő sorrendben. Valaki időközben megkeverte a tömb elemeit, de csak annyira, hogy minden egyes elem új helye az eredetitől legfeljebb 5 távolságra esik. Adjunk $O(n)$ idejű algoritmust az eredeti állapot helyreállítására!

6. Adott egy dobozban n különböző méretű anyacsavar, valamint egy másik dobozban a hozzájuk illő apacsavarok. Kizárólag a következő összehasonlítási lehetőségünk van: Egy apacsavarhoz hozzápróbálunk egy anyacsavart. A próbának háromféle kimenete lehet: apa $<$ anya, apa $=$ anya, vagy apa $>$ anya; annak megfelelően, hogy az apacsavar külső átmérője hogyan viszonyul az anyacsavar belső átmérőjéhez. Szeretnénk az anyacsavarokhoz megtalálni a megfelelő apacsavarokat. Adjunk erre a feladatra **átlagosan** $O(n \log n)$ összehasonlítást felhasználó módszert!

7. Adott egy $n \times n$-es mátrix. Adjunk $O(n^2 \log n)$ összehasonlítást használó algoritmust, amely eldönti, van-e két olyan sor, amelyeknek az első oszlopbeli elemei különböznek, viszont az összes többi oszlopban megegyeznek!

8. Hogyan működik a láda- és a (lexikografikus) radix-rendezés?

9. Rendezzük radix rendezéssel az acedf, adfce, ecdaf, acfde, edcaf, edcfa, caedf elemeket.

10. Egy azonos hosszú abc-szerint rendezett (ismétlődő elem nélküli) karaktersorozatokból álló lista kolexikografikusan rendezett, ha bármely két i és j elem esetén i pontosan akkor előzi meg j-t, ha az i lexikografikus ("normál") rendezés szerinti utolsó karaktere megelőzi j lexikografikusan utolsó elemét. (Pl.: *abc*, *abd*, *acd*, *bcd*, *abe*, *ace*, *ade*, *bce*, ...) Hogyan rendezhetnénk sorba a feltételnek megfelelően azonos hosszú, abc-szerint rendezett karaktersorozatokat. Mennyi ennek a költsége? Rendezzük eszerint ezeket: *cdf*, *bcf*, *ace*, *cef*, *cde*, *bcd*.

11. Vázoljunk egy $O(n)$ időigényű algoritmust (az időkorlát bizonyításával együtt) n olyan egész számból álló sorozat rendezésére, melynek elemei az

    (a) $\{1, \dots, 3n\}$ tartományba esnek!

    (b) $\{1, \dots, n^7 - 1\}$ tartományba esnek!