2-3 fák, hash-elés

Algoritmuselmélet

9. gyakorlat

2023.

1. Egy 2-3 fa kezdetben csak a 6, 8, 13 elemeket tárolja. Rajzoljuk le ezt a fát, majd szúrjuk be a 2, 5, 1 elemeket, végül töröljük a 8, 2 elemeket.
2. Egy különböző egész számokat tároló 2-3 fa gyökerében két útjelző van, a 101 és a 117. Legfeljebb hány elemet tárolhat a fa?
3. Az $[1, 178]$ intervallum összes egészei egy 2-3 fában helyezkednek el. Tudjuk, hogy a gyökérben két kulcs van, és az első kulcs a 17. Mi lehet a második?
4. Adott n darab intervallum $(a_1, b_1), (a_2, b_2), \ldots, (a_n, b_n)$; az intervallumok végpontjai racionális számok és szokás szerint $a_i < b_i$. Tegyük fel, hogy egyik intervallum sem tartalmazza teljes egészében a másikat. Az intervallumokat a kezdő a_i koordinátájuk szerint szervezett 2-3 fában tároljuk. Hogyan lehet ennek segítségével egy adott x pontra $O(\log n)$ lépésben meghatározni, hogy a megadottak között van-e olyan intervallum, ami az x pontot tartalmazza?
5. Vödrös hash-t használva akarunk számokat tárolni, a $h(k) = k \pmod{11}$ hash függvényt használva.
   (a) Szúrjuk be a kezdetben üres, 11-méretű hash-táblába a $2, 5, 12, 1, 3, 88, 23, 43, 10, 34$ elemeket.
   (b) Hogyan zajlanak a feltöltött táblában kERES(1), KERES(20), KERES(45) műveletek?
   (c) Hogyan zajlik a TÖRÖL(23) művelet?
6. Nyitott címzéssel hash-elünk egy kezdetben üres $M = 11$ méretű táblába a $h(x) = x \pmod{M}$ hash-függvénnyel lineáris próbával. Mi lesz a tábla állapota az egyes lépések után, ha a 11, 9, 99, 7, 18 külcsokat ebben a sorrendben beszúrjuk, majd töröljük a 99-et és végül beszúrjuk a 33-at?
7. A hash-függvény legyen $f(x) = x \pmod{M}$ és a táblaméret $M = 7$. Helyezzük el a táblában a 9, 4, 17, 13, 3 kulcsokat ebben a sorrendben kvadratikus maradék próbálást használva az ütközések feloldására, majd töröljük a 17 kulcsot. Hol ér véget a KERES(10) művelet?
8. Egy 7-méretű hash-táblába a $h(x) = x \pmod{7}$ hash-függvénnyel szúrunk be elemeket. Az ütközéseket kettős hash-eléssel oldjuk fel a $h'(x) = 5 - \big(x \pmod{5}\big)$ másodlagos hash-függvény segítségével. A táblába a 19, 26, 38, 33, 31 elemeket szúrjuk be ebben a sorrendben. Adjuk meg a hash-tábla állapotát minden beszúrás után!
9. Egy $M = 1000$ méretű tömbbe nyitott címzésű hash-elést végzünk a $h(x) = x \pmod{1000}$ hash-függvény segítségével. Valaki azt javasolta, hogy a lineáris próba helyett használjuk a $h_i(x) = i \cdot x \pmod{1000}$ ugrósorozatot ($i = 0, 1, \ldots, 999$). Jó próbasorozatot kapunk-e így?
10. A vödrös hash-elésnél az egyes vödrök tartalmát lapok egy-egy láncolt listájában tároljuk. Legyen a vödörkatalógus mérete M és a hash-táblában tárolt lapok összes száma L.
    (a) Legrosszabb esetben nagyságrendileg hány lépést kell tennünk egy keresés során?
    (b) Legrosszabb esetben nagyságrendileg hány lépést kell tennünk egy beszúrás során?

    Most tároljuk az egyes vödrök tartalmát láncolt lista helyett egy-egy 2-3 fában.

(c) Legrosszabb esetben nagyságrendileg hány lépést kell tennünk egy keresés során?

(d) Legrosszabb esetben nagyságrendileg hány lépést kell tennünk egy beszúrás során?

11. A $T[0:M]$ táblában $2n$ elemet helyeztünk el az első $3n$ helyen ($3n < M$) egy ismeretlen hash-függvény segítségével. A táblában minden $3i$ indexű hely üresen maradt ($0 \leq i < n$). Legfeljebb hány ütközés lehetett, ha az ütközések feloldására

    (a) lineáris próbálást,

    (b) kvadratikus maradék próbálást használtunk?

12. Előfordulhat-e nyílt címzéses hash esetén, hogy az $n > 3$ méretű táblában csak 3 elem van, de a keresés lépésszáma n?

13. Kvadratikus maradék próbával hash-elünk egy $M = 127$ méretű hash-táblába, a $K \pmod{M}$ hash-függvényt használva. A kulcsok a következő sorrendben érkeznek: $M, 2M, 3M, \ldots, M \cdot M$, ezzel a tábla meg is telik. Igaz-e, hogy az így kapott tömb egy kupacot reprezentál?

14. A $b_0, \ldots, b_n$ alakú $(n+1)$-hosszú bitsorozatokat akarjuk tárolni. Tudjuk, hogy a b_0 paritásbit (ami a sorozatban az egyesek számát párosra egészíti ki). Ha nyitott címzésű hash-elést használunk $h(x) = x \pmod{M}$ hash-függvénnyel és lineáris próbával, akkor $M = 2^n$ vagy $M = 2^n + 1$ méretű hash-tábla esetén lesz kevesebb ütközés?

15. Egy M-méretű hash-táblába $n < M$ elemet raktunk be nyitott címzéssel, kvadratikus próbával, a $h(x)$ hash-függvényt használva. Ennek során t_1 ütközés történt (egy elem beszúrása során több ütközés is lehetett). Ugyanezt az n elemet ugyanabban a sorrendben beszúrtuk egy M^2-éretű hash-táblába is, de most lineáris próbával, $M \cdot h(x) + 1$ hash-függvénnyel, és ekkor t_2 ütközés történt. Bizonyítsuk be, hogy $t_2 \leq t_1$.

16. Vázoljuk a 2-3 fának (és műveleteinek) egy olyan módosítását, amiben továbbra is van KERES, BESZÚR, TÖRÖL, MIN, MAX művelet, és ezeken kívül van még RANG és K-ADIK művelet is, ahol RANG(x) azt adja vissza, hogy a tárolt elemek között az x a rendezés szerint hányadik elem, a K-ADIK(i) pedig, hogy a rendezés szerint a tárolt elemek közül melyik az i-edik. A módosítás során a felsorolt szokásos műveletek lépésszámának nagyságrendeje ne változzon, és mindkét új művelet lépésszáma legyen $O(\log n)$, ahol n a tárolt elemek száma.

17. 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 az alábbi 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.