

BFák alkalmazása nagy adatbázisokban. Keresés, beszúrás, törlés megvalósítása

Bfák

A bináris fa hátrányai

- Ha rendezett adatokat fesszünk fel, akkor egy része elfajulhat, a keresés lelassul
- Ha sok adatot szeretnénk tárolni a fában, lehetséges, hogy nem fér el a memóriában. Ekkor lemezen kéne tárolni: a lemezről olvasás sokkal lassabb

A B-fákban a csúcsoknak sok gyereke lehet, akár több ezer is. A B-fák elágazási tényezője sokkal nagyobb, mint a bináris fáké, ezért magasságuk lényegesen kisebb.

Ha a B-fa egy X csúcsa $N[X]$ kulcsot tartalmaz, akkor az X-nek $N[X]+1$ gyereke van. Ha a B-fákban egy kulcsot keresünk, akkor $(N[X]+1)$ -féle választásunk lehet, mert a keresett kulcsot $N[X]$ kulccsal hasonlítjuk össze.

A B-fa magassága a benne lévő csúcsok számának növelésekor csak a csúcsszámok logaritmusával nő.

Tulajdonságai

- A kitöltöttségi faktor (a redundáns adatok száma) 50%-nál jobb a B-fa esetében
- Egy lemez-szektorba bele kell férjen a fa egy mezejének tartalma, így pl. a BlockRead eljárással gyors adatkezelés valósul meg
- Minden lap legfeljebb $2n$ tételt tartalmaz (n jelenti a B-fa rendjét)
- Minden lapon - a gyökérlapot kivéve - legalább n tétel van
- Egy lap lehet levél (utód nélküli) vagy $n+1$ utóddal rendelkezik, ahol n a kulcsok száma
- Minden levél-lap ugyanazon a szinten helyezkedik el

→ Kiegyensúlyozott kereső fák

→ Hatékonyság: $O(\lg(n))$, de a nagy fókusz miatt igen alacsony szorzótényezővel

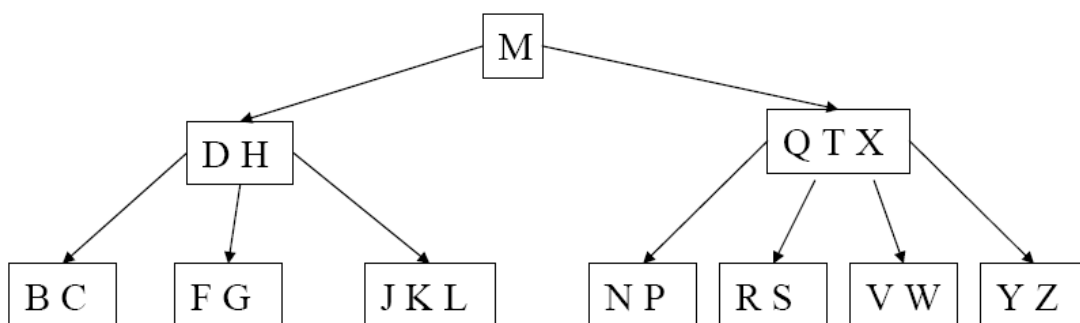
→ Alkalmazás: Lemezen tárolt adatbázisoknál

→ Mágneslemezen hatékony tárolásra lett tervezve

→ Ki/Be műveletek minimalizálása

→ Elágazási tényező: néhány ezerig is terjedhet

→ Időszükséglet = Ki/Be idő + memória számolás ideje



Egy csúcs n db kulcsot tartalmaz → a kulcstartomány felosztása → a csúcsnak $n+1$ gyereke van.

Levélcsúcs: egyetlen gyereke sincs

Lemzeműveletek modellezése

- x: egy adathivatkozás, nem tudni, hogy hol van
- Ha x a lemezen van, akkor x mezői nem elérhetők
- LemezrőlOlvas(x) ha lemezen van, akkor beolvassa. Egyébként hatástalan,
- LemezreÍr(x) kiírja a lemezre, ha történt módosítás. Egyébként hatástalan.
- Egy csúcs mérete = lemez blokkmérete. Ez az elágazási tényező felső korlátja is.

BFa Definíciója

A T BFa olyan gyökeres fa, amelyre:

1. Minden x csúcsnak a következő mezői vannak:
 - n: az x csúcsban tárolt kulcsok száma
 - n db kulcs úgy, hogy $kulcs[1] \leq \dots \leq kulcs[n]$
 - levél=Igaz, ha x levél, Hamis egyébként
 2. Ha x belső csúcs, akkor gyerek[i] vektor, ahol $i \leq n$ a gyerekekre mutató vektor. Ha x levél, akkor a vektor tartalma meghatározatlan
 3. Ha k_i egy kulcs, amelyik a gyerek[i] részében van, akkor: $k_1 \leq kulcs[1] \leq k_2 \leq kulcs[2] \leq \dots$
- Minden levél magassága egyenlő, ez a fa h magassága
 - Minimális fokszám: $t \geq 2$, maximális fokszám: $2t$
 - Minden nem gyökér csúcsnak legalább $t-1$ kulcsa van
 - Minden csúcsnak legfeljebb $2t-1$ kulcsa van
 - Egy csúcs telített, ha pontosan $2t-1$ kulcsa van
 - Gyökércsúcsnak legalább 1 kulcsa \square 2 gyereke van
 - Legegyszerűbb eset: $t=2 \square$ minden csúcsnak 2,3 vagy 4 gyereke lehet \square 2-3-4 fának nevezzük (gyakorlatban $t \gg 2$!)

BFa magassága

Tétel: Ha $n \geq 1$ és T egy n kulcsos BFa, magassága h és min.fokszáma $t \geq 2$ $h \leq \log_t(n+1)/2$

Biz: egy h magasságú bfa csúcsszáma akkor minimális, ha minden nem gyökér belső csúcsnak éppen $t-1$ kulcsa van. Ilyenkor:

$$n = 1 + 2t + 2t^2 + \dots + 2t^{h-1} = 1 + (t-1) \cdot \sum_{i=1}^h 2t^{i-1} = 1 + 2(t-1)(t^h - 1)/(t-1) = 2t^h - 1$$

Vagyis más fákra $n \geq 2t^h - 1 \rightarrow \log_t(n+1)/2 \geq h$

Keresés Bfában

BfaKeres(k)

i=1

while i ≤ n és k > kulcs[i]

i=i+1

if i ≤ n és k = kulcs[i] then

return (Me,i)

if levél then return NIL

else LemezrőlOlvas(gyerek[i])

return gyerek[i].BFaKeres(k)

Üres BFa készítése

ÚjBFa()

bfa=CsúcsotElhelyez()

bfa.levél=Igaz

bfa.n=0

LemezreÍr(bfa)

return(bfa)

Kulcs beszúrása Bfába

- Beszúrás mindig levélelemnél történik → a fa magassága nem változik → (egyfajta kiegyenlítetttség megmarad)
- Ha a beszúrási ponthoz vezető út bármelyik eleme telített, akkor felhasad két fele méretű csúcsra
- A fa nem a levelénél, hanem a gyökerénél nő

Kulcs törlése Bfából

- Hasonlít a beszúrásra, de kicsit bonyolultabb
- A minimális fokszámnál szélesebb csúcsokból törölünk. A keskenyebbeket egyesíteni próbáljuk
- A rekurzió során lefelé menően minden egyesíthető csúcsot egyesítünk, felfelé pedig törölünk

Kulcs törlése Bfából

Ha a k kulcs az x belső csúcsban nincs, akkor meghatározzuk annak a részfájának a csúcsát, amelyben benne lehet. A rekurzív hívást csak legalább t kulccsal (1 extra kulccsal) rendelkező részfákon hajtjuk végre, hogy legyen mit törölni. Ha mégis csak $t-1$ kulcsa lenne, akkor

- a. Ha van olyan testvére, amelynek legalább t gyereke van, akkor az x csúcsból a megelőző vagy rákövetkező elemet vigyünk a gyerekekbe, a testvéréből pedig 1 kulcsot vigyünk az x csúcsba.
- b. Ha nincs ilyen testvére, akkor egyesítsük a két testvért.
- c. Ha a gyökér két utolsó gyereket egyesítjük, akkor a gyökérkulcsot is vigyünk ide le, ilyenkor csökken a fa fokszáma