Fák
3. előadás

(Horváth Gyula és Szlávi Péter anyagai felhasználásával)

Fák

Bináris fa "fordított" ábrázolása, a levelektől vissza:

- Ha a bináris fa elemei címezhetőek is (pl. sorszámuk van), akkor elképzelhető egy olyan **statikusan láncolt** ábrázolás, amikor azt adjuk meg minden elemről, hogy ki van a fában fölötte.

```
Típus BFa=Tömb(1..Max,BFaelem)
      BFaelem=Rekord(érték: Elemtípus,
                     szülő: 0..Max)
```

Egy ilyen fára persze másféle műveleteket definiálhatunk.

Fák

```
Üres(bf):
  Ciklus i=1-től N-ig
    bf(i).szülő:=0
  Ciklus vége
Függvény vége.
Beilleszt(bf,a,b):  {a szülője b-nek}
  bf(b).szülő:=a
Függvény vége.
```

Fák

```
Ősszülő(bf,e):
  Ha bf(e).szülő=0 akkor Ősszülő:=e
  különben Ősszülő:=Ősszülő(bf,bf(e).szülő)
Függvény vége.
Őse?(bf,utód,ős):
  Ha utód=ős akkor Őse?:=igaz
  különben ha bf(utód).szülő=0 akkor Őse?:=hamis
  különben Őse?:=Őse?(bf,bf(utód).szülő,ős)
Függvény vége.
```

Megjegyzés: egyetlen fa esetén a bf tömb lehet globális változó is.

Fák

```
Ősszülő(bf,e):
  Ciklus amíg bf(e).szülő≠0
    e:=bf(e).szülő
  Ciklus vége
Függvény vége.
```

```
Őse?(bf,utód,ős):
  Ciklus amíg utód≠ős és bf(utód).szülő≠0
    utód:=bf(utód).szülő
  Ciklus vége
  Őse?:=utód=ős
Függvény vége.
```

Ugyanez ciklussal.

Fák

Bináris fa "fordított" ábrázolása, a levelektől vissza:

- Ha a bináris fa elemei címezhetőek is (pl. sorszámuk van), akkor elképzelhető egy olyan **dinamikusan láncolt** ábrázolás, amikor azt adjuk meg minden elemről, hogy ki van a fában fölötte.

```
Típus BFa=Tömb(1..Max,BFaelemMutató)
    BFaelem=Rekord(érték: Elemtípus,
                   szülő: BFaelemMutató)
```

Egy ilyen fára persze másféle művele-teket definiálhatunk.

Fák

```
Üres(bf):
  Ciklus i=1-től N-ig
    Lefoglal(bf(i),(i,sehova))
  Ciklus vége
Függvény vége.

Beilleszt(bf,a,b):  {a szülője b-nek}
  Tartalom(bf(b)).BFaelemMutató:=bf(a)
Függvény vége.
```

Nem bináris fák

A fa rekurzív adatszerkezet jellemzői:

- sokaság: azonos típusú elemekből áll;
- akár 0 db elemet tartalmazhat;
- Üres: rekurzív „nullelem”, kitüntetett konstans;
- Fraktál (=önhasonlóság) tulajdonság: a részei ugyanolyan szerkezetűek, mint az egész;
- nem lineárisan rendezett (azaz nem sorozatféle): bármely elemének 0, 1, 2... (közvetlen) rákövetkezője lehet;

- minden elemnek legfeljebb egy (köz-vetlen) előzője van.

Nem bináris fák

A fa rekurzív adatszerkezet ábrázolása:

```
Típus TFa=Rekord
          (elem: TElem
           ágak: Sorozat(TFa))
```

Egy speciális változat (ágszám felső korláttal):

```
Típus TFa=Rekord
           (elem: TElem
            db: Egész
            ág: Tömb(1..Max,TFa))
```

A sorozat lehet pl. lista.

Nem bináris fák

A fa rekurzív adatszerkezet műveletei:

```
Üres:Fa
üres?(Fa):Logikai
Egyeleműfa(Elem):Fa
gyerekszám(Fa):Egész
Beilleszt(Fa,Fa):Fa ∪ {NemDef}
Gyökérelem(Fa):Elem ∪ {NemDef}
Gyökérmódosít(Fa,Elem):Fa ∪ {NemDef}
Gyerek(Fa,Egész):Fa ∪ {NemDef}
```

Nem bináris fák

A fa műveletei megvalósítása dinamikus láncolással:

```
Üres(f):
  f:=sehova
Eljárás vége.
üres?(f):
  üres?:=(f=sehova)
Függvény vége.
Egyeleműfa(e):
  Lefoglal(f,(e,Üressorozat)); Egyeleműfa:=f
Függvény vége.
```

Nem bináris fák

A fa műveletei megvalósítása dinamikus láncolással:

```
gyerekszám(f):
  gyerekszám:=Elemszám(Tartalom(f).ágak)
Függvény vége.
Gyökérelem(f):
  Gyökérelem:=Tartalom(f).elem
Függvény vége.
Gyökérmódosít(f,e):
  Tartalom(f).elem:=e
Eljárás vége.
```

gyerekszám(f):
gyerekszám:=Tartalom(f).db
Függvény vége.

Nem bináris fák

Sorozattal ábrázolva:

```
Beilleszt(mire,mit):
  Tartalom(mire).ágak:=
                Végére(Tartalom(mire).ágak,mit)
Eljárás vége.
Gyerek(f,i):
  Ha i≤Elemszám(Tartalom(f).ágak)
     akkor Gyerek:=Tartalom(f).ágak(i)
  különben Gyerek:=Üres
Függvény vége.
```

Nem bináris fák

Tömbbel ábrázolva:

```
Beilleszt(mire,mit):
  Tartalom(mire).db:=Tartalom(mire).db+1
  Tartalom(mire).ág(Tartalom(mire).db):=mit
Eljárás vége.
Gyerek(f,i):
  Ha i≤Tartalom(f).db
     akkor Gyerek:=Tartalom(f).ág(i)
  különben Gyerek:=Üres
Függvény vége.
```

Nem bináris fák – alkalmazás

A fa bejárása (gyökérkezdő):

```
Bejárás(f):
  Ha nem üres?(f) akkor
           Feldolgoz(Tartalom(f).elem)
           Ciklus i=1-től Tartalom(f).db-ig
             Bejárás(Tartalom(f).ág(i))
           Ciklus vége
Függvény vége.
```

Nem bináris fák – alkalmazás

A fa bejárása (gyökérvégző):

```
Bejárás(f):
  Ha nem üres?(f) akkor
           Ciklus i=1-től Tartalom(f).db-ig
             Bejárás(Tartalom(f).ág(i))
           Ciklus vége
           Feldolgoz(Tartalom(f).elem)
Függvény vége.
```

Nem bináris fák – alkalmazás

A fa elemszáma

```
elemszám(f):
  Ha üres?(f) akkor elemszám:=0
  különben s:=1
           Ciklus i=1-től Tartalom(f).db-ig
             s:=s+elemszám(Tartalom(f).ág(i))
           Ciklus vége
           elemszám:=s
Függvény vége.
```

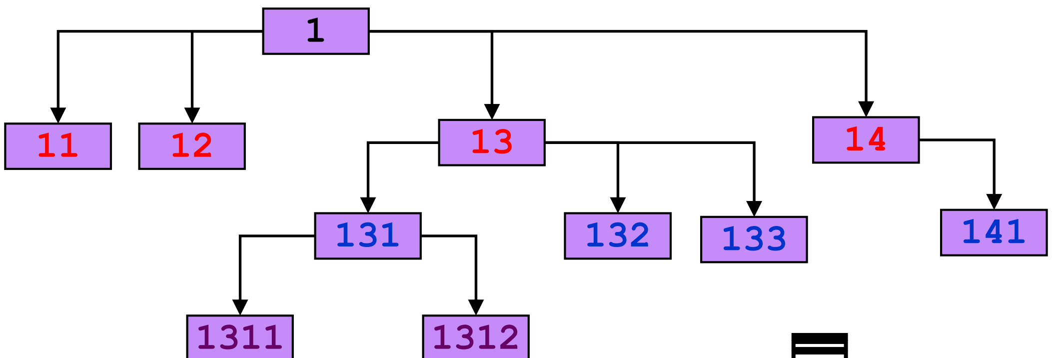
Nem bináris fák – alkalmazás

A fa magassága

```
magasság(f):
  Ha üres?(f) akkor magasság:=0
  különben max:=0
           Ciklus i=1-től Tartalom(f).db-ig
             m:=magasság(Tartalom(f).ág(i))
             Ha m>max akkor max:=m
           Ciklus vége
           magasság:=max+1
Függvény vége.
```

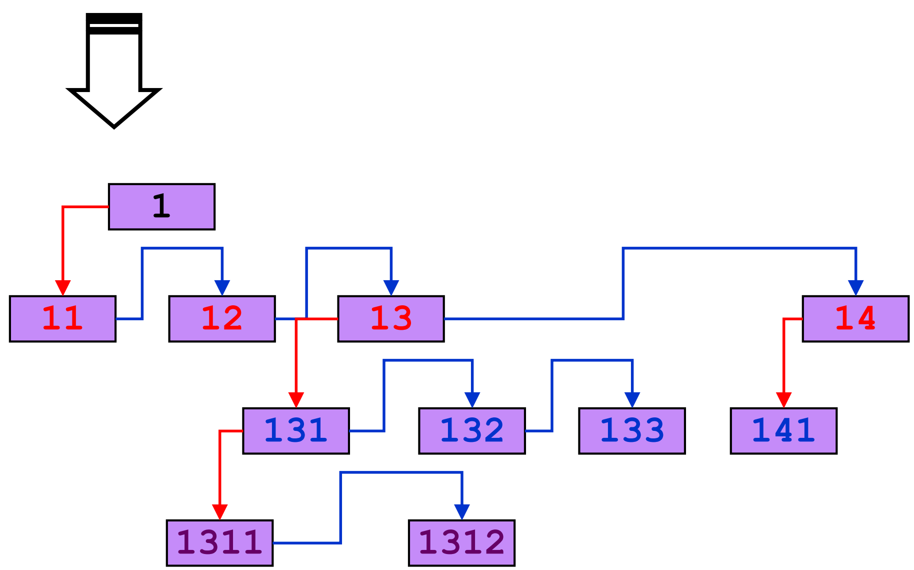
Nem bináris fák – binárisan

Bináris ábrázolás:



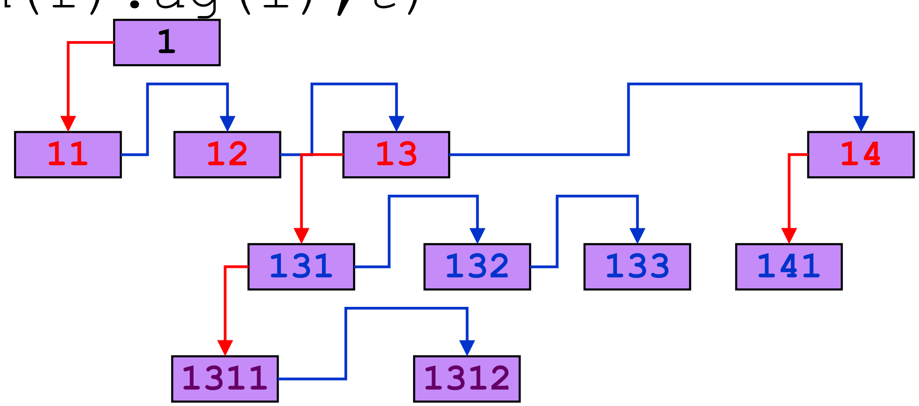
balra az 1. gyerek

jobbra a következő testvér



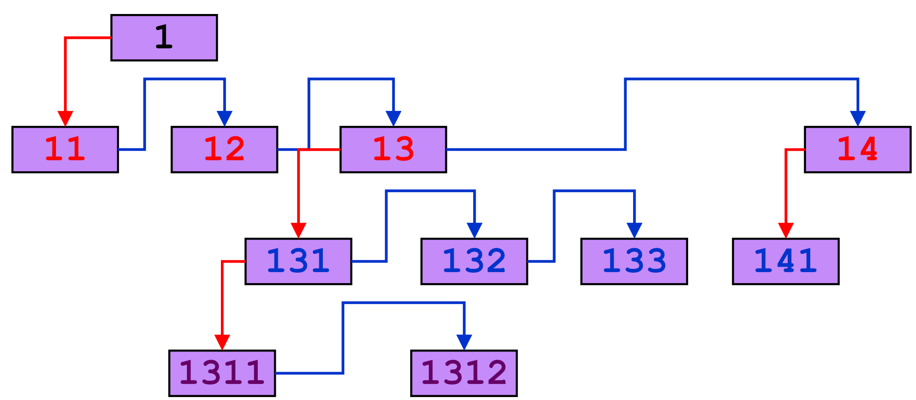
Nem bináris fák – bináris ábrázolásra alakítás

```
Átalakít(f,bf):
  Egyelemûfa(bf,Tartalom(f).elem)
  Ha Tartalom(f).db>0 akkor
     Balraillieszt(bf,Tartalom(f).ág(1));
     t:=balgyerek(bf)
     Átalakít(Tartalom(f).ág(1),t)
     Ciklus i=2-től Tartalom(f).db-ig
       Jobbraillieszt(t,Tartalom(f).ág(1))
       t:=jobbgyerek(t);
       Átalakít(Tartalom(f).ág(i),t)
     Ciklus vége
Függvény vége.
```



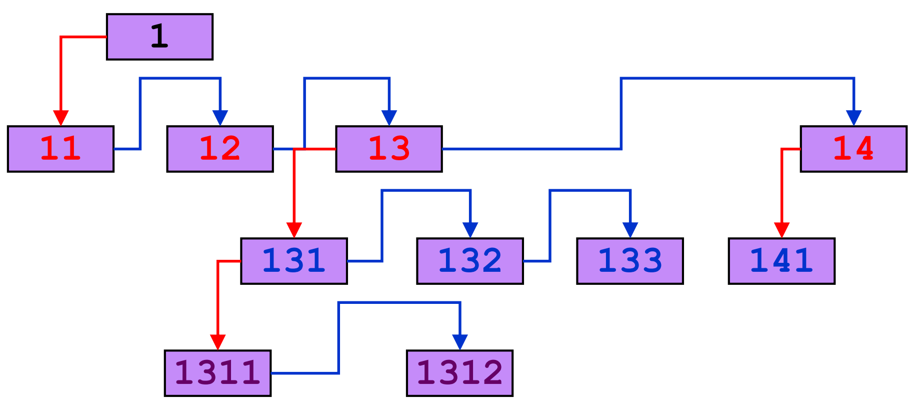
Nem bináris fák – binárisan

```
Gyerek(f,i):
   Ha i=1 akkor Gyerek:=Tartalom(f).bal
   különben Gyerek:=Testvér(Tartalom(f).bal,i)
Függvény vége.
Testvér(f,i):
   Ha i=2 akkor Testvér:=Tartalom(f).jobb
   különben Testvér:=Testvér(Tartalom(f).jobb,i-1)
Függvény vége.
```



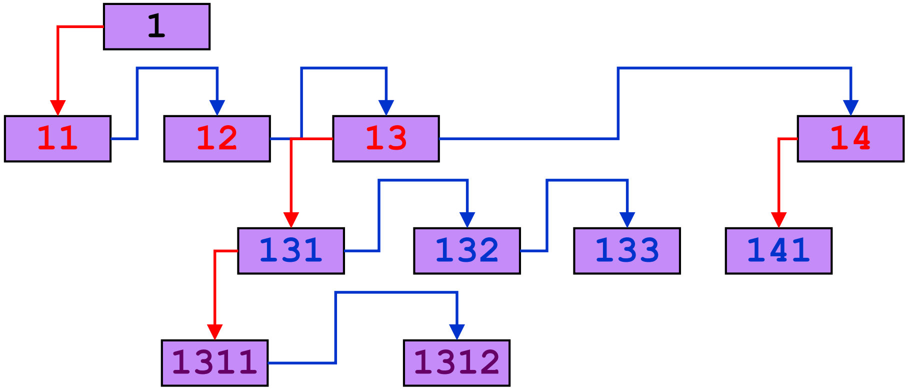
Nem bináris fák – binárisan

```
Elsőgyerek(f):
  Elsőgyerek:=Tartalom(f).bal
Függvény vége.
Következőgyerek(gy):
  Következőgyerek:=Tartalom(gy).jobb
Függvény vége.
Vanméggyerek(gy):
  Vanméggyerek:=(Tartalom(gy).jobb≠sehova)
Függvény vége.
```



Nem bináris fák – binárisan

```
Gyerek(f,i):
  gy:=Elsőgyerek(f)
  Ha i=1 akkor Gyerek:=gy
      különben Ciklus j=2-től i-ig
                  gy:=Következőgyerek(gy)
               Ciklus vége
               Gyerek:=gy
Függvény vége.
```



Nem bináris fák – binárisan

Fa "fordított" ábrázolása, a levelektől vissza:

- Ha a fa elemei címezhetőek is (pl. sorszámuk van), akkor elképzelhető egy olyan statikusan láncolt ábrázolás, amikor azt adjuk meg minden elemről, hogy ki van a fában fölötte.

```
Típus Fa=Tömb(1..Max,Faelem)
      Faelem=Rekord(érték: Elemtípus,
                    szülő: 0..Max)
```

Ez ugyanaz, mint a bináris fáknál!

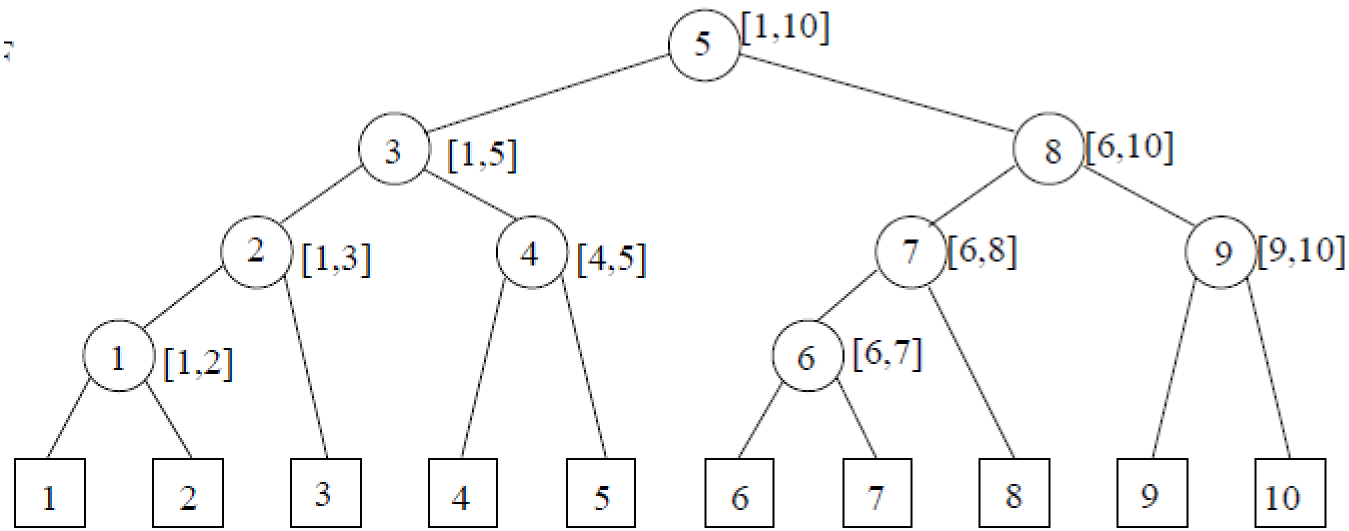
Kérdezőfa

A kérdezőfa egy olyan bináris fa, amelyre teljesülnek az alábbi tulajdonságok:

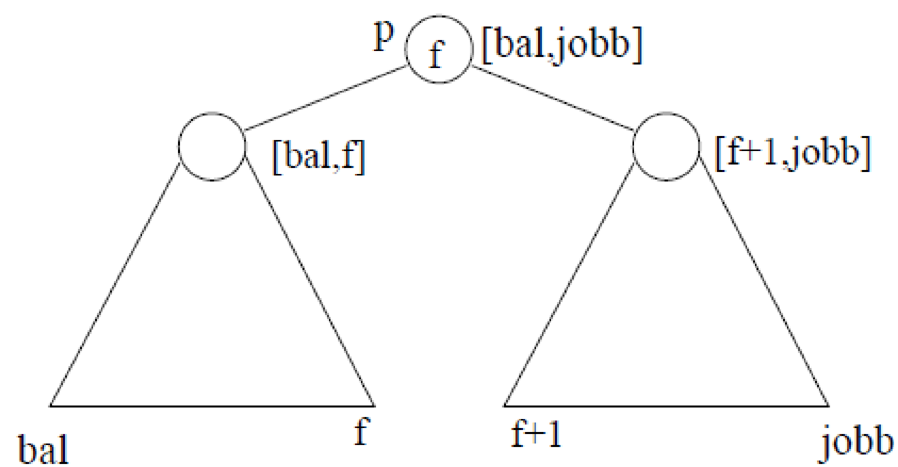
- A fának n levele van, amelyek balról jobbra sorrendben az 1,…,n számokat tartalmazzák.
- A fának n-1 belső pontja van, mindegyiknek 2 gyereke.
- Minden p belső pont a p bal-részfájában levő levélértékek maximumát tartalmazza.
- Minden kérdéshez költségek tartoznak, vagy a kérdések száma korlátozott vagy a nem válaszok száma korlátozott...

A kérdezőfát úgy építjük fel, hogy a kérdések összköltsége minimális legyen!

Kérdezőfa



5 [1,10]
3 [1,5]
8 [6,10]
2 [1,3]
4 [4,5]
7 [6,8]
9 [9,10]
1 [1,2]
6 [6,7]
1
2
3
4
5
6
7
8
9
10



p
f
[bal,jobb]
[bal,f]
[f+1,jobb]
bal
f
f+1
jobb

Kérdezőfa

A kérdezőfa használata:

- Induljunk ki a fa gyökeréből (az 1-es elemből)!
- Amíg az aktuális pont nem levél, kérdezzünk rá az aktuális ponthoz tartozó értékre (a keresett érték ≤-e nála)!
- Ha a válasz igen, akkor lépjünk a fában balra, egyébként pedig jobbra!

Statikus láncolással:

```
Típus Kérdezőfa=Tömb(1..Max,Faelem)
      Faelem=Rekord(érték,ár: Egész,
                    bal,jobb: 0..max)
```

Kérdezőfa

```
Kérdés(kf,p,S):
  p:=1; S:=0
  Ciklus amíg kf(p).bal>0 {vagy kf(p.jobb)>0}
    S:=S+kf(p).érték; Ki: p; Be: V
    Ha V="igen" akkor p:=kf(p).bal
             különben p:=kf(p).jobb
  Ciklus vége
Eljárás vége.
```

Kérdezőfa

Kérdezőfa előállítása kérdésköltségek esetén:

- Jelölje K(x) a kérdezőfában a gyökértől az x levélig haladó úton a belső pontok kérdéseihez tartozó értékek összegét.
- Ha a gondolt szám x, akkor a kitalálásának költsége K(x).
- Az a kérdezőfa optimális, amelyre a K(x) értékek maximuma a lehető legkisebb.

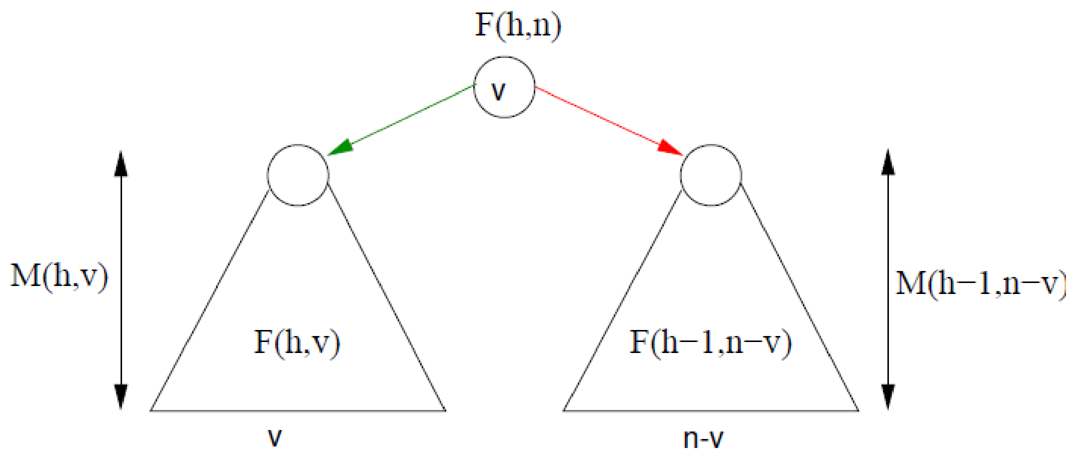
A kérdezőfa előállítása:

dinamikus programozás!

Kérdezőfa

h-hibázó optimális kérdezőfa

Legyen $M(h,n)$ az n elemre h-hibázó optimális (legkisebb magasságú) kérdezőfa magassága, $F(h,n)$ pedig az optimális fa.



$1<v<n$

$$M(h,n)=1+\max(M(h,v), M(h-1,n-v))$$

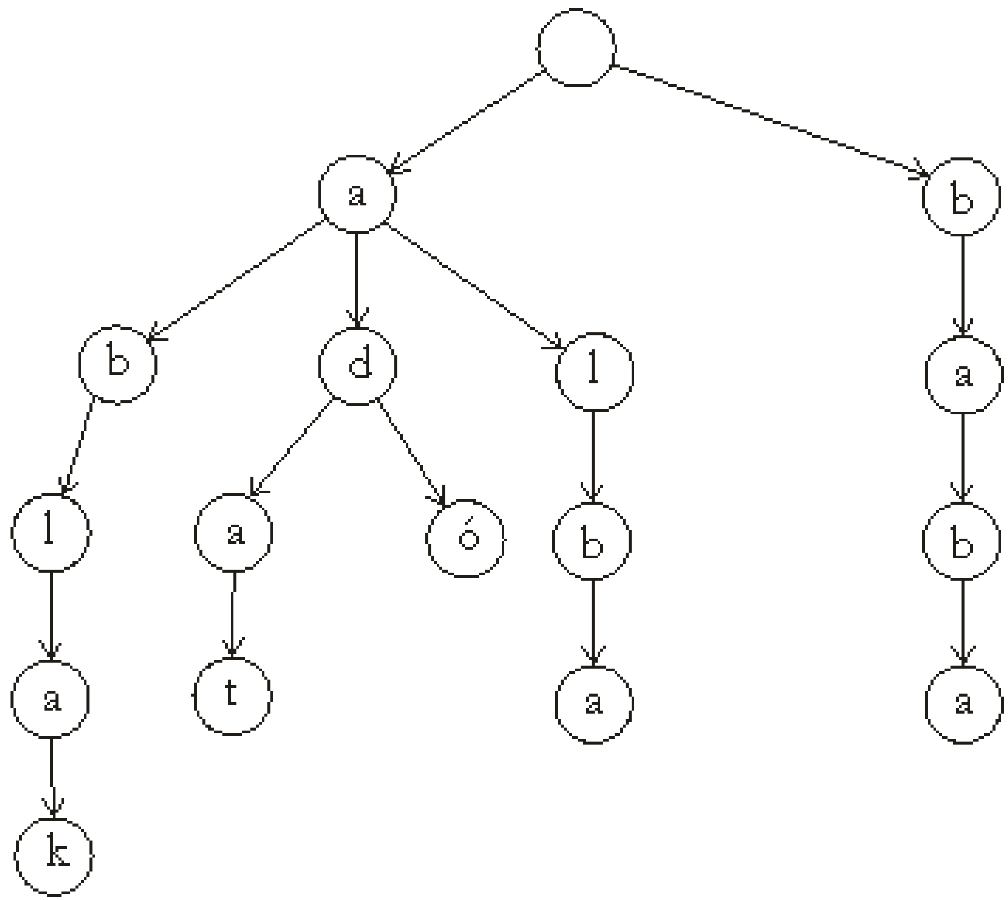
A kérdezőfa előállítása:

dinamikus programozás!

Szófák

A szófa:

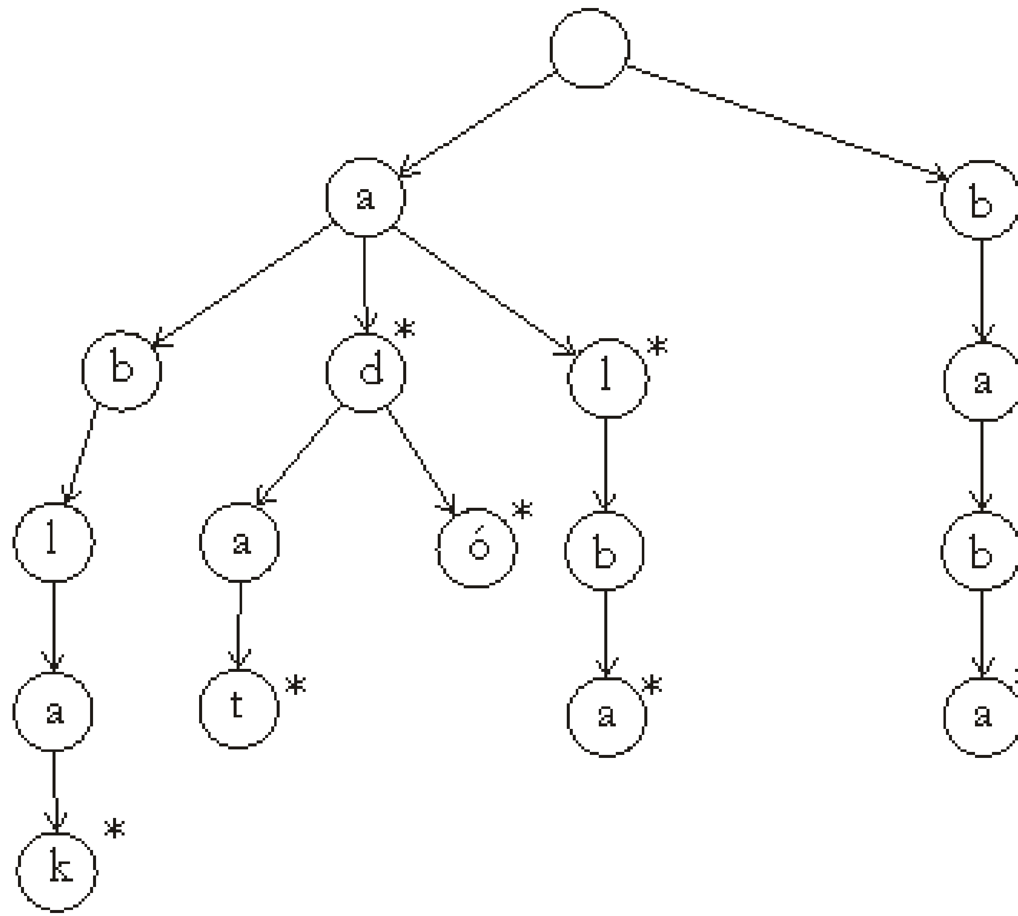
- nem bináris fa
- a gyökere üres
- a szavak felülről lefelé olvashatók
- hol a szó vége?
- hány gyerek lehet?



Szófák

A szófa:

- jelöljük a szóvégeket!



Szófák

A szófa ábrázolása:

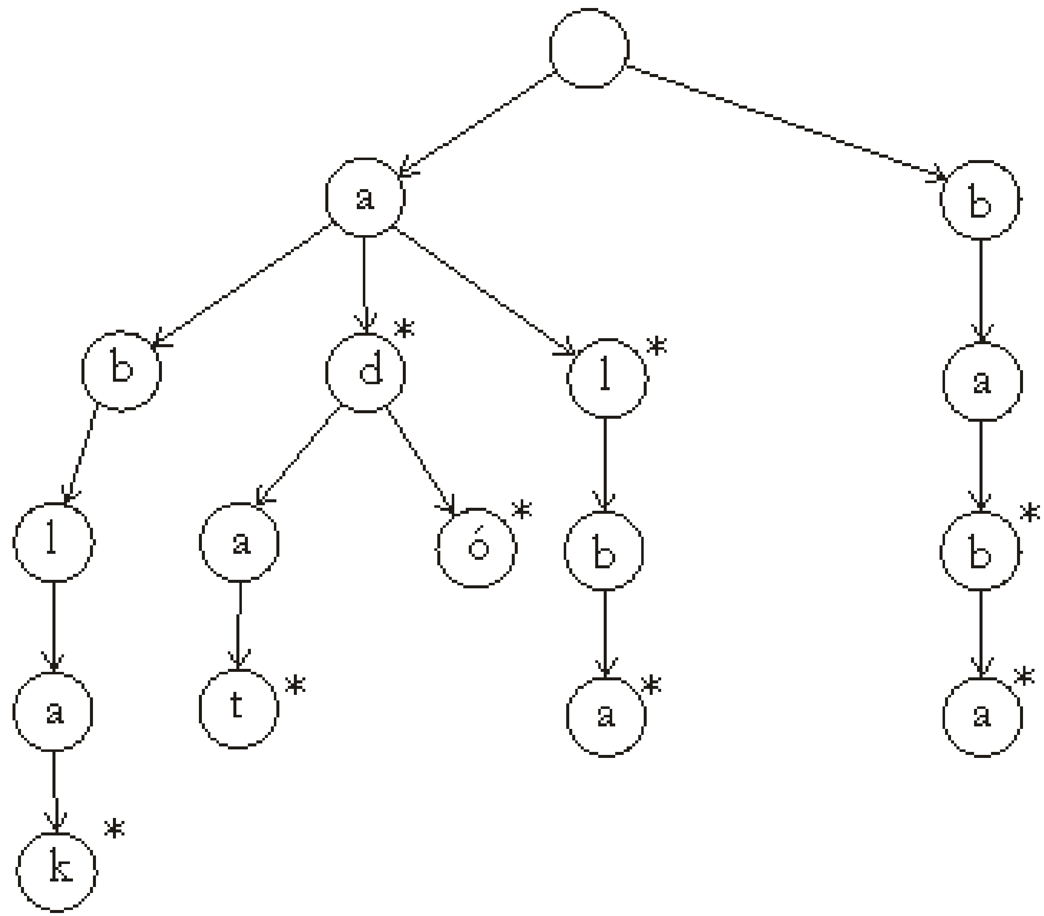
- indexeljük a következő betűvel a szófa további részét;
- jelezzük, hogy szó végén vagyunk-e!

```
Típus SzóFa=Rekord(betű:Tömb('a'..'z':Szófa,
                         vége:Logikai)
```

Szófák

Szó keresése esetei:

- adó
- bab
- adós
- alma
- abl



Szófák

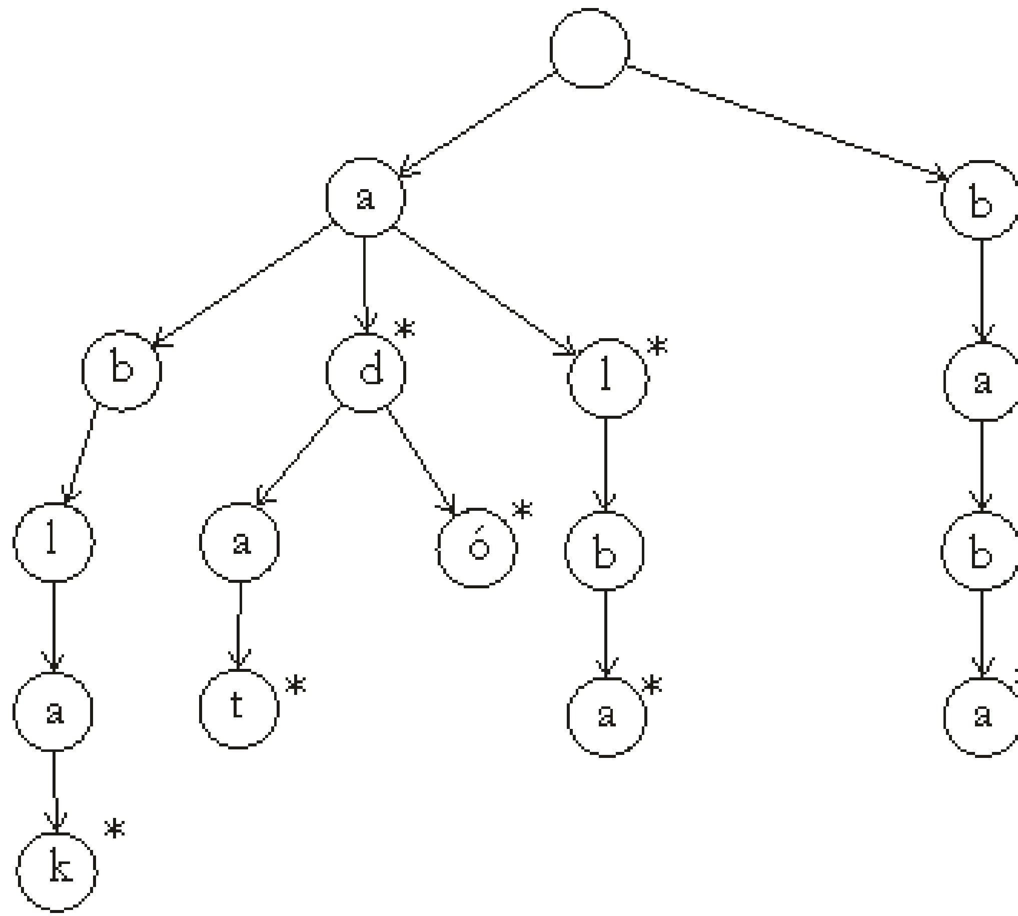
Szó keresése:

```
Bennevan?(szf,s):
  Ha Tartalom(szf).vége és üres?(s)
     akkor Bennevan?:=igaz
  különben ha üres?(s) akkor Bennevan?:=hamis
  különben ha üres?(Tartalom(szf).betű(első(s)))
     akkor Bennevan?:=hamis
  különben Bennevan?:=Bennevan?(Tartalom(szf).
                  betű(első(s)),elsőnélküli(s))
Függvény vége.
```

Szófák

Szó beillesztés esetei:

- ada
- adós
- alma



Szófák

Szó beillesztése:

```
Beilleszt(szf,s):
  Ha üres?(s) akkor Tartalom(szf).vége:=igaz
  különben
    Ha üres?(Tartalom(szf).betű(első(s)))
       akkor Lefoglal(Tartalom(szf).betű(első(s))
    Beilleszt(Tartalom(szf).betű(első(s)),
              elsőnélküli(s))
Függvény vége.
```

Szófák

Problémák:

- minden elemhez annyi mutató kell, ahány betű van az ábécében;
- kérdéses a magyar ábécével való indexelés;
- sok faelemnél nincs is elágazás, azaz egyetlen ág meg tovább.

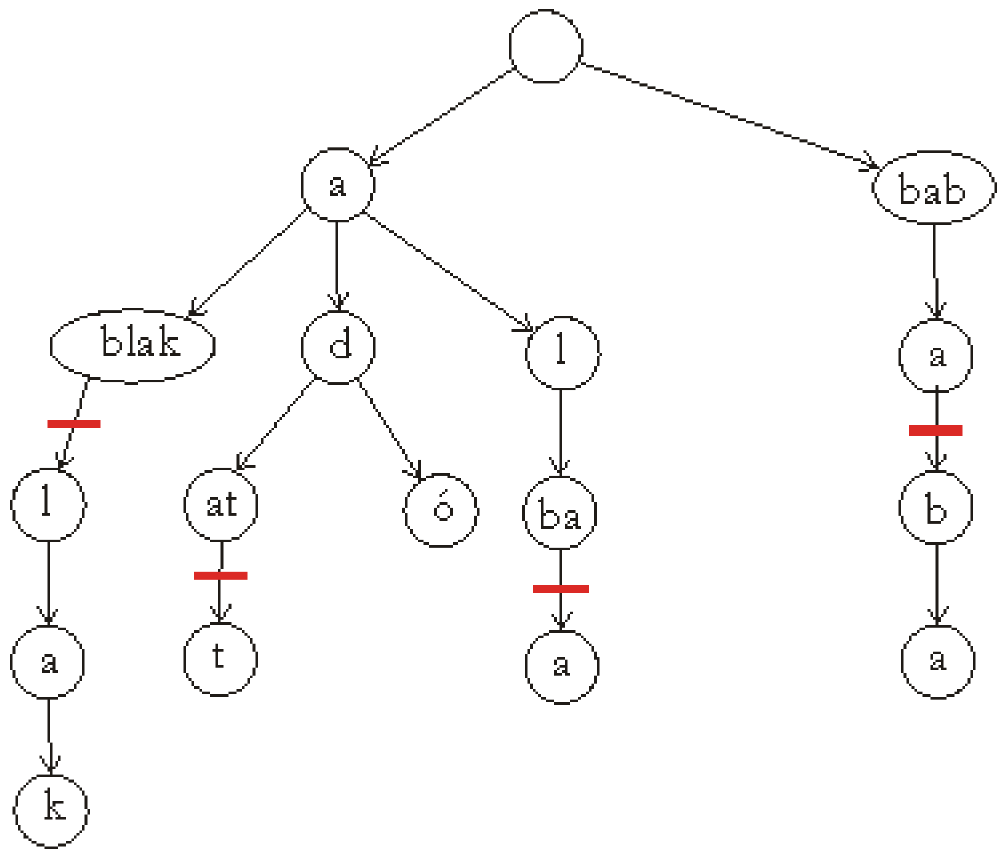
Ötlet: Indexelés helyett logaritmikus keresés rendezett tömbben.

Kérdés: Lineáris is elég rendezetlen tömbben?

Szófák

A szófa tömörítése:

- vonjuk össze az elágazás nélküli csomópontokat!



Fák

3. előadás vége