

# Dinamikus programozás

Hajnal Péter

Bolyai Intézet, TTIK, SZTE, Szeged

2021. ősz

# Az alapötlet

# Az alapötlet

- A dinamikus programozás alapötlete egyszerű. Az alapfeladatot egy nagyobb „feladat-sokaságba” ágyazzuk.

# Az alapötlet

- A dinamikus programozás alapötlete egyszerű. Az alapfeladatot egy nagyobb „feladat-sokaságba” ágyazzuk.
- Az alapfeladat a sokaság egyik feladata. A sokaságnak azonban lesznek nagyon egyszerű esetei, amelyek megválaszolása senkit sem állít kihívás elé.

# Az alapötlet

- A dinamikus programozás alapötlete egyszerű. Az alapfeladatot egy nagyobb „feladat-sokaságba” ágyazzuk.
- Az alapfeladat a sokaság egyik feladata. A sokaságnak azonban lesznek nagyon egyszerű esetei, amelyek megválaszolása senkit sem állít kihívás elé.
- A sokaság feladatainak lesz egy természetes sorrendje (ez lehet rendezés, de lehet részben rendezés is).

# Az alapötlet

- A dinamikus programozás alapötlete egyszerű. Az alapfeladatot egy nagyobb „feladat-sokaságba” ágyazzuk.
- Az alapfeladat a sokaság egyik feladata. A sokaságnak azonban lesznek nagyon egyszerű esetei, amelyek megválaszolása senkit sem állít kihívás elé.
- A sokaság feladatainak lesz egy természetes sorrendje (ez lehet rendezés, de lehet részben rendezés is).
- A feladatok sorrendben való elvégése egyszerű lesz.

# Az alapötlet

- A dinamikus programozás alapötlete egyszerű. Az alapfeladatot egy nagyobb „feladat-sokaságba” ágyazzuk.
- Az alapfeladat a sokaság egyik feladata. A sokaságnak azonban lesznek nagyon egyszerű esetei, amelyek megválaszolása senkit sem állít kihívás elé.
- A sokaság feladatainak lesz egy természetes sorrendje (ez lehet rendezés, de lehet részben rendezés is).
- A feladatok sorrendben való elvégzése egyszerű lesz. A korábban megoldott feladatokra alapozva az elsőre nehéz kérdések is könnyen megválaszolhatók lesznek.

# Az ötlet „matematikai tartalma”



# Az ötlet „matematikai tartalma”

- A kihívás a jó feladat-sokaság megválasztása.

# Az ötlet „matematikai tartalma”

- A kihívás a jó feladat-sokaság megválasztása. Ha ezt jól választottuk meg, akkor feladataink „felgöngyölítése” gyakran rutinból megoldható.

# Az ötlet „matematikai tartalma”

- A kihívás a jó feladat-sokaság megválasztása. Ha ezt jól választottuk meg, akkor feladataink „felgöngyölítése” gyakran rutinból megoldható.
- Egy algoritmikus probléma-sereg dinamikus programozással történő megoldása nagy hasonlóságot mutat egy állítás-sereg teljes indukciós bizonyításával.

# Dinamikus programozás vs rekurzió

# Dinamikus programozás vs rekurzió

- A dinamikus programozást a teljes indukcióhoz „hasonlítottuk”.

# Dinamikus programozás vs rekurzió

- A dinamikus programozást a teljes indukcióhoz „hasonlítottuk”.
- Igazából egy rekurzióról van szó.

# Dinamikus programozás vs rekurzió

- A dinamikus programozást a teljes indukcióhoz „hasonlítottuk”.
- Igazából egy rekurzióról van szó.
- Lényeges különbség van a dinamikus programozás és emögött lévő rekurziós gondolat programozásilag leírt rekurzív kódolása között.

# Dinamikus programozás vs rekurzió

- A dinamikus programozást a teljes indukcióhoz „hasonlítottuk”.
- Igazából egy rekurzióról van szó.
- Lényeges különbség van a dinamikus programozás és emögött lévő rekurziós gondolat programozásilag leírt rekurzív kódolása között.
- Formális leírás helyett nézzünk példákat.



# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1,

# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1, 1,

# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1, 1, 2,

# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1, 1, 2, 3,

# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1, 1, 2, 3, 5,

# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1, 1, 2, 3, 5, 8,

# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1, 1, 2, 3, 5, 8, 13,



# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1, 1, 2, 3, 5, 8, 13, 21,

# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1, 1, 2, 3, 5, 8, 13, 21, 34,

# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1, 1, 2, 3, 5, 8, 13, 21, 34, 55,

# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89,

# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144,

# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233,

# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377,

# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610,



# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987,

# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597,

# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584,

# A Fibonacci-számok definíciója

## Definíció: Fibonacci-számok

$$F_1 = F_2 = 1,$$

ha  $n > 2$ , akkor

$$F_n = F_{n-1} + F_{n-2}.$$

## Példa az első Fibonacci-számokra

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, ...

# A Fibonacci-számok és a dinamikus programozás

# A Fibonacci-számok és a dinamikus programozás

- Az  $F_{18}$  Fibonacci-számot a következő „programmal” számolhatjuk ki:

# A Fibonacci-számok és a dinamikus programozás

- Az  $F_{18}$  Fibonacci-számot a következő „programmal” számolhatjuk ki:

```
var F(1..18): natural;  
for i=1 to 18 do  
    if n=1 or 2 then F(i)=1  
    if i>2 then F(i)=F(i-1)+ F(i-2);  
print F(18)
```

# A Fibonacci-számok és a dinamikus programozás

- Az  $F_{18}$  Fibonacci-számot a következő „programmal” számolhatjuk ki:

```
var F(1..18): natural;  
for i=1 to 18 do  
    if n=1 or 2 then F(i)=1  
    if i>2 then F(i)=F(i-1)+ F(i-2);  
print F(18)
```

- Ez a dinamikus programozás logikáját követi.



# A Fibonacci-számok és a dinamikus programozás

- Az  $F_{18}$  Fibonacci-számot a következő „programmal” számolhatjuk ki:

```
var F(1..18): natural;  
for i=1 to 18 do  
    if n=1 or 2 then F(i)=1  
    if i>2 then F(i)=F(i-1)+ F(i-2);  
print F(18)
```

- Ez a dinamikus programozás logikáját követi. Egy 10 hosszú tömbbel dolgozik.

# A Fibonacci-számok és a dinamikus programozás

- Az  $F_{18}$  Fibonacci-számot a következő „programmal” számolhatjuk ki:

```
var F(1..18): natural;  
for i=1 to 18 do  
    if n=1 or 2 then F(i)=1  
    if i>2 then F(i)=F(i-1)+ F(i-2);  
print F(18)
```

- Ez a dinamikus programozás logikáját követi. Egy 10 hosszú tömbbel dolgozik. A futás végén mind a 10 szám ott lesz a memóriában,

# A Fibonacci-számok és a dinamikus programozás

- Az  $F_{18}$  Fibonacci-számot a következő „programmal” számolhatjuk ki:

```
var F(1..18): natural;  
for i=1 to 18 do  
    if n=1 or 2 then F(i)=1  
    if i>2 then F(i)=F(i-1)+ F(i-2);  
print F(18)
```

- Ez a dinamikus programozás logikáját követi. Egy 10 hosszú tömbbel dolgozik. A futás végén mind a 10 szám ott lesz a memóriában, pedig számunkra csak  $F_{10}$  fontos.

# A Fibonacci-számok és a dinamikus programozás

- Az  $F_{18}$  Fibonacci-számot a következő „programmal” számolhatjuk ki:

```
var F(1..18): natural;  
for i=1 to 18 do  
    if n=1 or 2 then F(i)=1  
    if i>2 then F(i)=F(i-1)+ F(i-2);  
print F(18)
```

- Ez a dinamikus programozás logikáját követi. Egy 10 hosszú tömbbel dolgozik. A futás végén mind a 10 szám ott lesz a memóriában, pedig számunkra csak  $F_{10}$  fontos.
- A dinamikus programozási algoritmusok sebessége attól függ hány részfeladatot tartalmaz feladat-seregünk.

# A Fibonacci-számok és a dinamikus programozás

- Az  $F_{18}$  Fibonacci-számot a következő „programmal” számolhatjuk ki:

```
var F(1..18): natural;  
for i=1 to 18 do  
    if n=1 or 2 then F(i)=1  
    if i>2 then F(i)=F(i-1)+ F(i-2);  
print F(18)
```

- Ez a dinamikus programozás logikáját követi. Egy 10 hosszú tömbbel dolgozik. A futás végén mind a 10 szám ott lesz a memóriában, pedig számunkra csak  $F_{10}$  fontos.
- A dinamikus programozási algoritmusok sebessége attól függ hány részfeladatot tartalmaz feladat-seregünk.
- A dinamikus programozás formailag gyakran úgy jelentkezik, hogy egy számsorozatot, vagy egy táblázatot, vagy bonyolultabb szám-sereget töltünk ki meghatározott sorrendben.

# A Fibonacci-számok és a matematikai rekurzió

# A Fibonacci-számok és a matematikai rekurzió

- A Fibonacci-számokat a következő program is kiszámolja:

# A Fibonacci-számok és a matematikai rekurzió

- A Fibonacci-számokat a következő program is kiszámolja:

```
Fibonacci(n):  if n>2 return Fibonacci(n-1)+ F(n-2)
else return 1;
print Fibonacci(18).
```



# A Fibonacci-számok és a matematikai rekurzió

- A Fibonacci-számokat a következő program is kiszámolja:

```
Fibonacci(n):  if n>2 return Fibonacci(n-1)+ F(n-2)
else return 1;
print Fibonacci(18).
```

- Mit csinál ezzel a gép? Egy bonyolult gépi szintű feloldás történik.

# A Fibonacci-számok és a matematikai rekurzió

- A Fibonacci-számokat a következő program is kiszámolja:

```
Fibonacci(n):  if n>2 return Fibonacci(n-1)+ F(n-2)
else return 1;
print Fibonacci(18).
```

- Mit csinál ezzel a gép? Egy bonyolult gépi szintű feloldás történik.
- Emiatt a rekurzív program futása lassabb lesz (sok újra számolás történik).

# Szünet



# Az alapkérdés

# Az alapkérdés

A legnagyobb független ponthalmaz megkeresése egy fában

Adott egy  $T$  fa. Határozzuk meg a legnagyobb független halmazának méretét.

# Az alapkérdés

A legnagyobb független ponthalmaz megkeresése egy fában

Adott egy  $T$  fa. Határozzuk meg a legnagyobb független halmazának méretét.

Fa

Egy gráf fa egy körmentes, összefüggő gráf.

# Az alapkérdés

A legnagyobb független ponthalmaz megkeresése egy fában

Adott egy  $T$  fa. Határozzuk meg a legnagyobb független halmazának méretét.

Fa

Egy gráf fa egy körmentes, összefüggő gráf.

Független csúcshalmaz gráfban

Egy  $F$  csúcshalmaz független, ha nincs olyan él amely mindkét végpontja  $F$ -beli.

# Gyökeres fák



# Gyökeres fák

- Először jelöljük ki egy speciális  $r \in V(T)$  csúcsot, amit gyökérnek nevezünk.

# Gyökeres fák

- Először jelöljük ki egy speciális  $r \in V(T)$  csúcsot, amit gyökérnek nevezünk.
- A  $(T, r)$  gyökeres fa új elnevezéseket indukál.

# Gyökeres fák

- Először jelöljük ki egy speciális  $r \in V(T)$  csúcsot, amit gyökérnek nevezünk.
- A  $(T, r)$  gyökeres fa új elnevezéseket indukál.
- A csúcshalmaz diszjunkt osztályokba sorolható a gyökértől vett távolság szerint.

# Gyökeres fák

- Először jelöljük ki egy speciális  $r \in V(T)$  csúcsot, amit gyökérnek nevezünk.
- A  $(T, r)$  gyökeres fa új elnevezéseket indukál.
- A csúcshalmaz diszjunkt osztályokba sorolható a gyökértől vett távolság szerint.
- Legyen  $G_i$  a gyökértől  $i$  távolságra lévő csúcsok halmaza.

# Gyökeres fák

- Először jelöljük ki egy speciális  $r \in V(T)$  csúcsot, amit gyökérnek nevezünk.
- A  $(T, r)$  gyökeres fa új elnevezéseket indukál.
- A csúcshalmaz diszjunkt osztályokba sorolható a gyökértől vett távolság szerint.
- Legyen  $G_i$  a gyökértől  $i$  távolságra lévő csúcsok halmaza.  
 $G_0 = \{r\}$ ,

# Gyökeres fák

- Először jelöljük ki egy speciális  $r \in V(T)$  csúcsot, amit gyökérnek nevezünk.
- A  $(T, r)$  gyökeres fa új elnevezéseket indukál.
- A csúcshalmaz diszjunkt osztályokba sorolható a gyökértől vett távolság szerint.
- Legyen  $G_i$  a gyökértől  $i$  távolságra lévő csúcsok halmaza.  
 $G_0 = \{r\}$ ,  $G_1$  a gyökér szomszédait tartalmazó csúcshalmaz,

# Gyökeres fák

- Először jelöljük ki egy speciális  $r \in V(T)$  csúcsot, amit gyökérnek nevezünk.
- A  $(T, r)$  gyökeres fa új elnevezéseket indukál.
- A csúcshalmaz diszjunkt osztályokba sorolható a gyökértől vett távolság szerint.
- Legyen  $G_i$  a gyökértől  $i$  távolságra lévő csúcsok halmaza.  $G_0 = \{r\}$ ,  $G_1$  a gyökér szomszédait tartalmazó csúcshalmaz,  $G_2$  a gyökér másodsomszédainak halmaza.

# Gyökeres fák

- Először jelöljük ki egy speciális  $r \in V(T)$  csúcsot, amit gyökérnek nevezünk.
- A  $(T, r)$  gyökeres fa új elnevezéseket indukál.
- A csúcshalmaz diszjunkt osztályokba sorolható a gyökértől vett távolság szerint.
- Legyen  $G_i$  a gyökértől  $i$  távolságra lévő csúcsok halmaza.  $G_0 = \{r\}$ ,  $G_1$  a gyökér szomszédait tartalmazó csúcshalmaz,  $G_2$  a gyökér másodsomszédainak halmaza.

Minden  $e \in E(T)$  él esetén van olyan  $i \in \mathbb{N}$ , hogy  $e$  az  $x \in G_i$  és  $y \in G_{i+1}$  csúcsokat kösse össze.



# Gyökeres fák

- Először jelöljük ki egy speciális  $r \in V(T)$  csúcst, amit gyökérnek nevezünk.
- A  $(T, r)$  gyökeres fa új elnevezéseket indukál.
- A csúcshalmaz diszjunkt osztályokba sorolható a gyökértől vett távolság szerint.
- Legyen  $G_i$  a gyökértől  $i$  távolságra lévő csúcsok halmaza.  $G_0 = \{r\}$ ,  $G_1$  a gyökér szomszédait tartalmazó csúcshalmaz,  $G_2$  a gyökér másodsomszédainak halmaza.

Minden  $e \in E(T)$  él esetén van olyan  $i \in \mathbb{N}$ , hogy  $e$  az  $x \in G_i$  és  $y \in G_{i+1}$  csúcsokat kösse össze. Ebben az esetben  $x$  az  $e$  élnek megfelelő szülő csúcs, míg  $y$  a gyerek csúcs.

# Gyökeres fák

- Először jelöljük ki egy speciális  $r \in V(T)$  csúcst, amit gyökérnek nevezünk.
  - A  $(T, r)$  gyökeres fa új elnevezéseket indukál.
  - A csúcshalmaz diszjunkt osztályokba sorolható a gyökértől vett távolság szerint.
  - Legyen  $G_i$  a gyökértől  $i$  távolságra lévő csúcsok halmaza.  $G_0 = \{r\}$ ,  $G_1$  a gyökér szomszédait tartalmazó csúcshalmaz,  $G_2$  a gyökér másodsomszédainak halmaza.
- Minden  $e \in E(T)$  él esetén van olyan  $i \in \mathbb{N}$ , hogy  $e$  az  $x \in G_i$  és  $y \in G_{i+1}$  csúcsokat kösse össze. Ebben az esetben  $x$  az  $e$  élnek megfelelő szülő csúcs, míg  $y$  a gyerek csúcs. Azt mondjuk, hogy  $x$  gyereke  $y$ ,  $y$ -nak  $x$  a szülője.

# Gyökeres fák (folytatás)

# Gyökeres fák (folytatás)

- Ha  $x$  nem a gyökér, akkor  $x$ -ből tehetünk egy egyértelmű lépést a gyökér felé.

# Gyökeres fák (folytatás)

- Ha  $x$  nem a gyökér, akkor  $x$ -ből tehetünk egy egyértelmű lépést a gyökér felé. Ezzel  $x$  egyetlen szülőjébe (gyakori az *apa* elnevezés, ekkor a gyerekre a *fiú* egy alternatív terminológia) jutunk.

# Gyökeres fák (folytatás)

- Ha  $x$  nem a gyökér, akkor  $x$ -ből tehetünk egy egyértelmű lépést a gyökér felé. Ezzel  $x$  egyetlen szülőjébe (gyakori az *apa* elnevezés, ekkor a gyerekre a *fiú* egy alternatív terminológia) jutunk.
- $G_1$  tartalmazza a gyökér gyerekeit/fiait,  $G_2$  tartalmazza a gyökér unokáit.

# Gyökeres fák (folytatás)

- Ha  $x$  nem a gyökér, akkor  $x$ -ből tehetünk egy egyértelmű lépést a gyökér felé. Ezzel  $x$  egyetlen szülőjébe (gyakori az *apa* elnevezés, ekkor a gyerekre a *fiú* egy alternatív terminológia) jutunk.
- $G_1$  tartalmazza a gyökér gyerekeit/fiait,  $G_2$  tartalmazza a gyökér unokáit.
- A gyökér a fa *ősapja*, ennek nincs szülője.

# Gyökeres fák (folytatás)

- Ha  $x$  nem a gyökér, akkor  $x$ -ből tehetünk egy egyértelmű lépést a gyökér felé. Ezzel  $x$  egyetlen szülőjébe (gyakori az *apa* elnevezés, ekkor a gyerekre a *fiú* egy alternatív terminológia) jutunk.
- $G_1$  tartalmazza a gyökér gyerekeit/fiait,  $G_2$  tartalmazza a gyökér unokáit.
- A gyökér a fa *ősapja*, ennek nincs szülője.
- Ha egy csúcsnak nincs gyereke, akkor *levélnek* nevezzük.



# Gyökeres fák (folytatás)

- Ha  $x$  nem a gyökér, akkor  $x$ -ből tehetünk egy egyértelmű lépést a gyökér felé. Ezzel  $x$  egyetlen szülőjébe (gyakori az *apa* elnevezés, ekkor a gyerekre a *fiú* egy alternatív terminológia) jutunk.
- $G_1$  tartalmazza a gyökér gyerekeit/fiait,  $G_2$  tartalmazza a gyökér unokáit.
- A gyökér a fa *ősapja*, ennek nincs szülője.
- Ha egy csúcsnak nincs gyereke, akkor *levélnek* nevezzük.
- Ha  $y$ -ból a gyökér felé haladva elérjük az  $x$  csúcsot, akkor  $x$  az  $y$  csúcs felmenője,  $y$  az  $x$  leszármazottja.

# Gyökeres fák (folytatás)

- Ha  $x$  nem a gyökér, akkor  $x$ -ből tehetünk egy egyértelmű lépést a gyökér felé. Ezzel  $x$  egyetlen szülőjébe (gyakori az *apa* elnevezés, ekkor a gyerekre a *fiú* egy alternatív terminológia) jutunk.
- $G_1$  tartalmazza a gyökér gyerekeit/fiait,  $G_2$  tartalmazza a gyökér unokáit.
- A gyökér a fa *ősapja*, ennek nincs szülője.
- Ha egy csúcsnak nincs gyereke, akkor *levélnak* nevezzük.
- Ha  $y$ -ból a gyökér felé haladva elérjük az  $x$  csúcsot, akkor  $x$  az  $y$  csúcs felmenője,  $y$  az  $x$  leszármazottja.
- Az  $r$  gyökér minden más csúcsnak a felmenője,

# Gyökeres fák (folytatás)

- Ha  $x$  nem a gyökér, akkor  $x$ -ből tehetünk egy egyértelmű lépést a gyökér felé. Ezzel  $x$  egyetlen szülőjébe (gyakori az *apa* elnevezés, ekkor a gyerekre a *fiú* egy alternatív terminológia) jutunk.
- $G_1$  tartalmazza a gyökér gyerekeit/fiait,  $G_2$  tartalmazza a gyökér unokáit.
- A gyökér a fa *ősapja*, ennek nincs szülője.
- Ha egy csúcsnak nincs gyereke, akkor *levélnek* nevezzük.
- Ha  $y$ -ból a gyökér felé haladva elérjük az  $x$  csúcsot, akkor  $x$  az  $y$  csúcs felmenője,  $y$  az  $x$  leszármazottja.
- Az  $r$  gyökér minden más csúcsnak a felmenője,  $r$ -nek minden nem-gyökér csúcs leszármazottja.

# Gyökeres fák (folytatás)

- Ha  $x$  nem a gyökér, akkor  $x$ -ből tehetünk egy egyértelmű lépést a gyökér felé. Ezzel  $x$  egyetlen szülőjébe (gyakori az *apa* elnevezés, ekkor a gyerekre a *fiú* egy alternatív terminológia) jutunk.
- $G_1$  tartalmazza a gyökér gyerekeit/fiait,  $G_2$  tartalmazza a gyökér unokáit.
- A gyökér a fa *ősapja*, ennek nincs szülője.
- Ha egy csúcsnak nincs gyereke, akkor *levélnek* nevezzük.
- Ha  $y$ -ból a gyökér felé haladva elérjük az  $x$  csúcsot, akkor  $x$  az  $y$  csúcs felmenője,  $y$  az  $x$  leszármazottja.
- Az  $r$  gyökér minden más csúcsnak a felmenője,  $r$ -nek minden nem-gyökér csúcs leszármazottja. Egy  $\ell$  levélcsúcs esetén  $\ell$ -nek nincs leszármazottja.

# Gyökeres fák (folytatás)

# Gyökeres fák (folytatás)

- Egy  $x$  csúcs meghatároz/„generál” egy gyökeres részfát, amely csúcsai  $x$  és leszármazottjai, élei a  $T$  fa ezen csúcsai között haladó élei, gyökere  $x$ .

# Gyökeres fák (folytatás)

- Egy  $x$  csúcs meghatároz/„generál” egy gyökeres részfát, amely csúcsai  $x$  és leszármazottjai, élei a  $T$  fa ezen csúcsai között haladó élei, gyökere  $x$ .
- Ezt a gyökeres részfát  $T_x$ -szel jelöljük.

# Gyökeres fák (folytatás)

- Egy  $x$  csúcs meghatároz/„generál” egy gyökeres részfát, amely csúcsai  $x$  és leszármazottjai, élei a  $T$  fa ezen csúcsai között haladó élei, gyökere  $x$ .
- Ezt a gyökeres részfát  $T_x$ -szel jelöljük.
- Egy  $n$  csúcsú fának  $n$  darab csúcs által generált rész gyökeres fája van



# Gyökeres fák (folytatás)

- Egy  $x$  csúcs meghatároz/„generál” egy gyökeres részfát, amely csúcsai  $x$  és leszármazottjai, élei a  $T$  fa ezen csúcsai között haladó élei, gyökere  $x$ .
- Ezt a gyökeres részfát  $T_x$ -szel jelöljük.
- Egy  $n$  csúcsú fának  $n$  darab csúcs által generált rész gyökeres fája van (a gyökér az eredeti gyökeres fát generálja,

# Gyökeres fák (folytatás)

- Egy  $x$  csúcs meghatároz/„generál” egy gyökeres részfát, amely csúcsai  $x$  és leszármazottjai, élei a  $T$  fa ezen csúcsai között haladó élei, gyökere  $x$ .
- Ezt a gyökeres részfát  $T_x$ -szel jelöljük.
- Egy  $n$  csúcsú fának  $n$  darab csúcs által generált rész gyökeres fája van (a gyökér az eredeti gyökeres fát generálja, az  $\ell$  levél egy egyetlen csúcsból álló fát generál).

# Gyökeres fák (folytatás)

- Egy  $x$  csúcs meghatároz/„generál” egy gyökeres részfát, amely csúcsai  $x$  és leszármazottjai, élei a  $T$  fa ezen csúcsai között haladó élei, gyökere  $x$ .
- Ezt a gyökeres részfát  $T_x$ -szel jelöljük.
- Egy  $n$  csúcsú fának  $n$  darab csúcs által generált rész gyökeres fája van (a gyökér az eredeti gyökeres fát generálja, az  $\ell$  levél egy egyetlen csúcsból álló fát generál).
- Egy gyökeres fa mélysége a leghosszabb gyökér-levél út hossza.

# Gyökeres fák (folytatás)

- Egy  $x$  csúcs meghatároz/„generál” egy gyökeres részfát, amely csúcsai  $x$  és leszármazottjai, élei a  $T$  fa ezen csúcsai között haladó élei, gyökere  $x$ .
- Ezt a gyökeres részfát  $T_x$ -szel jelöljük.
- Egy  $n$  csúcsú fának  $n$  darab csúcs által generált rész gyökeres fája van (a gyökér az eredeti gyökeres fát generálja, az  $\ell$  levél egy egyetlen csúcsból álló fát generál).
- Egy gyökeres fa mélysége a leghosszabb gyökér-levél út hossza. A 0 mélységű gyökeres fa egyetlen csúcsból áll, a gyökérből, ami az ősapa és egyben levélcsúcs is.

# Gyökeres fák (folytatás)

- Egy  $x$  csúcs meghatároz/„generál” egy gyökeres részfát, amely csúcsai  $x$  és leszármazottjai, élei a  $T$  fa ezen csúcsai között haladó élei, gyökere  $x$ .
- Ezt a gyökeres részfát  $T_x$ -szel jelöljük.
- Egy  $n$  csúcsú fának  $n$  darab csúcs által generált rész gyökeres fája van (a gyökér az eredeti gyökeres fát generálja, az  $\ell$  levél egy egyetlen csúcsból álló fát generál).
- Egy gyökeres fa mélysége a leghosszabb gyökér-levél út hossza. A 0 mélységű gyökeres fa egyetlen csúcsból áll, a gyökérből, ami az ősapa és egyben levélcsúcs is.
- A fenti szleng/terminológia onnan ered, hogy egy gyökeres fára egy családfa a legjobb példa.

# Gyökeres fák (folytatás)

- Egy  $x$  csúcs meghatároz/„generál” egy gyökeres részfát, amely csúcsai  $x$  és leszármazottjai, élei a  $T$  fa ezen csúcsai között haladó élei, gyökere  $x$ .
- Ezt a gyökeres részfát  $T_x$ -szel jelöljük.
- Egy  $n$  csúcsú fának  $n$  darab csúcs által generált rész gyökeres fája van (a gyökér az eredeti gyökeres fát generálja, az  $\ell$  levél egy egyetlen csúcsból álló fát generál).
- Egy gyökeres fa mélysége a leghosszabb gyökér-levél út hossza. A 0 mélységű gyökeres fa egyetlen csúcsból áll, a gyökérből, ami az ősapa és egyben levélcsúcs is.
- A fenti szleng/terminológia onnan ered, hogy egy gyökeres fára egy családfa a legjobb példa. Az Árpád-ház családfájában a fenti elnevezések mindennapi értelmet kapnak.

# A dinamika leírása

# A dinamika leírása

- Megjegyzésünk után az alapfeladatra úgy gondolunk, hogy inputja egy  $(T, r)$  gyökeres fa.



# A dinamika leírása

- Megjegyzésünk után az alapfeladatra úgy gondolunk, hogy inputja egy  $(T, r)$  gyökeres fa. A független halmazok fogalmát a gyökér nem befolyásolja, csupán kibővíti „nyelvünket”.

# A dinamika leírása

- Megjegyzésünk után az alapfeladatra úgy gondolunk, hogy inputja egy  $(T, r)$  gyökeres fa. A független halmazok fogalmát a gyökér nem befolyásolja, csupán kibővíti „nyelvünket”.

## A feladatsereg

$$\{\mathcal{F}_x\}_{x \in V(T)},$$

ahol  $\mathcal{F}_x$  a  $(T_x, x)$  gyökeres fára vonatkozó feladat, azaz a legnagyobb független csúcshalmaz méretének megválaszolása.

# A dinamika leírása

- Megjegyzésünk után az alapfeladatra úgy gondolunk, hogy inputja egy  $(T, r)$  gyökeres fa. A független halmazok fogalmát a gyökér nem befolyásolja, csupán kibővíti „nyelvünket”.

## A feladatsereg

$$\{\mathcal{F}_x\}_{x \in V(T)},$$

ahol  $\mathcal{F}_x$  a  $(T_x, x)$  gyökeres fára vonatkozó feladat, azaz a legnagyobb független csúcshalmaz méretének megválaszolása.

- Tehát  $|V(T)|$  darab részfeladatot definiáltunk.

## A feladatok felgöngyölítési sorrendje

A mélység szerinti növekvő sorrend.

# A kezdeti „algorithmus”

Algorithmus-vázlet a legnagyobb független csúcshalmaz megkeresésére

# A kezdeti „algorithmus”

## Algorithmus-vázlet a legnagyobb független csúcshalmaz megkeresésére

(0) Azaz azokkal a feladatokkal kezdjük, ahol  $(T_x, x)$  0 mélységű, azaz  $x$  az eredeti fa egy levele.

# A kezdeti „algorithmus”

## Algorithmus-vázlet a legnagyobb független csúcshalmaz megkeresésére

(0) Azaz azokkal a feladatokkal kezdjük, ahol  $(T_x, x)$  0 mélységű, azaz  $x$  az eredeti fa egy levele. Ekkor a kérdésre a válasz: 1.

# A kezdeti „algorithmus”

## Algorithmus-vázlet a legnagyobb független csúcshalmaz megkeresésére

(0) Azaz azokkal a feladatokkal kezdjük, ahol  $(T_x, x)$  0 mélységű, azaz  $x$  az eredeti fa egy levele. Ekkor a kérdésre a válasz: 1.

// Ekkor  $(T_x, x)$  egy csúcsú, 0 élű gráf, amelyben  $\{x\}$  egyben független halmaz is (az  $\emptyset$  mellett), azaz a legnagyobb független halmaz mérete 1.

# A kezdeti „algorithmus”

## Algorithmus-vázlet a legnagyobb független csúcshalmaz megkeresésére

(0) Azaz azokkal a feladatokkal kezdjük, ahol  $(T_x, x)$  0 mélységű, azaz  $x$  az eredeti fa egy levele. Ekkor a kérdésre a válasz: 1.

// Ekkor  $(T_x, x)$  egy csúcshalmaz, 0 élű gráf, amelyben  $\{x\}$  egyben független halmaz is (az  $\emptyset$  mellett), azaz a legnagyobb független halmaz mérete 1.

(Göngyölyítés) Csúcsainkat mélység szerinti sorban vizsgáljuk.



# A kezdeti „algoritmus”

## Algoritmus-vázlet a legnagyobb független csúcshalmaz megkeresésére

(0) Azaz azokkal a feladatokkal kezdjük, ahol  $(T_x, x)$  0 mélységű, azaz  $x$  az eredeti fa egy levele. Ekkor a kérdésre a válasz: 1.

// Ekkor  $(T_x, x)$  egy csúcshalmaz, 0 élű gráf, amelyben  $\{x\}$  egyben független halmaz is (az  $\emptyset$  mellett), azaz a legnagyobb független halmaz mérete 1.

(Göngyölyítés) Csúcsainkat mélység szerinti sorban vizsgáljuk. Tegyük fel, hogy feladataink között  $(T_x, x)$  a soros.

# A kezdeti „algoritmus”

## Algoritmus-vázlet a legnagyobb független csúcshalmaz megkeresésére

(0) Azaz azokkal a feladatokkal kezdjük, ahol  $(T_x, x)$  0 mélységű, azaz  $x$  az eredeti fa egy levele. Ekkor a kérdésre a válasz: 1.

// Ekkor  $(T_x, x)$  egy csúcsú, 0 élű gráf, amelyben  $\{x\}$  egyben független halmaz is (az  $\emptyset$  mellett), azaz a legnagyobb független halmaz mérete 1.

(Göngyöltés) Csúcsainkat mélység szerinti sorban vizsgáljuk. Tegyük fel, hogy feladataink között  $(T_x, x)$  a soros.

// A kisebb mélységű gyökeres fákra vonatkozó problémákat már megoldottuk.

# A kezdeti „algoritmus”

## Algoritmus-vázlet a legnagyobb független csúcshalmaz megkeresésére

(0) Azaz azokkal a feladatokkal kezdjük, ahol  $(T_x, x)$  0 mélységű, azaz  $x$  az eredeti fa egy levele. Ekkor a kérdésre a válasz: 1.

// Ekkor  $(T_x, x)$  egy csúcsú, 0 élű gráf, amelyben  $\{x\}$  egyben független halmaz is (az  $\emptyset$  mellett), azaz a legnagyobb független halmaz mérete 1.

(Göngyöltés) Csúcsainkat mélység szerinti sorban vizsgáljuk. Tegyük fel, hogy feladataink között  $(T_x, x)$  a soros.

// A kisebb mélységű gyökeres fákra vonatkozó problémákat már megoldottuk.

A korábbi eredményekből számoljuk ki a soros kérdésre adandó választ.

# A kezdeti „algoritmus”

## Algoritmus-vázlet a legnagyobb független csúcshalmaz megkeresésére

(0) Azaz azokkal a feladatokkal kezdjük, ahol  $(T_x, x)$  0 mélységű, azaz  $x$  az eredeti fa egy levele. Ekkor a kérdésre a válasz: 1.

// Ekkor  $(T_x, x)$  egy csúcsú, 0 élű gráf, amelyben  $\{x\}$  egyben független halmaz is (az  $\emptyset$  mellett), azaz a legnagyobb független halmaz mérete 1.

(Göngyöltés) Csúcsainkat mélység szerinti sorban vizsgáljuk. Tegyük fel, hogy feladataink között  $(T_x, x)$  a soros.

// A kisebb mélységű gyökeres fákra vonatkozó problémákat már megoldottuk.

A korábbi eredményekből számoljuk ki a soros kérdésre adandó választ.

(Output)  $(T_r, r)$  kérdésre adott választ jelentsük be outputként.

# A korábbi eredményekből történő számolás alapötlete

# A korábbi eredményekből történő számolás alapötlete

- A soros  $(T_x, x)$  gyökeres fa független halmazait osszuk két osztályba:

# A korábbi eredményekből történő számolás alapötlete

- A soros  $(T_x, x)$  gyökeres fa független halmazait osszuk két osztályba:
  - (a) Az  $x$ -et nem tartalmazók.
  - (b) Az  $x$ -et tartalmazók.

# A korábbi eredményekből történő számolás alapötlete

- A soros  $(T_x, x)$  gyökeres fa független halmazait osszuk két osztályba:

- (a) Az  $x$ -et nem tartalmazók.
- (b) Az  $x$ -et tartalmazók.

A két fajta független halmazokra küéöm-külön határozzuk meg a legnagyobb méretét.



# (a) eset

# (a) eset

- Legyen  $f_1, \dots, f_d$  az  $x$  csúcs fiai.

## (a) eset

- Legyen  $f_1, \dots, f_d$  az  $x$  csúcs fiai.

### Észrevétel

Ekkor  $T_{f_1}, T_{f_2}, \dots, T_{f_d}$  részfákban tetszőlegesen/függetlenül vehetünk egy független halmazt és ezek uniója egy (a) eset alá tartozó független halmazt ad.

## (a) eset

- Legyen  $f_1, \dots, f_d$  az  $x$  csúcs fiai.

### Észrevétel

Ekkor  $T_{f_1}, T_{f_2}, \dots, T_{f_d}$  részfákban tetszőlegesen/függetlenül vehetünk egy független halmazt és ezek uniója egy (a) eset alá tartozó független halmazt ad.

Másrészt minden (a) eset alá tartozó független halmazt megkapunk így.

## (a) eset

- Legyen  $f_1, \dots, f_d$  az  $x$  csúcs fiai.

### Észrevétel

Ekkor  $T_{f_1}, T_{f_2}, \dots, T_{f_d}$  részfákban tetszőlegesen/függetlenül vehetünk egy független halmazt és ezek uniója egy (a) eset alá tartozó független halmazt ad.

Másrészt minden (a) eset alá tartozó független halmazt megkapunk így.

- A független választások miatt egyszerű az elemszám maximalizálása:

## (a) eset

- Legyen  $f_1, \dots, f_d$  az  $x$  csúcs fiai.

### Észrevétel

Ekkor  $T_{f_1}, T_{f_2}, \dots, T_{f_d}$  részfákban tetszőlegesen/függetlenül vehetünk egy független halmazt és ezek uniója egy (a) eset alá tartozó független halmazt ad.

Másrészt minden (a) eset alá tartozó független halmazt megkapunk így.

- A független választások miatt egyszerű az elemszám maximalizálása: Mindegyik  $T_{f_i}$ -ből egy legnagyobb méretű független halmazt kell választanunk.

## (a) eset

- Legyen  $f_1, \dots, f_d$  az  $x$  csúcs fiai.

### Észrevétel

Ekkor  $T_{f_1}, T_{f_2}, \dots, T_{f_d}$  részfákban tetszőlegesen/függetlenül vehetünk egy független halmazt és ezek uniója egy (a) eset alá tartozó független halmazt ad.

Másrészt minden (a) eset alá tartozó független halmazt megkapunk így.

- A független választások miatt egyszerű az elemszám maximalizálása: Mindegyik  $T_{f_i}$ -ből egy legnagyobb méretű független halmazt kell választanunk. Ezek az  $M_i$  méretek rendezésünkre állnak,

## (a) eset

- Legyen  $f_1, \dots, f_d$  az  $x$  csúcs fiai.

### Észrevétel

Ekkor  $T_{f_1}, T_{f_2}, \dots, T_{f_d}$  részfákban tetszőlegesen/függetlenül vehetünk egy független halmazt és ezek uniója egy (a) eset alá tartozó független halmazt ad.

Másrészt minden (a) eset alá tartozó független halmazt megkapunk így.

- A független választások miatt egyszerű az elemszám maximalizálása: Mindegyik  $T_{f_i}$ -ből egy legnagyobb méretű független halmazt kell választanunk. Ezek az  $M_i$  méretek rendezésünkre állnak, mert  $(T_x, x)$  mélysége nagyobb mint  $(T_{f_i}, f_i)$  mélysége.



## (a) eset

- Legyen  $f_1, \dots, f_d$  az  $x$  csúcs fiai.

### Észrevétel

Ekkor  $T_{f_1}, T_{f_2}, \dots, T_{f_d}$  részfákban tetszőlegesen/függetlenül vehetünk egy független halmazt és ezek uniója egy (a) eset alá tartozó független halmazt ad.

Másrészt minden (a) eset alá tartozó független halmazt megkapunk így.

- A független választások miatt egyszerű az elemszám maximalizálása: Mindegyik  $T_{f_i}$ -ből egy legnagyobb méretű független halmazt kell választanunk. Ezek az  $M_i$  méretek rendezésünkre állnak, mert  $(T_x, x)$  mélysége nagyobb mint  $(T_{f_i}, f_i)$  mélysége.
- Az aktuális feladatra adott válasz  $\sum_i M_i$ .

# (b) eset

## (b) eset

- Legyen  $u_1, \dots, u_D$  az  $x$  csúcs unokái.

## (b) eset

- Legyen  $u_1, \dots, u_D$  az  $x$  csúcs unokái.

### Észrevétel

Ekkor  $T_{u_1}, T_{u_2}, \dots, T_{u_D}$  részfákban tetszőlegesen/függetlenül vehetünk egy független halmazt és ezek uniója,  $x$ -szel kibővítve egy (b) eset alá tartozó független halmazt ad.

## (b) eset

- Legyen  $u_1, \dots, u_D$  az  $x$  csúcs unokái.

### Észrevétel

Ekkor  $T_{u_1}, T_{u_2}, \dots, T_{u_D}$  részfákban tetszőlegesen/függetlenül vehetünk egy független halmazt és ezek uniója,  $x$ -szel kibővítve egy (b) eset alá tartozó független halmazt ad.

Másrészt minden (b) eset alá tartozó független halmaz előáll így.

## (b) eset

- Legyen  $u_1, \dots, u_D$  az  $x$  csúcs unokái.

### Észrevétel

Ekkor  $T_{u_1}, T_{u_2}, \dots, T_{u_D}$  részfákban tetszőlegesen/függetlenül vehetünk egy független halmazt és ezek uniója,  $x$ -szel kibővítve egy (b) eset alá tartozó független halmazt ad.

Másrészt minden (b) eset alá tartozó független halmaz előáll így.

- Mindegyik  $T_{u_i}$ -ből egy legnagyobb méretű független halmazt kell választanunk.

## (b) eset

- Legyen  $u_1, \dots, u_D$  az  $x$  csúcs unokái.

### Észrevétel

Ekkor  $T_{u_1}, T_{u_2}, \dots, T_{u_D}$  részfákban tetszőlegesen/függetlenül vehetünk egy független halmazt és ezek uniója,  $x$ -szel kibővítve egy (b) eset alá tartozó független halmazt ad.

Másrészt minden (b) eset alá tartozó független halmaz előáll így.

- Mindegyik  $T_{u_i}$ -ből egy legnagyobb méretű független halmazt kell választanunk. Ezek a  $\mu_i$  méretek rendezésünkre állnak,

## (b) eset

- Legyen  $u_1, \dots, u_D$  az  $x$  csúcs unokái.

### Észrevétel

Ekkor  $T_{u_1}, T_{u_2}, \dots, T_{u_D}$  részfákban tetszőlegesen/függetlenül vehetünk egy független halmazt és ezek uniója,  $x$ -szel kibővítve egy (b) eset alá tartozó független halmazt ad.

Másrészt minden (b) eset alá tartozó független halmaz előáll így.

- Mindegyik  $T_{u_i}$ -ből egy legnagyobb méretű független halmazt kell választanunk. Ezek a  $\mu_i$  méretek rendezésünkre állnak, mert  $(T_x, x)$  mélysége nagyobb mint  $(T_{u_i}, u_i)$  mélysége.



## (b) eset

- Legyen  $u_1, \dots, u_D$  az  $x$  csúcs unokái.

### Észrevétel

Ekkor  $T_{u_1}, T_{u_2}, \dots, T_{u_D}$  részfákban tetszőlegesen/függetlenül vehetünk egy független halmazt és ezek uniója,  $x$ -szel kibővítve egy (b) eset alá tartozó független halmazt ad.

Másrészt minden (b) eset alá tartozó független halmaz előáll így.

- Mindegyik  $T_{u_i}$ -ből egy legnagyobb méretű független halmazt kell választanunk. Ezek a  $\mu_i$  méretek rendezésünkre állnak, mert  $(T_x, x)$  mélysége nagyobb mint  $(T_{u_i}, u_i)$  mélysége.
- Az aktuális feladatra adott válasz  $1 + \sum_i \mu_i$ .

# Az algoritmus

Algoritmus a legnagyobb független csúcshalmaz megkeresésére

# Az algoritmus

## Algoritmus a legnagyobb független csúcshalmaz megkeresésére

(0) Azaz azokkal a feladatokkal kezdjük, ahol  $(T_x, x)$  0 mélységű, azaz  $x$  az eredeti fa egy levele.

# Az algoritmus

## Algoritmus a legnagyobb független csúcshalmaz megkeresésére

(0) Azaz azokkal a feladatokkal kezdjük, ahol  $(T_x, x)$  0 mélységű, azaz  $x$  az eredeti fa egy levele. Ekkor a kérdésre a válasz: 1.

# Az algoritmus

## Algoritmus a legnagyobb független csúcshalmaz megkeresésére

(0) Azaz azokkal a feladatokkal kezdjük, ahol  $(T_x, x)$  0 mélységű, azaz  $x$  az eredeti fa egy levele. Ekkor a kérdésre a válasz: 1.

// Ekkor  $(T_x, x)$  egy csúcsú, 0 élű gráf, amelyben  $\{x\}$  egyben független halmaz is (az  $\emptyset$  mellett), azaz a legnagyobb független halmaz mérete 1.

# Az algoritmus

## Algoritmus a legnagyobb független csúcshalmaz megkeresésére

(0) Azaz azokkal a feladatokkal kezdjük, ahol  $(T_x, x)$  0 mélységű, azaz  $x$  az eredeti fa egy levele. Ekkor a kérdésre a válasz: 1.

// Ekkor  $(T_x, x)$  egy csúcsú, 0 élű gráf, amelyben  $\{x\}$  egyben független halmaz is (az  $\emptyset$  mellett), azaz a legnagyobb független halmaz mérete 1.

(Göngyöltés) Csúcsainkat mélység szerinti sorban vizsgáljuk.

# Az algoritmus

## Algoritmus a legnagyobb független csúcshalmaz megkeresésére

(0) Azaz azokkal a feladatokkal kezdjük, ahol  $(T_x, x)$  0 mélységű, azaz  $x$  az eredeti fa egy levele. Ekkor a kérdésre a válasz: 1.

// Ekkor  $(T_x, x)$  egy csúcsú, 0 élű gráf, amelyben  $\{x\}$  egyben független halmaz is (az  $\emptyset$  mellett), azaz a legnagyobb független halmaz mérete 1.

(Göngyöltés) Csúcsainkat mélység szerinti sorban vizsgáljuk. Tegyük fel, hogy feladataink között  $(T_x, x)$  a soros.

# Az algoritmus

## Algoritmus a legnagyobb független csúcshalmaz megkeresésére

(0) Azaz azokkal a feladatokkal kezdjük, ahol  $(T_x, x)$  0 mélységű, azaz  $x$  az eredeti fa egy levele. Ekkor a kérdésre a válasz: 1.

// Ekkor  $(T_x, x)$  egy csúcsú, 0 élű gráf, amelyben  $\{x\}$  egyben független halmaz is (az  $\emptyset$  mellett), azaz a legnagyobb független halmaz mérete 1.

(Göngyöltés) Csúcsainkat mélység szerinti sorban vizsgáljuk. Tegyük fel, hogy feladataink között  $(T_x, x)$  a soros.

// A kisebb mélységű gyökeres fákra vonatkozó problémákat már megoldottuk.



# Az algoritmus

## Algoritmus a legnagyobb független csúcshalmaz megkeresésére

(0) Azaz azokkal a feladatokkal kezdjük, ahol  $(T_x, x)$  0 mélységű, azaz  $x$  az eredeti fa egy levele. Ekkor a kérdésre a válasz: 1.

// Ekkor  $(T_x, x)$  egy csúcsú, 0 élű gráf, amelyben  $\{x\}$  egyben független halmaz is (az  $\emptyset$  mellett), azaz a legnagyobb független halmaz mérete 1.

(Göngyöltés) Csúcsainkat mélység szerinti sorban vizsgáljuk. Tegyük fel, hogy feladataink között  $(T_x, x)$  a soros.

// A kisebb mélységű gyökeres fákra vonatkozó problémákat már megoldottuk.

A korábbi eredményekből számoljuk ki  $\max\{\sum_i M_i, 1 + \sum_i \mu_i\}$ -t.

# Az algoritmus

## Algoritmus a legnagyobb független csúcshalmaz megkeresésére

(0) Azaz azokkal a feladatokkal kezdjük, ahol  $(T_x, x)$  0 mélységű, azaz  $x$  az eredeti fa egy levele. Ekkor a kérdésre a válasz: 1.

// Ekkor  $(T_x, x)$  egy csúcsú, 0 élű gráf, amelyben  $\{x\}$  egyben független halmaz is (az  $\emptyset$  mellett), azaz a legnagyobb független halmaz mérete 1.

(Göngyöltés) Csúcsainkat mélység szerinti sorban vizsgáljuk. Tegyük fel, hogy feladataink között  $(T_x, x)$  a soros.

// A kisebb mélységű gyökeres fákra vonatkozó problémákat már megoldottuk.

A korábbi eredményekből számoljuk ki  $\max\{\sum_i M_i, 1 + \sum_i \mu_i\}$ -t. Ez lesz a soros kérdésre adandó választ.

# Az algoritmus

## Algoritmus a legnagyobb független csúcshalmaz megkeresésére

(0) Azaz azokkal a feladatokkal kezdjük, ahol  $(T_x, x)$  0 mélységű, azaz  $x$  az eredeti fa egy levele. Ekkor a kérdésre a válasz: 1.

// Ekkor  $(T_x, x)$  egy csúcsú, 0 élű gráf, amelyben  $\{x\}$  egyben független halmaz is (az  $\emptyset$  mellett), azaz a legnagyobb független halmaz mérete 1.

(Göngyöltés) Csúcsainkat mélység szerinti sorban vizsgáljuk. Tegyük fel, hogy feladataink között  $(T_x, x)$  a soros.

// A kisebb mélységű gyökeres fákra vonatkozó problémákat már megoldottuk.

A korábbi eredményekből számoljuk ki  $\max\{\sum_i M_i, 1 + \sum_i \mu_i\}$ -t. Ez lesz a soros kérdésre adandó választ.

(Output)  $(T_r, r)$  kérdésre adott választ jelentsük be outputként.

# Végső megjegyzés

Könnyű az algoritmust átlakítani úgy, hogy ne csak egy maximális méretet adjon meg,

# Végső megjegyzés

Könnyű az algoritmust átlakítani úgy, hogy ne csak egy maximális méretet adjon meg, de egy legnagyobb független csúcshalmaz legyen az output.

# Szünet



# Az alapkérdés

Legnagyobb konvex helyzetű részhalmaz megkeresése egy véges síkbeli ponthalmazban.

Adott  $n$  darab pont a síkon:  $\mathcal{P} = \{P_i(x_i, y_i)\}_{i=1}^n$ .

# Az alapkérdés

Legnagyobb konvex helyzetű részhalmoz megkeresése egy véges síkbeli ponthalmazban.

Adott  $n$  darab pont a síkon:  $\mathcal{P} = \{P_i(x_i, y_i)\}_{i=1}^n$ .

// Feltesszük, hogy pontjaink  $x$ -koordinátái különbözőek.



# Az alapkérdés

Legnagyobb konvex helyzetű részhalmaz megkeresése egy véges síkbeli ponthalmazban.

Adott  $n$  darab pont a síkon:  $\mathcal{P} = \{P_i(x_i, y_i)\}_{i=1}^n$ .

// Feltesszük, hogy pontjaink  $x$ -koordinátái különbözőek.

Határozzuk meg azt a legnagyobb  $\mathcal{R}$  részhalmazát, amelyek konvex helyzetű.

# Az alapkérdés

Legnagyobb konvex helyzetű részhalmaz megkeresése egy véges síkbeli ponthalmazban.

Adott  $n$  darab pont a síkon:  $\mathcal{P} = \{P_i(x_i, y_i)\}_{i=1}^n$ .

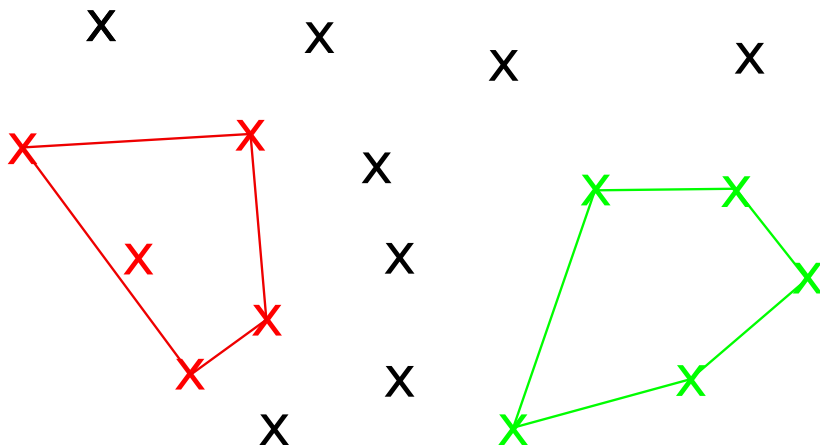
// Feltesszük, hogy pontjaink  $x$ -koordinátái különbözőek.

Határozzuk meg azt a legnagyobb  $\mathcal{R}$  részhalmazát, amelyek konvex helyzetű.

**Definíció: konvex helyzet**

$\mathcal{R} \subset \mathbb{R}^2$  véges ponthalmaz konvex helyzetű, ha egy konvex  $|\mathcal{R}|$ -szög csúcshalmaza.

## Példa



**Figure:** Egy véges ponthalmaz és egy öt elemű része (piros), amely nem konvex helyzetű, továbbá egy másik öt elemű része (zöld), amely konvex helyzetű.

# Kezdeti lépések: seprés

# Kezdeti lépések: seprés

- Először is ponthalmazunkat balról jobbra végigpásztázzuk/seperjük. Pontosabban az  $x$  koordinátákat sorbarendezzük.

# Kezdeti lépések: seprés

- Először is ponthalmazunkat balról jobbra végigpásztázzuk/seperjük. Pontosabban az  $x$  koordinátákat sorbarendezzük.
- // Az egyszerűség kedvéért feltettük, hogy  $i < j$  esetén  $x_i < x_j$ .

# Kezdeti lépések: seprés

- Először is ponthalmazunkat balról jobbra végigpásztázzuk/seperjük. Pontosabban az  $x$  koordinátákat sorbarendezzük.
- // Az egyszerűség kedvéért feltettük, hogy  $i < j$  esetén  $x_i < x_j$ .
- $\mathcal{O}(n \log n)$  összehasonlítással megoldható ez a része algoritmusunknak.

# Kezdeti lépések: sapka/csésze problémára való szétszedés



# Kezdeti lépések: sapka/csésze problémára való szétszedés

- Másodszor is  $k$  konvex helyzetű csúcs konvex burka egy konvex  $k$ -szög.

# Kezdeti lépések: sapka/csésze problémára való szétszedés

- Másodszor is  $k$  konvex helyzetű csúcs konvex burka egy konvex  $k$ -szög.
- Ennek kerületének két pontját kiemeljük: a minimális és a maximális  $x$  koordinátájú csúcsot ( $m$  és  $M$ ).

# Kezdeti lépések: sapka/csésze problémára való szétszedés

- Másodszor is  $k$  konvex helyzetű csúcs konvex burka egy konvex  $k$ -szög.
- Ennek kerületének két pontját kiemeljük: a minimális és a maximális  $x$  koordinátájú csúcsot ( $m$  és  $M$ ).
- Ez a kerületet, mint körszerűen rendezett véges csúcshalmazt két „ívre” osztja

# Kezdeti lépések: sapka/csésze problémára való szétszedés

- Másodszor is  $k$  konvex helyzetű csúcs konvex burka egy konvex  $k$ -szög.
- Ennek területének két pontját kiemeljük: a minimális és a maximális  $x$  koordinátájú csúcsot ( $m$  és  $M$ ).
- Ez a területet, mint körszerűen rendezett véges csúcshalmazt két „ívre” osztja (a két kiemelt csúcsot mindkettőhöz hozzáértjük): egy felsőre és egy alsóra.

# Kezdeti lépések: sapka/csésze problémára való szétszedés

- Másodszor is  $k$  konvex helyzetű csúcs konvex burka egy konvex  $k$ -szög.
- Ennek kerületének két pontját kiemeljük: a minimális és a maximális  $x$  koordinátájú csúcsot ( $m$  és  $M$ ).
- Ez a kerületet, mint körszerűen rendezett véges csúcshalmazt két „ívre” osztja (a két kiemelt csúcsot mindkettőhöz hozzáértjük): egy felsőre és egy alsóra.
- A felső ív csúcsai is konvex helyzetűek, azaz a konvex burkukhoz minden pont hozzájárul mint csúcs.

# Kezdeti lépések: sapka/csésze problémára való szétszedés

- Másodszor is  $k$  konvex helyzetű csúcs konvex burka egy konvex  $k$ -szög.
- Ennek kerületének két pontját kiemeljük: a minimális és a maximális  $x$  koordinátájú csúcsot ( $m$  és  $M$ ).
- Ez a kerületet, mint körszerűen rendezett véges csúcshalmazt két „ívre” osztja (a két kiemelt csúcsot mindkettőhöz hozzáértjük): egy felsőre és egy alsóra.
- A felső ív csúcsai is konvex helyzetűek, azaz a konvex burkukhoz minden pont hozzájárul mint csúcs. Sőt,  $mM$  egy oldal lesz és az egész konvex burok ezen oldal felett fekszik.

# Kezdeti lépések: sapka/csésze problémára való szétszedés

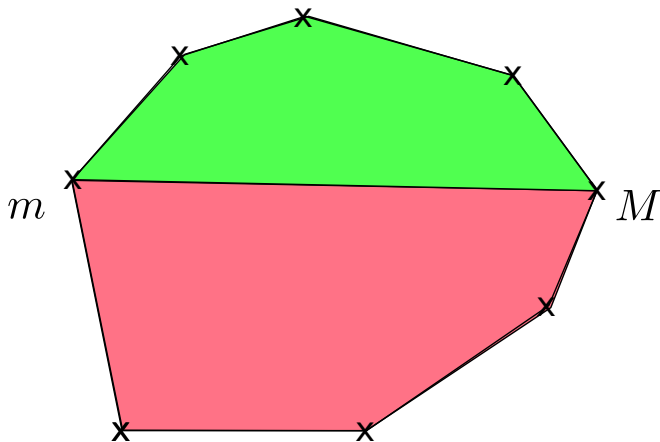
- Másodszor is  $k$  konvex helyzetű csúcs konvex burka egy konvex  $k$ -szög.
- Ennek kerületének két pontját kiemeljük: a minimális és a maximális  $x$  koordinátájú csúcsot ( $m$  és  $M$ ).
- Ez a kerületet, mint körszerűen rendezett véges csúcshalmazt két „ívre” osztja (a két kiemelt csúcsot mindkettőhöz hozzáértjük): egy felsőre és egy alsóra.
- A felső ív csúcsai is konvex helyzetűek, azaz a konvex burkukhoz minden pont hozzájárul mint csúcs. Sőt,  $mM$  egy oldal lesz és az egész konvex burok ezen oldal felett fekszik. Azt mondjuk a felső ív egy *sapka* (angolul *cap*).

# Kezdeti lépések: sapka/csésze problémára való szétszedés

- Másodszor is  $k$  konvex helyzetű csúcs konvex burka egy konvex  $k$ -szög.
- Ennek kerületének két pontját kiemeljük: a minimális és a maximális  $x$  koordinátájú csúcsot ( $m$  és  $M$ ).
- Ez a kerületet, mint körszerűen rendezett véges csúcshalmazt két „ívre” osztja (a két kiemelt csúcsot mindkettőhöz hozzáértjük): egy felsőre és egy alsóra.
- A felső ív csúcsai is konvex helyzetűek, azaz a konvex burkukhoz minden pont hozzájárul mint csúcs. Sőt,  $mM$  egy oldal lesz és az egész konvex burok ezen oldal felett fekszik. Azt mondjuk a felső ív egy *sapka* (angolul *cap*).
- Hasonlóan kezelhető az alsó ív. Erre azt mondjuk, hogy *csésze* (angolul *cup*).



# Példa



**Figure:** Egy konvex helyzetű ponthalmaz, a kiemelt két csúccsal. A zöld rész a sapka összetevő konvex burka, a piros a csésze összetevő konvex burka.

# Kezdei lépések: elég a sapkákat nézni

# Kezdei lépések: elég a sapkákat nézni

- Ha minden  $i < j$  esetén tudjuk az  $P_i$ -vel kezdődő és  $P_j$ -vel végződő leghosszabb sapka, illetve csésze méretét, akkor az alapkérdést is megválaszoljuk.

# Kezdei lépések: elég a sapkákat nézni

- Ha minden  $i < j$  esetén tudjuk az  $P_i$ -vel kezdődő és  $P_j$ -vel végződő leghosszabb sapka, illetve csésze méretét, akkor az alapkérdést is megválaszoljuk.
- Ez az ötlet már egy „dinamikus programozásos” ötlet.

# Kezdei lépések: elég a sapkákat nézni

- Ha minden  $i < j$  esetén tudjuk az  $P_i$ -vel kezdődő és  $P_j$ -vel végződő leghosszabb sapka, illetve csésze méretét, akkor az alapkérdést is megválaszoljuk.
- Ez az ötlet már egy „dinamikus programozásos” ötlet.
- A tervezés eeső lépése mellett arra használjuk, hogy az alapfeladatot „elfelezzük”, csak a legnagyobb sapkaméret meghatározását tárgyaljuk adott  $P_i$  és  $P_j$  pontok között.

# Kezdei lépések: elég a sapkákat nézni

- Ha minden  $i < j$  esetén tudjuk az  $P_i$ -vel kezdődő és  $P_j$ -vel végződő leghosszabb sapka, illetve csésze méretét, akkor az alapkérdést is megválaszoljuk.
- Ez az ötlet már egy „dinamikus programozásos” ötlet.
- A tervezés eőső lépése mellett arra használjuk, hogy az alapfeladatot „elfelezzük”, csak a legnagyobb sapkaméret meghatározását tárgyaljuk adott  $P_i$  és  $P_j$  pontok között.
- Természetesen az alábbi ötletek a legnagyobb csészeméret meghatározásához is használhatók.

# Kezdei lépések: elég a sapkákat nézni

- Ha minden  $i < j$  esetén tudjuk az  $P_i$ -vel kezdődő és  $P_j$ -vel végződő leghosszabb sapka, illetve csésze méretét, akkor az alapkérdést is megválaszoljuk.
- Ez az ötlet már egy „dinamikus programozásos” ötlet.
- A tervezés eeső lépése mellett arra használjuk, hogy az alapfeladatot „elfelezzük”, csak a legnagyobb sapkaméret meghatározását tárgyaljuk adott  $P_i$  és  $P_j$  pontok között.
- Természetesen az alábbi ötletek a legnagyobb csészeméret meghatározásához is használhatók.
- Így az eredeti feladatot megoldó algoritmus kiolvasható az alábbiakból.

# $P_i$ -től $P_j$ -be vezető leghosszabb sapka meghatározásának problémája dinamikus programozással



# $P_i$ -től $P_j$ -be vezető leghosszabb sapka meghatározásának problémája dinamikus programozással

- A jelölés leegyszerűsítése miatt feltesszük, hogy  $i = 1$  és  $j = n$ .

## A feladatsereg

Legyen  $\mathcal{F}_{k,\ell,s}$  ( $k \leq \ell < s$ ) az a feladat, amely megadja a leghosszabb  $P_k$ -val kezdődő,  $P_s$ -ben végződő, oda közvetlenül  $P_\ell$ -ből jutó sapka problémáját.

# $P_i$ -től $P_j$ -be vezető leghosszabb sapka meghatározásának problémája dinamikus programozással

- A jelölés leegyszerűsítése miatt feltesszük, hogy  $i = 1$  és  $j = n$ .

## A feladatsereg

Legyen  $\mathcal{F}_{k,\ell,s}$  ( $k \leq \ell < s$ ) az a feladat, amely megadja a leghosszabb  $P_k$ -val kezdődő,  $P_s$ -ben végződő, oda közvetlenül  $P_\ell$ -ből jutó sapka problémáját.

- Így  $\mathcal{O}(n^3)$  részfeladatot definiáltunk.

## A felgöngyölítési sorrend

$\ell - k$  szerint növekvő sorrendben kezeljük a problémákat.

# $P_i$ -től $P_j$ -be vezető leghosszabb sapka meghatározásának problémája dinamikus programozással

- A jelölés leegyszerűsítése miatt feltesszük, hogy  $i = 1$  és  $j = n$ .

## A feladatsereg

Legyen  $\mathcal{F}_{k,\ell,s}$  ( $k \leq \ell < s$ ) az a feladat, amely megadja a leghosszabb  $P_k$ -val kezdődő,  $P_s$ -ben végződő, oda közvetlenül  $P_\ell$ -ből jutó sapka problémáját.

- Így  $\mathcal{O}(n^3)$  részfeladatot definiáltunk.

## A felgöngyöltési sorrend

$\ell - k$  szerint növekvő sorrendben kezeljük a problémákat.

- A legegyszerűbb/kezdő problémák az  $\mathcal{F}_{k,k,s}$  problémák. Ekkor a  $P_k, P_s$  két pont egy sapkát alkot, ami az optimális sapka.

# A göngyölítési lépés

# A göngyölítési lépés

- Tegyük fel, hogy  $k < \ell$  és  $\mathcal{F}_{k,\ell,s}$  problémával foglalkozunk, de az összes  $\mathcal{F}_{k_0,\ell_0,s_0}$  probléma megoldása ismert, ha  $\ell_0 - k_0 < k - \ell$ .

# A göngyölítési lépés

- Tegyük fel, hogy  $k < \ell$  és  $\mathcal{F}_{k,\ell,s}$  problémával foglalkozunk, de az összes  $\mathcal{F}_{k_0,\ell_0,s_0}$  probléma megoldása ismert, ha  $\ell_0 - k_0 < k - \ell$ .
- Csak azok a feladatok érdekesek, ahol a  $P_k, P_\ell, P_s$  hármas sapkát alkot (ez nem volt probléma a  $K = \ell < s$  esetben).

# A göngyölítési lépés

- Tegyük fel, hogy  $k < \ell$  és  $\mathcal{F}_{k,\ell,s}$  problémával foglalkozunk, de az összes  $\mathcal{F}_{k_0,\ell_0,s_0}$  probléma megoldása ismert, ha  $\ell_0 - k_0 < k - \ell$ .
- Csak azok a feladatok érdekesek, ahol a  $P_k, P_\ell, P_s$  hármas sapkát alkot (ez nem volt probléma a  $K = \ell < s$  esetben). Ezt a továbbiakban feltesszük.

# A göngyölítési lépés

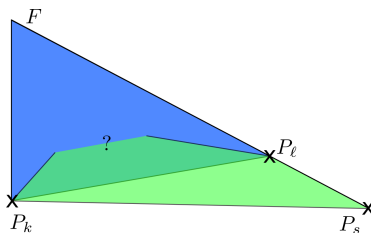
- Tegyük fel, hogy  $k < \ell$  és  $\mathcal{F}_{k,\ell,s}$  problémával foglalkozunk, de az összes  $\mathcal{F}_{k_0,\ell_0,s_0}$  probléma megoldása ismert, ha  $\ell_0 - k_0 < k - \ell$ .
- Csak azok a feladatok érdekesek, ahol a  $P_k, P_\ell, P_s$  hármas sapkát alkot (ez nem volt probléma a  $K = \ell < s$  esetben). Ezt a továbbiakban feltesszük.
- Tippeljük meg milyen csúcs lehet a sapka hátulról harmadik (azaz a  $P_\ell$  előtti) pontja.



# A göngyölítési lépés

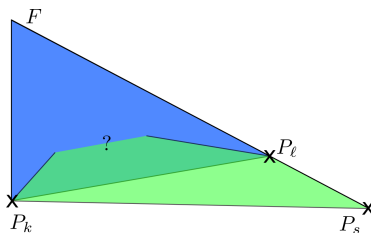
- Tegyük fel, hogy  $k < \ell$  és  $\mathcal{F}_{k,\ell,s}$  problémával foglalkozunk, de az összes  $\mathcal{F}_{k_0,\ell_0,s_0}$  probléma megoldása ismert, ha  $\ell_0 - k_0 < k - \ell$ .
- Csak azok a feladatok érdekesek, ahol a  $P_k, P_\ell, P_s$  hármas sapkát alkot (ez nem volt probléma a  $K = \ell < s$  esetben). Ezt a továbbiakban feltesszük.
- Tippeljük meg milyen csúcs lehet a sapka hátulról harmadik (azaz a  $P_\ell$  előtti) pontja.
- Ez a csúcs geometriailag egy háromszögből kerül ki: A  $P_k$ -n átmenő függőleges egyenestől balra kell lennie a sapka hiányzó részének. A  $P_k P_\ell$  egyenes felett és a  $P_\ell P_s$  egyenes alatt. A kijelölt háromszög határán nem feket a csúcsunk, de kivételt alkot  $P_k$ , ami előfordulhat, mint közvetlenül  $P_\ell$  előtt álló csúcs.

## title



**Figure:**  $F$  a  $P_i$ -n áthaladó függőleges egyenes és a  $P_\ell P_s$  egyenes metszéspontja. A kék  $P_i P_\ell F$  háromszögben kell megkeresni az optimális sapkát.

## title



**Figure:**  $F$  a  $P_i$ -n áthaladó függőleges egyenes és a  $P_\ell P_s$  egyenes metszéspontja. A kék  $P_i P_\ell F$  háromszögben kell megkeresni az optimális sapkát.

Vesszük  $P_k P_\ell F$  háromszögbe eső pontjainkat (ezek  $P_k$  és még esetleg további belső pontok a háromszögből). Mindegyik  $P_\ell$ -pontra a korábbi munkánkból kiolvassuk  $\mathcal{F}_{k, \ell-1, \ell}$  megoldását. Ezekből a számokból a legnagyobbhoz 1-et hozzáadva kapjuk  $\mathcal{F}_{k, \ell, s}$  megoldását.

# Végső megjegyzések

# Végső megjegyzések

- Fontos, hogy az  $\mathcal{O}(n^3)$  feladat közül a már megoldottak megoldását tároljuk és ebből olvassuk ki ami szükséges számunkra.

# Végső megjegyzések

- Fontos, hogy az  $\mathcal{O}(n^3)$  feladat közül a már megoldottak megoldását tároljuk és ebből olvassuk ki ami szükséges számunkra.
- Könnyű az algoritmust átlakítani úgy, hogy ne csak egy maximális méretet adjon meg,

# Végső megjegyzések

- Fontos, hogy az  $\mathcal{O}(n^3)$  feladat közül a már megoldottak megoldását tároljuk és ebből olvassuk ki ami szükséges számunkra.
- Könnyű az algoritmust átlakítani úgy, hogy ne csak egy maximális méretet adjon meg, de egy legnagyobb konvex helyzetű pontthalmaz legyen az output.

# Vége van!

Köszönöm a figyelmet!