

Föreläsning 32 i ADK

# Komplexitetsklasser

Viggo Kann

KTH

# Komplexitetsklasser

- $\mathcal{P}$  – Problem som kan lösas i polynomisk tid
- $\mathcal{NP}$  – Problem som kan lösas i icke-deterministisk polynomisk tid

## Komplementklass:

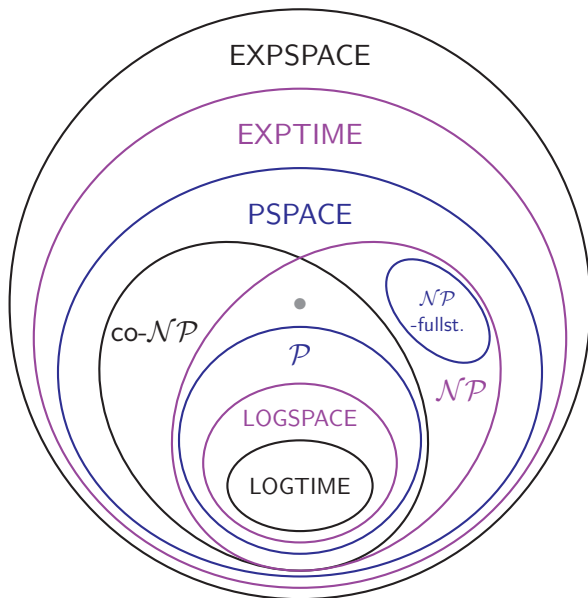
- $\text{co-}\mathcal{NP}$  – Ett problem  $A \in \text{co-}\mathcal{NP}$  om komplementproblemet till  $A \in \mathcal{NP}$ , t.ex. “Är det så att det **inte** finns någon hamiltonsk cykel?”

## Andra tidsklasser:

- LOGTIME – Problem som kan lösas i logaritmisk tid
- EXPTIME – Problem som kan lösas i exponentiell tid

## Minnesklasser:

- LOGSPACE – Problem som kan lösas med tillgång till logaritmiskt mycket minne (oavsett tid)
- PSPACE – Problem som kan lösas med tillgång till polynomiskt mycket minne
- EXPSPACE – Problem som kan lösas med tillgång till exponentiellt mycket minne



## QBF-kvantifierad boolesk formel

**Indata:** En boolesk formel  $\varphi(x_1, x_2, \dots, x_n)$

**Fråga:** Är följande kvantifierade formel sann?

$$\exists x_1 \forall x_2 \exists x_3 \dots Q x_n \quad \varphi(x_1, x_2, \dots, x_n)$$

Där  $Q$  är  $\exists$  om  $n$  är udda och  $\forall$  om  $n$  är jämnt

# Problem i PSPACE

## Bevis av att QBF $\in$ PSPACE

```
function EVALQBF( $i, n, \varphi, \{x_1, \dots, x_{i-1}\}$ )  
  if  $i > n$  then return  $\varphi(x_1, \dots, x_n)$   
   $x_i \leftarrow \text{false}$   
   $\text{fval} \leftarrow \text{EVALQBF}(i + 1, n, \varphi, \{x_1, \dots, x_i\})$   
   $x_i \leftarrow \text{true}$   
   $\text{tval} \leftarrow \text{EVALQBF}(i + 1, n, \varphi, \{x_1, \dots, x_i\})$   
  if ODD( $i$ ) then return  $\text{fval} \vee \text{tval}$   
  else return  $\text{fval} \wedge \text{tval}$ 
```

▷  $\exists$

▷  $\forall$

Anropet  $\text{EVALQBF}(1, n, \varphi, \{\})$  löser QBF i  $\mathcal{O}(n^2)$  minne.

Det går att visa att QBF är PSPACE-fullständigt

# Visa att $\text{LOGSPACE} \subseteq \mathcal{P}$

$Q \in \text{LOGSPACE} \Leftrightarrow \exists$  turingmaskin som löser  $Q$  på ett arbetsband med  $\mathcal{O}(\log n)$  rutor ( $n =$  antal bitar i indata)

## Räkna antalet möjliga konfigurationer för en sådan turingmaskin!

Antal tillstånd hos turingmaskinen:  $\mathcal{O}(1)$

Antal positioner för läshuvudet på indatabandet:  $n$

Antal positioner för huvudet på arbetsbandet:  $\mathcal{O}(\log n)$

Antal möjliga innehåll på arbetsbandet:  $\underbrace{3^{\mathcal{O}(\log n)}}_{\text{om alfabetet är } \{0, 1, B\}}$

Totalt antal möjliga konfigurationer:  $\mathcal{O}(n \log n) \cdot 3^{\mathcal{O}(\log n)}$

**Turingmaskinen kan inte vara i samma konfiguration flera gånger för då blir det en oändlig slinga**

Alltså: en övre gräns för tidskomplexiteten är  
 $\mathcal{O}(n \log n) \cdot 3^{\mathcal{O}(\log n)} = \mathcal{O}(n^{\mathcal{O}(1)})$

d.v.s. polynomisk tid

# Problem i EXPTIME

## Generaliserat schack

**Indata:** En position på ett  $n \times n$ -schackbräde med en svart kung, en vit kung och ett antal svarta och vita bönder, hästar, löpare, torn och damer

**Fråga:** Finns det någon säker vinnande strategi för vit från denna position om man spelar enligt vanliga schackregler?

### Bevis av att Generaliserat schack $\in$ EXPTIME:

- Varje ruta på schackbrädet innehåller antingen:
  - En svart pjäs (6 olika möjligheter)
  - En vit pjäs (6 olika möjligheter)
  - Ingenting (En möjlighet)
- Antalet möjliga positioner i ett spel begränsas av  $2 \cdot (6 + 6 + 1)^{n^2} = 2 \cdot 13^{n^2}$ 
  - tvåan kommer från att det antingen är svarts eller vits tur
- Samma position återkommer aldrig (i en vinnande strategi)
- Storleken på spelträdet blir alltså inte mer än  $2 \cdot 13^{n^2}$

## Fortsättningskurser med ADK som förkunskapskrav:

- DD2440 Avancerade algoritmer, period 1-2, obligatorisk för datalogimastern
- DD2445 Komplexitetsteori, period 1, vartannat (udda) år
- DD2542 Seminariekurs i teoretisk datalogi, algoritmer och komplexitet, period 1 jämna år, alternerar med DD2445
- DD2448 Kryptografins grunder, period 4
- DD2458 Problemlösning och programmering under press, nästa gång i period 3-4 2023