Graph-Spiel *ist PSPACE-vollständig*

Graph-Spiel := $\{(G,s) : G=(V,E)$ gerichteter Graph, $s \in V$, Spieler 1 besitzt eine Gewinnstrategie $\}$

Übung: Graph-Spiel $\in$ PSPACE

Zeige nun: 3QBF $\leqslant_p$ Graph-Spiel, d.h.

- gegeben Φ=„$\exists x_1\ \forall x_2\ \exists x_3\ \ldots\ \exists x_n: \varphi(x_1,\ldots,x_n)$",
  - oBdA. sind die Variablen alternierend quantifiziert
  - $\varphi = C_1 \wedge \ldots \wedge C_k$ mit 3 Literalen in jeder Klausel
- berechne in polynomieller Zeit (G,s) mit:
  - Φ wahr $\Leftrightarrow$ Spieler 1 hat Gewinnstrategie auf (G,s)

3QBF $\leqslant_p$ Graph-Spiel

Gegeben Φ=

„$\exists x_1\ \forall x_2\ \exists x_3 \ldots \exists x_n$: $\varphi(x_1,\ldots,x_n)$“

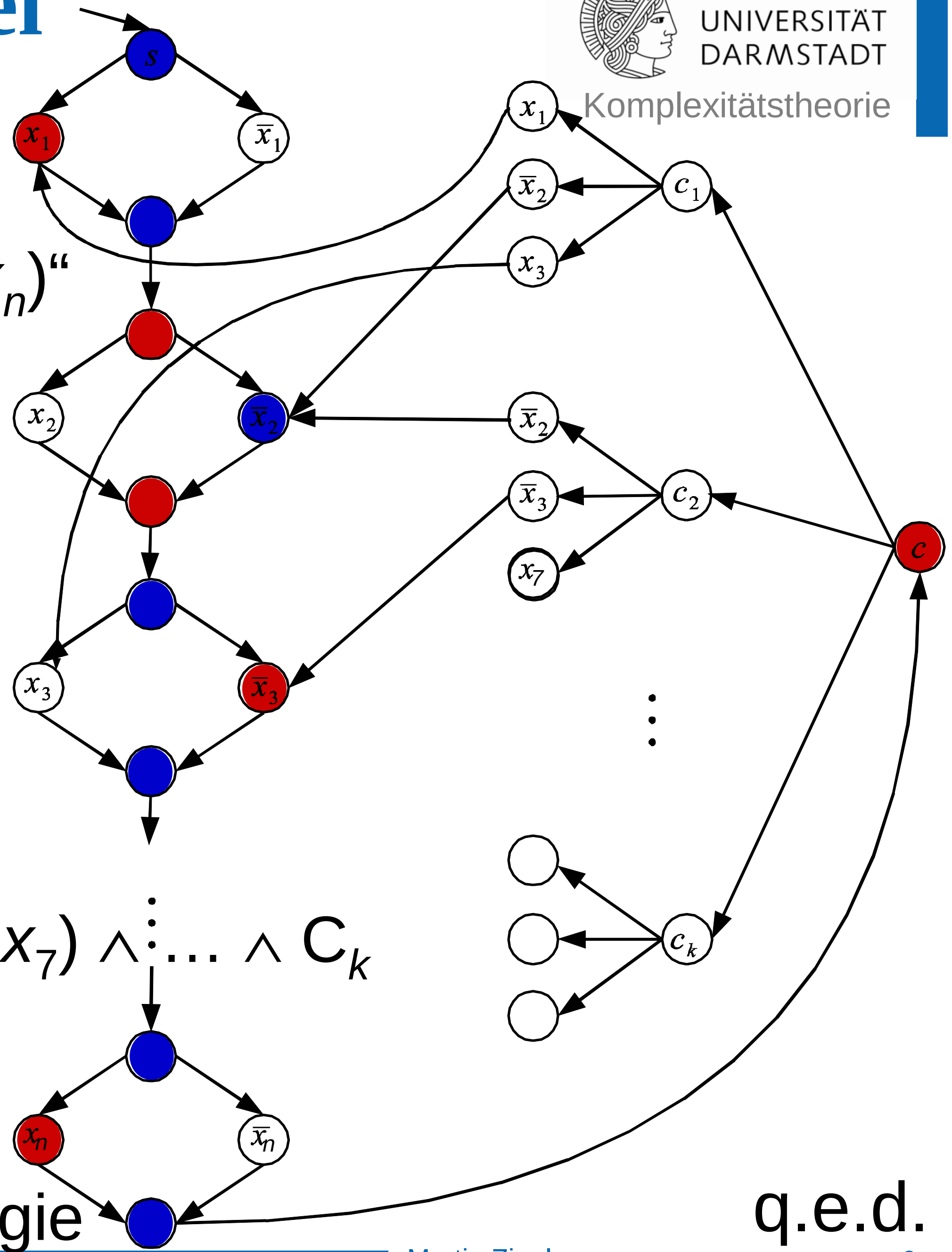
- $\varphi = C_1 \wedge \ldots \wedge C_k$ mit 3 Literalen in jeder Klausel

berechne (G,s) mit:

- Φ wahr ⇔ Spieler 1 hat Gewinnstrategie auf (G,s)

Beispiel für

$\varphi = (x_1 \vee \underline{x}_2 \vee x_3) \wedge (\underline{x}_2 \vee \underline{x}_3 \vee x_7) \wedge \ldots \wedge C_k$

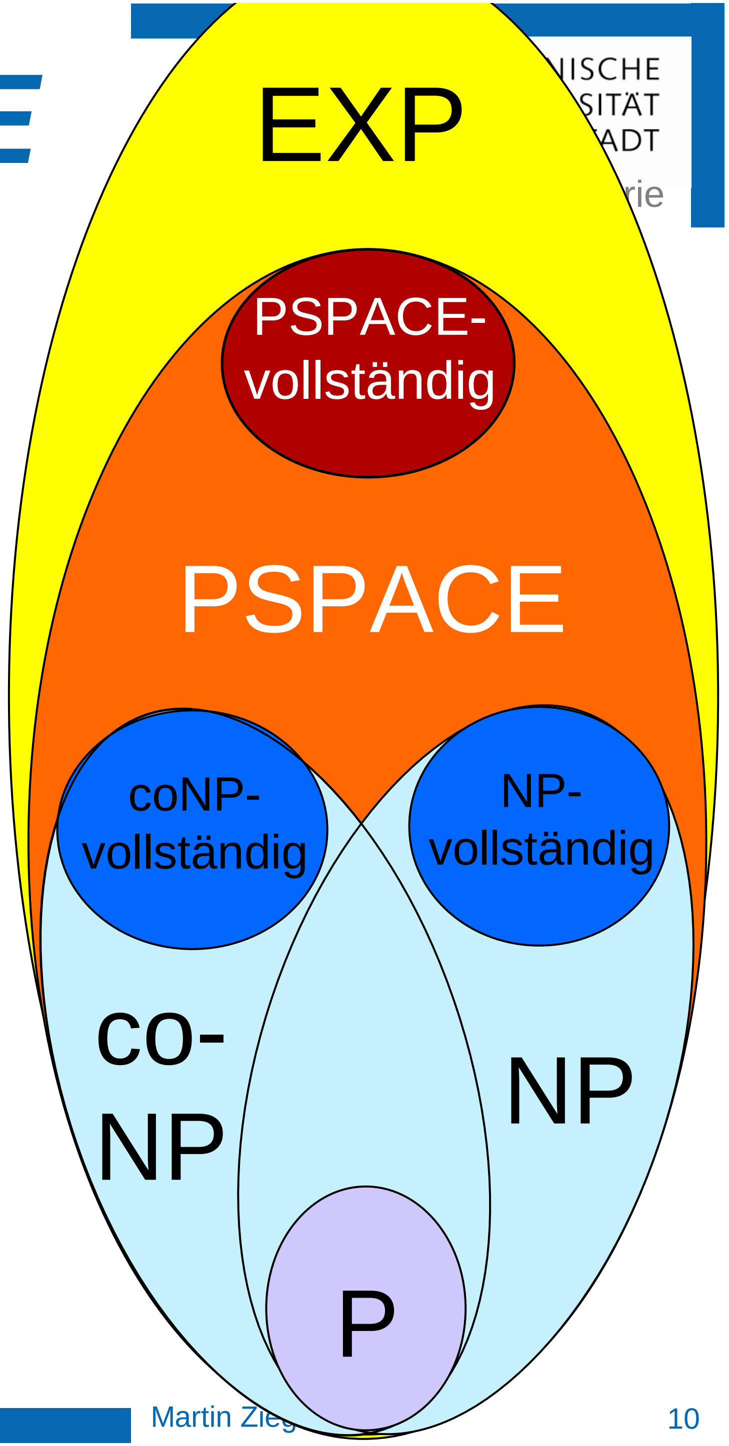


- Φ sei wahr
- Spieler 1 habe Gewinnstrategie

q.e.d.

P, NP und PSPACE

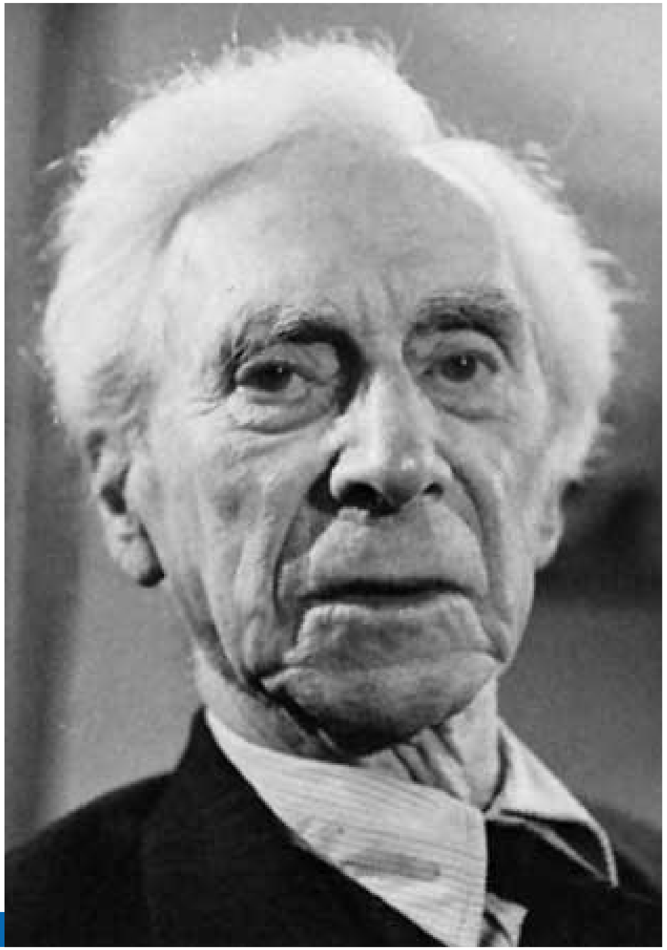
- **P** = $\bigcup_k$ DTIME(n^k)
- **NP** = $\bigcup_k$ NTIME(n^k)
- P $\subseteq$ NP √ •P $\subseteq$ coNP √
- P = NP ? • NP = coNP ?
- **PSPACE** = $\bigcup_k$ DSPACE(n^k)
- NP, coNP $\subseteq$ PSPACE √
- NP = PSPACE ?
- P = PSPACE ?
- **EXP** := $\bigcup_k$ DTIME(2^{n^k})
- PSPACE $\subseteq$ EXP
- **Theorem** (später): P ≠ EXP



Das Wesen der Wissenschaft

Bertrand Russel (1872-1970):

„Darin besteht das Wesen der Wissenschaft:
Zuerst denkt man an etwas, das wahr sein könnte.
Dann sieht man nach, ob es der Fall ist
und im allgemeinen ist es nicht der Fall.“



Erweiterung:

Oft kann man es nicht beweisen
noch widerlegen.

Damit muß man umgehen (lernen)!

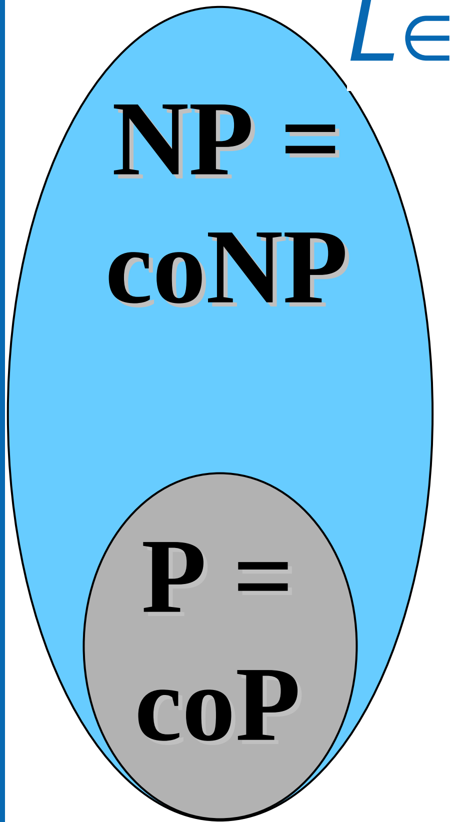
Szenarien für P≠NP

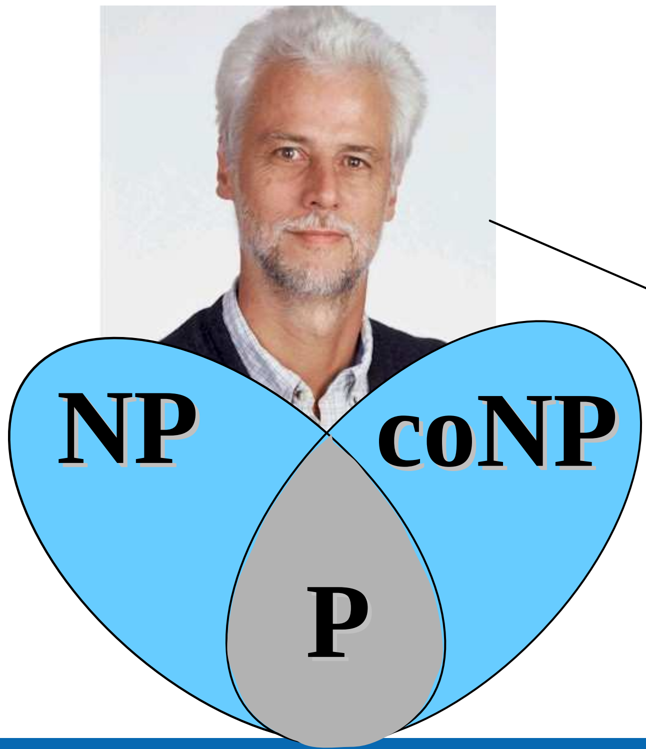
coNP := { L : das Komplement von L liegt in **NP** }

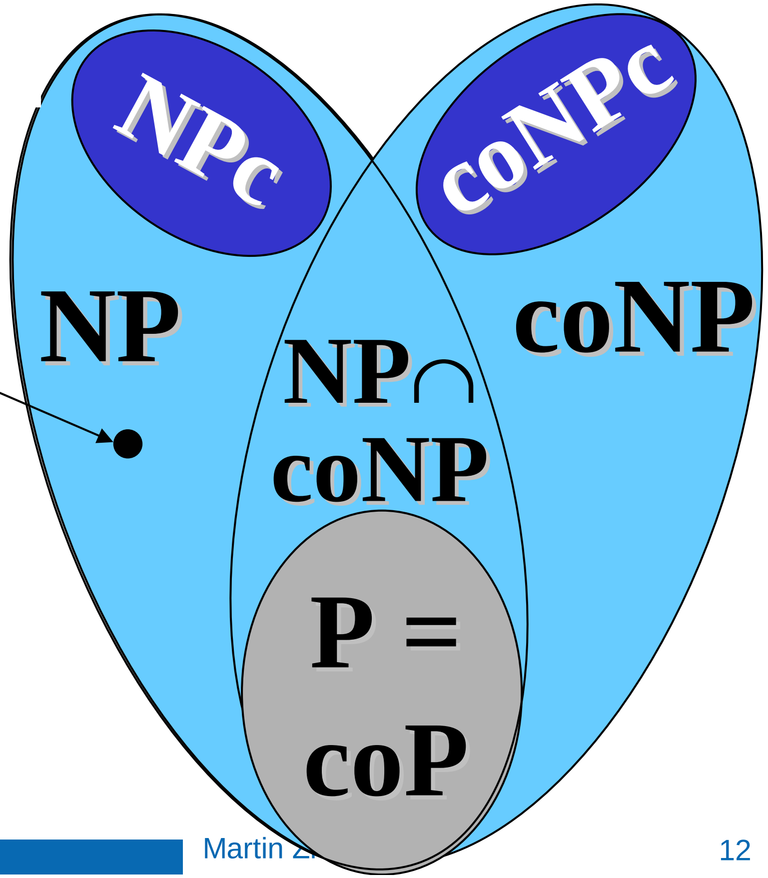
unSAT ={Bool‘Formel $\varphi(x_1,\ldots,x_n)$ falsch $\forall x$} **coNPc**

Theorem (Ladner 1975): Falls **P≠NP**, so gibt es

$L \in$ **NP** \ (**P**$\cup$**NPc**)







2.8 Der Satz von Savitch

Zeit: $\mathbf{P} \subseteq \mathbf{NP}$ √ $\mathbf{P} = \mathbf{NP}$? $1 Mio

Platz: $\mathbf{PSPACE} \subseteq \mathbf{NPSPACE}$ √

Walter Savitch (1970): $\mathbf{PSPACE} = \mathbf{NPSPACE}$

Genauer gilt:

$\mathbf{NSPACE}(s(n)) \subseteq \mathbf{DSPACE}(s(n)^2)$
sofern $s(n) \geq n$ platzkonstruierbar ist.

Def: …d.h. wenn es eine $O(s(n))$-platzbeschränkte DTM gibt, die bei Eingabe „1^n" die Binärdarstellung von $s(n)$ ausgibt.

Beispiele: Polynome, Übung…

Akzeptanz als Erreichbarkeit

Sei M (nicht-/deterministische) TM,
$s(n)$-platzbeschränkt und $t(n)$-zeitbeschränkt.

Erinnerung: $L(M) \leqslant_p \mathrm{QBF}$:

$\mathrm{succ}_\ell(U,U'')$:= «U'' ist Nachfolgekonfig. von U, die nach höchstens 2^ℓ Rechenschritten erreicht werden kann»

$\Leftrightarrow \exists U'$: $\mathrm{succ}_{\ell-1}(U,U') \wedge \mathrm{succ}_{\ell-1}(U',U'')$ $\Leftrightarrow$

$\Big(\exists U' \, \forall V,W$: $((V=U \wedge W=U') \vee (V=U' \wedge W=U'')) \Rightarrow \mathrm{succ}_{\ell-1}(V,W)\Big)$

Dann rekursiv weiter…

Hier: determin. $s(n)^2$-platzbeschränkter Algorithmus

Algorithmus „erreichbar"

erreichbar(U,U'',ℓ)

// Ist U'' von U aus

// in $\leq 2^\ell$ Schritten erreichbar?

– Falls ℓ=0, **`return`**($U=U'' \vee U \vdash U''$)

– Falls $\ell\geq$1, teste für alle U' rekursiv:

- erreichbar(U,U' ,ℓ-1) <u>und</u>
- erreichbar(U',U'',ℓ-1)
- Falls beides ja, **`return(true)`**

– **`return(false)`**

Platzbedarf: $f(\ell) \leq O(S) + f(\ell-1) \leq O(S\cdot\ell)$

NSPACE($s(n)$)⊆DSPACE($s(n)^2$)

NTM N sei $s(n)$-platzbeschränkt;

DTM M simuliert N auf Eingabe $\underline{w}$:

- Berechne $S:=s(|\underline{w}|)$
  und markiere so viel Platz auf dem Band.
  [Nach Vor. „*platzkonstruierbar*" geht das!]
- Berechne $\ell:=\log(T(|\underline{w}|)) \leq c\cdot S$
- und U:=Startkonfig von N (S Zellen),
- U'':=Endkonfig von N (o.B.d.A. eindeutig)
- Aufruf: erreichbar(U,U'',ℓ). Platz $O(S\cdot\ell)$

3.2 Classes **L**, **NL**, and **L**-reductions

L ⊆ **NL** ⊆ **P** ⊆ **NP** ⊆ **PSPACE**=**NPSPACE** ⊆ **EXP**

L := { problems decidable using O(log *n*) tape cells }
but already the input uses $n=|\underline{x}|$ tape symbols.

Def: DTM/NTM with separate *read-only* input tape, count only working/output tape usage

→ **L**, **NL**

Exercise: **NL** ⊆ **P**

Example: $\{ 0^m 1^m : m \in \mathbb{N} \}$

- check the form 0*1*
- count #0s
- and #1s;
- compare these numbers:

only O(log m) tape cells *in addition* to the input!

First Problem in **NL**

TECHNISCHE UNIVERSITÄT DARMSTADT
Komplexitätstheorie

- directed s-t-connectivity:

  $\{ \langle G,s,t\rangle : G=(V,E)$ directed graph,
  $s,t\in V$ with a path from s to t $\}$

  Algorithm directedReachable(V,E,s,t,ℓ):

  – while $\ell>0$ and $s\neq t$ do
    - follow some edge $(s,u)\in E$ leaving from s
    - let $s:=u$, $\ell:=\ell-1$

  – if $s=t$ return(`true`), else return(`false`).

- $\ell:=|V|-1$, s, u each use $O(\log|V|)$ bits.