PSPACE *und* QBF

Bsp: $\varphi(x,y,z) = (x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)$

SAT = alle erfüllbaren booleschen Formeln

$\Leftrightarrow$ alle existenziell quantifizierten wahren BF

z.B. $\Phi =$ „$\exists x\, \exists y\, \exists z\ \ \varphi(x,y,z)$“

QBF := { alle (beliebig) vollständig quantifizier-ten wahren boolesche Formeln Φ }

Bsp: $\Phi =$ „ $\exists x\, \forall y\, \exists z\ \ \varphi(x,y,z)$“

entscheidbar in polynom. Platz

Allgemein:

$$Q_1 x_1\, Q_2 x_2\, Q_3 x_3 \ldots Q_n x_n : \varphi(x_1, x_2, x_3, \ldots, x_n)$$

2.6.7 QBF ist PSPACE-schwer

Gezeigt: Für jedes $L \in NP$ gilt: $L \leqslant_p SAT$

Thm: Für jedes $L \in PSPACE$ gilt: $L \leqslant_p QBF$

d.h. QBF ist PSPACE-schwer

(und damit PSPACE-vollständig)

Warum zeigt Cook-Levin nicht:
"SAT ist PSPACE-schwer?"

M akzeptiert $w \Leftrightarrow \Phi$ ist erfüllbar (nur $\exists$-Quantoren)

Φ hat Länge $O(S \cdot T)$, wobei S,T=Platz/Zeit von M

QBF ist PSPACE-schwer: Beweis

$\text{succ}_\ell(V_1,V_2)$:= „V_2 ist Nachfolgekonfig. von V_1, die nach höchstens 2^ℓ Rechenschritten erreicht werden kann“

Für $t(n)$-zeitbeschr. & $s(n)$-platzbeschr. Maschine M:

$\underline{w} \in L(M) \Leftrightarrow \exists V,V' \in \{0,1\}^{O(s(|\underline{w}|))}$:

$\text{start}(V,\underline{w}) \wedge \text{accept}(V') \wedge \text{succ}_{\log t(|\underline{w}|)}(V,V') \wedge \text{legal}(V) \wedge \text{legal}(V')$

Erinnerung an den Beweis von Cook-Levin:

$\text{legal}(V)$:= “V beschreibt legale Konfiguration (von M)”

$\text{start}(V,\underline{w})$:= “V beschreibt Startkonfigur. von M auf w”

$\text{accept}(V)$:= “V beschreibt eine akzept. Konfiguration”

$\text{succ}(V_1,V_2)$:= “V_2 ist *direkte* Nachfolgekonfig. von V_1”

QBF ist PSPACE-schwer: Beweis

$\mathrm{succ}_\ell(V_1,V_2)$:= „V_2 ist Nachfolgekonfig. von V_1, die nach höchstens 2^ℓ Rechenschritten erreicht werden kann“

Für $t(n)$-zeitbeschr. & $s(n)$-platzbeschr. Maschine M:

$\underline{w}\in \mathrm{L}(M) \Leftrightarrow \exists V,V'\in\{0,1\}^{O(s(|\underline{w}|))}$: $s(n)=\mathrm{poly}(n),\ t(n)=2^{O(s(n))}$

$\mathrm{start}(V,\underline{w}) \wedge \mathrm{accept}(V') \wedge \mathrm{succ}_{\log t(|\underline{w}|)}(V,V') \wedge \mathrm{legal}(V) \wedge \mathrm{legal}(V')$

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

Aber succ_ℓ so aufzuschreiben, erfordert zu viel Platz&Zeit andererseits noch kein „$\forall$“ benutzt… Also weiter:

$\mathrm{succ}_{\ell'}(V_1,V_2) \wedge \mathrm{succ}_{\ell'}(V_2,V_3) \Leftrightarrow$

$\big(\forall U,U'\colon ((U=V_1 \wedge U'=V_2) \vee (U=V_2 \wedge U'=V_3)) \Rightarrow \mathrm{succ}_{\ell'}(U,U')\big)$

jetzt Rekursion ab $\ell:=\log t(|\underline{w}|)$: succ_ℓ, $\mathrm{succ}_{\ell-1}$, $\mathrm{succ}_{\ell-2}$

3QBF ebenfalls PSPACE-schwer

- Ergibt quantifizierte Formel welcher Länge? / berechenbar in welcher Laufzeit?
- Wie viele Quantoren welcher Sorte?

QBF := vollständig quantifizierte wahre boolesche Formeln Φ; z.B. „$\exists x\, \forall y\, \exists z\;\; \varphi(x,y,z)$“

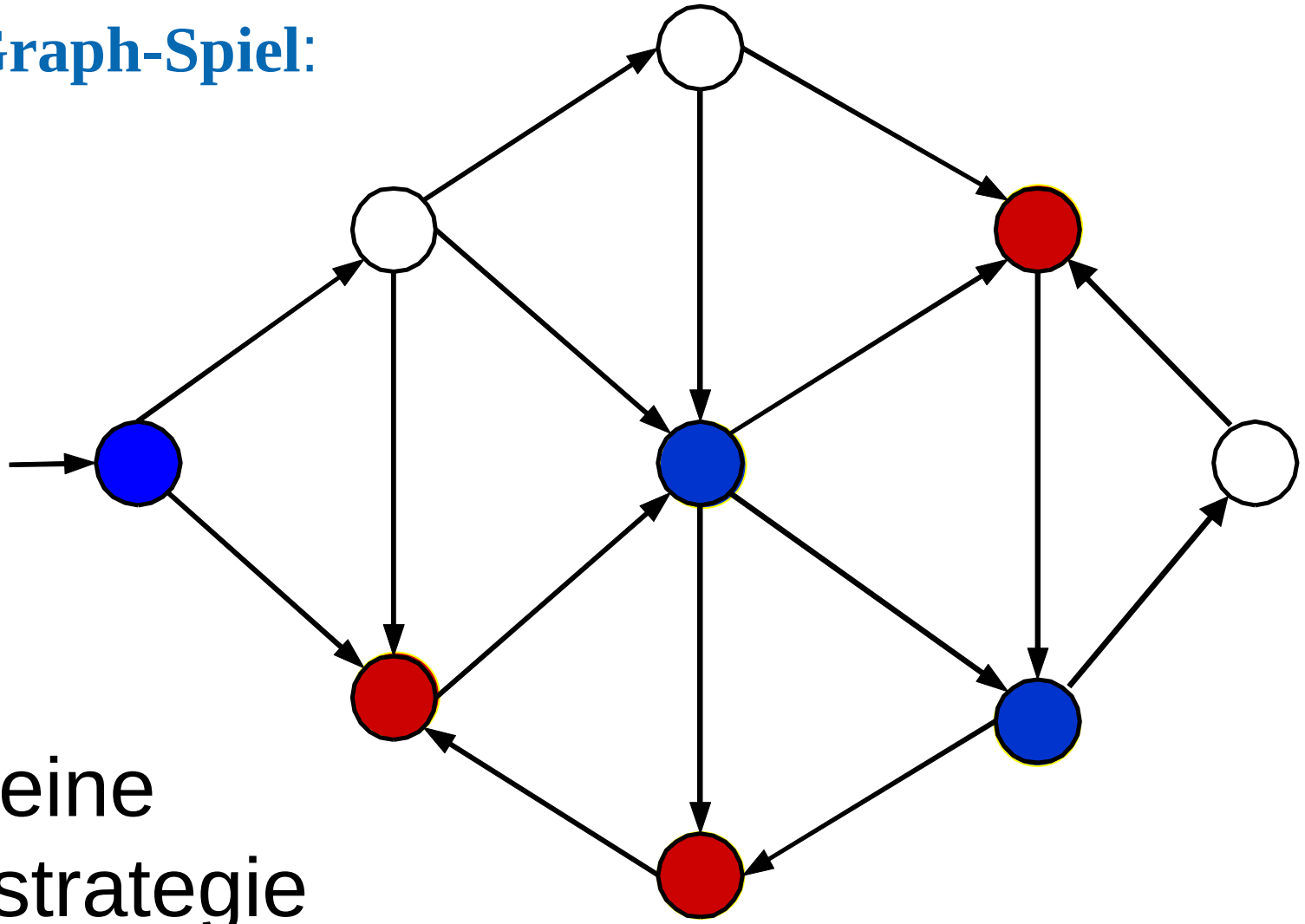
- o.B.d.A. Quantoren „$\exists x_1\, \forall x_2\, \exists x_3\, \forall x_4\, \ldots \exists x_n$“
- o.B.d.A. φ in 3KNF

$$\mathrm{succ}_{\ell'}(V_1,V_2) \wedge \mathrm{succ}_{\ell'}(V_2,V_3) \Leftrightarrow$$
$$\big(\forall U,U'\colon ((U{=}V_1 \wedge U'{=}V_2) \vee (U{=}V_2 \wedge U'{=}V_3)) \Rightarrow \mathrm{succ}_{\ell'}(U,U')\big)$$

jetzt Rekursion ab $\ell{:=}\log t(|\underline{w}|)$: succ_ℓ, $\mathrm{succ}_{\ell-1}$, $\mathrm{succ}_{\ell-2}$

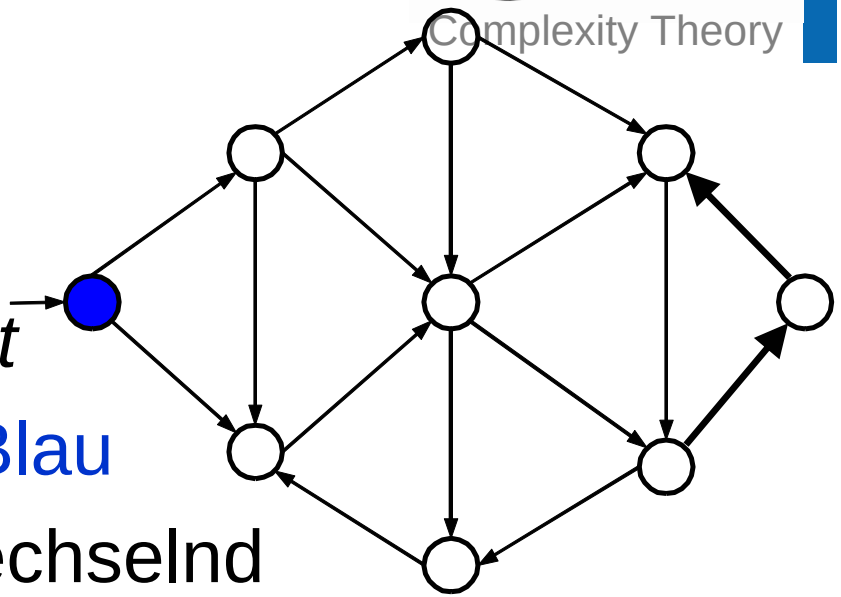
Bewiesen: Es gibt eine PSPACE-vollständige Sprache

Jetzt ein **Graph-Spiel**:



Rot hat eine Gewinnstrategie

Graph-Spiel



Spielregeln:

- gerichteter Graph *G*
- Startknoten *s*, *aktiv*, *markiert*
- zwei SpielerInnen Rot und Blau
- wählen und markieren abwechselnd
- einen neuen *aktiven* Knoten
  - Ziel einer Kante vom bisherigen
  - noch *nicht markiert*.
- Wer nicht mehr ziehen kann, verliert; der andere gewinnt.

Frage: *Gibt es eine Gewinnstrategie für Rot?*

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=

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

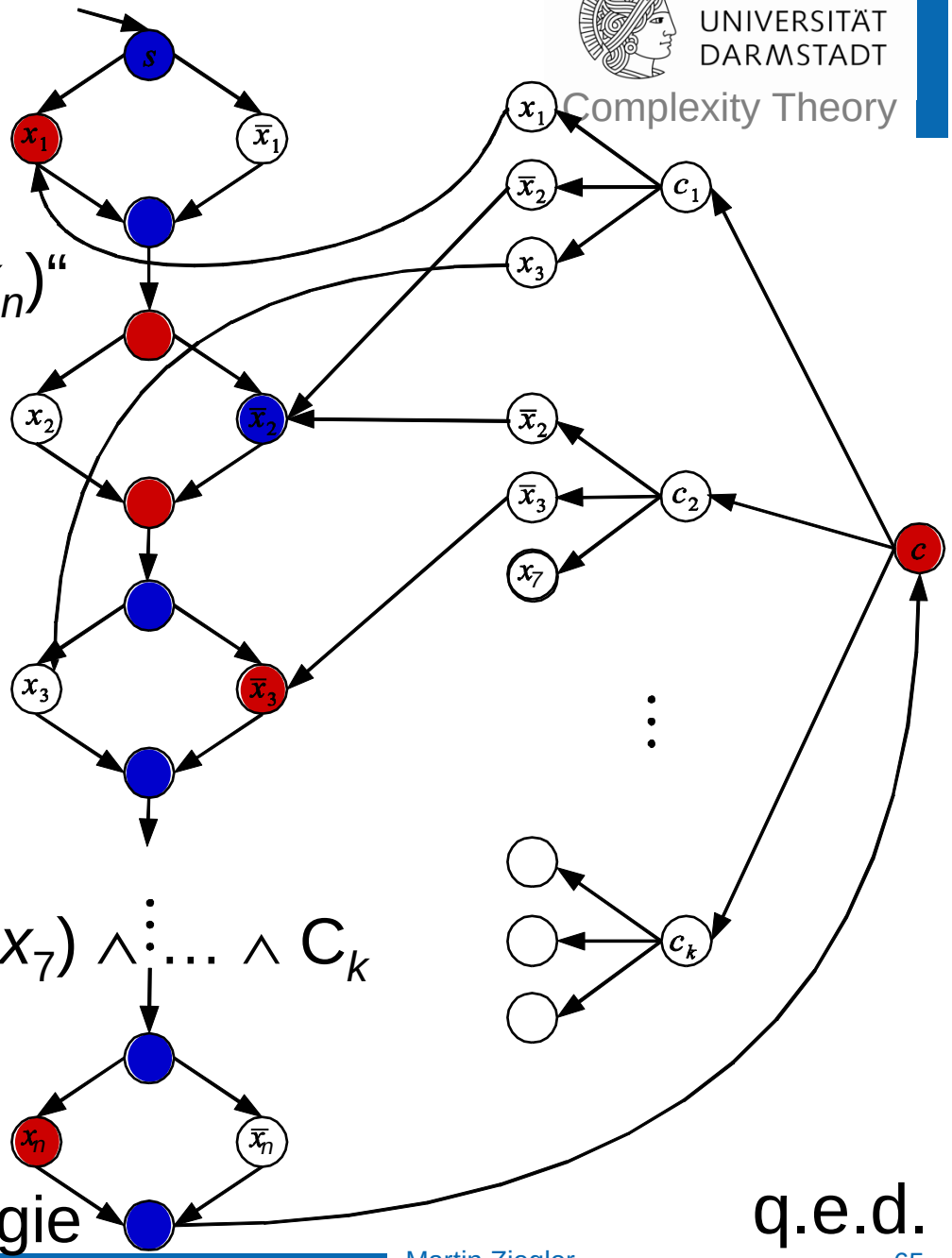
- $\varphi = C_1 \wedge \ldots \wedge C_k$ mit 3 Literalen in jeder Klausel

berechne (G,s) mit:

- Φ wahr $\Leftrightarrow$ 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

$\mathbf{P} = \bigcup_k \mathrm{DTIME}(n^k)$

$\mathbf{NP} = \bigcup_k \mathrm{NTIME}(n^k)$

$P \subseteq NP$ √

$P = NP$?

- $P \subseteq coNP$ √
- $NP = coNP$?

$\mathbf{PSPACE} = \bigcup_k \mathrm{DSPACE}(n^k)$

$NP, coNP \subseteq PSPACE$ √

$NP = PSPACE$?

$P = PSPACE$?

$\mathbf{EXP} := \bigcup_k \mathrm{DTIME}(2^{n^k})$

$PSPACE \subseteq EXP$

Theorem (später): $P \neq EXP$