Polynomiell berechenbare Funktionen

Definition 3.18 $f : \Sigma^* \to \Gamma^*$ **heißt polynomiell berechenbar, wenn es eine DTM M mit polynomieller Laufzeit gibt, die bei jeder Eingabe** $w \in \Sigma^*$ **im akzeptierenden Zustand und mit Bandinhalt f(w) hält.**

Dabei werden $\triangleright$ **und** $\sqcup$**‘s ignoriert.**

Lemma 3.19 **Seien** $f : \Sigma^* \to \Gamma^*$ **und** $g : \Gamma^* \to \Xi^*$ **polynomiell berechenbar. Dann ist auch** $g \circ f : \Sigma^* \to \Xi^*$ **polynomiell berechenbar.**

Polynomielle Reduktionen

Definition 3.20 Seien A,B Sprachen. A heißt auf B **polynomiell reduzierbar**, wenn es eine polynomiell berechenbare Funktion f gibt mit

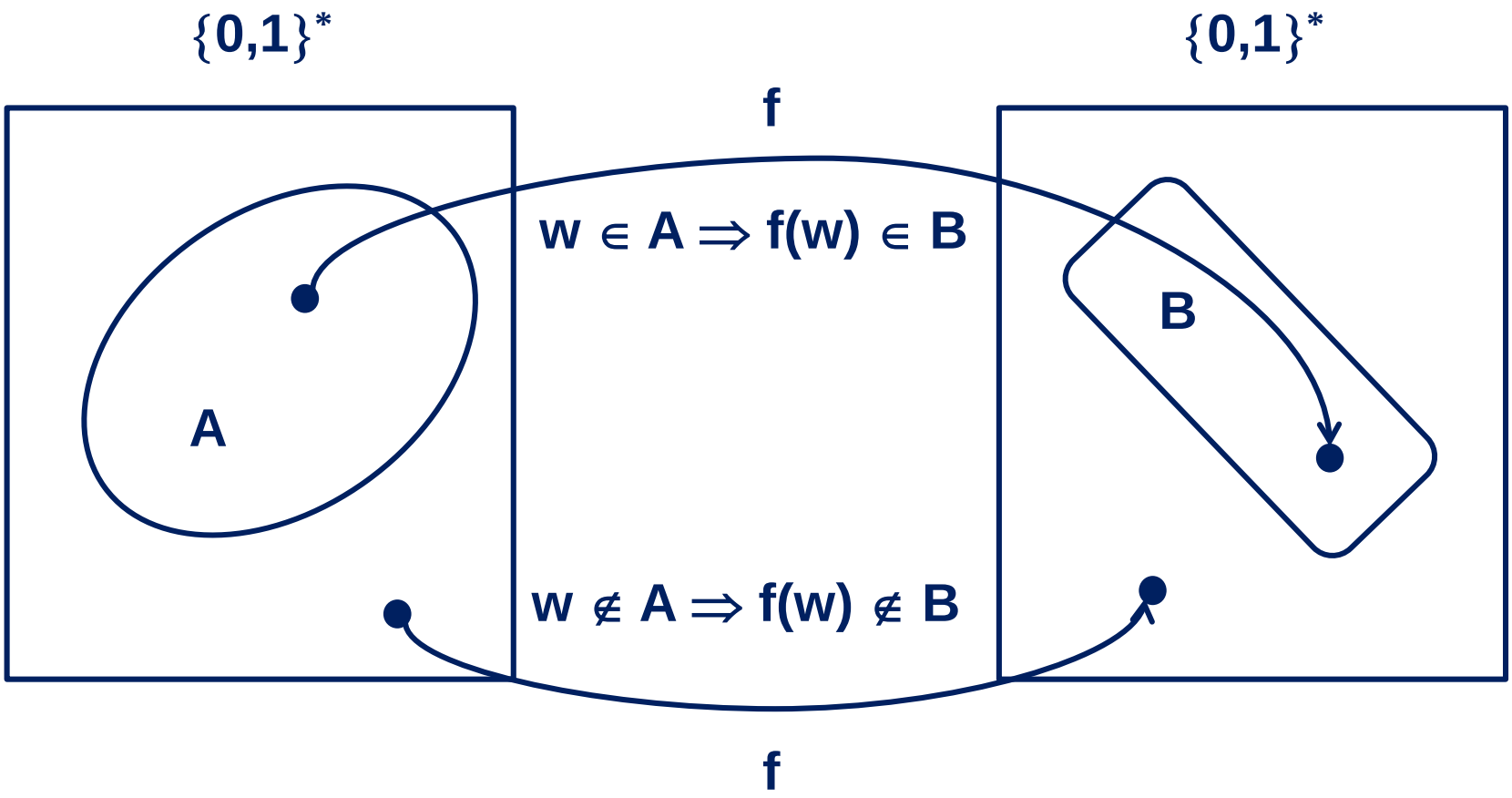
$$w \in A \Leftrightarrow f(w) \in B \text{ für alle } w.$$

Die Funktion wird in diesem Fall eine **polynomielle Reduktion** genannt.

Ist A auf B polynomiell reduzierbar, so schreiben wir

$$A \leq_p B.$$

Reduktionen – Graphische Darstellung



Zwei einfache Sprachen

$$PAL := \{ w \in \{0,1\}^* \mid w \text{ ist ein Palindrom.} \}$$

$$XOR := \{ (a,b,c) \in \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^* \mid a,b,c \text{ haben die gleiche Länge und } a \oplus b = c. \}$$

$$f : \{0,1\}^* \rightarrow \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^*$$

$$w \rightarrow (w, w^R, 0^{|w|})$$

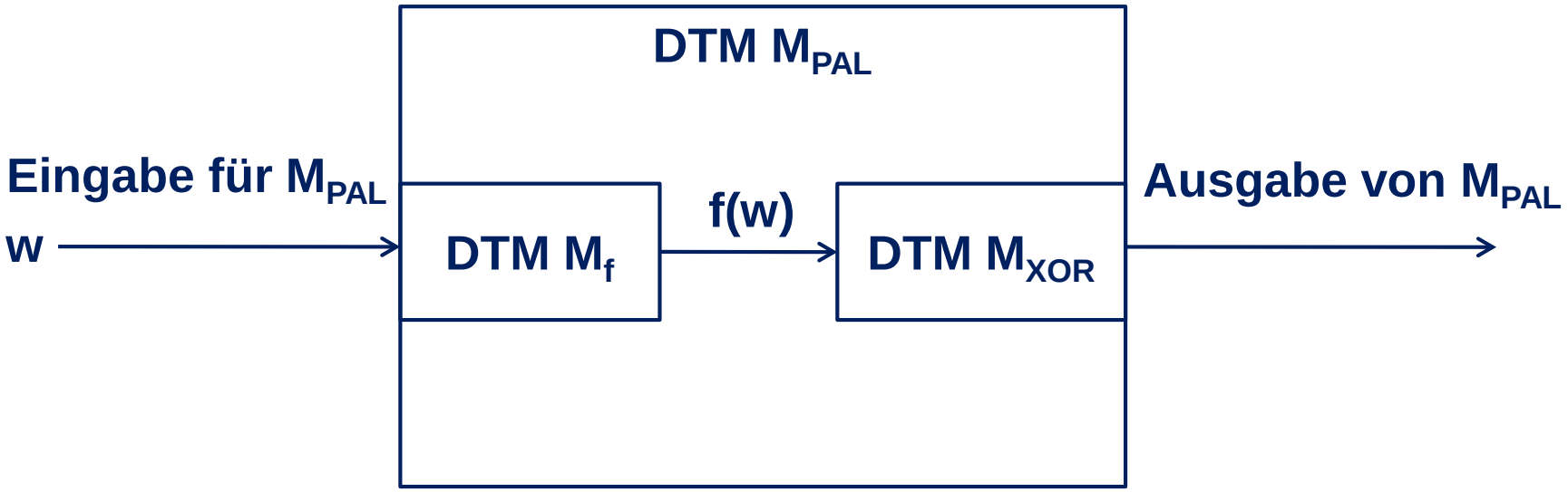
Von XOR und f zu PAL

M_{PAL} bei Eingabe $w \in \{0,1\}^*$:

1. Berechne mit M_f das Tripel $f(w) = (w, w^R, 0^{|w|})$.
2. Simuliere M_{XOR} mit Eingabe f(w).
3. Falls M_{XOR} die Eingabe f(w) akzeptiert, akzeptiere w.
4. Falls M_{XOR} die Eingabe f(w) ablehnt, lehne w ab.

M_{XOR} entscheidet XOR, M_f berechnet f.

Graphische Darstellung von M_P



Da f polynomiell berechenbar ist, ist also PAL $\leq_p$ XOR

Eigenschaften polynomieller Reduktionen

Satz 3.21 Gilt $A \leq_p B$ und $B \in P$, so folgt $A \in P$.

Da $PAL \leq_p XOR$ und $XOR \in P$, ist damit auch $PAL \in P$.

Von B und f zu A

M_A bei Eingabe $w \in \{0,1\}^*$:

1. Berechne mit M_f die Folge f(w).
2. Simuliere M_B mit Eingabe f(w).
3. Akzeptiere w genau dann, wenn M_B f(w) akzeptiert.

- M_f, M_B polynomielle DTMs.
- M_f berechnet f.
- M_B entscheidet B.

Eigenschaften polynomieller Reduktionen

Satz 3.21 Gilt $A \leq_p B$ und $B \in P$, so folgt $A \in P$.

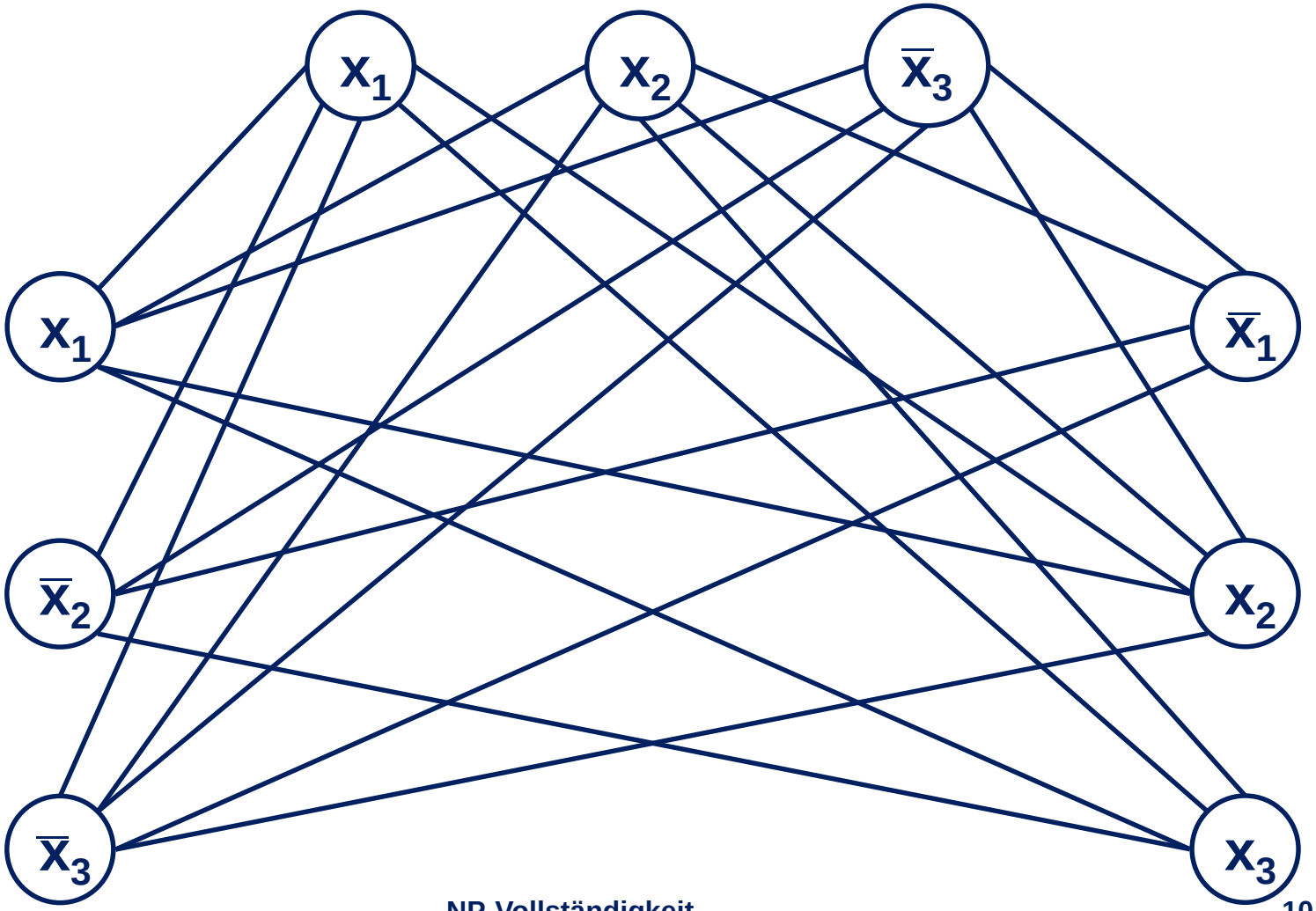
Lemma 3.19 Seien $f: \Sigma^* \to \Gamma^*$ und $g: \Gamma^* \to \Xi^*$ polynomiell berechenbar. Dann ist auch $g \circ f : \Sigma^* \to \Xi^*$ polynomiell berechenbar.

Lemma 3.22 Gilt $A \leq_p B$ und $B \leq_p C$, so gilt $A \leq_p C$.

Satz 3.23 3SAT ist auf Clique polynomiell reduzierbar, d.h., $3SAT \leq_p$ Clique.

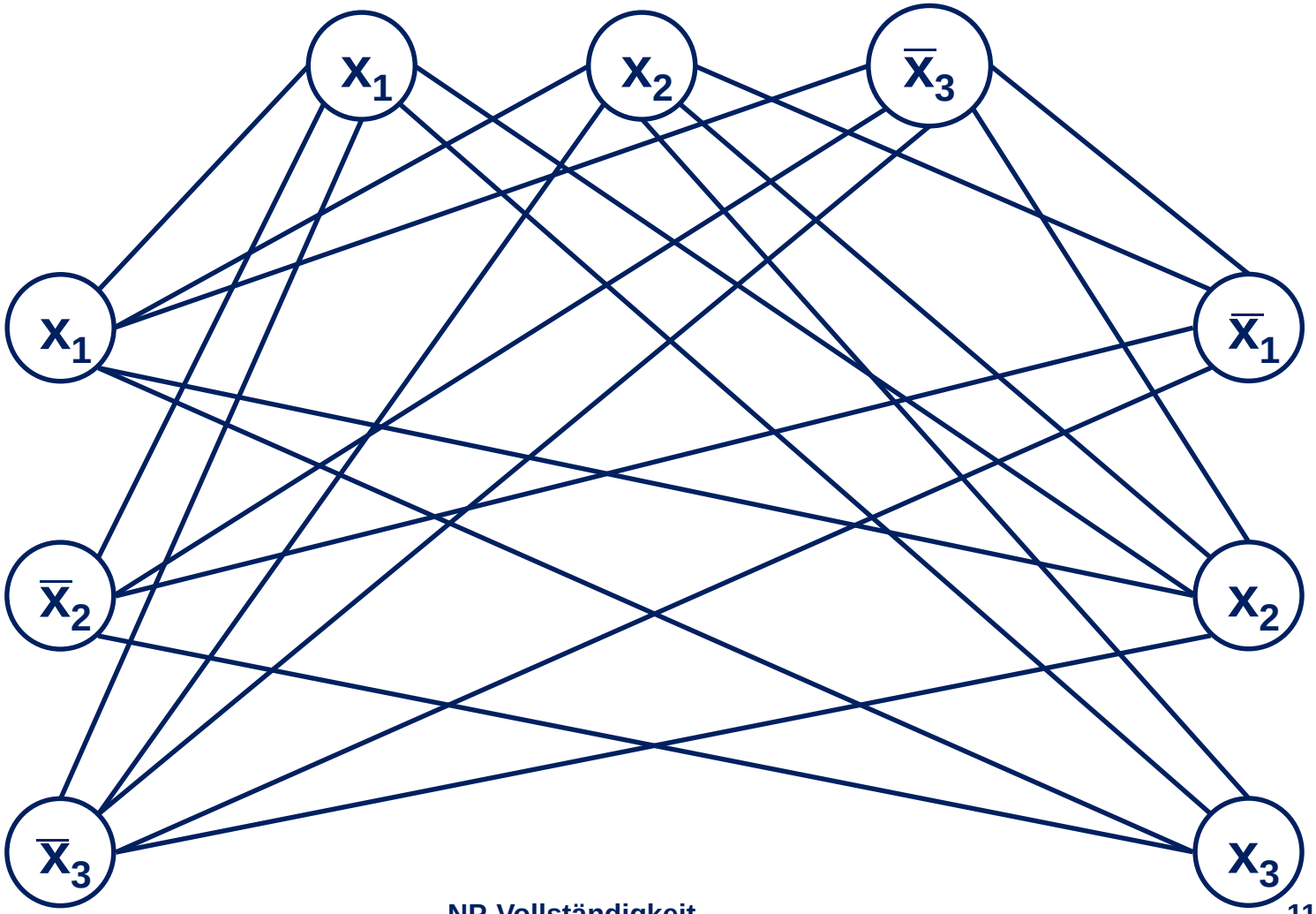
Reduktion 3SAT auf Clique

$$\varphi(x_1, x_2, x_3) = (x_1 \vee \overline{x}_2 \vee \overline{x}_3) \wedge (x_1 \vee x_2 \vee \overline{x}_3) \wedge (\overline{x}_1 \vee x_2 \vee x_3)$$



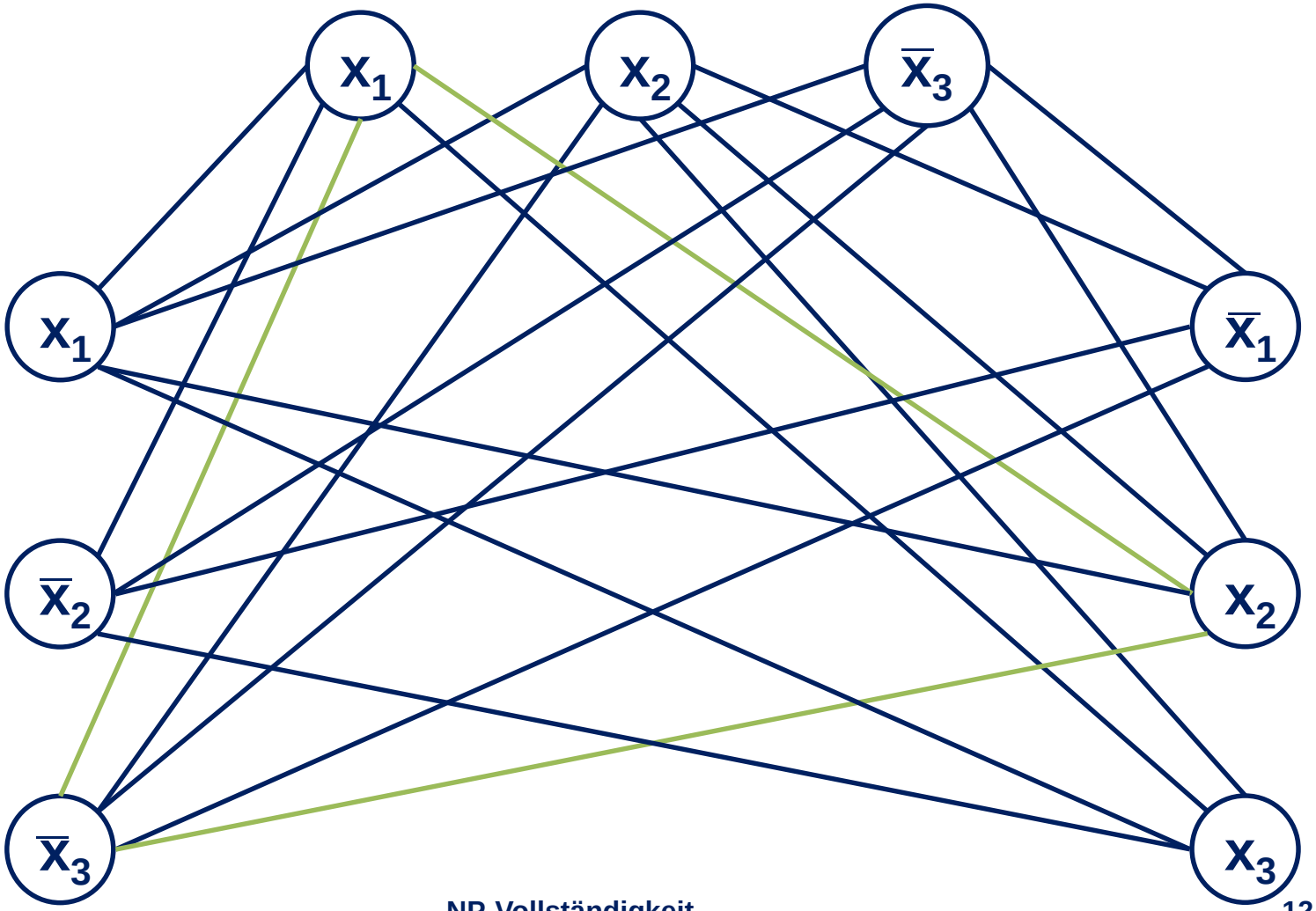
Reduktion 3SAT auf Clique

$$\varphi(x_1, x_2, x_3) = (x_1 \vee \overline{x}_2 \vee \overline{x}_3) \wedge (x_1 \vee x_2 \vee \overline{x}_3) \wedge (\overline{x}_1 \vee x_2 \vee x_3)$$



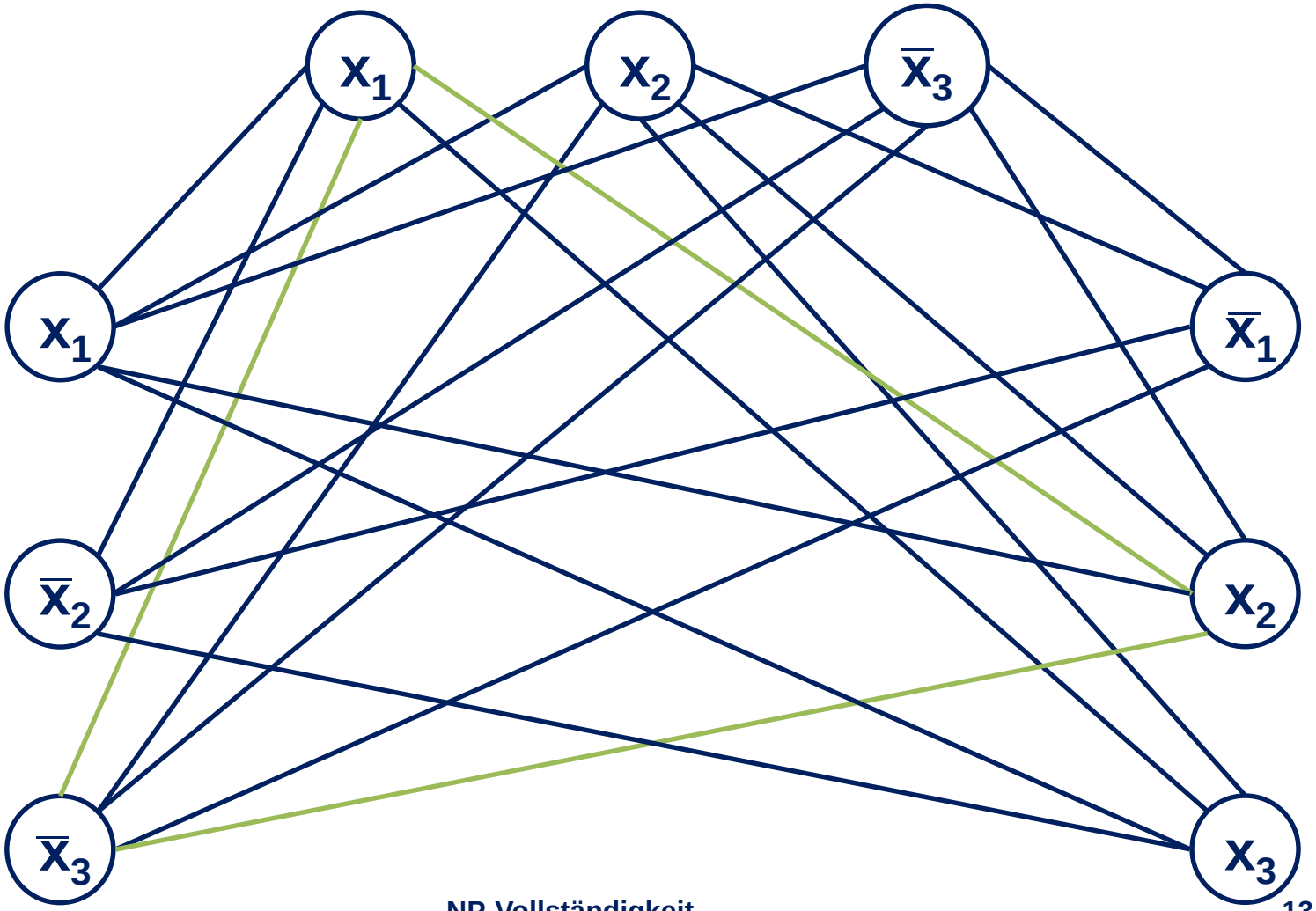
Reduktion 3SAT auf Clique

$$\varphi(x_1, x_2, x_3) = (x_1 \vee \overline{x}_2 \vee \overline{x}_3) \wedge (x_1 \vee x_2 \vee \overline{x}_3) \wedge (\overline{x}_1 \vee x_2 \vee x_3)$$



Reduktion 3SAT auf Clique

$$\varphi(x_1, x_2, x_3) = (x_1 \vee \overline{x}_2 \vee \overline{x}_3) \wedge (x_1 \vee x_2 \vee \overline{x}_3) \wedge (\overline{x}_1 \vee x_2 \vee x_3)$$



NP-Vollständigkeit - Definition

Definition 3.24 **Eine Sprache L heißt NP-vollständig, wenn sie die beiden folgenden Bedingungen erfüllt.**

1. **L ist in NP enthalten.**
2. **Für jede Sprache A in NP gibt es eine polynomielle Reduktion von A auf L.**

NP-Vollständigkeit - Eigenschaften

Satz 3.25 **Ist L NP-vollständig und ist $L \in P$, so gilt $P = NP$.**

Satz 3.26 **Ist $A \in NP$ und gilt $L \leq_p A$ für eine NP-vollständige Sprache L, so ist A NP-vollständig.**

Satz 3.23 **3SAT ist auf Clique polynomiell reduzierbar, d.h., $3SAT \leq_p$ Clique.**

Also: **Ist 3SAT NP-vollständig, dann auch Clique.**

Der Satz von Cook-Levin

Satz 3.27 **SAT ist NP-vollständig.**

Satz 3.29 **3SAT ist NP-vollständig.**

Beweis von Satz 3.27

SAT $\in$ NP: wissen wir bereits

Wir kennen noch keine NP-vollständige Sprache, also müssen wir zeigen: für alle L $\in$ NP gilt L $\leq_p$ SAT

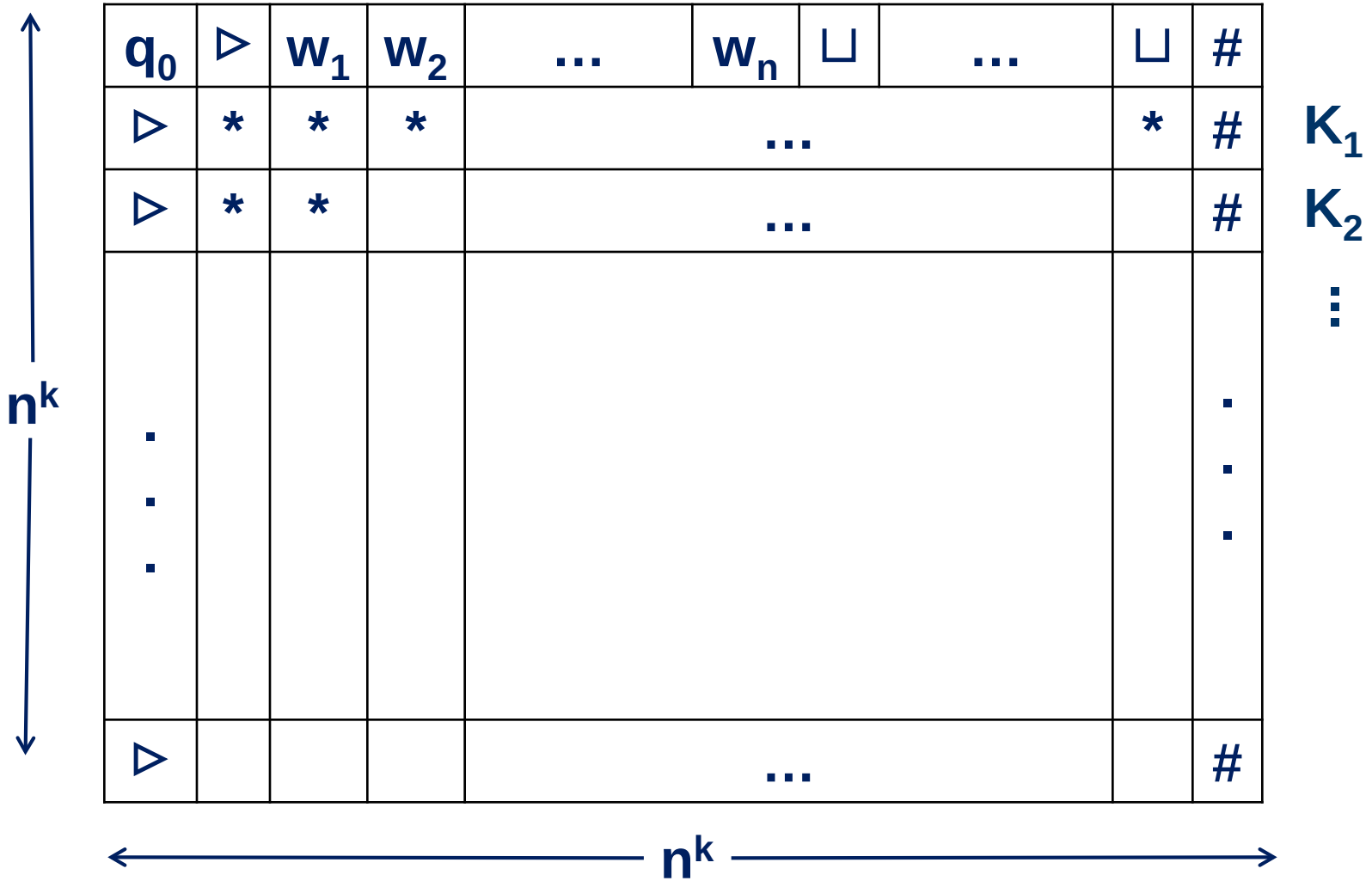
Konfigurationen und Berechnungstabellen

- Sei N eine NTM.
- Eingabe für N ist w, dann heißt $q_0 \triangleright w$ **Startkonfiguration**.
- $K = \alpha q \beta$ heißt **akzeptierende** Konfiguration, falls $q = q_{accept}$.
- $K = \alpha q \beta$ heißt **ablehnende** Konfiguration, falls $q = q_{reject}$.
- **Rechnung** von N bei Eingabe w: Folge $K_1, K_2, \ldots$ von Konfigurationen.
- Es gibt mehrere Rechnungen von N bei Eingabe w, abhängig von den ausgewählten Rechenschritten.
- Darstellung von Rechnung durch **Berechnungstabelle**.

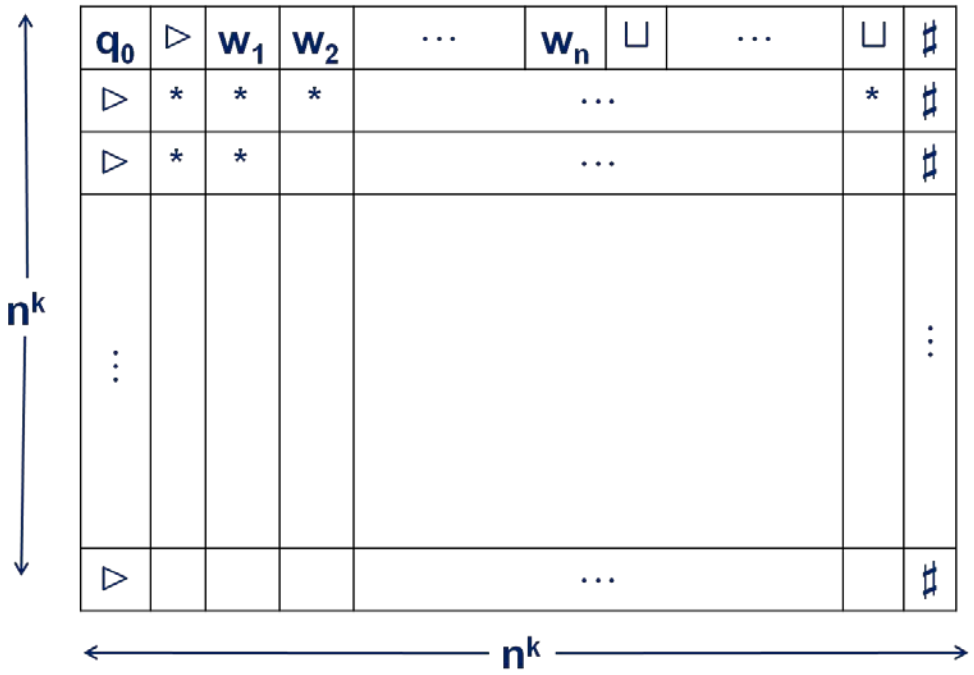
Konfigurationen und Berechnungstabellen

- Laufzeit von NTM N sei $n^k - 2$ (existiert für $L \in NP$).
- Konfigurationen $\alpha q \beta$ bestehen aus höchstens $n^k - 1$ Symbolen aus $Q \cup \Gamma$.
- Ergänzen Konfigurationen $\alpha q \beta$ um Endsymbol #.
- $|\alpha q \beta \#| \leq n^k$.
- Jede Rechnung besteht aus höchstens n^k solcher Konfigurationen.
- Fassen Konfigurationen einer Rechnung in $n^k \times n^k$ Tabelle zusammen.

Berechnungstabellen

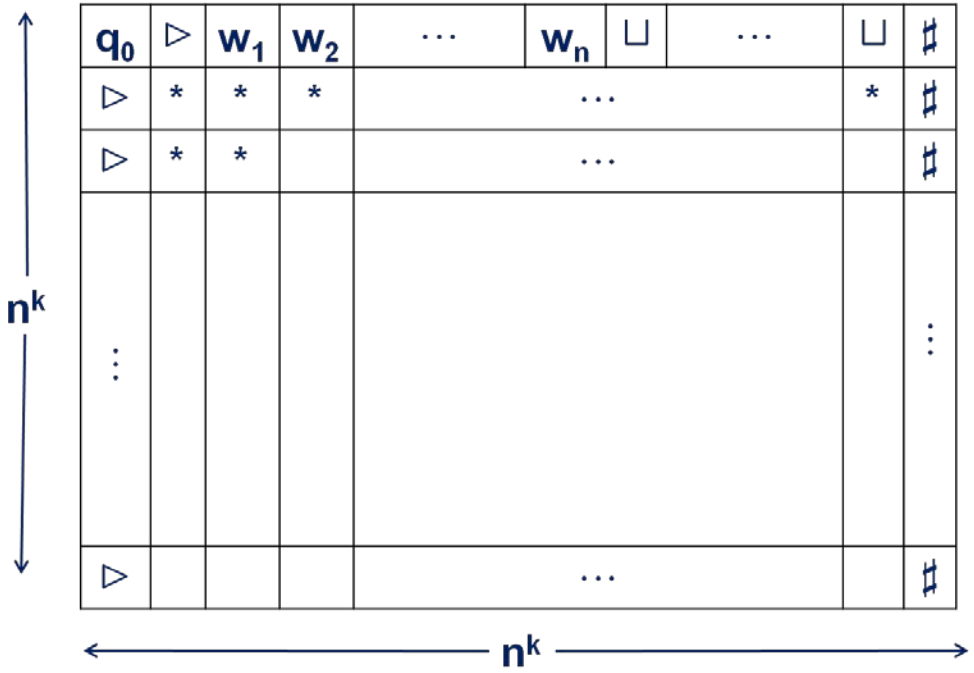


Berechnungstabellen



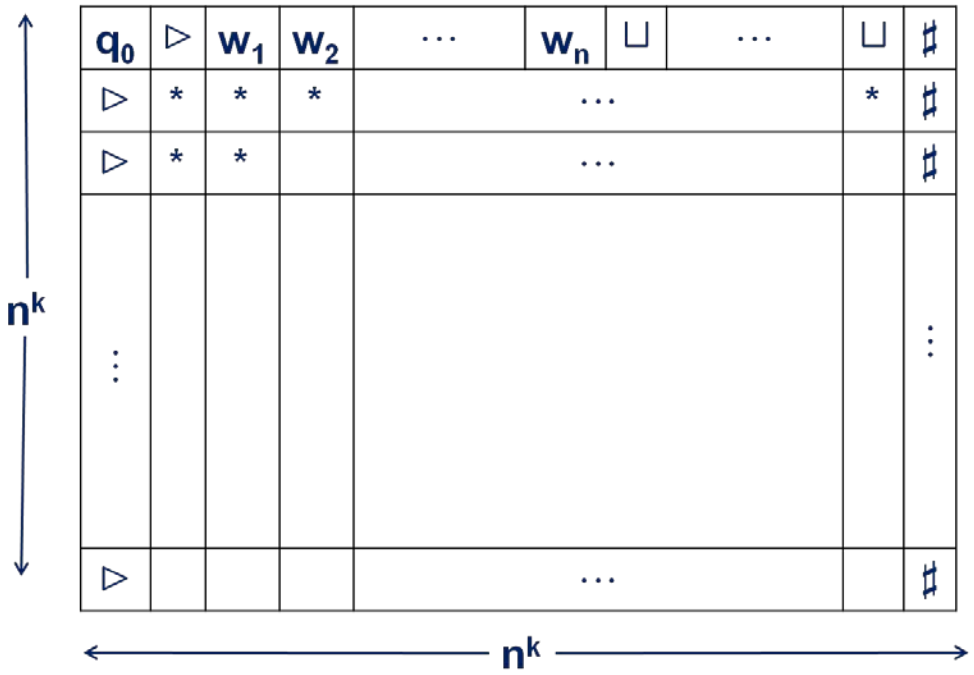
- **Berechnungstabelle heißt akzeptierend, falls sie das Symbol q_{accept} enthält.**

Berechnungstabellen



Beobachtung: **w wird von N akzeptiert genau dann, wenn es eine akzeptierende Berechnungstabelle von N bei Eingabe w gibt.**

Von Tabellen zu Formeln



Ziel: Gegeben NTM $N = (Q,\Sigma,\Gamma,\delta)$ und $w \in \Sigma^*$, konstruieren wir Boolesche Formel ϕ, so dass ϕ genau dann erfüllbar ist, wenn es eine akzeptierende Berechnungstabelle von N bei Eingabe w gibt.

Struktur der Formel

- $C := Q \cup \Gamma \cup \{\#\}$
- Boolesche Variablen $x_{i,j,s}$, $1 \leq i,j \leq n^k$, $s \in C$.
- Interpretation
  - ✓ pro Eintrag in Berechnungstabelle und pro möglichem Symbol eine Variable.
  - ✓ $x_{i,j,s} = 1 :\Leftrightarrow$ Eintrag an Position (i,j) der Berechnungstabelle ist $s \in C$.

Die Struktur der Formel

$$\phi = \phi_{Start} \wedge \phi_{accept} \wedge \phi_{Eintrag} \wedge \phi_{move}$$

- ϕ_{Start} : 1. Zeile entspricht Startkonfiguration.
- ϕ_{accept} : Berechnungstabelle ist akzeptierend.
- $\phi_{Eintrag}$: Einträge korrekt formatiert.
- ϕ_{move} : Aufeinander folgende Zeilen entsprechen Konfiguration und Nachfolgekonfiguration.

Die Struktur der Formel

$$\phi_{Start} = x_{1,1,q_0} \wedge x_{1,2,\triangleright} \wedge x_{1,3,w_1} \wedge x_{1,4,w_2} \wedge \dots \wedge x_{1,n+2,w_n} \wedge x_{1,n+3,\sqcup} \wedge \dots \wedge x_{1,n^k-1,\sqcup} \wedge x_{1,n^k,\#}$$

Die Struktur der Formel

$$\phi_{Start} = x_{1,1,q_0} \wedge x_{1,2,\triangleright} \wedge x_{1,3,w_1} \wedge x_{1,4,w_2} \wedge \dots \wedge x_{1,n+2,w_n} \wedge x_{1,n+3,\sqcup} \wedge \dots \wedge x_{1,n^k-1,\sqcup} \wedge x_{1,n^k,\#}$$

$$\phi_{accept} = \bigvee_{1 \le i,j \le n^k} x_{i,j,q_{accept}}$$

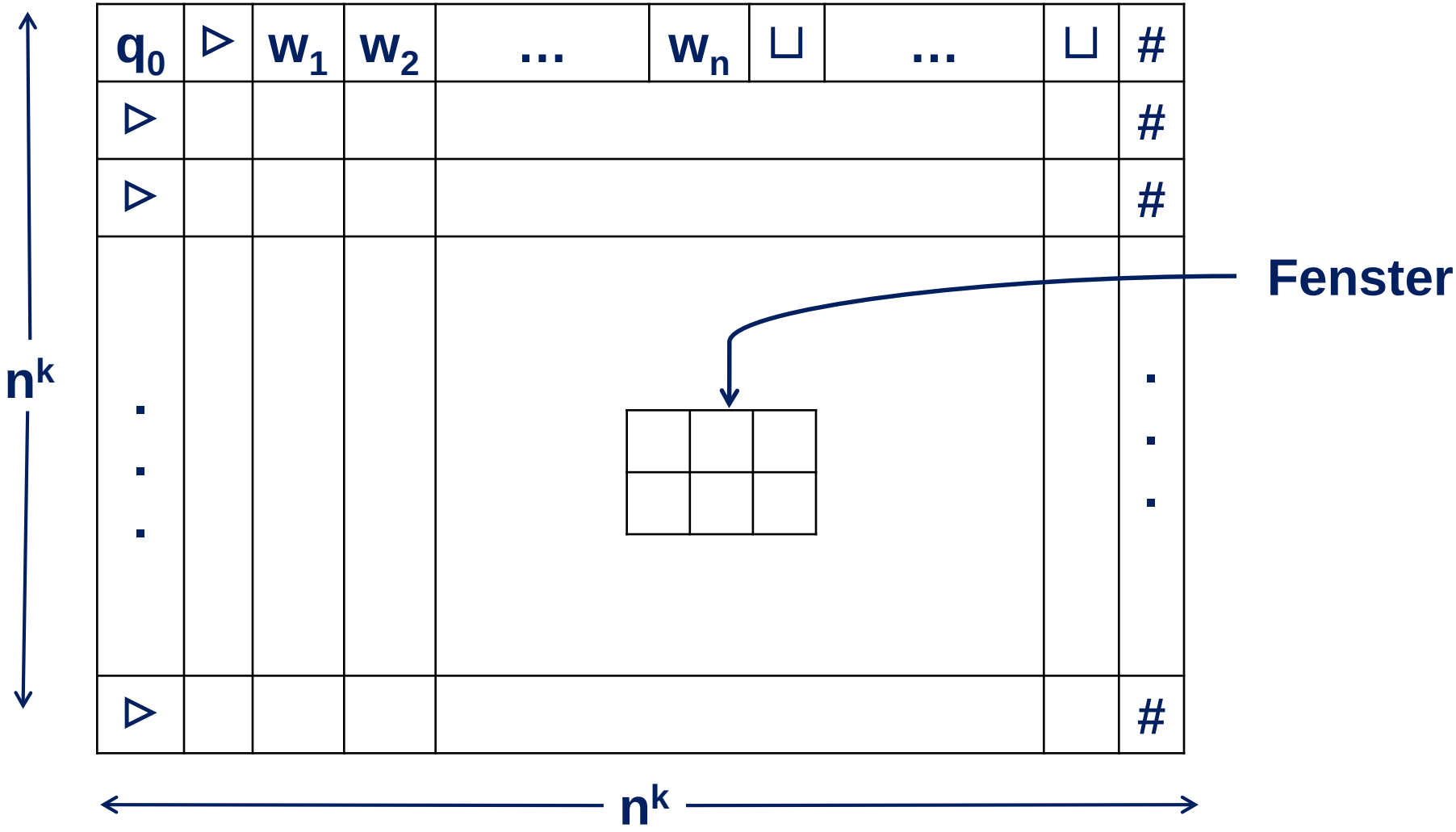
Die Struktur der Formel

$$\phi_{Start} = x_{1,1,q_0} \wedge x_{1,2,\triangleright} \wedge x_{1,3,w_1} \wedge x_{1,4,w_2} \wedge \dots \wedge x_{1,n+2,w_n} \wedge x_{1,n+3,\sqcup} \wedge \dots \wedge x_{1,n^k-1,\sqcup} \wedge x_{1,n^k,\#}$$

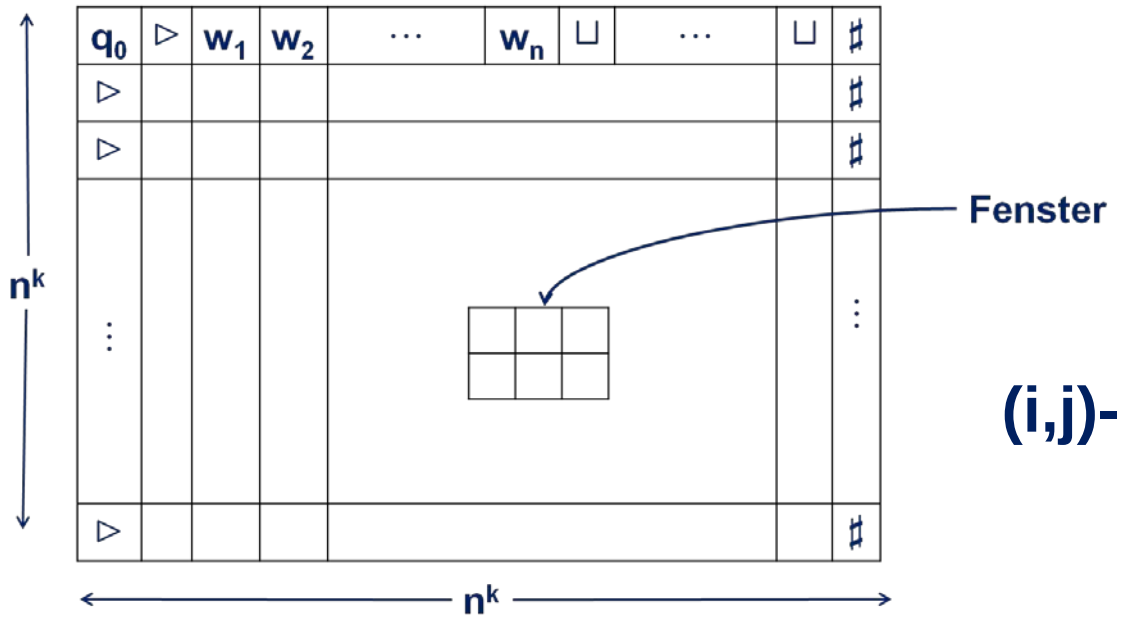
$$\phi_{accept} = \bigvee_{1 \le i,j \le n^k} x_{i,j,q_{accept}}$$

$$\phi_{Eintrag} = \bigwedge_{1 \le i,j \le n^k} \left[\left(\bigvee_{s \in C} x_{i,j,s} \right) \wedge \left(\bigwedge_{s,t \in C,\, s \neq t} (\overline{x}_{i,j,s} \vee \overline{x}_{i,j,t}) \right) \right]$$

Berechnungstabellen



Fenster in Berechnungstabellen

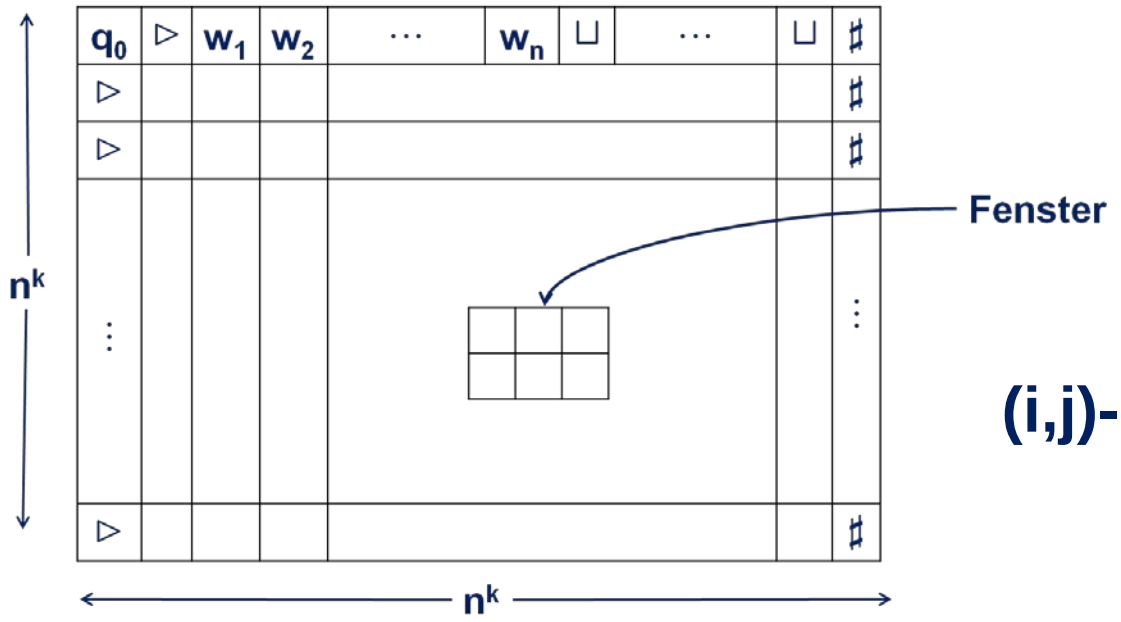


(i,j)-Fenster:

	(i,j)	

- **(i,j)-Fenster für $1 \leq i \leq n^k-1$, $2 \leq j \leq n^k-1$.**
- **Insgesamt $(n^k-1)\cdot(n^k-2)$ Fenster.**

Fenster in Berechnungstabellen



(i,j)-Fenster:

	(i,j)	

a_1	a_2	a_3
a_4	a_5	a_6

- **Fenster mit Belegungen $a_i \in C$ heißt legal, wenn es der Übergangsfunktion von N nicht widerspricht.**
- **Anzahl möglicher Belegungen $|C|^6$.**

Legale Fenster

1. $\delta(q_1, a) = \{(q_1,b,R)\}$
2. $\delta(q_1, b) = \{(q_2,c,L),(q_2,a,R)\}$

(g)

a	q_{accept}	b
a	q_{accept}	b

(a)

a	q_1	b
q_2	a	c

(b)

a	q_1	b
a	a	q_2

(c)

a	a	q_1
a	a	b

(d)

▷	b	a
▷	b	a

(e)

a	b	a
a	b	q_2

(f)

b	b	b
c	b	b

Illegale Fenster

1. $\delta(q_1, a) = \{(q_1,b,R)\}$
2. $\delta(q_1, b) = \{(q_2,c,L),(q_2,a,R)\}$

(a)

a	b	a
a	a	a

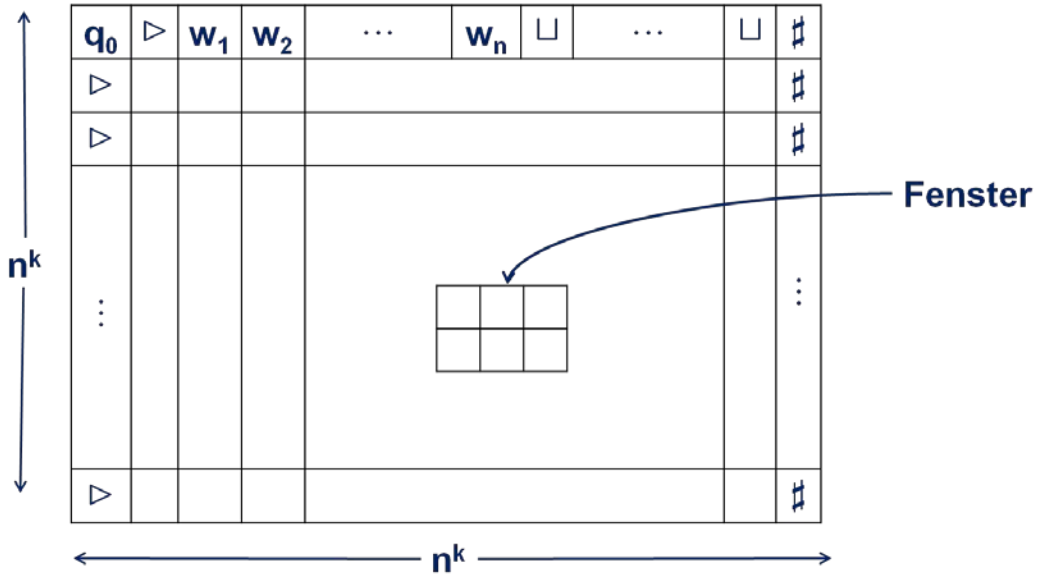
(b)

a	q_1	b
q_1	a	a

(c)

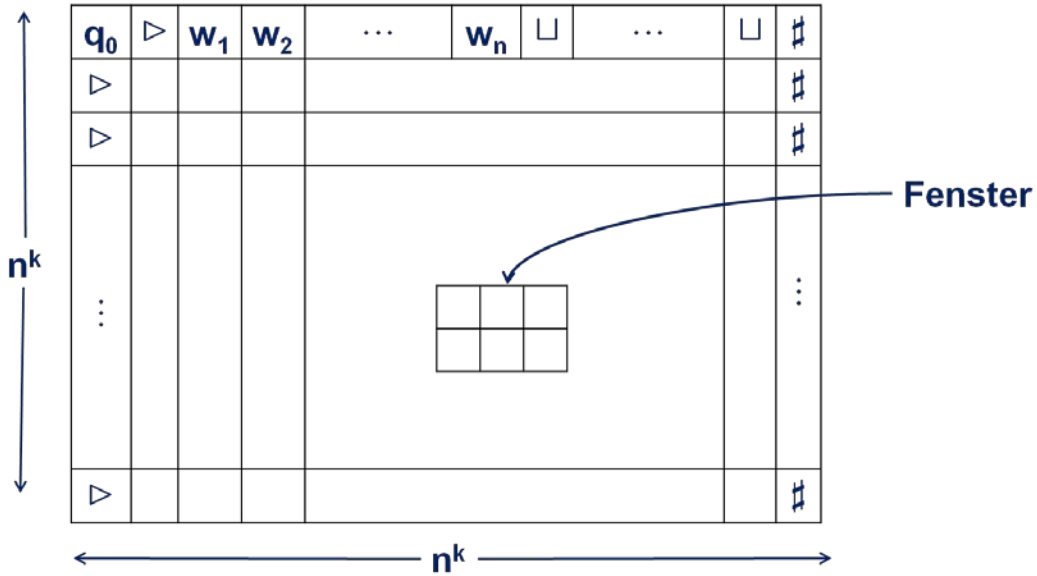
a	q_1	b
q_2	b	q_2

Fenster in Berechnungstabellen



$$\phi_{move} = \bigwedge_{\substack{1 \le i \le n^k - 1 \\ 2 \le j \le n^k - 1}} \text{(Das (i,j)-Fenster ist legal)}$$

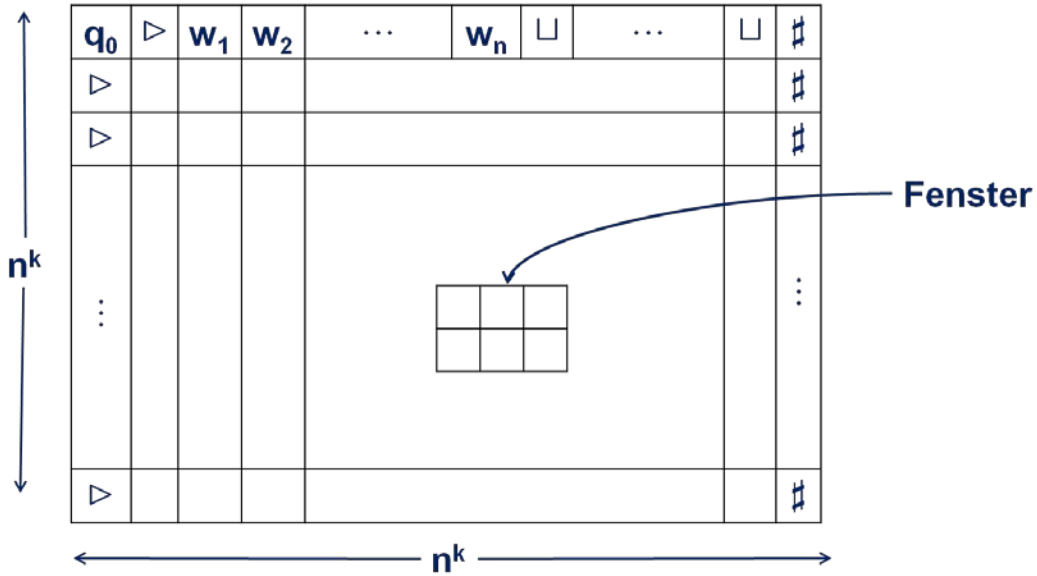
Fenster in Berechnungstabellen



$$\phi_{move} = \bigwedge_{\substack{1 \le i \le n^k - 1 \\ 2 \le j \le n^k - 1}} \text{(Das (i,j)-Fenster ist legal)}$$

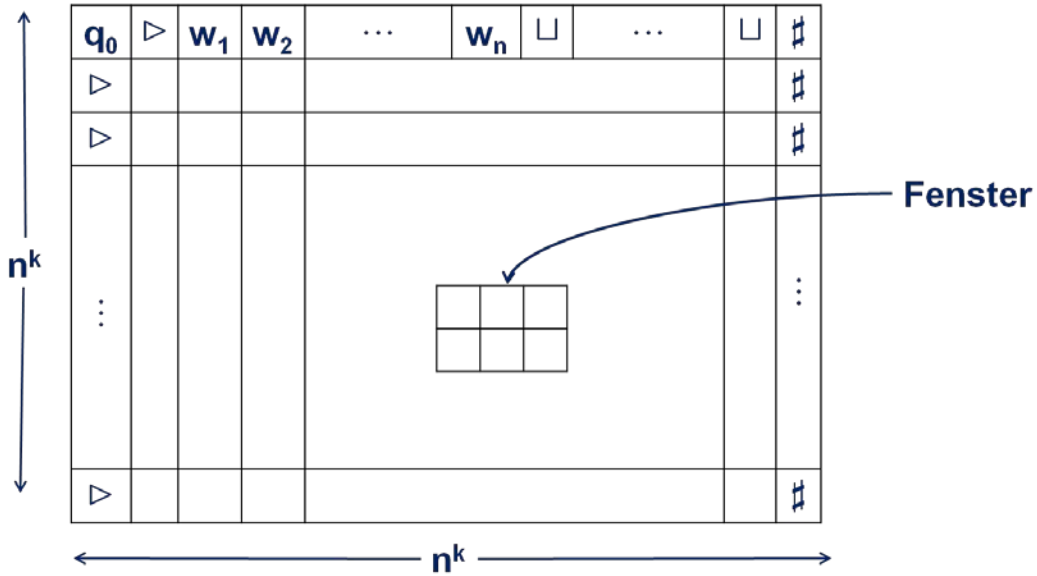
$$\phi_{move} = \bigwedge_{\substack{1 \le i \le n^k - 1 \\ 2 \le j \le n^k - 1}} \bigvee_{\substack{(a_1,\ldots,a_6) \\ \text{Ist legales} \\ \text{Fenster}}} (x_{i,j-1,a_1} \wedge x_{i,j,a_2} \wedge x_{i,j+1,a_3} \wedge x_{i+1,j-1,a_4} \wedge x_{i+1,j,a_5} \wedge x_{i+1,j+1,a_6})$$

Fenster in Berechnungstabellen



Lemma 3.28 **Ist die erste Zeile die Startkonfiguration von N bei Eingabe w und ist jedes Fenster der Tabelle legal, so ist jede Zeile eine Nachfolgekonfiguration der Konfiguration der vorangegangenen Zeile und die Tabelle ist eine Berechnungstabelle von N bei Eingabe w.**

Fenster in Berechnungstabellen



Beweis von Lemma 3.28

- ϕ_{Start} garantiert, dass Zeile 1 eine legale Startkonfiguration sein muss.
- Zeile i legale Konfiguration und alle (i,j)-Fenster legal

  $\Rightarrow$ Zeile i+1 legale Konfiguration

Die Struktur der Formel

$$\phi = \phi_{Start} \wedge \phi_{accept} \wedge \phi_{Eintrag} \wedge \phi_{move}$$

$$\phi_{Start} = x_{1,1,q_0} \wedge x_{1,2,\triangleright} \wedge x_{1,3,w_1} \wedge x_{1,4,w_2} \wedge \dots \wedge x_{1,n+2,w_n} \; x_{1,n+3,\sqcup} \wedge \dots \wedge x_{1,n^k-1,\sqcup} \wedge x_{1,n^k,\#}$$

$$\phi_{accept} = \bigvee_{1 \le i,j \le n^k} x_{i,j,q_{accept}}$$

$$\phi_{Eintrag} = \bigwedge_{1 \le i,j \le n^k} \left[\left(\bigvee_{s \in C} x_{i,j,s} \right) \wedge \left(\bigwedge_{s,t \in C,\, s \neq t} (\overline{x}_{i,j,s} \vee \overline{x}_{i,j,t}) \right) \right]$$

$$\phi_{move} = \bigwedge_{\substack{1 \le i \le n^k-1 \\ 2 \le j \le n^k-1}} \; \bigvee_{\substack{(a_1,\dots,a_6) \\ \text{Ist legales} \\ \text{Fenster}}} (x_{i,j-1,a_1} \wedge x_{i,j,a_2} \wedge x_{i,j+1,a_3} \wedge x_{i+1,j-1,a_4} \wedge x_{i+1,j,a_5} \wedge x_{i+1,j+1,a_6})$$

Größe der Formel

$$\phi = \phi_{Start} \wedge \phi_{accept} \wedge \phi_{Eintrag} \wedge \phi_{move}$$

- ϕ_{Start} : n^k Literale.
- ϕ_{accept} : n^{2k} Literale.
- $\phi_{Eintrag}$: $O(n^{2k})$ Literale.
- ϕ_{move} : $O(n^{2k})$ Literale.

Zusammenfassung – SAT NP-vollständig

- Wir wollten zeigen: für alle $L \in NP$ gilt $L \leq_p SAT$.
- Betrachte **beliebiges** $L \in NP$.
- Dann **existiert** NTM N für L mit polynomieller Laufzeit, also $O(n^k)$ für ein konstantes k, bzw. maximal $c \cdot n^k$ Laufzeit für Konstanten c,k.
- Dann **existiert** eine DTM M_f, die aus jedem $w \in \Sigma^*$ in **polynomieller Zeit** eine Boolesche Formel $\phi(w)$ baut mit (s. Ziel auf Folie 22): $w \in L$ (d.h. es existiert akzeptierende Rechnung von N auf w) $\Leftrightarrow$ $\phi(w)$ erfüllbar (d.h. $\phi(w) \in SAT$)
- Also ist $L \leq_p SAT$.

3SAT ist NP-vollständig

Satz 3.29 **3SAT ist NP-vollständig.**

Ziel **Schreiben** $\phi = \phi_{\text{Start}} \wedge \phi_{\text{accept}} \wedge \phi_{\text{Eintrag}} \wedge \phi_{\text{move}}$ **in 3-KNF.**

Zunächst ϕ in KNF.

Die Struktur der Formel

$$\phi = \phi_{Start} \wedge \phi_{accept} \wedge \phi_{Eintrag} \wedge \phi_{move}$$

$$\phi_{Start} = x_{1,1,q_0} \wedge x_{1,2,\triangleright} \wedge x_{1,3,w_1} \wedge x_{1,4,w_2} \wedge \dots \wedge x_{1,n+2,w_n}$$
$$x_{1,n+3,\sqcup} \wedge \dots \wedge x_{1,n^k-1,\sqcup} \wedge x_{1,n^k,\#}$$

$$\phi_{accept} = \bigvee_{1 \le i,j \le n^k} x_{i,j,q_{accept}}$$

ϕ_{Start}, ϕ_{accept} **sind bereits in KNF.**

Die Struktur der Formel

$$\phi_{\text{Eintrag}} = \bigwedge_{1 \le i,j \le n^k} \left[\left(\bigvee_{s \in C} x_{i,j,s} \right) \wedge \left(\bigwedge_{s,t \in C,\, s \neq t} (\overline{x}_{i,j,s} \vee \overline{x}_{i,j,t}) \right) \right]$$

$$= \bigwedge_{1 \le i,j \le n^k} \left(\bigvee_{s \in C} x_{i,j,s} \right) \wedge \bigwedge_{1 \le i,j \le n^k} \left(\bigwedge_{\substack{s,t \in C, \\ s \neq t}} (\overline{x}_{i,j,s} \vee \overline{x}_{i,j,t}) \right)$$

Damit ϕ_{Eintrag} ebenfalls in KNF.

Die Struktur der Formel

$$\phi_{move} = \bigwedge_{\substack{1\le i\le n^k-1 \\ 2\le j\le n^k-1}} \bigvee_{\substack{(a_1,\ldots,a_6) \\ \text{Ist legales} \\ \text{Fenster}}} (x_{i,j-1,a_1} \wedge x_{i,j,a_2} \wedge x_{i,j+1,a_3} \wedge x_{i+1,j-1,a_4} \wedge x_{i+1,j,a_5} \wedge x_{i+1,j+1,a_6})$$

Größe von

$$\phi_{Fenster} = \bigvee_{\substack{(a_1,\ldots,a_6) \\ \text{Ist legales} \\ \text{Fenster}}} (x_{i,j-1,a_1} \wedge x_{i,j,a_2} \wedge x_{i,j+1,a_3} \wedge x_{i+1,j-1,a_4} \wedge x_{i+1,j,a_5} \wedge x_{i+1,j+1,a_6})$$

hängt nur von N, nicht von w ab, ist also konstant.

Bekannt: für jede Boolesche Formel ϕ gilt es eine äquivalente Boolesche Formel ϕ' in KNF.

Die Struktur der Formel

$$\phi_{move} = \bigwedge_{\substack{1\le i\le n^k-1 \\ 2\le j\le n^k-1}} \bigvee_{\substack{(a_1,\ldots,a_6) \\ \text{Ist legales} \\ \text{Fenster}}} (x_{i,j-1,a_1} \wedge x_{i,j,a_2} \wedge x_{i,j+1,a_3} \wedge x_{i+1,j-1,a_4} \wedge x_{i+1,j,a_5} \wedge x_{i+1,j+1,a_6})$$

Größe von

$$\phi_{Fenster} = \bigvee_{\substack{(a_1,\ldots,a_6) \\ \text{Ist legales} \\ \text{Fenster}}} (x_{i,j-1,a_1} \wedge x_{i,j,a_2} \wedge x_{i,j+1,a_3} \wedge x_{i+1,j-1,a_4} \wedge x_{i+1,j,a_5} \wedge x_{i+1,j+1,a_6})$$

hängt nur von N, nicht von w ab, ist also konstant.

Wir schreiben $\phi_{Fenster}$ und damit ϕ_{move} in KNF.

Größe bleibt polynomiell in $|w|$.

Von KNF zu 3-KNF

Wir ersetzen

$$a_1 \vee a_2 \vee \cdots \vee a_l$$

durch erfüllbarkeitsäquivalente 3-KNF Formel

$$(a_1 \vee a_2 \vee z_1) \wedge (\overline{z}_1 \vee a_3 \vee z_2) \wedge (\overline{z}_2 \vee a_4 \vee z_3) \wedge \cdots$$
$$\cdots \wedge (\overline{z}_{l-4} \vee a_{l-2} \vee z_{l-3}) \wedge (\overline{z}_{l-3} \vee a_{l-1} \vee a_l)$$

z_i neue Variablen, paarweise verschieden.

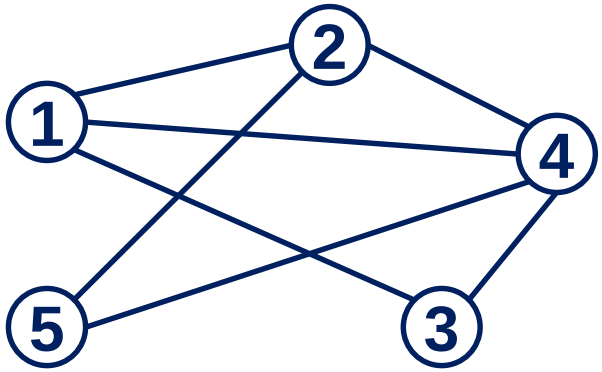
Clique ist NP-vollständig

Satz 3.23 **3SAT ist auf Clique polynomiell reduzierbar, d.h., 3SAT $\leq_p$ Clique.**

Satz 3.30 **Clique ist NP-vollständig.**

Graphen und Knotenüberdeckung

- **G = (V, E) ungerichteter Graph.**
- **U $\subseteq$ V heißt Knotenüberdeckung, wenn für alle e $\in$ E gilt e $\cap$ U $\neq\emptyset$.**
- **U heißt k-Knotenüberdeckung, wenn U Knotenüberdeckung und |U| = k.**

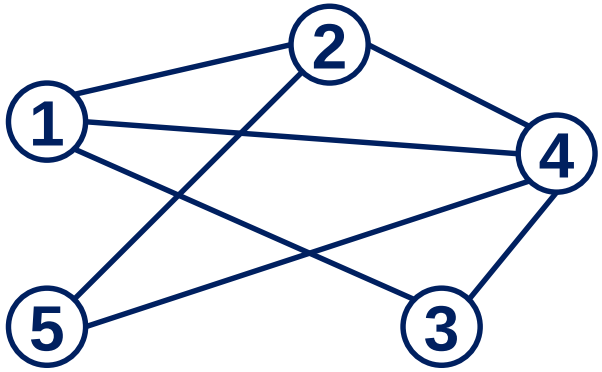


U = {2, 3, 4} ist 3-Knotenüberdeckung.

Die Sprache Knotenüberdeckung

Knotenüberdeckung := $\{\langle G, k\rangle |$ G besitzt k-Knotenüberdeckung$\}$

$G_0 :=$



$\langle G_0, 3\rangle \in$ Knotenüberdeckung

Knotenüberdeckung ist NP-vollständig

Satz 3.31 **Knotenüberdeckung ist NP-vollständig.**

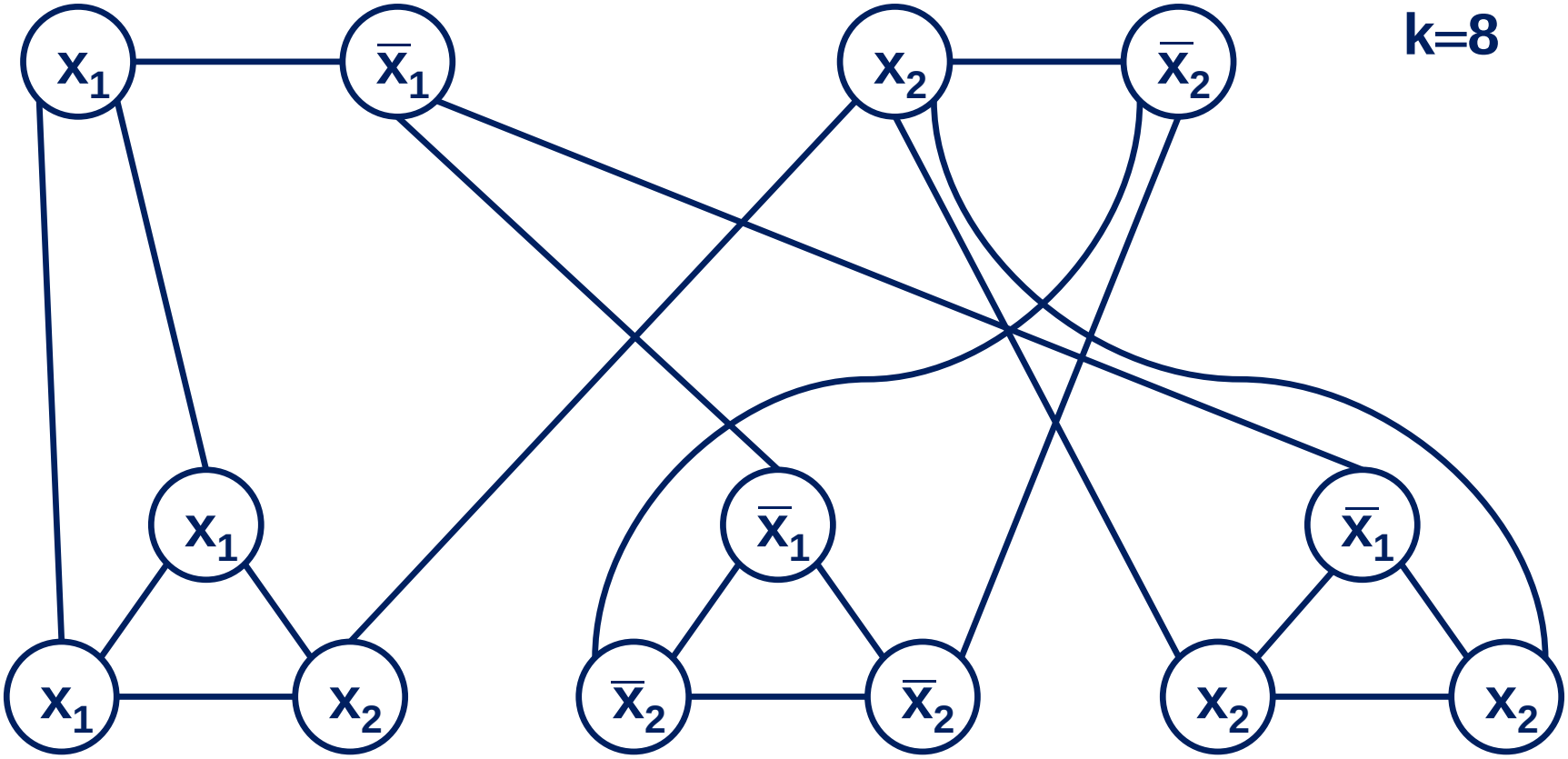
- **Knotenüberdeckung $\in$ NP.** $\surd$
- **3SAT $\leq_p$ Knotenüberdeckung.**

Reduktion 3SAT auf Knotenüberdeckung

$$f = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_2) \wedge (\overline{x}_1 \vee x_2 \vee x_2)$$

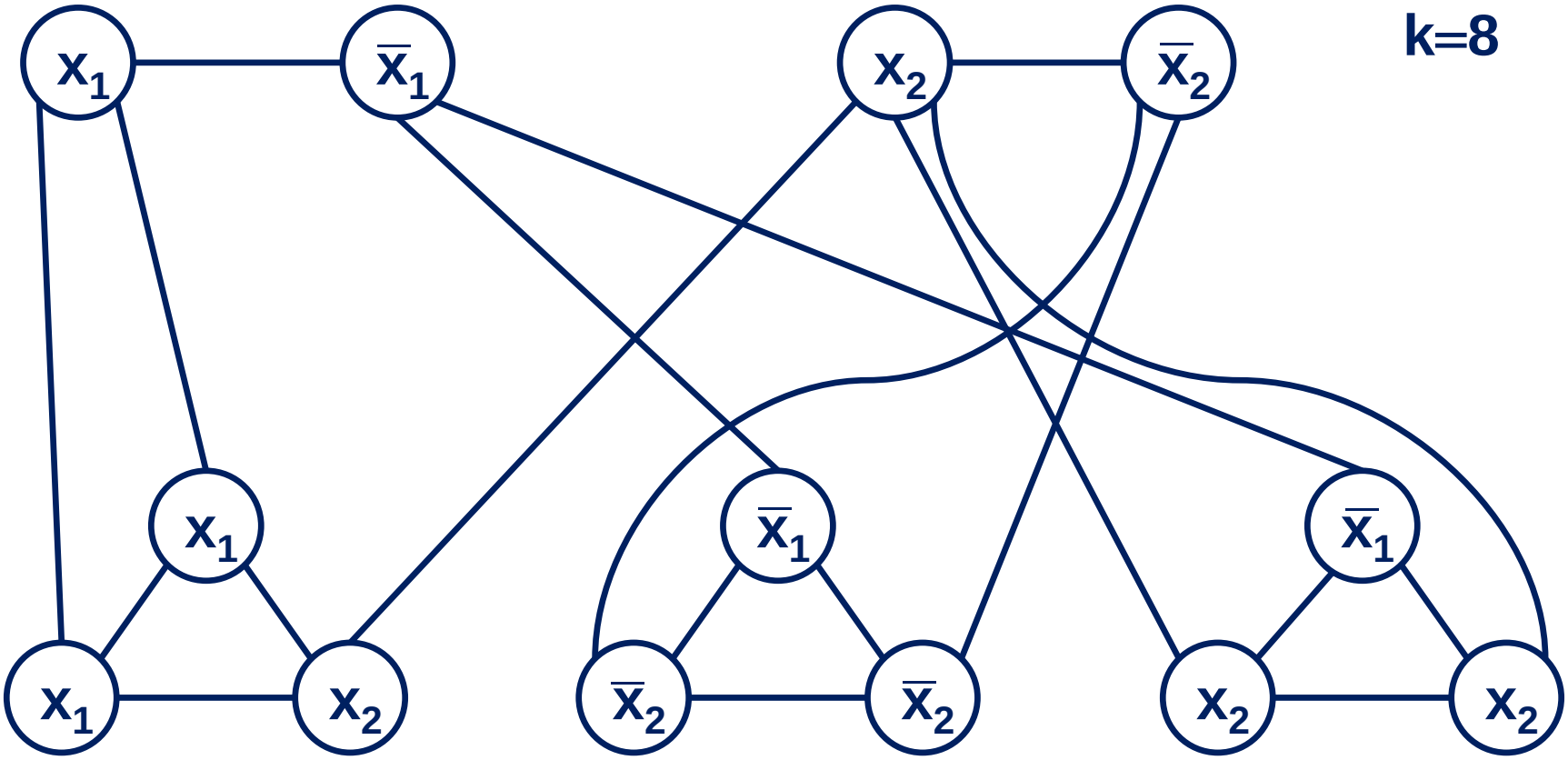
Reduktion 3SAT auf Knotenüberdeckung

$$f = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_2) \wedge (\overline{x}_1 \vee x_2 \vee x_2)$$



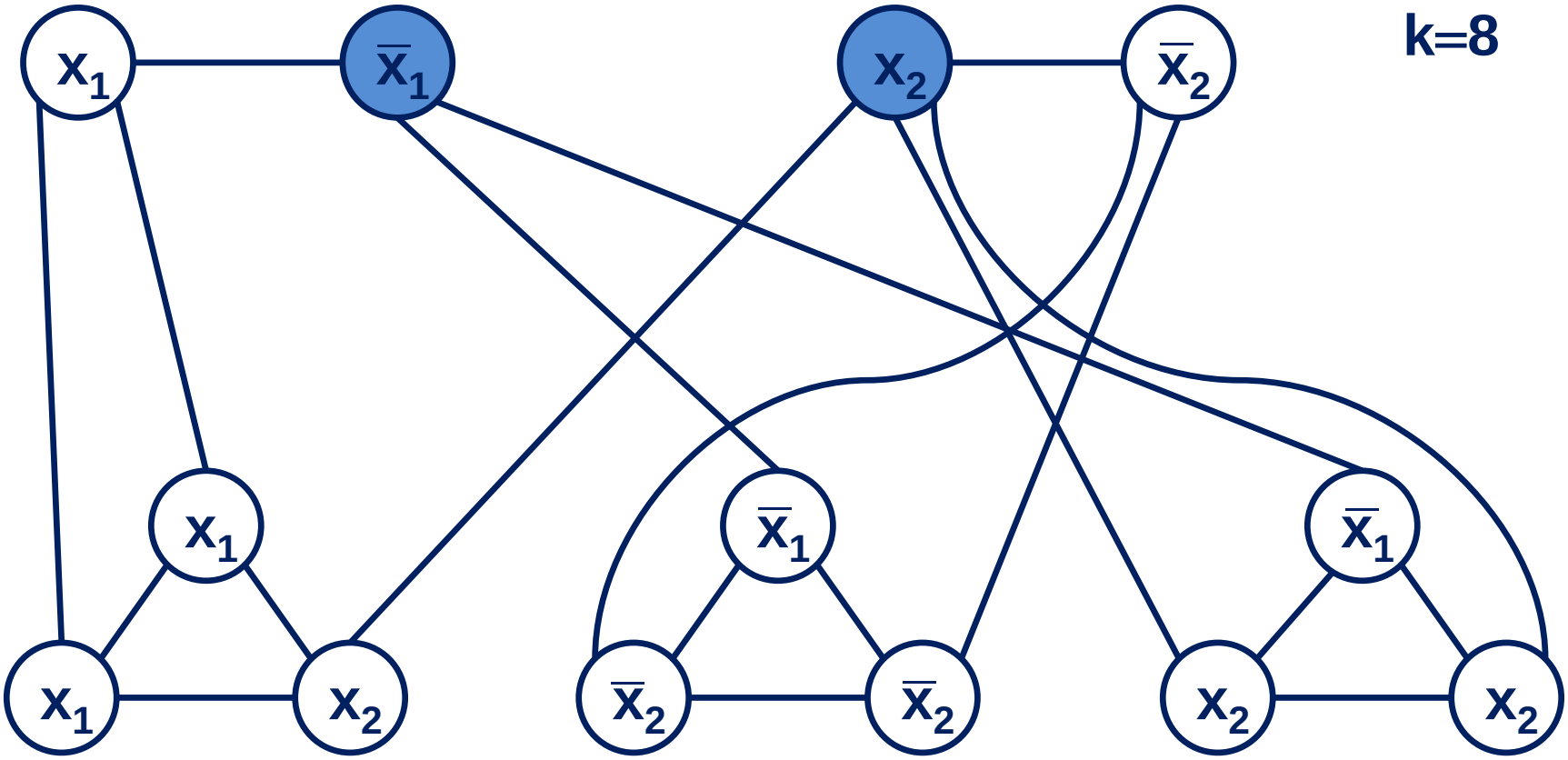
Reduktion 3SAT auf Knotenüberdeckung

$$f = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_2) \wedge (\overline{x}_1 \vee x_2 \vee x_2)$$



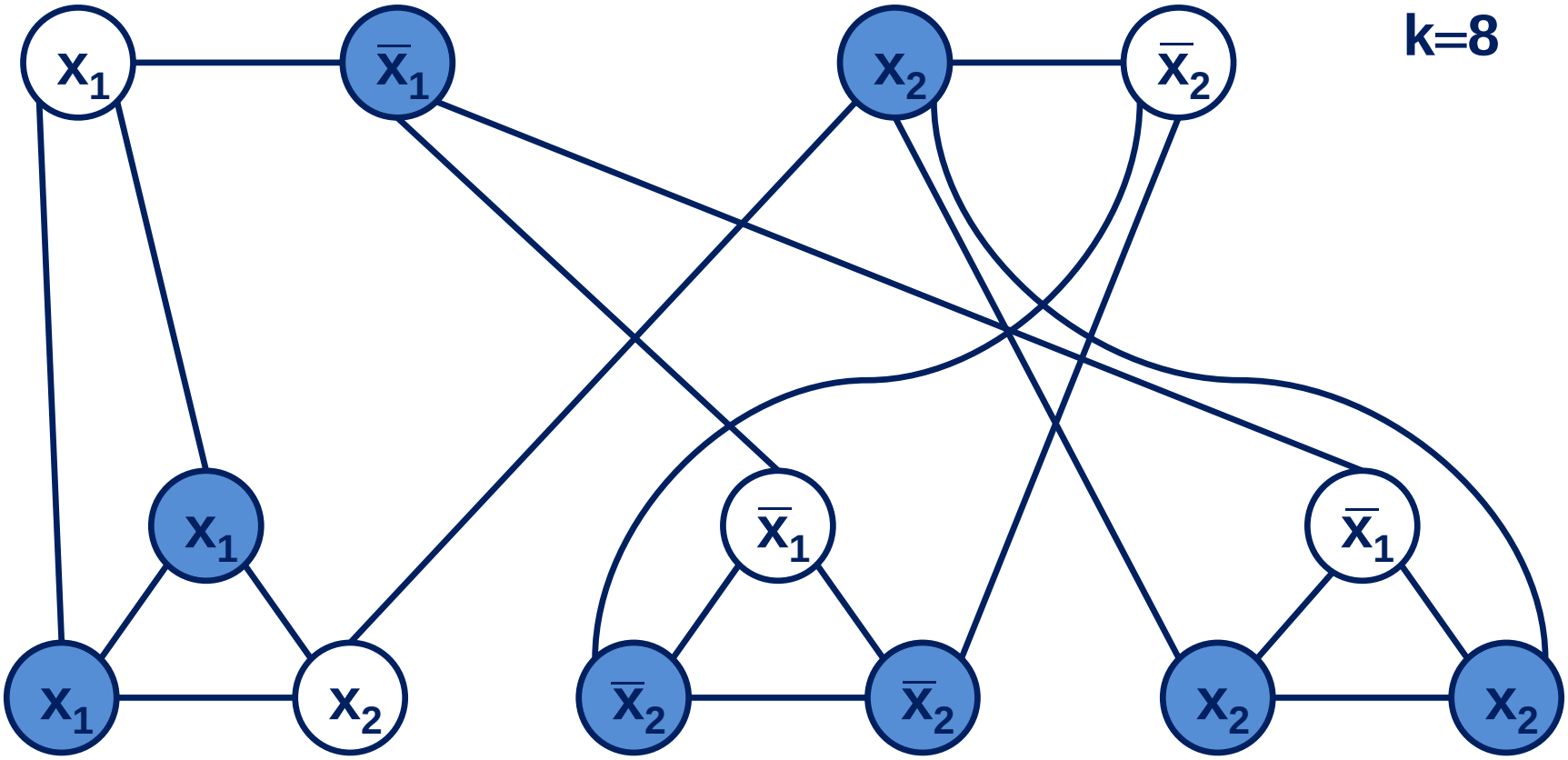
Reduktion 3SAT auf Knotenüberdeckung

$$f = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_2) \wedge (\overline{x}_1 \vee x_2 \vee x_2)$$



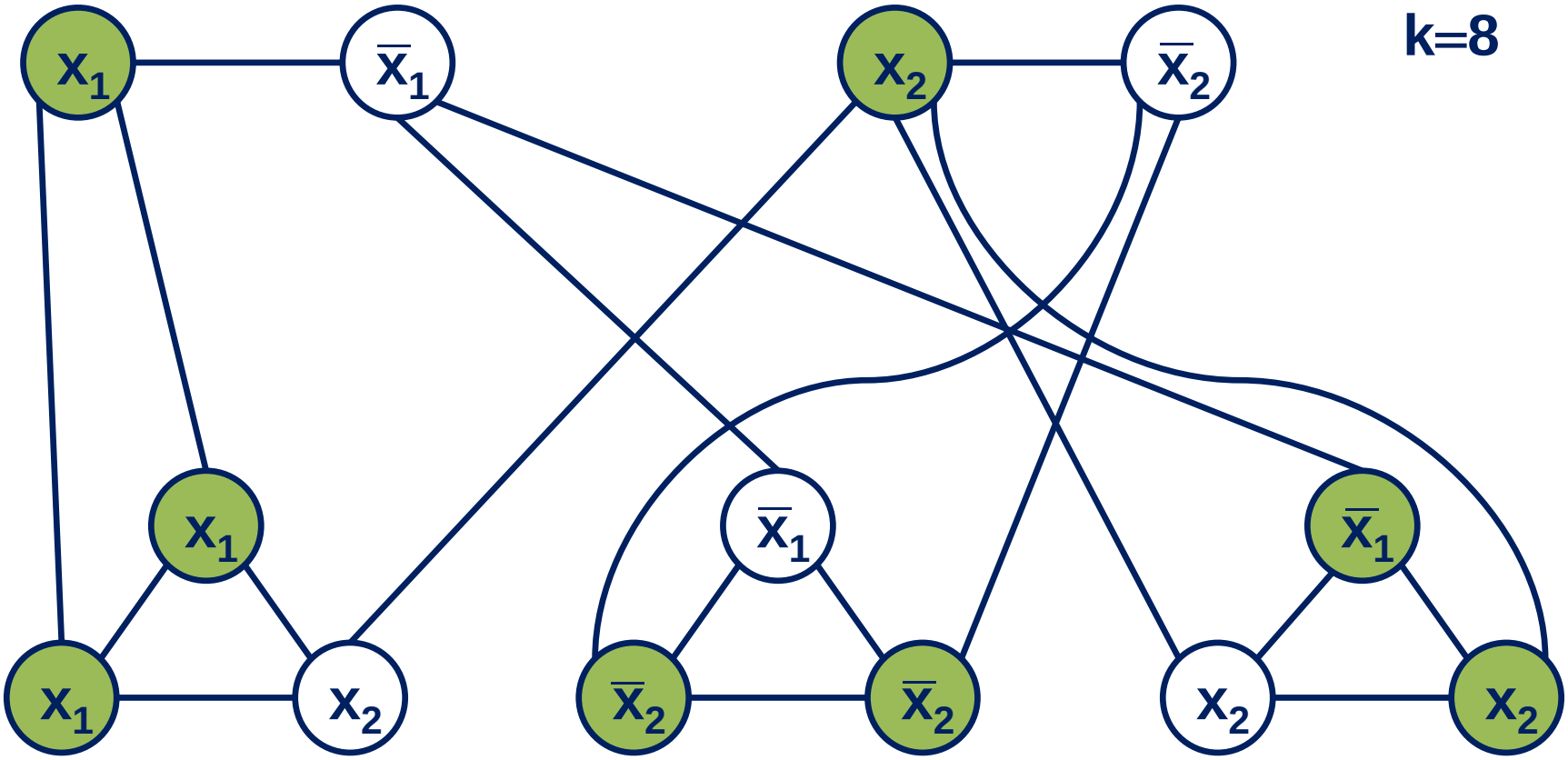
Reduktion 3SAT auf Knotenüberdeckung

$$f = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_2) \wedge (\overline{x}_1 \vee x_2 \vee x_2)$$



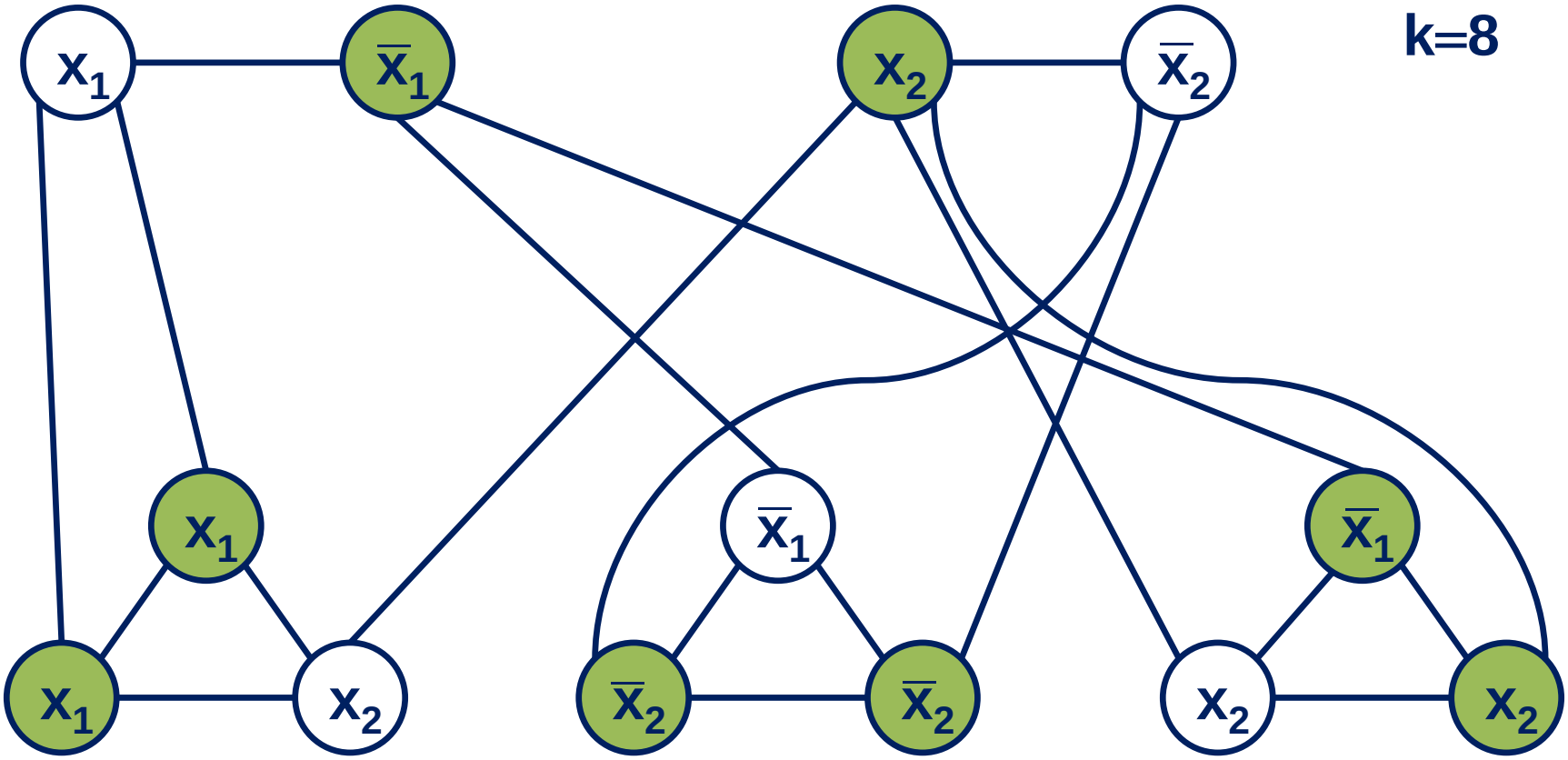
Reduktion 3SAT auf Knotenüberdeckung

$$f = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_2) \wedge (\overline{x}_1 \vee x_2 \vee x_2)$$



Reduktion 3SAT auf Knotenüberdeckung

$$f = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_2) \wedge (\overline{x}_1 \vee x_2 \vee x_2)$$



Rucksackproblem und SubsetSum

$$RS_{ent} := \left\{ \langle G, W, g, w \rangle \;\middle|\; \begin{array}{l} G = \{g_1, \ldots, g_n\}, W = \{w_1, \ldots, w_n\} \text{ und es} \\ \text{existiert ein } S \subseteq \{1, \ldots, n\} \text{ mit } \sum_{i \in S} g_i \leq g \\ \text{und } \sum_{i \in S} w_i \geq w. \end{array} \right\}$$

Rucksackproblem und SubsetSum

$$RS_{ent} := \left\{ \langle G, W, g, w \rangle \;\middle|\; \begin{array}{l} G = \{g_1, \ldots, g_n\}, W = \{w_1, \ldots, w_n\} \text{ und es} \\ \text{existiert ein } S \subseteq \{1, \ldots, n\} \text{ mit } \sum_{i \in S} g_i \leq g \\ \text{und } \sum_{i \in S} w_i \geq w. \end{array} \right\}$$

$$SubsetSum := \left\{ \langle S, t \rangle \;\middle|\; \begin{array}{l} S = \{s_1, \ldots, s_n\} \subset \mathbf{N},\ t \in \mathbf{N} \text{ und es existiert} \\ \text{ein } T \subseteq \{1, \ldots, n\} \text{ mit } \sum_{i \in T} s_i = t. \end{array} \right\}$$

Rucksackproblem und SubsetSum

Lemma 3.32 **SubsetSum ist polynomiell reduzierbar auf RS_{ent}.**

Satz 3.33 **SubsetSum ist NP-vollständig.**

Daher RS_{ent} ist NP-vollständig.

Reduktion 3SAT auf SubsetSum

$$f = (x_1 \vee \overline{x}_2 \vee x_3) \wedge (\overline{x}_1 \vee x_2 \vee \overline{x}_3)$$

	x_1	x_2	x_3	C_1	C_2
y_1	1	0	0	1	0
z_1	1	0	0	0	1
y_2	0	1	0	0	1
z_2	0	1	0	1	0
y_3	0	0	1	1	0
z_3	0	0	1	0	1
g_1	0	0	0	1	0
h_1	0	0	0	1	0
g_2	0	0	0	0	1
h_2	0	0	0	0	1
t	1	1	1	3	3

Reduktion 3SAT auf SubsetSum

$$f = (x_1 \vee \overline{x}_2 \vee x_3) \wedge (\overline{x}_1 \vee x_2 \vee \overline{x}_3)$$

	x_1	x_2	x_3	C_1	C_2
y_1	1	0	0	1	0
z_1	1	0	0	0	1
y_2	0	1	0	0	1
z_2	0	1	0	1	0
y_3	0	0	1	1	0
z_3	0	0	1	0	1
g_1	0	0	0	1	0
h_1	0	0	0	1	0
g_2	0	0	0	0	1
h_2	0	0	0	0	1
t	1	1	1	3	3

Reduktion 3SAT auf SubsetSum

$$f = (x_1 \vee \overline{x}_2 \vee x_3) \wedge (\overline{x}_1 \vee x_2 \vee \overline{x}_3)$$

	x_1	x_2	x_3	C_1	C_2
y_1	1	0	0	1	0
z_1	1	0	0	0	1
y_2	0	1	0	0	1
z_2	0	1	0	1	0
y_3	0	0	1	1	0
z_3	0	0	1	0	1
g_1	0	0	0	1	0
h_1	0	0	0	1	0
g_2	0	0	0	0	1
h_2	0	0	0	0	1
t	1	1	1	3	3

Reduktion 3SAT auf SubsetSum

$$f = (x_1 \vee \overline{x}_2 \vee x_3) \wedge (\overline{x}_1 \vee x_2 \vee \overline{x}_3)$$

	x_1	x_2	x_3	C_1	C_2
y_1	1	0	0	1	0
z_1	1	0	0	0	1
y_2	0	1	0	0	1
z_2	0	1	0	1	0
y_3	0	0	1	1	0
z_3	0	0	1	0	1
g_1	0	0	0	1	0
h_1	0	0	0	1	0
g_2	0	0	0	0	1
h_2	0	0	0	0	1
t	1	1	1	3	3

Reduktion 3SAT auf SubsetSum

$$f = (x_1 \vee \overline{x}_2 \vee x_3) \wedge (\overline{x}_1 \vee x_2 \vee \overline{x}_3)$$

	x_1	x_2	x_3	C_1	C_2
y_1	1	0	0	1	0
z_1	1	0	0	0	1
y_2	0	1	0	0	1
z_2	0	1	0	1	0
y_3	0	0	1	1	0
z_3	0	0	1	0	1
g_1	0	0	0	1	0
h_1	0	0	0	1	0
g_2	0	0	0	0	1
h_2	0	0	0	0	1
t	1	1	1	3	3

Reduktion 3SAT auf SubsetSum

$$f = (x_1 \vee \overline{x}_2 \vee x_3) \wedge (\overline{x}_1 \vee x_2 \vee \overline{x}_3)$$

	x_1	x_2	x_3	C_1	C_2
y_1	1	0	0	1	0
z_1	1	0	0	0	1
y_2	0	1	0	0	1
z_2	0	1	0	1	0
y_3	0	0	1	1	0
z_3	0	0	1	0	1
g_1	0	0	0	1	0
h_1	0	0	0	1	0
g_2	0	0	0	0	1
h_2	0	0	0	0	1
t	1	1	1	3	3

Das Problem des Handlungsreisenden

Satz 3.34 **TSP_{ent} ist NP-vollständig.**

Polynomielle Reduktion vom Hamiltonkreis Problem (wird in den Übungen behandelt), für welches eine (recht komplexe) polynomielle Reduktion von SAT existiert.