Grundbegriffe der Theoretischen Informatik

Sommersemester 2018 - Thomas Schwentick

Teil D: Komplexitätstheorie

19: NP-vollständige Probleme

Version von: 10. Juli 2018 (13:01)

Ein erstes NP-vollständiges Problem

- In Teil C der Vorlesung haben wir für die Sprache TM-DIAG die Unentscheidbarkeit durch Diagonalisierung bewiesen und dann alle weiteren unentscheidbaren Probleme durch (direkte oder indirekte) Reduktionen von TM-DIAG gewonnen
- Hinsichtlich **NP**-vollständiger Probleme gehen wir ähnlich vor

- Wir zeigen als nächstes, dass das Problem SAT **NP**-vollständig ist ☞ Satz von Cook
- Für weitere Probleme weisen wir die **NP**-Vollständigkeit dann durch (direkte oder indirekte) polynomielle Reduktionen von SAT nach

Inhalt

Satz von Cook (1/14)

Satz 19.1 [Cook 71]

SAT ist **NP**-vollständig

Beweisskizze

- SAT $\in$ **NP**: ✓

- SAT ist **NP**-schwierig:
  - Wir zeigen, dass für jedes $L \in$ **NP** gilt: $L \leqslant_p$ SAT

- Sei L dazu eine beliebige Sprache aus **NP**
- Sei $M = (Q, \Gamma, \delta, q_1)$ eine (1-String)-TM, die L mit polynomieller Zeitschranke n^k nichtdeterministisch entscheidet
- Sei w eine Eingabe für M und n die Länge von w

Beweisskizze (Forts.)

- Wir zeigen, wie aus w eine KNF-Formel φ konstruiert werden kann, so dass gilt:
  M akzeptiert w nichtdeterministisch
  $\iff \varphi$ ist erfüllbar

- Genauer: wir zeigen, dass es zu jeder Zusatzeingabe y, für die w von M akzeptiert wird, eine erfüllende Belegung α für φ gibt, und umgekehrt

- **Wichtige Idee:**
  - Wir betrachten die Berechnung von M bei Eingabe w als eine *Berechnungstabelle*
    - Die Variablen von φ entsprechen den einzelnen Einträgen der Tabelle
    - Jede Variablenbelegung α der Variablen von φ entspricht einer „ausgefüllten“ Tabelle
  - φ soll genau dann wahr werden, wenn α eine akzeptierende Berechnung repräsentiert

Satz von Cook (2/14): Beispiel-TM

Beispiel

- Wie betrachten als Beispiel eine TM M, die die Sprache L aller Strings über $\{a, b\}$, die *kein Palindrom* sind, akzeptiert

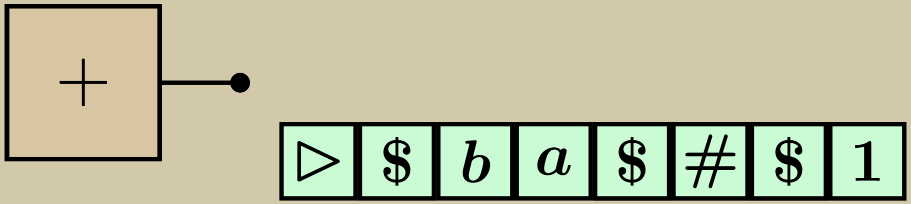
- M erwartet Eingaben der Art $w\#y$
  - w ist der zu überprüfende String ☞ $n \stackrel{\text{def}}{=} |w|$
  - y ist eine Zusatzeingabe der Art 0^k1
- M akzeptiert genau dann, wenn die $(k+1)$-te Position von w von der $(n-k)$-ten Position verschieden ist

Bemerkung

- L ist in **NP**, aber natürlich auch in **P**
- Wir verwenden ein so einfaches Beispiel, weil es sich detailliert auf einer Folie unterbringen lässt

Satz von Cook (3/14): Beispiel-Berechnung

Beispiel

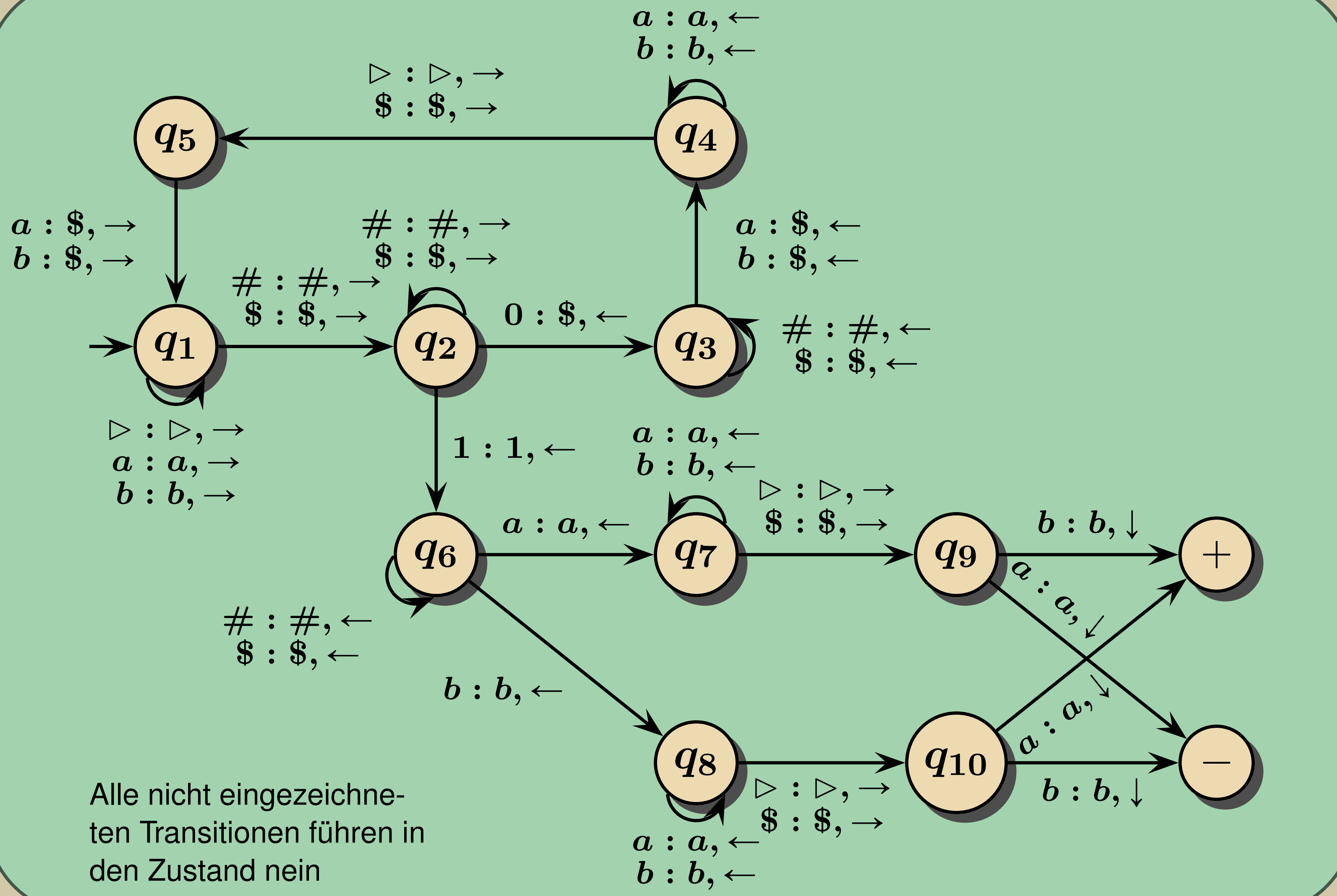


- M erwartet Eingaben der Art $w\#y$

- M bewegt den Kopf zuerst nach rechts auf das erste Zeichen von y
- Falls dort eine 0 steht, wird sie durch $\$$ überschrieben und M läuft nach links und überschreibt dabei das letzte und dann das erste Zeichen von w mit $\$$
- Dann läuft M wieder nach rechts zum nächsten Zeichen von y
- Wenn dort eine 0 steht, werden wieder das letzte und erste noch nicht veränderte Zeichen von w mit $\$$ überschrieben
- Wenn dort eine 1 steht, werden das letzte und erste noch nicht veränderte Zeichen von w verglichen
- Sind diese beiden Zeichen verschieden, so akzeptiert M, andernfalls lehnt M ab

Satz von Cook (4/14): Beispiel-TM als Diagramm

Beispiel



Alle nicht eingezeichneten Transitionen führen in den Zustand nein

Satz von Cook (5/14): Beispiel einer Berechnungstabelle

Beispiel: Tabelle für TM-Beispiel

t	B	Z	P
0	$\triangleright abaa\#01$	q_1	0
1	$\triangleright abaa\#01$	q_1	1
2	$\triangleright abaa\#01$	q_1	2
3	$\triangleright abaa\#01$	q_1	3
4	$\triangleright abaa\#01$	q_1	4
5	$\triangleright abaa\#01$	q_1	5
6	$\triangleright abaa\#01$	q_2	6
7	$\triangleright abaa\#\$1$	q_3	5
8	$\triangleright abaa\#\$1$	q_3	4
9	$\triangleright aba\$\#\$1$	q_4	3
10	$\triangleright aba\$\#\$1$	q_4	2
11	$\triangleright aba\$\#\$1$	q_4	1
12	$\triangleright aba\$\#\$1$	q_4	0
13	$\triangleright aba\$\#\$1$	q_5	1

Beispiel (Forts.)

t	B	Z	P
14	$\triangleright \$ba\$\#\$1$	q_1	2
15	$\triangleright \$ba\$\#\$1$	q_1	3
16	$\triangleright \$ba\$\#\$1$	q_1	4
17	$\triangleright \$ba\$\#\$1$	q_2	5
18	$\triangleright \$ba\$\#\$1$	q_2	6
19	$\triangleright \$ba\$\#\$1$	q_2	7
20	$\triangleright \$ba\$\#\$1$	q_6	6
21	$\triangleright \$ba\$\#\$1$	q_6	5
22	$\triangleright \$ba\$\#\$1$	q_6	4
23	$\triangleright \$ba\$\#\$1$	q_6	3
24	$\triangleright \$ba\$\#\$1$	q_7	2
25	$\triangleright \$ba\$\#\$1$	q_7	1
26	$\triangleright \$ba\$\#\$1$	q_9	2
27	$\triangleright \$ba\$\#\$1$	ja	2

Satz von Cook (6/14): Positionen in der Berechnungstabelle

Beweisskizze (Forts.)

- Zur Vereinfachung nehmen wir oBdA an, dass die Zusatzeingabe genau die Länge $n^k - n - 1$ hat und nur die Zeichen 0 und 1 verwendet
- Es gilt also:
  - $w\#y$ hat genau n^k Zeichen
  - Linker Rand: Position 0
  - w: Positionen $1, \ldots, n$
  - $\#$ an Position $n + 1$
  - y: Positionen $n + 2, \ldots, n^k$

- Klar: in n^k Schritten kann sich der Kopf der Turing-Maschine nicht über y hinaus bewegen

- Sei $\Gamma = \{\sigma_1, \ldots, \sigma_l\}$ das Arbeitsalphabet von M
- Sei $Q = \{q_1, \ldots, q_m\}$ die Zustandsmenge von M inklusive ja, nein und h

Satz von Cook (7/14): Variablen von φ

Beweisskizze (Forts.)

- Die Formel φ verwendet die folgenden aussagenlogischen Variablen:

Variablen	Indizes	Intendierte Bedeutung
$Z_{t,q}$	$t = 0, \ldots, n^k$ $q \in Q$	$\alpha(Z_{t,q}) = 1 \iff$ nach t Schritten befindet sich M im Zustand q
$P_{t,i}$	$t = 0, \ldots, n^k$ $i = 0, \ldots, n^k$	$\alpha(P_{t,i}) = 1 \iff$ nach t Schritten befindet sich der Kopf von M auf Position i
$B_{t,i,\sigma}$	$t = 0, \ldots, n^k$ $i = 0, \ldots, n^k$ $\sigma \in \Gamma$	$\alpha(B_{t,i,\sigma}) = 1$ $\iff$ nach t Schritten befindet sich auf Position i das Zeichen σ

Satz von Cook (8/14): Belegung der Variablen

Beispiel

- Wir werfen einen Blick auf die Variablenbelegung α, die der Berechnungstabelle der Beispielberechnung entspricht
- Wir betrachten nur die Variablen, die die Konfiguration der TM nach 10 Schritten repräsentieren:
  - $t = 10$, $Z = q_4$, $P = 2$
  - $B = \triangleright aba\$\#\$1$

Bemerkung

✎ Im Gegensatz zur Konstruktion im Beweis ist die Länge der Eingabe im Beispiel nicht gleich der Anzahl der Berechnungsschritte

- Deshalb werden hier, der allgemeinen Semantik von TMs entsprechend, hinter der Eingabe Blanks ergänzt

Beispiel (Forts.)

- $\alpha(Z_{10,q_4}) = 1$
- $\alpha(Z_{10,p}) = 0$, für $p \neq q_4$

- $\alpha(P_{10,2}) = 1$
- $\alpha(P_{10,i}) = 0$, für $i \neq 2$

- $\alpha(B_{10,0,\triangleright}) = 1$
- $\alpha(B_{10,1,a}) = 1$
- $\alpha(B_{10,2,b}) = 1$
- $\alpha(B_{10,3,a}) = 1$
- $\alpha(B_{10,4,\$}) = 1$
- $\alpha(B_{10,5,\#}) = 1$
- $\alpha(B_{10,6,\$}) = 1$
- $\alpha(B_{10,7,1}) = 1$
- $\alpha(B_{10,i,\sqcup}) = 1$, für alle $i > 7$
- $\alpha(B_{10,i,\sigma}) = 0$, für alle übrigen i, σ

Satz von Cook (9/14): Konsistenzbedingungen

Beweisskizze (Forts.)

- φ ist aus mehreren Teilformeln zusammengesetzt:

$$\varphi = \varphi_K \wedge \varphi_A \wedge \varphi_D \wedge \varphi_E$$

- φ_K soll sicherstellen, dass α überhaupt eine Tabelle repräsentiert, also jeder Eintrag der Tabelle genau einmal vorhanden ist

☞ Konsistenzbedingungen

- φ_A drückt aus, dass die erste Zeile der Tabelle der Startkonfiguration entspricht

☞ Anfangsbedingungen

- φ_D drückt aus, dass aufeinander folgende Zeilen der Tabelle mit der Transitionsfunktion verträglich sind

☞ Transitionsbedingungen

- φ_E drückt aus, dass die Berechnung akzeptiert

☞ Endbedingung

Satz von Cook (10/14): Konsistenzbedingungen

Beweisskizze (Forts.)

- φ_K drückt für die von α kodierte Tabelle aus:
  - Zu jedem Zeitpunkt ist der Zustand von M eindeutig
  - Zu jedem Zeitpunkt ist die Position des Kopfes von M eindeutig bestimmt
  - Zu jedem Zeitpunkt steht an jeder Stringposition genau ein Zeichen

Beweisskizze (Forts.)

- Wir verwenden dabei die Hilfsformel
$$\psi_{\text{unique}}(x_1,\ldots,x_s) \overset{\text{def}}{=} \Big(\bigvee_{i=1}^{s} x_i\Big) \wedge \Big(\bigwedge_{i\neq j}(\neg x_i \vee \neg x_j)\Big),$$
die wahr wird, wenn $\alpha(x_i) = 1$, für genau ein i

- $$\varphi_K \overset{\text{def}}{=} \bigwedge_t \psi_{\text{unique}}(Z_{t,q_1},\ldots,Z_{t,q_m}) \wedge \bigwedge_t \psi_{\text{unique}}(P_{t,0},\ldots,P_{t,n^k}) \wedge \bigwedge_{t,i} \psi_{\text{unique}}(B_{t,i,\sigma_1},\ldots,B_{t,i,\sigma_l})$$

Satz von Cook (11/14): Anfangsbedingungen

Beweisskizze (Forts.)

- φ_A beschreibt die Situation von M zum Zeitpunkt 0:

- $\varphi_A \stackrel{\text{def}}{=}$

$$Z_{0,q_1} \wedge P_{0,0} \wedge B_{0,0,\triangleright} \wedge \bigwedge_{i=1}^{n} B_{0,i,w[i]}$$
$$\wedge B_{0,n+1,\#} \wedge \bigwedge_{i=n+2}^{n^k} (B_{0,i,0} \vee B_{0,i,1})$$

- Zu beachten:
  - Die Formel drückt unter anderem aus, dass die Zusatzeingabe nur aus Nullen und Einsen besteht
  - Die Formel legt die Eingabe w fest
    - ∗ Dies ist die einzige Teilformel von φ, die wirklich von w abhängt
    - ∗ Die anderen Teilformeln hängen allenfalls von der Länge n von w ab
  - Die Formel legt *nicht* die Zusatzeingabe y fest

Satz von Cook (12/14): Transitionsbedingungen

Beweisskizze (Forts.)

- φ_D beschreibt die Beziehung zwischen den aufeinander folgenden Konfigurationen

- $\varphi_D \overset{\text{def}}{=} \varphi_{D_1} \wedge \varphi_{D_2}$, wobei:
  - φ_{D_1} beschreibt, was sich an der Stelle, an der sich der Kopf der Turing-Maschine befindet, ändert

- $\varphi_{D_1} \overset{\text{def}}{=}$

$$\bigwedge_{t,i,p,\sigma} [(Z_{t,p} \wedge P_{t,i} \wedge B_{t,i,\sigma}) \rightarrow (Z_{t+1,q} \wedge P_{t+1,i+d} \wedge B_{t+1,i,\tau})]$$

- Dabei sind q, τ, d jeweils durch $\delta(p,\sigma) = (q,\tau,d)$ gegeben
  - d wird hier als Zahl in $\{-1, 0, 1\}$ interpretiert ($\leftarrow \equiv -1, \downarrow \equiv 0, \rightarrow \equiv 1$)

✎ φ_{D_1} ist die einzige Teilformel von φ, die von der Transitionsfunktion δ abhängt

Beispiel

- Die Formel φ_{D_1} ist zu lang, um sie für das Beispiel ganz anzugeben
- Deshalb hier nur ein kleiner Ausschnitt für $t = 8$, $i = 3$, $p = q_7$
- Es gilt in der TM
  - $\delta(q_7, a) = (q_7, a, \leftarrow)$
  - $\delta(q_7, b) = (q_7, b, \leftarrow)$
  - $\delta(q_7, \$) = (q_9, \$, \rightarrow)$

- Die entsprechende Teilformel lautet dann:

$$[(Z_{8,q_7} \wedge P_{8,3} \wedge B_{8,3,a}) \rightarrow (Z_{9,q_7} \wedge P_{9,2} \wedge B_{9,3,a})] \wedge$$
$$[(Z_{8,q_7} \wedge P_{8,3} \wedge B_{8,3,b}) \rightarrow (Z_{9,q_7} \wedge P_{9,2} \wedge B_{9,3,b})] \wedge$$
$$[(Z_{8,q_7} \wedge P_{8,3} \wedge B_{8,3,\$}) \rightarrow (Z_{9,q_9} \wedge P_{9,4} \wedge B_{9,3,\$})]$$

Satz von Cook (13/14): Transitionsbedingungen (Forts.)

Beweisskizze (Forts.)

- Die Teilformeln
$$[(Z_{t,p} \wedge P_{t,i} \wedge B_{t,i,\sigma}) \rightarrow (Z_{t+1,q} \wedge P_{t+1,i+d} \wedge B_{t+1,i,\tau})]$$
von φ_{D_1} sind noch nicht in KNF, lassen sich aber äquivalent umformen:
$$(\neg Z_{t,p} \vee \neg P_{t,i} \vee \neg B_{t,i,\sigma} \vee Z_{t+1,q}) \wedge$$
$$(\neg Z_{t,p} \vee \neg P_{t,i} \vee \neg B_{t,i,\sigma} \vee P_{t+1,i+d}) \wedge$$
$$(\neg Z_{t,p} \vee \neg P_{t,i} \vee \neg B_{t,i,\sigma} \vee B_{t+1,i,\tau})$$

- Ein technisches Detail: Was ist, wenn M weniger als n^k Schritte macht?
  - Dann wird in der Berechnungstabelle die Endkonfiguration ab der entsprechenden Zeile immer wiederholt
  - Die Formel wird analog gebildet (als wäre $\delta(\text{ja}, \sigma) = (\text{ja}, \sigma, \downarrow)$)

Beweisskizze (Forts.)

- φ_{D_2} drückt aus, dass sich der String an allen übrigen Positionen nicht verändert:

- $\varphi_{D_2} \stackrel{\text{def}}{=} \bigwedge_{t,i,\sigma} ((\neg P_{t,i} \wedge B_{t,i,\sigma}) \rightarrow B_{t+1,i,\sigma})$
- In konjunktiver Normalform:
$$\varphi_{D_2} \stackrel{\text{def}}{=} \bigwedge_{t,i,\sigma} (P_{t,i} \vee \neg B_{t,i,\sigma} \vee B_{t+1,i,\sigma})$$

Beispiel

- Für $t = 8$ und $i = 3$ ergibt sich also:
$$(P_{8,3} \vee \neg B_{8,3,a} \vee B_{9,3,a}) \wedge$$
$$(P_{8,3} \vee \neg B_{8,3,b} \vee B_{9,3,b}) \wedge$$
$$\cdots$$
$$(P_{8,3} \vee \neg B_{8,3,\$} \vee B_{9,3,\$}) \wedge$$
$$(P_{8,3} \vee \neg B_{8,3,\triangleright} \vee B_{9,3,\triangleright})$$

Satz von Cook (14/14): Abschluss des Beweises

Beweisskizze (Forts.)

- Endbedingung:
  - φ_E drückt aus, dass die letzte Konfiguration den ja-Zustand hat:
    $$\varphi_E = Z_{n^k,\text{ja}}$$

- *Behauptung: Die Größe von φ ist polynomiell in n:*
  - Größe der einzelnen Teilformeln:

φ_K	$\mathcal{O}(n^{3k})$
φ_A	$\mathcal{O}(n^k)$
φ_D	$\mathcal{O}(n^{2k})$
φ_E	$\mathcal{O}(1)$

✎ $|\psi_{\text{unique}}(x_1,\ldots,x_s)| = \mathcal{O}(s^2)$

  - Zu beachten: m und l sind durch M bestimmt und damit konstant

➡ die Größe von φ ist polynomiell in n

- Und: alle Teilformeln sind in konjunktiver Normalform, also auch ihre Konjunktion

Beweisskizze (Forts.)

- Noch zu zeigen: $w \in L \iff \varphi$ erfüllbar
- Also: Zu w gibt es genau dann eine Zusatzeingabe y, die M zum Akzeptieren bringt, wenn φ erfüllbar ist

- Falls es ein solches y gibt, können alle Variablen gemäß ihrer intendierten Bedeutung mit Wahrheitswerten belegt werden

➡ φ wird wahr

- Umgekehrt: Zu jeder erfüllenden Belegung α von φ lässt sich eine Zusatzeingabe y und eine akzeptierende Berechnung von M bei Eingabe w und Zusatzeingabe y konstruieren

➡ $w \in L$

- Nachweis jeweils durch Induktion nach t

- Damit ist die Beweisskizze des Satzes von Cook vollendet

Inhalt

Einleitung

- Wir wissen jetzt also: SAT ist **NP**-vollständig
- Ihre große Bedeutung hat die **NP**-Vollständigkeit erst durch den Nachweis erlangt, dass viele andere algorithmische Probleme **NP**-vollständig sind
- Die erste größere Menge solcher Probleme wurde von Karp 1972 vorgestellt
  - Die Arbeit enthält alle im Folgenden betrachteten Probleme
  - Die hier vorgestellten Beweise sind aber zum Teil anders

- Wie schon gesagt, werden wir ähnlich wie im Falle der unentscheidbaren Probleme in Teil C vorzugehen:
  - Ausgehend von SAT zeigen wir die **NP**-Vollständigkeit der anderen Probleme jeweils mit Hilfe einer *einzelnen* polynomiellen Reduktion

- Zunächst vergewissern wir uns aber, dass dieser Ansatz wirklich funktioniert
- Dazu zeigen wir, dass wir wie folgt schließen können
  - Wenn alle **NP**-Probleme polynomiell reduzierbar auf eine Sprache L' sind
  - und L' polynomiell auf eine Sprache L reduzierbar ist
  - dann sind auch alle **NP**-Probleme polynomiell reduzierbar auf L

- Wir müssen also zeigen, dass $\leqslant_p$ eine transitive Relation ist

Reduktionen und NP-Vollständigkeit (1/2)

Lemma 19.2

- Seien $L_1, L_2, L_3 \subseteq \Sigma^*$ Sprachen
- Falls $L_1 \leqslant_p L_2$ und $L_2 \leqslant_p L_3$, so gilt auch $L_1 \leqslant_p L_3$

Beweisskizze

- Sei f_1 eine polynomielle Reduktion von L_1 auf L_2 und f_2 eine polynomielle Reduktion von L_2 auf L_3

- Behauptung: Die durch $f(w) \stackrel{\text{def}}{=} f_2(f_1(w))$ definierte Funktion ist eine polynomielle Reduktion von L_1 auf L_3

Beweisskizze (Forts.)

- f ist eine Reduktion:
  - Für alle Strings $w \in \Sigma^*$ gilt:
    $w \in L_1$
    $\iff f_1(w) \in L_2$
    $\iff f(w) = f_2(f_1(w)) \in L_3$

- f kann in polynomieller Zeit berechnet werden:
  - Seien n^i und n^j Zeitschranken für die Berechnung von f_1 und f_2

  ➨ Dann kann $f(w)$ in
  $$|w|^i + |f_1(w)|^j \leqslant |w|^i + (|w|^i)^j = \mathcal{O}(|w|^{ij})$$
  Schritten berechnet werden

- Die binäre Relation $\leqslant_p$ zwischen Sprachen ist also transitiv

Reduktionen und NP-Vollständigkeit (2/2)

- Die Möglichkeit des Nachweises der **NP**-Vollständigkeit durch eine einzelne Reduktion wird nun durch das folgende Lemma eröffnet

Lemma 19.3

- Ist L' **NP**-schwierig und gilt $L' \leqslant_p L$, so ist auch L **NP**-schwierig

Beweisskizze

- Sei $L'' \in$ **NP** beliebig

➨ $L'' \leqslant_p L'$

☞ da L' **NP**-schwierig

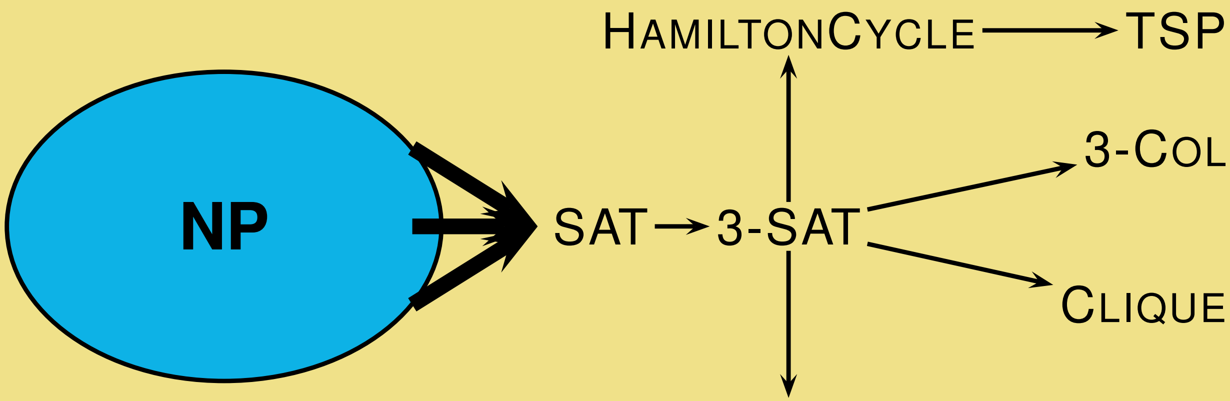
- Wegen $L' \leqslant_p L$ folgt $L'' \leqslant_p L$

☞ gemäß Lemma 19.2

➨ L **NP**-schwierig

- Um nachzuweisen, dass ein Problem L **NP**-vollständig ist, genügt es also zu zeigen:
  - $L \in$ **NP**
  - $L' \leqslant_p L$ für ein **NP**-vollständiges Problem L'

- Die Reduktionen, die wir in diesem Kapitel betrachten werden, sind in der folgenden Abbildung zusammengefasst:



Polynomielle Reduktionen: Rezept

- Die folgenden Beweise für Aussagen der Art $L_1 \leqslant_p L_2$ verlaufen alle nach demselben Muster
- Sie gehen in vier Schritten vor

(1) Definiere eine Reduktionsfunktion f, die Eingaben für L_1 auf Eingaben für L_2 abbildet

(2) Zeige, dass f in polynomieller Zeit berechnet werden kann

✎ Das ist meistens ziemlich offensichtlich

(3) Zeige: wenn $w \in L_1$ dann ist auch $f(w) \in L_2$

- Beweise dazu, dass aus jeder Lösung y_1 für w eine Lösung y_2 für $f(w)$ konstruiert werden kann

(4) Zeige: wenn $f(w) \in L_2$ dann ist auch $w \in L_1$

- Beweise dazu, dass aus jeder Lösung y_2 für $f(w)$ eine Lösung y_1 für w konstruiert werden kann

- Daraus folgt dann $L_1 \leqslant_p L_2$

Inhalt

SAT $\leqslant_p$ 3-SAT (1/4)

Proposition 19.4

$$\text{SAT} \leqslant_p \text{3-SAT}$$

- Bei dieser Reduktion verwenden wir für lange Klauseln eine ähnliche Idee wie beim Entfernen langer rechter Seiten bei der Umwandlung kontextfreier Grammatiken in Chomsky-Normalform...

Beispiel

- Für $\varphi = (x_1 \vee \neg x_2) \wedge (x_2 \vee \neg x_4 \vee x_7 \vee \neg x_8 \vee x_1 \vee \neg x_5)$ sei:

$$f(\varphi) \stackrel{\text{def}}{=} (x_1 \vee \neg x_2 \vee \neg x_2) \wedge (x_2 \vee \neg x_4 \vee y_1^2) \wedge (\neg y_1^2 \vee x_7 \vee y_2^2) \wedge (\neg y_2^2 \vee \neg x_8 \vee y_3^2) \wedge (\neg y_3^2 \vee x_1 \vee \neg x_5)$$

- Die erfüllende Belegung $\theta : x_1 \mapsto 0, x_2 \mapsto 0, x_4 \mapsto 1, x_7 \mapsto 0, x_8 \mapsto 0, x_5 \mapsto 1$ kann zu einer erfüllenden Belegung von $f(\varphi)$ erweitert werden durch: $y_1^2 \mapsto 1, y_2^2 \mapsto 1, y_3^2 \mapsto 0$

SAT $\leqslant_p$ 3-SAT (2/4)

Beweisskizze

- Sei $\varphi = K_1 \wedge \cdots \wedge K_m$ eine KNF-Formel

(1) $f(\varphi) \stackrel{\text{def}}{=} \chi_1 \wedge \cdots \wedge \chi_m$, wobei die 3-KNF-Formeln χ_i wie folgt definiert sind

- Sei $K_i = L_1 \vee \cdots \vee L_j$ (mit Literalen L_ℓ)
- Wenn $j = 1$, dann $\chi_i \stackrel{\text{def}}{=} L_1 \vee L_1 \vee L_1$
- Wenn $j = 2$, dann $\chi_i \stackrel{\text{def}}{=} L_1 \vee L_2 \vee L_2$
- Wenn $j = 3$, dann $\chi_i \stackrel{\text{def}}{=} K_i$
- Wenn $j > 3$ verwenden wir $j - 3$ neue Variablen $y_1^i, \ldots, y_{j-3}^i$ und definieren

$$\begin{aligned}\chi_i \stackrel{\text{def}}{=} {} & (L_1 \vee L_2 \vee y_1^i) \wedge \\ & (\neg y_1^i \vee L_3 \vee y_2^i) \wedge \\ & \vdots \\ & (\neg y_{j-4}^i \vee L_{j-2} \vee y_{j-3}^i) \wedge \\ & (\neg y_{j-3}^i \vee L_{j-1} \vee L_j)\end{aligned}$$

(2) $f(\varphi)$ kann in quadratischer Zeit in $|\varphi|$ berechnet werden

SAT $\leqslant_p$ 3-SAT **(3/4)**

Beweisskizze (Forts.)

(3) φ erfüllbar $\Rightarrow f(\varphi)$ erfüllbar:

- Sei θ eine Belegung mit $\theta \models \varphi$

➡ für jedes $i \leqslant m$ gilt: $\theta \models K_i$

- Wir zeigen, dass wir θ zu einer Belegung θ' erweitern können (die auch für die neuen Variablen definiert ist), so dass, für jedes i gilt: $\theta' \models \chi_i$

- Da die neuen Variablen jeweils nur in *einer* Teilformel χ_i vorkommen, können wir die Erweiterung von θ' für jedes χ_i einzeln definieren

- Sei also $i \leqslant m$ und
$$K_i = L_1 \vee \cdots \vee L_j$$

- Falls $j \leqslant 3$, muss θ für χ_i nicht erweitert werden

Beweisskizze (Forts.)

- Sei nun $j > 3$

- Wenn θ eines der beiden ersten Literale von K_i wahr macht ($\theta \models L_1$ oder $\theta \models L_2$), können alle neuen Variablen mit 0 belegt werden:
  - $\theta'(y^i_\ell) \stackrel{\text{def}}{=} 0$, für alle $\ell \leqslant j-3$

- Wenn nicht, aber θ eines der beiden letzten Literale von K_i wahr macht ($\theta \models L_{j-1}$ oder $\theta \models L_j$), können alle neuen Variablen auf 1 gesetzt werden
  - $\theta'(y^i_\ell) \stackrel{\text{def}}{=} 1$, für alle $\ell \leqslant j-3$

- Andernfalls sei $p \leqslant j$ die kleinste Zahl mit $\theta \models L_p$ und wir setzen
  - $\theta'(y^i_\ell) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{für alle } \ell \leqslant p-2 \\ 0 & \text{für alle } \ell \geqslant p-1 \end{cases}$

- In allen Fällen folgt dann: $\theta' \models \chi_i$

SAT $\leqslant_p$ 3-SAT (4/4)

Beweisskizze (Forts.)

(4) $f(\varphi)$ erfüllbar $\Rightarrow$ φ erfüllbar:

- Sei θ' eine erfüllende Belegung für $f(\varphi)$

➡ für jedes $i \leqslant m$ gilt: $\theta' \models \chi_i$

- Wir definieren $\theta(x_p) \stackrel{\text{def}}{=} \theta'(x_p)$, für alle Variablen x_p, die in φ vorkommen

- Zu zeigen: für jedes $i \leqslant m$ gilt: $\theta \models K_i$

- Sei also $K_i = L_1 \vee \cdots \vee L_j$ eine Klausel von φ

- Falls $j \leqslant 3$, ist K_i äquivalent zu χ_i und deshalb folgt $\theta \models K_i$ direkt aus $\theta' \models \chi_i$

Beweisskizze (Forts.)

- Beobachtung: falls $j > 3$ können nicht alle Klauseln von χ_i durch Literale mit Variablen y_q^i wahr gemacht werden
  - denn: nach Konstruktion kann jede Variable nur eine Klausel wahr machen, es sind aber weniger neue Variablen als Klauseln

➡ für ein $p \leqslant j$ muss gelten $\theta' \models L_p$

➡ $\theta \models K_i$

Folgerung 19.5

3-SAT ist **NP**-vollständig

Inhalt

3-SAT $\leqslant_p$ CLIQUE (1/5)

- Die Reduktionen 3-COL $\leqslant_p$ SAT und SAT $\leqslant_p$ 3-SAT waren nicht allzu kompliziert
  - Dass sich die Korrektheit einer 3-Färbung eines Graphen in einer aussagenlogischen Formel „kodieren“ lässt, ist nicht allzu überraschend

- Wir wollen jetzt zeigen: 3-SAT $\leqslant_p$ CLIQUE
  - Das ist schon weniger nahe liegend
  - Wie sollen Variablen, Wahrheitsbelegungen und Klauseln in einen Graphen kodiert werden?
  - Das ist deutlich komplizierter

- Grob gesagt, ist die Korrespondenz zwischen Formeln und Graphen in dieser Reduktion wie folgt:
  - Jedes *Vorkommen* eines Literals entspricht einem Knoten im Graphen
  - Kanten zwischen Knoten drücken aus, dass die entsprechenden Literale sich nicht widersprechen
  - Cliquen im Graphen entsprechen dann Mengen simultan erfüllbarer Literale

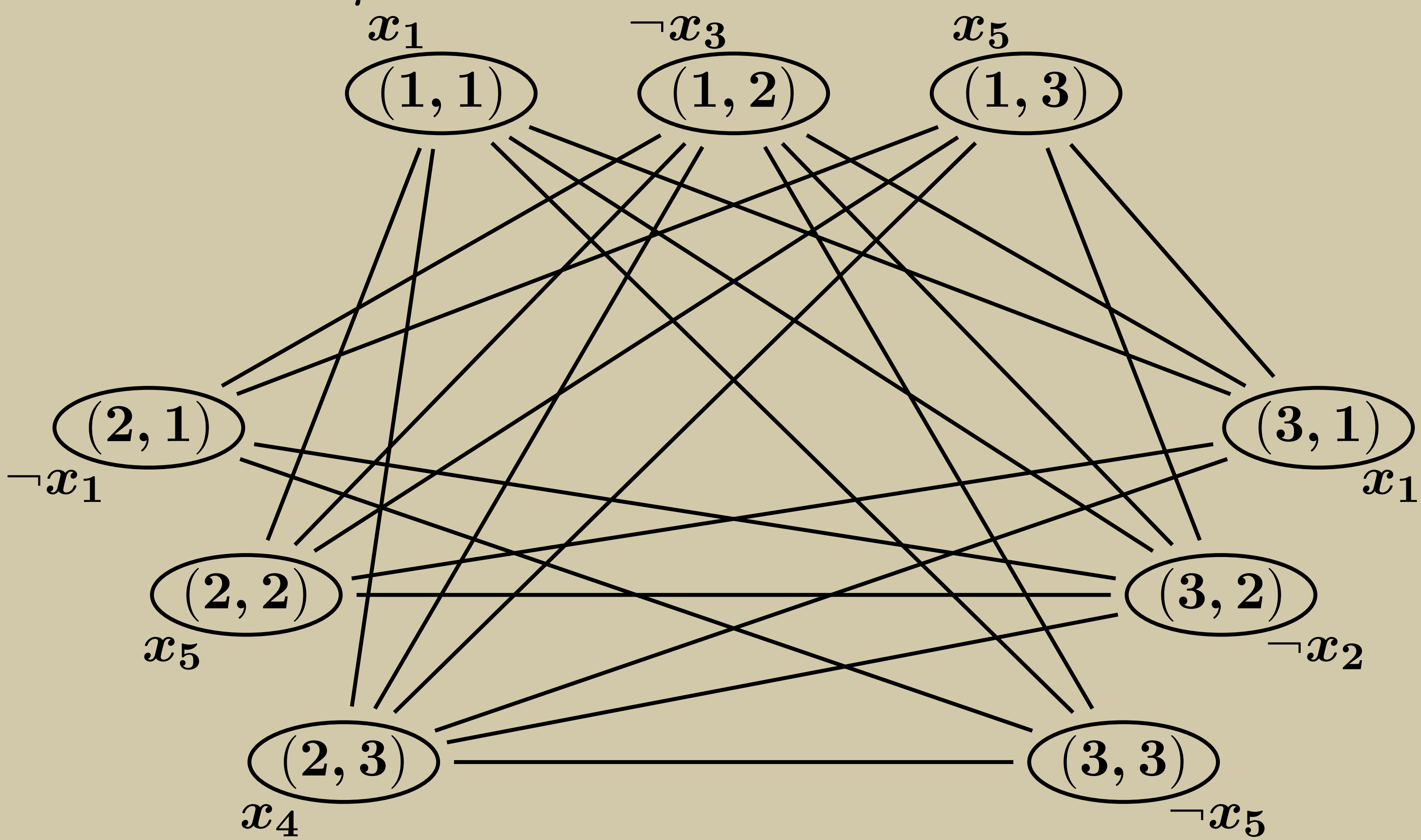
Proposition 19.6

3-SAT $\leqslant_p$ CLIQUE

3-SAT $\leqslant_p$ CLIQUE (2/5)

Illustration der Reduktion von 3-SAT auf CLIQUE

- Beispielformel $\varphi = (x_1 \vee \neg x_3 \vee x_5) \wedge (\neg x_1 \vee x_5 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee \neg x_5)$
- Beispielgraph G_φ:



3-SAT $\leqslant_p$ CLIQUE (3/5)

Beweisskizze

(1) Sei φ eine Formel in KNF mit m Klauseln zu je 3 Literalen

- Wir konstruieren einen Graphen G mit $3m$ Knoten so dass gilt:
  G hat eine Clique der Größe $m \iff \varphi$ ist erfüllbar

- Sei $\varphi = (L_{11} \vee L_{12} \vee L_{13}) \wedge \cdots \wedge (L_{m1} \vee L_{m2} \vee L_{m3})$

- Sei $G = (V, E)$ mit
  - $V = \{(i,j) \mid 1 \leqslant i \leqslant m, 1 \leqslant j \leqslant 3\}$
  - $E = \{((i,j),(k,l)) \mid i \neq k$ und $L_{ij} \neq \neg L_{kl}\}$

- G hat also einen Knoten für jedes Vorkommen eines Literals in einer Klausel von φ
- Zwei Knoten sind miteinander verbunden, wenn ihre Literale
  - nicht in derselben Klausel sind *und*
  - sich nicht direkt widersprechen, d.h. nicht einer Variablen x_i und ihrer Negation $\neg x_i$ entsprechen
- Wir definieren $f(\varphi) \stackrel{\text{def}}{=} (G, m)$

(2) f kann in quadratischer Zeit in $|\varphi|$ berechnet werden

3-SAT $\leqslant_p$ Clique (4/5)

Beweisskizze (Forts.)

(3) φ erfüllbar $\Rightarrow$ G hat eine m-Clique

- Sei θ eine erfüllende Belegung für φ
- Dann gibt es für jedes $i \leqslant m$ ein $j_i \in \{1, 2, 3\}$, so dass $\theta \models L_{ij_i}$

➨ Die Literale $L_{1j_1}, \dots, L_{mj_m}$ sind in verschiedenen Klauseln und widersprechen sich nicht

➨ Ihre zugehörigen Knoten bilden eine Clique der Größe m in G

➨ (3)

Beweisskizze (Forts.)

(4) G hat eine m-Clique $\Rightarrow$ φ erfüllbar

- Sei C eine m-Clique von G
- C besteht aus Knoten zu m Literalen $L_{1j_1}, \dots, L_{mj_m}$ aus verschiedenen Klauseln
- Da alle diese Literale miteinander verbunden sind, gibt es keine Variable x_p, für die sowohl x_p als auch $\neg x_p$ in $\{L_{1j_1}, \dots, L_{mj_m}\}$ vorkommt

➨ θ kann so definiert werden, dass alle Literale $L_{1j_1}, \dots, L_{mj_m}$ wahr werden

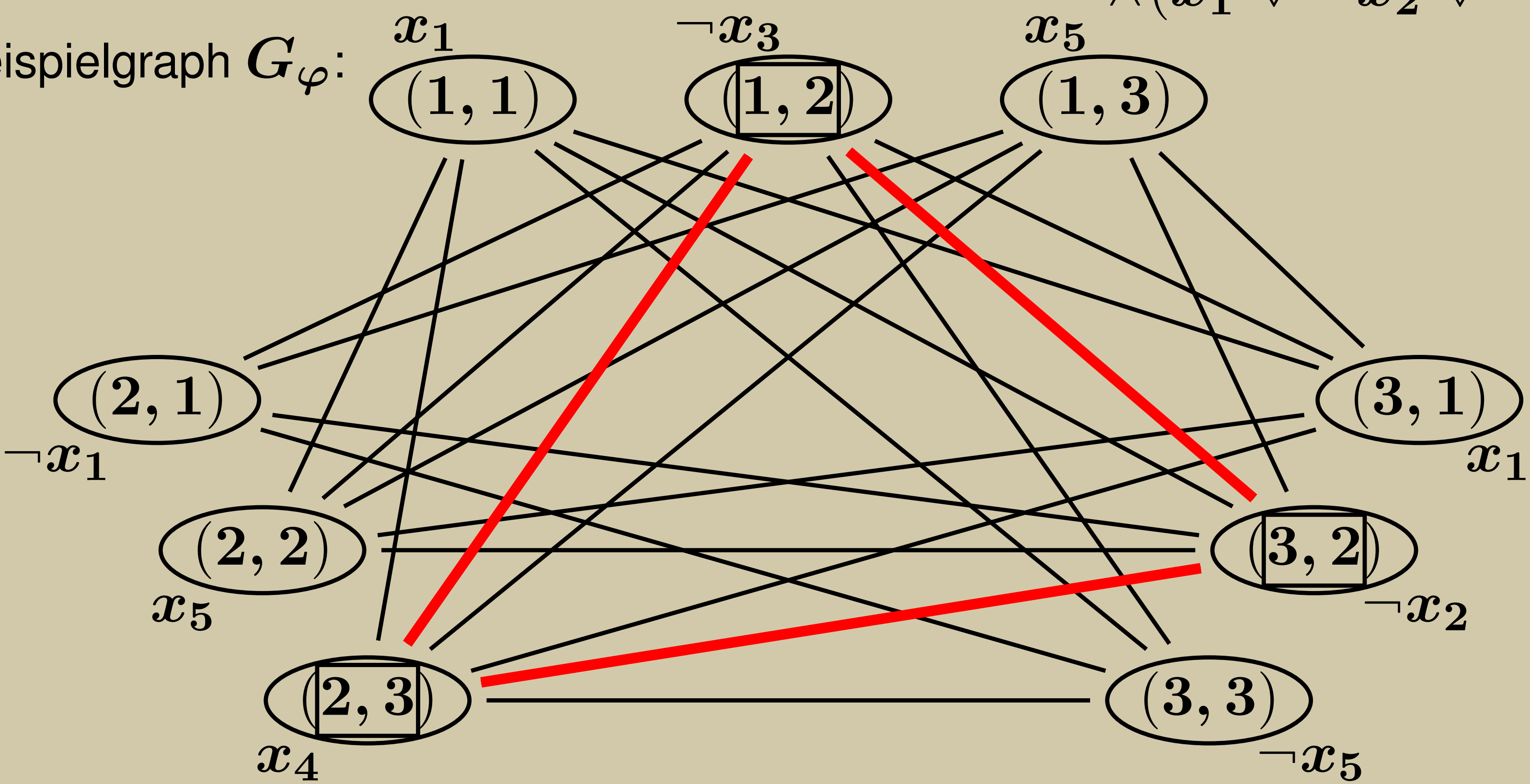
➨ θ macht in jeder Klausel mindestens ein Literal wahr

➨ (4)

3-SAT $\leqslant_p$ CLIQUE (5/5)

Illustration der Reduktion von 3-SAT auf CLIQUE

- Beispielformel $\varphi = (x_1 \lor \neg x_3 \lor x_5) \land (\neg x_1 \lor x_5 \lor x_4)$
  $\land (x_1 \lor \neg x_2 \lor \neg x_5)$
- Beispielgraph G_φ:



- Die Knoten $(1,2)$, $(2,3)$ und $(3,2)$ bilden eine Clique in G_φ

➨ Die Literale $\neg x_3$, $\neg x_2$ und x_4 widersprechen sich nicht

➨ Sie induzieren deshalb eine erfüllende Belegung von φ:
  - $\theta(x_2) = 0, \theta(x_3) = 0, \theta(x_4) = 1$
  - Der Rest ist frei wählbar, z.B.: $\theta(x_1) = 0, \theta(x_5) = 1$

Inhalt

3-SAT $\leqslant_p$ 3-COL (1/9)

- Bei der nächsten Reduktion 3-SAT $\leqslant_p$ 3-COL wird wieder zu jeder 3KNF-Formel ein Graph (und eine Zahl) konstruiert
- Die Korrespondenz zwischen den Bestandteilen des Graphen und der Formel ist jedoch anders

- Wir assoziieren
  - jede aussagenlogische Variable x_i mit jeweils zwei Knoten x_i und $\neg x_i$ des Graphen,
  - Wahrheitswerte mit Farben,
  - und Klauseln mit speziell konstruierten Teilgraphen, die nur dann korrekt gefärbt werden können, wenn die entsprechende Klausel durch die gegebene Wahrheitsbelegung wahr wird

Proposition 19.7

- 3-SAT $\leqslant_p$ 3-COL

3-SAT $\leqslant_p$ 3-COL (2/9)

Beweisskizze

(1) Sei $\varphi = K_1 \wedge \cdots \wedge K_m$ eine 3KNF-Formel mit Variablen $x_1, \ldots, x_n$ und Klauseln $K_1, \ldots, K_m$
- Mit L_{i1}, L_{i2}, L_{i3} seien jeweils die drei Literale der Klausel K_i bezeichnet
- Also: $K_i = L_{i1} \vee L_{i2} \vee L_{i3}$

- Die Reduktion konstruiert einen Graphen G mit folgenden Knoten:
  - Für jedes $i \leqslant n$ je ein Knoten x_i und $\neg x_i$ (die *Literalknoten*)
  - Für jedes $i \leqslant m$ fünf Knoten b_i, c_i, d_i, e_i, g_i pro Klausel K_i (die *Klauselknoten*)
  - Drei Knoten t, u, n

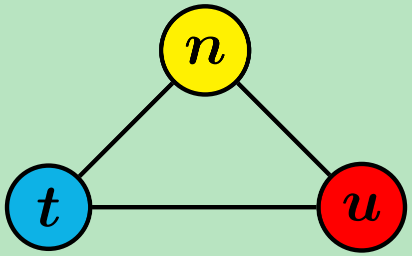
- Intention: G ist genau dann mit rot, blau und gelb zulässig färbbar, wenn φ erfüllbar ist
- Wir definieren dann $f(\varphi) \stackrel{\text{def}}{=} G$

✎ Literale kommen im Folgenden in φ und als Knoten in G vor!

3-SAT $\leqslant_p$ 3-Col (3/9)

Beweisskizze (Forts.)

- Unabhängig von der gegebenen Formel enthält G die Kanten
  $$(u,t),(u,n),(t,n):$$

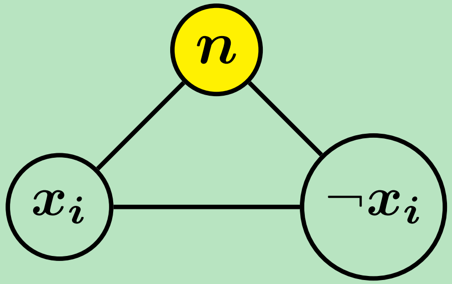


- Intention: t, u, n müssen verschieden gefärbt werden

- Für den Nachweis der Reduktionseigenschaft werden wir im Folgenden oBdA davon ausgehen, dass zulässige Färbungen t blau, u rot und n gelb färben
- Intuitiv soll blau „wahr“ und rot „unwahr“ entsprechen ☞ $t \equiv$ *true*, $u \equiv$ *untrue*
- (gelb ist „neutral“)

Beweisskizze (Forts.)

- Für jedes i enthält G die Kanten
  $$(n,x_i),(n,\neg x_i),(\neg x_i,x_i):$$

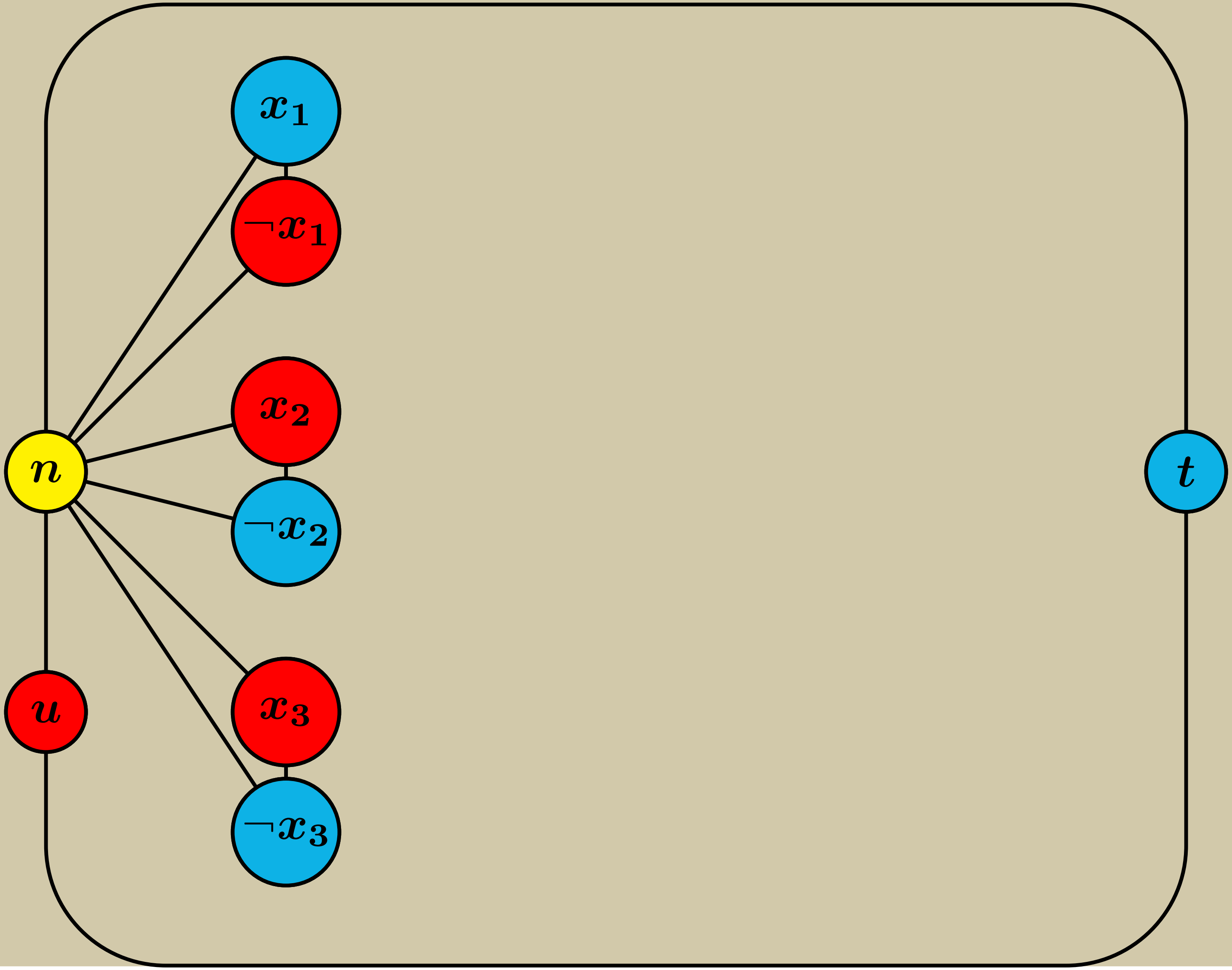


- Intention:
  - Die Knoten x_i und $\neg x_i$ müssen jeweils verschieden gefärbt werden ☞ klar
  - Da n gelb ist, müssen dafür die Farben rot und blau verwendet werden

- Wir erweitern unsere Intention entsprechend:
  - Falls x_i in G blau gefärbt wird, soll dies $\theta(x_i) = 1$ entsprechen
  - Falls $\neg x_i$ in G blau gefärbt wird, soll dies $\theta(x_i) = 0$ entsprechen

Beispiel (Forts.)

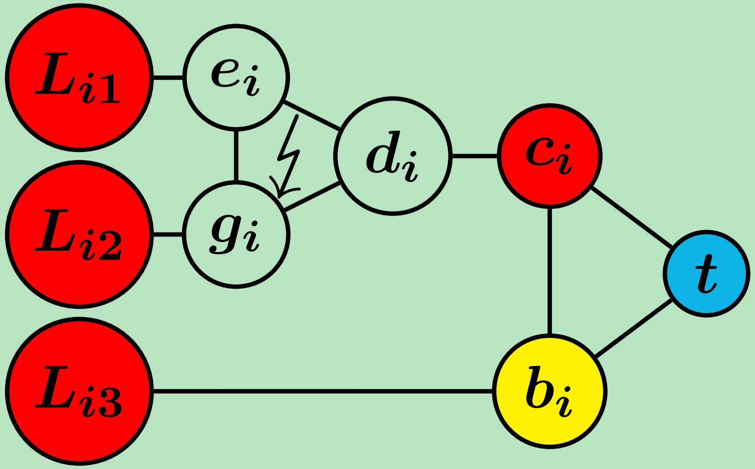
- Beispielformel $\varphi = (x_1 \lor \neg x_2 \lor x_3) \land (\neg x_1 \lor x_2 \lor \neg x_3) \land (x_1 \lor x_2 \lor \neg x_3)$
- Beispielgraph G (im Aufbau):



$$\theta(x_1) = 1$$
$$\theta(x_2) = 0$$
$$\theta(x_3) = 0$$

Beweisskizze (Forts.)

- Schließlich verwenden wir für jede Klausel K_i mit Literalen L_{i1}, L_{i2}, L_{i3} einen Teilgraphen H_i der folgenden Form

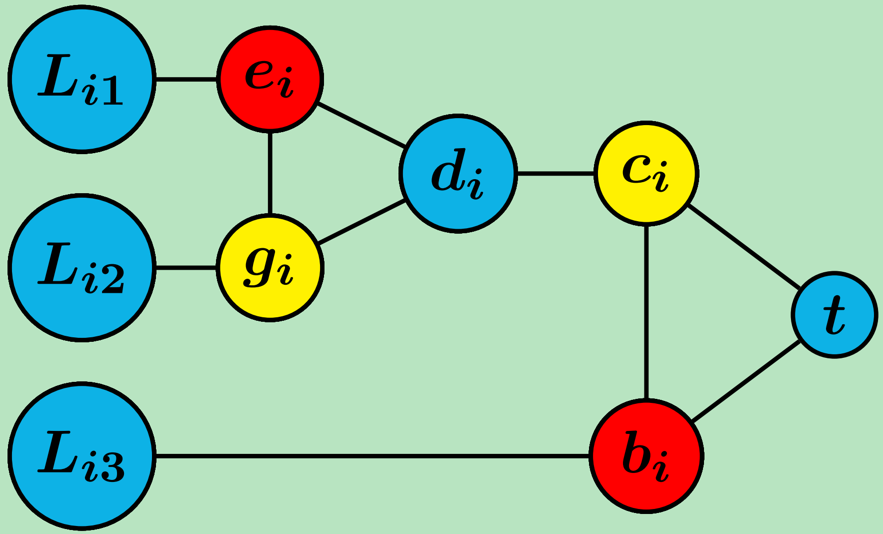


- Dabei sind b_i, c_i, d_i, e_i, g_i neue Knoten, die nur in H_i vorkommen

- Beobachtung: Sind alle drei Knoten L_{i1}, L_{i2}, L_{i3} rot, so kann dieser Teilgraph nicht zulässig gefärbt werden

Beweisskizze (Forts.)

- Umgekehrt kann jede Färbung, die mindestens einen der Knoten L_i blau färbt, zu einer zulässigen Färbung des Teilgraphen erweitert werden:

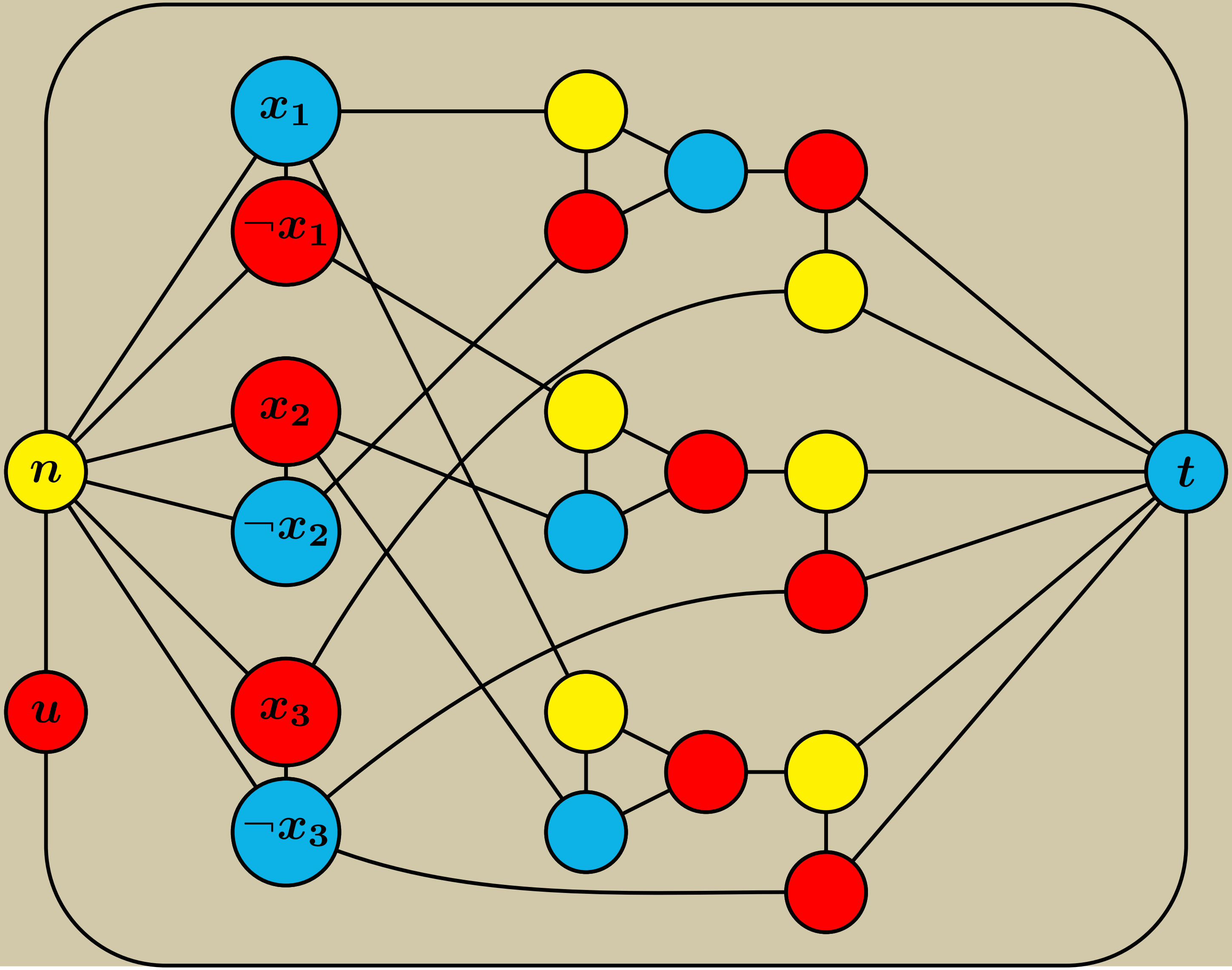


- Wir nennen den Teilgraphen H_i mit den Knoten b_i, c_i, d_i, e_i, g_i, den **Klausel-Teilgraphen** zu K_i

- Die Knoten L_{i1}, L_{i2}, L_{i3} nennen wir die **mit H_i verbundenen** Literalknoten

Beispiel (Forts.)

- Beispielformel $\varphi = (x_1 \lor \neg x_2 \lor x_3) \land (\neg x_1 \lor x_2 \lor \neg x_3) \land (x_1 \lor x_2 \lor \neg x_3)$
- Beispielgraph G:



$$\theta(x_1) = 1$$
$$\theta(x_2) = 0$$
$$\theta(x_3) = 0$$

Beweis von 3-SAT $\leqslant_p$ 3-COL (7/9)

Beweisskizze (Forts.)

- G hat also insgesamt die folgenden Kanten:
  - $(u,t), (u,n), (t,n)$
  - Für jedes $i \leqslant n$: $(n,x_i), (n,\neg x_i), (\neg x_i, x_i)$
  - Für jedes $i \leqslant m$:
    $(L_{i1}, e_i), (L_{i2}, g_i), (L_{i3}, b_i), (e_i, g_i), (e_i, d_i),$
    $(g_i, d_i), (d_i, c_i), (c_i, b_i), (c_i, t), (b_i, t)$

(2) $f(\varphi) = G$ kann in polynomieller Zeit berechnet werden

$\text{3-SAT} \leqslant_p \text{3-Col}$ (8/9)

Beweisskizze (Forts.)

(3) φ erfüllbar $\Rightarrow G$ 3-färbbar

- Sei $\theta : \{x_1, \ldots, x_n\} \to \{0, 1\}$ eine erfüllende Belegung für φ
- Wir konstruieren eine zulässige Färbung c:
  - $c(t) \overset{\text{def}}{=}$ blau, $c(u) \overset{\text{def}}{=}$ rot, $c(n) \overset{\text{def}}{=}$ gelb
  - $c(x_i) \overset{\text{def}}{=}$ blau, falls $\theta(x_i) = 1$, andernfalls rot
  - $c(\neg x_i) \overset{\text{def}}{=}$ blau, falls $\theta(x_i) = 0$, andernfalls rot

- Da θ jede Klausel K_i wahr macht, hat jeder Teilgraph H_i mindestens einen blauen Literal-Knoten

➨ Jeder Teilgraph H_i kann zulässig gefärbt werden

- Bezüglich der übrigen Kanten ist die entstehende Färbung ebenfalls zulässig

➨ $f(\varphi)$ ist 3-färbbar

➨ (3)

3-SAT $\leqslant_p$ 3-Col (9/9)

Beweis (Forts.)

(4) G 3-färbbar $\Rightarrow \varphi$ erfüllbar

- Sei c eine zulässige Färbung von G
- OBdA: $c(t)$ = blau, $c(u)$ = rot, $c(n)$ = gelb
  - Sonst benennen wir die Farben um

➨ Für jedes $i \leqslant n$ gilt:
  - $c(x_i)$ = blau und $c(\neg x_i)$ = rot oder
  - $c(x_i)$ = rot und $c(\neg x_i)$ = blau

Beweis (Forts.)

- Definiere θ durch
  - $\theta(x_i) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{falls } c(x_i) = \text{blau} \\ 0 & \text{falls } c(x_i) = \text{rot} \end{cases}$

- Für jedes Literal L gilt also: $\theta \models L \iff c(L) = \text{blau}$

- Da jeder Klauselgraph H_i zulässig gefärbt ist, muss jeweils mindestens einer der Literalknoten L_{i1}, L_{i2}, L_{i3} blau sein

➨ für jede Klausel K_i wird mindestens eines der Literale L_{i1}, L_{i2}, L_{i3} wahr

➨ θ ist eine erfüllende Belegung von φ

➨ φ erfüllbar

➨ (4)

Präsentation von Reduktionsbeweisen

- In der Darstellung von Reduktionen (z.B.: von 3-SAT auf 3-COL) vermischen sich meist mehrere Aspekte

- Die Beschreibung der Reduktionsfunktion
  - im Beispiel: der Funktion
    $$f : \varphi \mapsto G_\varphi$$
- Die Beschreibung der Intention der Reduktion, also z.B.
  - die Korrespondenz zwischen Färbungen und Wahrheitsbelegungen („falls Knoten x_i blau wird, ist $\theta(x_i) = 1$“)
  - die Färbbarkeitseigenschaften der elementaren Dreiecke und der Klauselgraphen

- Diese Vermischung erscheint schwer vermeidlich, da andernfalls die Konstruktion kaum zu verstehen wäre

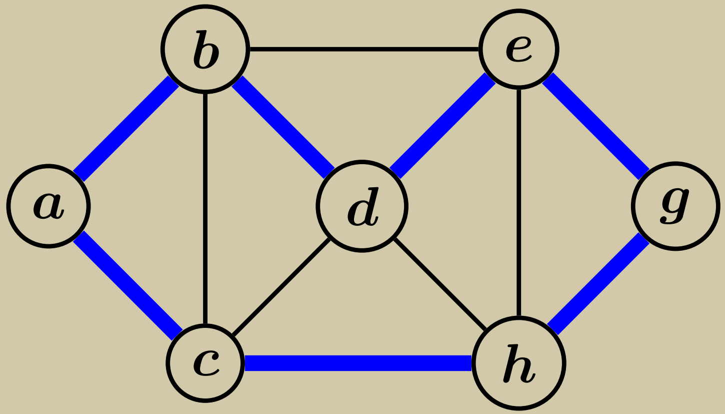
- Wichtig ist aber, dass Sie sich klarmachen, dass die Reduktionsfunktion f selbst *weder eine erfüllende Belegung noch eine zulässige Färbung konstruiert*

- Es besteht lediglich ein *Zusammenhang* zwischen erfüllenden Belegungen und zulässigen Färbungen
  - Mit Hilfe dieses Zusammenhangs beweisen wir dann in den Schritten (3) und (4), dass f eine Reduktion ist

Inhalt

Zur Erinnerung: Hamilton-Kreise

Beispiel



Definition (HAMILTONCYCLE)

Gegeben: Ungerichteter Graph G

Frage: Gibt es einen geschlossenen Weg in G, der jeden Knoten genau einmal besucht?

3-SAT $\leqslant_p$ HAMILTONCYCLE (1/9)

- Bei der folgenden Reduktion von 3-SAT auf HAMILTONCYCLE entspricht jede Variable x_i der Formel einem Knoten v_i des Graphen

- Jeder Knoten v_i hat zwei ausgehende Kanten, die zu den beiden Literalen x_i und $\neg x_i$ korrespondieren
- Jede Wahrheitsbelegung entspricht dann der Auswahl einer Menge von Kanten für die Literale, die sie wahr macht

- Zu jeder Klausel der Formel gibt es einen Teilgraphen
  - Diese Klauselgraphen sorgen dafür, dass sich eine Menge von „Ausgangskanten“ genau dann zu einem Hamiltonkreis erweitern lässt, wenn die entsprechende Wahrheitsbelegung erfüllend ist

Proposition 19.8

3-SAT $\leqslant_p$ HAMILTONCYCLE

3-SAT $\leqslant_p$ HAMILTONCYCLE (2/9)

Beweisskizze

- Wir zeigen zuerst: 3-SAT $\leqslant_p$ GHAMILTONCYCLE
  - GHAMILTONCYCLE: Gegeben ein *gerichteter* Graph G, hat G einen *gerichteten* Hamiltonkreis?

(1) Sei $\varphi = (L_{11} \vee L_{12} \vee L_{13}) \wedge \cdots \wedge (L_{m1} \vee L_{m2} \vee L_{m3})$

- Seien $x_1, \ldots, x_n$ die Variablen von φ

- Der Graph G hat $n + 6m$ Knoten:
  - Die Knoten $v_1, \ldots, v_n$ repräsentieren die Variablen von φ
  - Die Teilgraphen $G_1, \ldots, G_m$ mit Knoten u_{ij} und u'_{ij}, $1 \leqslant i \leqslant m$, $1 \leqslant j \leqslant 3$, repräsentieren die Klauseln von φ

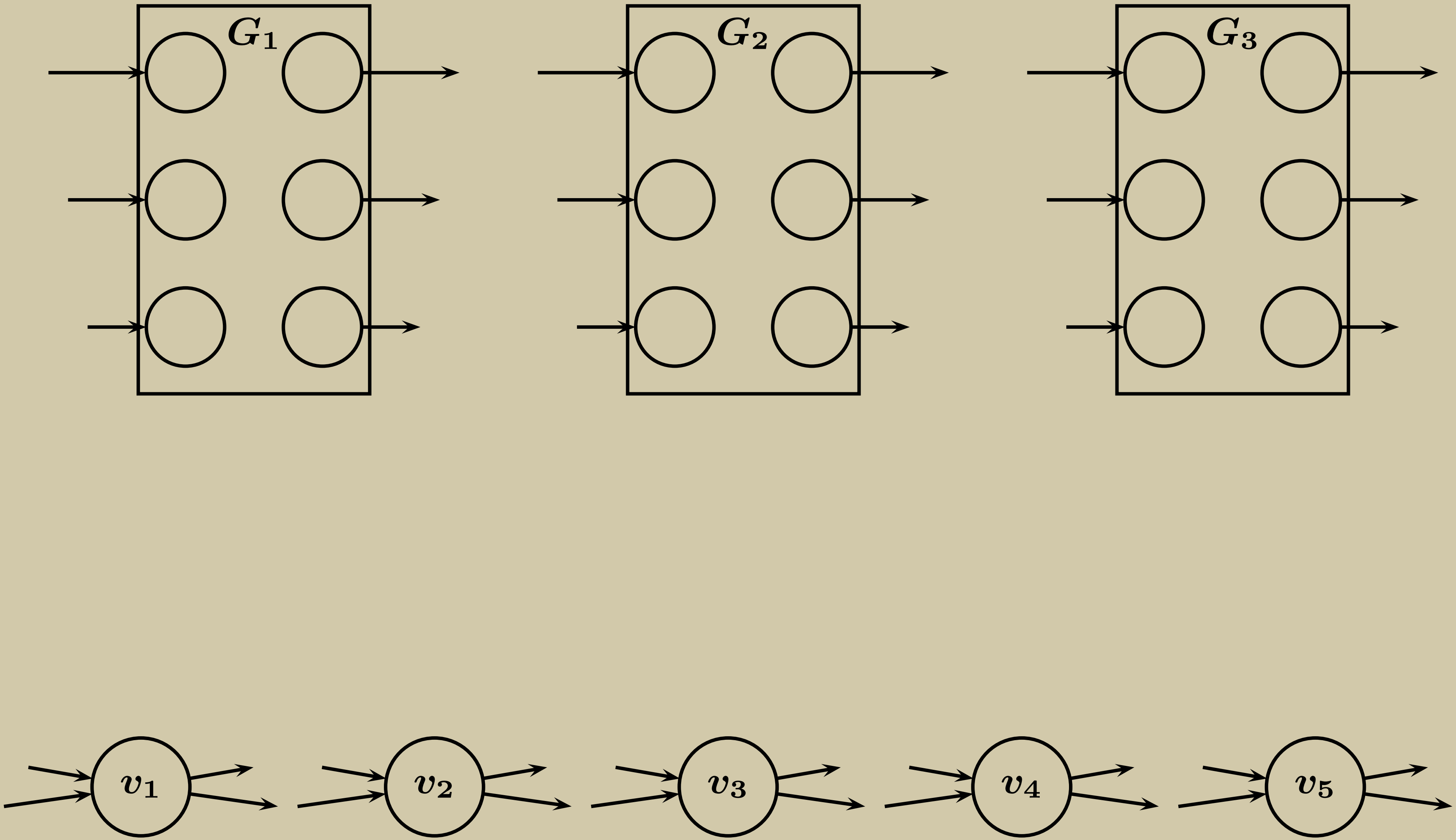
- Jeder Knoten v_i hat 2 eingehende und 2 ausgehende Kanten
  - Intention: die obere ausgehende Kante entspricht dem Wahrheitswert 1

- Die Teilgraphen G_j haben je 3 eingehende und 3 ausgehende Kanten

3-SAT $\leqslant_p$ HAMILTONCYCLE (3/9)

Beispiel

- Beispiel-Formel: $\varphi = (x_1 \vee \neg x_3 \vee x_5) \wedge (\neg x_1 \vee x_5 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee \neg x_5)$
- Beispiel-Graph G:



3-SAT $\leqslant_p$ HAMILTONCYCLE (4/9)

Beweisskizze (Forts.)

- Sei i fest und seien $K_{j_1},\dots,K_{j_l}$ die Klauseln, in denen x_i *positiv* vorkommt, und zwar als $L_{j_1 b_1},\dots,L_{j_l b_l}$
  - D.h.: x_i kommt in Klausel K_{j_t} an der b_t-ten Stelle vor, für $t \leqslant l$

- Dann hat G Kanten
  - vom ersten Ausgang von Knoten v_i zum b_1-ten Eingang von G_{j_1},
  - vom b_t-ten Ausgang von G_{j_t} zum b_{t+1}-ten Eingang von $G_{j_{t+1}}$, für alle $t < l$, und
  - vom b_l-ten Ausgang von G_{j_l} zum ersten Eingang von v_{i+1} (für $i = n$ zum ersten Eingang von v_1).
- Wir nennen diese Kantenmenge den **Literalweg** zu x_i

Beweisskizze (Forts.)

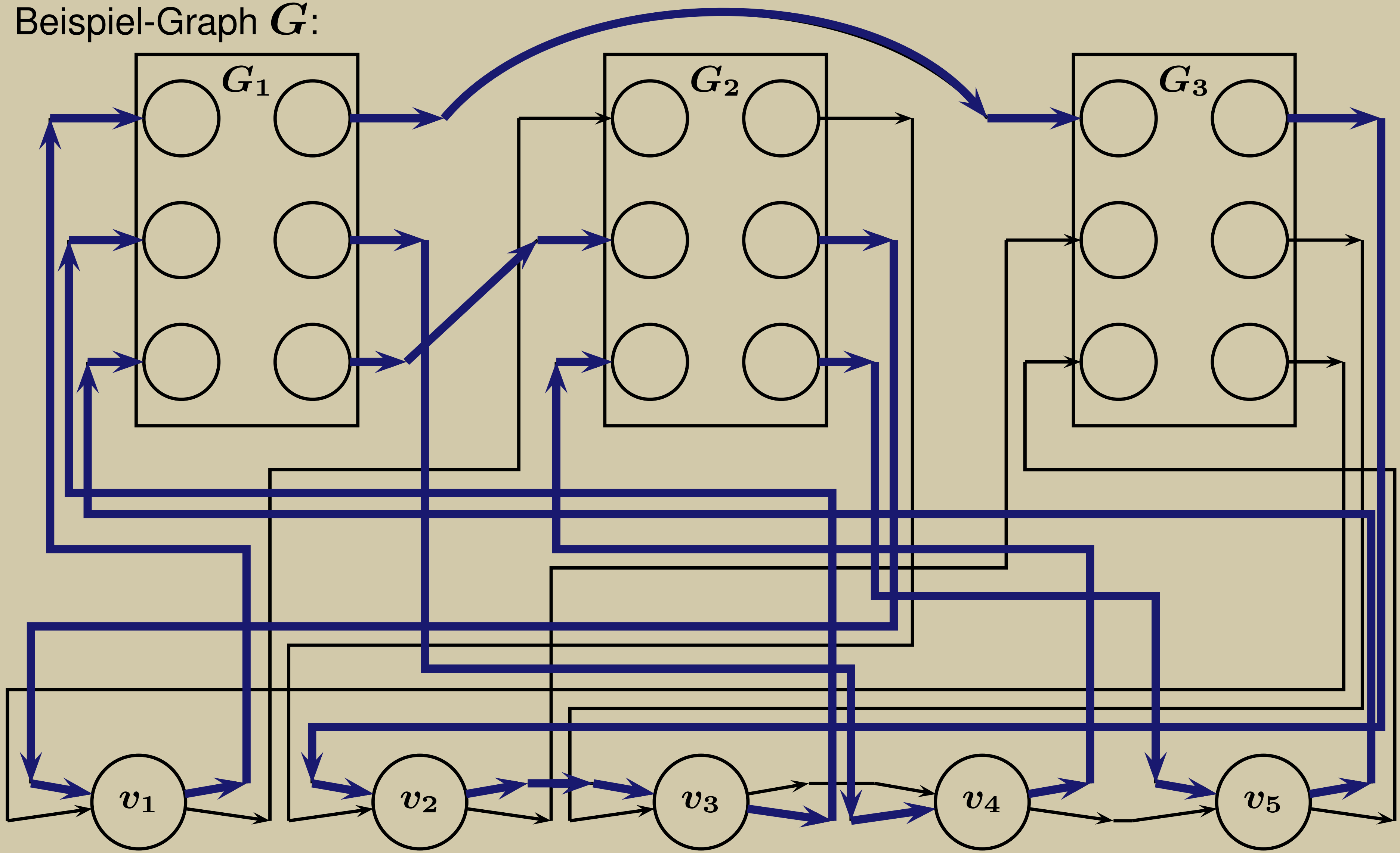
- Analog gibt es einen Literalweg zu $\neg x_i$, der im unteren Ausgang von v_i beginnt und alle Teilgraphen G_j mit *negativen* Vorkommen von x_i durchläuft
- Für jede Variable hat G also zwei Literalwege

- Intention: wenn eine Wahrheitsbelegung ein Literal wahr macht, dann besucht der Literalweg des Literals die Klauselgraphen aller Klauseln, die durch das Literal wahr werden

3-SAT $\leqslant_p$ HAMILTONCYCLE (5/9)

Beispiel

- Beispiel-Formel: $\varphi = (x_1 \vee \neg x_3 \vee x_5) \wedge (\neg x_1 \vee x_5 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee \neg x_5)$
  Belegung: $\theta(x_1) = 1$, $\theta(x_2) = 1$, $\theta(x_3) = 0$, $\theta(x_4) = 1$, $\theta(x_5) = 1$
- Beispiel-Graph G:



3-SAT $\leqslant_p$ HAMILTONCYCLE (6/9)

Beweisskizze (Forts.)

- Konstruktion der Teilgraphen G_i:

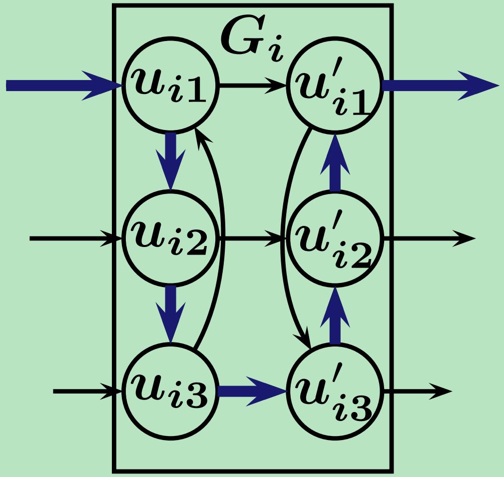
- Intention: die Knoten von G_i lassen sich auf jede nicht-leere Menge von Literalwegen durch G_i verteilen
  ☞ egal, ob es ein, zwei oder drei sind

- u_{i1}, u_{i2}, u_{i3}: Eingangsknoten, mit jeweils einer eingehenden Kante
- $u'_{i1}, u'_{i2}, u'_{i3}$: Ausgangsknoten, mit je einer ausgehenden Kante

- G_i hat folgende innere Kanten:
  - $(u_{i1}, u'_{i1}), (u_{i2}, u'_{i2}), (u_{i3}, u'_{i3})$
  - $(u_{i1}, u_{i2}), (u_{i2}, u_{i3}), (u_{i3}, u_{i1})$
  - $(u'_{i2}, u'_{i1}), (u'_{i3}, u'_{i2}), (u'_{i1}, u'_{i3})$

- Die Kanten zwischen den Knoten u'_{ij} sind gegenläufig zu den Kanten zwischen den Knoten u_{ij}!

Beweisskizze (Forts.)

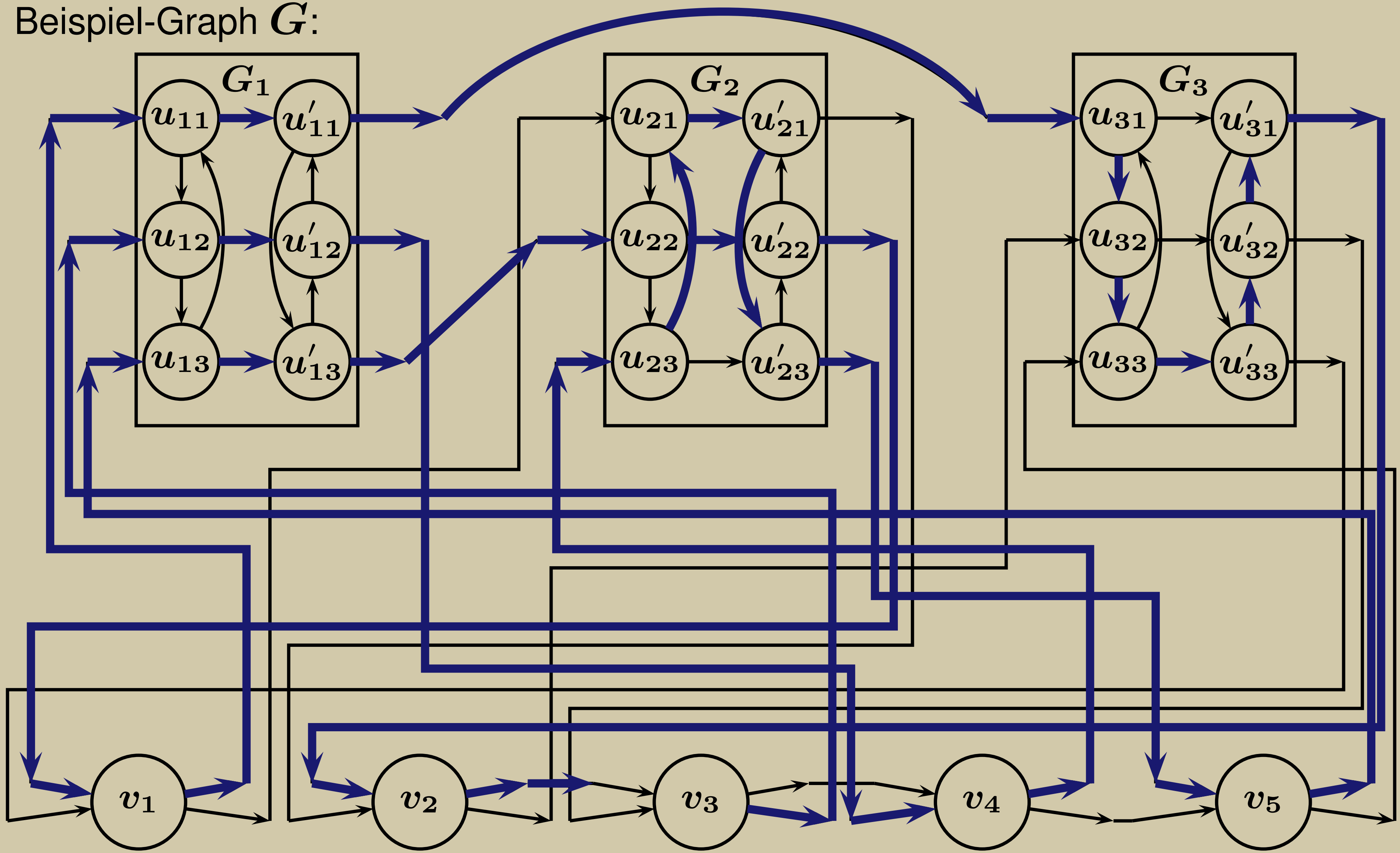


- Es gilt: für jede nicht-leere Menge S von Eingangskanten von G_i gibt es eine Menge von Wegen innerhalb G_i, die
  - G_i in der Menge S betritt,
  - alle Knoten aus G_i genau einmal besuchen,
  - und G_i in der zu S korrespondierenden Kantenmenge verlässt
- Und: jeder Hamiltonpfad, der G_i in u_{ij} betritt, muss G_i auch in u'_{ij} verlassen

- $f(\varphi) \stackrel{\text{def}}{=} G$

Beispiel (Forts.)

- Beispiel-Formel: $\varphi = (x_1 \vee \neg x_3 \vee x_5) \wedge (\neg x_1 \vee x_5 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee \neg x_5)$
  Belegung: $\theta(x_1) = 1$, $\theta(x_2) = 1$, $\theta(x_3) = 0$, $\theta(x_4) = 1$, $\theta(x_5) = 1$
- Beispiel-Graph G:



3-SAT $\leqslant_p$ HamiltonCycle (8/9)

Beweisskizze (Forts.)

(2) f ist in polynomieller Zeit berechenbar

(3) φ erfüllbar $\Rightarrow G$ hat gerichteten Hamilton-Kreis

- Sei θ erfüllende Belegung für φ
- Zur Konstruktion des Hamiltonkreises H wird zunächst für jedes Literal L mit $\theta(L) = 1$ der Literalweg zu L gewählt
- Da θ erfüllend ist, wird jedes G_i durch mindestens einen dieser Literalwege besucht
- Nach Konstruktion der G_i können Innenkanten so gewählt werden, dass alle Knoten genau einmal erreicht werden (und der Weg ist zusammenhängend)

➨ H ist Hamilton-Kreis

3-SAT $\leqslant_p$ HamiltonCycle **(9/9)**

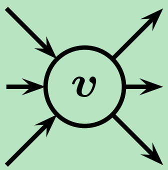
Beweisskizze (Forts.)

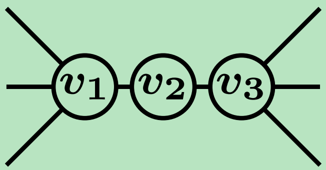
(4) G hat gerichteten Hamilton-Kreis $\Rightarrow \varphi$ ist erfüllbar

- Sei H ein Hamiltonkreis
- Falls die obere von v_i ausgehende Kante in H ist, setze $\theta(x_i) \stackrel{\text{def}}{=} 1$ sonst $\theta(x_i) \stackrel{\text{def}}{=} 0$
- Da jeder Klauselgraph durchlaufen wird, muss jede Klausel durch mindestens ein Literal wahr gemacht werden

➨ φ erfüllbar

- Noch zu zeigen: GHamiltonCycle $\leqslant_p$ HamiltonCycle

- Ersetze dazu 

durch 

- Dann gilt:
  Der gerichtete Graph hat einen Hamilton-Kreis $\Longleftrightarrow$
  der ungerichtete Graph hat einen Hamilton-Kreis

HamiltonCycle $\leqslant_p$ TSP

Satz 19.9

HamiltonCycle $\leqslant_p$ TSP

Beweisskizze

(1) Sei $G = (V, E)$ mit Knotenmenge $V = \{v_1, \ldots, v_n\}$

- Sei $f(G) \stackrel{\text{def}}{=} (s_1, \ldots, s_n, d, n)$, mit
  - $d(s_i, s_j) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{für } (v_i, v_j) \in E \\ 2 & \text{für } (v_i, v_j) \notin E \end{cases}$

(2) ✓

- Es ist leicht zu sehen:
  - Jede TSP-Reise hat mindestens die Gesamtentfernung n
  - Jeder Hamiltonkreis von G entspricht einer TSP-Reise der Gesamtstrecke n und umgekehrt

➡ (3), (4)

Inhalt

3-SAT $\leqslant_p$ SUBSETSUM $\leqslant_p$ KNAPSACK (1/6)

- Für den Nachweis, dass KNAPSACK **NP**-schwierig ist, verwenden wir das folgende Problem als Zwischenschritt

Definition (SUBSETSUM)

Gegeben: Menge S natürlicher Zahlen und eine Zielzahl k

Frage: Gibt es $T \subseteq S$ mit $\sum_{a \in T} a = k$?

- Im Gegensatz zu den bisherigen Reduktionen wird in der folgenden Reduktion die Kodierung von Informationen in Zahlen eine Rolle spielen

3-SAT $\leqslant_p$ SUBSETSUM $\leqslant_p$ KNAPSACK **(2/6)**

Proposition 19.10

3-SAT $\leqslant_p$ SUBSETSUM

Beweisskizze

(1) Sei $\varphi = K_1 \wedge \cdots \wedge K_m$, mit Variablen aus $\{x_1, \ldots, x_n\}$

- Wir „kodieren“ φ durch $(m+n)$-stellige Dezimalzahlen:
  - Die ersten m Stellen entsprechen den m Klauseln
    ☞ vorderer Teil
  - Die letzten n Stellen entsprechen den n Variablen
    ☞ hinterer Teil
  - $k \stackrel{\text{def}}{=} \underbrace{44\cdots44}_{m}\underbrace{11\cdots11}_{n}$
  - Intention: φ erfüllbar $\iff$
    es gibt eine Menge T mit $\sum_{a \in T} a = k$

- Für jedes Literal gibt es in S eine Zahl: a_i für x_i und b_i für $\neg x_i$
- Intention: $\theta(x_i) = 1$ entspricht der Wahl von a_i,
  $\theta(x_i) = 0$ entspricht der Wahl von b_i

Beweisskizze (Forts.)

- a_i, b_i haben eine 1 an der i-ten Position im hinteren Teil (und an allen anderen hinteren Positionen eine 0)
- An der j-ten Stelle des vorderen Teils hat a_i die Ziffer $\ell \overset{\text{def}}{\Leftrightarrow} x_i$ kommt ℓ-mal in K_j vor

☞ analog für b_i und $\neg x_i$

- Jede Wahrheitsbelegung θ korrespondiert also zu einer Zahlenmenge, deren Summe
  - im hinteren Teil an jeder Position eine 1, und
  - im vorderen Teil je Klausel-Position die Anzahl ihrer wahren Literale hat

➡ $\theta \models \varphi \iff$ die Summe hat im vorderen Teil an jeder Position eine Zahl zwischen 1 und 3

- Damit im vorderen Teil überall eine 4 (und damit genau k) erreicht werden kann, gibt es in S je Klausel von φ zwei weitere Zahlen:
  - y_j: vorne an Position j eine 1, sonst 0
  - z_j: vorne an Position j eine 2, sonst 0

Beispiel

- Beispiel-Formel

$$\varphi = (x_1 \vee \neg x_3 \vee x_5) \wedge (\neg x_1 \vee x_5 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee \neg x_5)$$

- $k = 444\,11111$

$$a_1 = 101\,10000 \qquad b_1 = 010\,10000$$
$$a_2 = 000\,01000 \qquad b_2 = 001\,01000$$
$$a_3 = 000\,00100 \qquad b_3 = 100\,00100$$
$$a_4 = 010\,00010 \qquad b_4 = 000\,00010$$
$$a_5 = 110\,00001 \qquad b_5 = 001\,00001$$

$$y_1 = 100\,00000$$
$$y_2 = 010\,00000$$
$$y_3 = 001\,00000$$
$$z_1 = 200\,00000$$
$$z_2 = 020\,00000$$
$$z_3 = 002\,00000$$

3-SAT $\leqslant_p$ SubsetSum $\leqslant_p$ Knapsack (4/6)

Vervollständigung des Beispiels

- Beispiel-Formel $\varphi = (x_1 \vee \neg x_3 \vee x_5) \wedge (\neg x_1 \vee x_5 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee \neg x_5)$
- Zahlen:

$$\begin{aligned} a_1 &= 101\,10000 & b_1 &= 010\,10000 & y_1 &= 100\,00000 \\ a_2 &= 000\,01000 & b_2 &= 001\,01000 & y_2 &= 010\,00000 \\ a_3 &= 000\,00100 & b_3 &= 100\,00100 & y_3 &= 001\,00000 \\ a_4 &= 010\,00010 & b_4 &= 000\,00010 & z_1 &= 200\,00000 \\ a_5 &= 110\,00001 & b_5 &= 001\,00001 & z_2 &= 020\,00000 \\ & & & & z_3 &= 002\,00000 \end{aligned}$$

- Die Wahrheitsbelegung
  - $\theta(x_1) = \theta(x_2) = \theta(x_4) = \theta(x_5) = 1$
  - $\theta(x_3) = 0$

  entspricht der Auswahl a_1, a_2, b_3, a_4, a_5
- Summe: $321\,11111$
- Wähle zusätzlich: y_1, z_2, y_3, z_3
- Gesamtsumme: $444\,11111$

3-SAT $\leqslant_p$ SUBSETSUM $\leqslant_p$ KNAPSACK (5/6)

Beweisskizze (Forts.)

(1) $f(\varphi) \stackrel{\text{def}}{=} (S, k)$, wobei S die Menge der Zahlen der a_i, b_i, y_j, z_j ist

(2) ✓

(3) φ ist erfüllbar $\Rightarrow f(\varphi)$ hat eine Lösung

- Sei θ eine Belegung, die φ wahr macht
- Falls $\theta(x_i) = 1$: wähle a_i
- Falls $\theta(x_i) = 0$: wähle b_i

➨ Die Summe der Zahlen ist im ersten Teil an jeder Position mindestens 1 und höchstens 3, im zweiten Teil genau 1

- Deshalb: falls in der bisherigen Summe im ersten Teil an der i-ten Stelle
  - eine 3 ist: wähle y_i
  - eine 2 ist: wähle z_i
  - eine 1 ist: wähle y_i und z_i

➨ die Gesamtsumme ergibt genau k

Beweisskizze (Forts.)

(4) $f(\varphi)$ hat eine Lösung $\Rightarrow \varphi$ ist erfüllbar

- Sei eine Menge $T \subseteq S$ mit Summe k gewählt

➨ Für jedes i ist entweder a_i oder b_i gewählt

➞ Definiere eine Variablenbelegung wie folgt:

$$\theta(x_i) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{falls } a_i \text{ gewählt ist} \\ 0 & \text{falls } b_i \text{ gewählt ist} \end{cases}$$

- Klar: die Summe der ausgewählten a_i und b_i hat an jeder vorderen Stelle mindestens 1

➨ Die Variablenbelegung macht jede einzelne Klausel und damit φ wahr

3-SAT $\leqslant_p$ SUBSETSUM $\leqslant_p$ KNAPSACK **(6/6)**

Proposition 19.11

SUBSETSUM $\leqslant_p$ KNAPSACK

Beweisskizze

- Sei S eine Menge von Zahlen und k eine Zielzahl

(1) Sei $f(S, k)$ die Eingabe für KNAPSACK mit
  - je einem Gegenstand mit Gewicht und Wert a, für jede Zahl $a \in S$, und
  - Gewichts- und Wertschranke k

- Es ist leicht zu sehen, dass
  (2) f in polynomieller Zeit berechnet werden kann, und
  (3,4) $(S, k) \in$ SUBSETSUM genau dann gilt, wenn $f(S, k) \in$ KNAPSACK

Gesamtergebnis

- Da wir für alle in diesem Kapitel betrachteten Probleme schon gezeigt haben, dass sie in **NP** sind, folgt aus den in diesem Kapitel gezeigten Resultaten der folgende Satz

19.12

- Die folgenden Probleme sind **NP**-vollständig:
  - 3-SAT
  - 3-COL
  - CLIQUE
  - HAMILTONCYCLE
  - TSP
  - SUBSETSUM
  - KNAPSACK

Zusammenfassung

- Es gibt Tausende von **NP**-vollständigen Problemen
- Das erste **NP**-vollständige Problem, SAT, haben wir durch den Satz von Cook gewonnen

- **NP**-Vollständigkeitsbeweise verwenden zumeist polynomielle Reduktionen **von** anderen, schon als **NP**-vollständig bekannten, Problemen
- Solche Reduktionsbeweise lassen sich in einer kanonischen Struktur darstellen

Literaturhinweise

Satz von Cook: Stephen A. Cook. The complexity of theorem-proving procedures. In *STOC*, pages 151–158, 1971

- Enthält viele weitere „klassische“ **NP**-vollständige Probleme

Andere NP-vollständige Probleme R.M. Karp. Reducibility among combinatorial problems. In R.E. Miller and J.W. Thatcher, editors, *Complexity of Computer Computa- tions*. Plenum, New York, 1972

Gerade Pfade: A. S. Lapaugh and C. H. Papadimitriou. The even-path problem for graphs and digraphs. *Networks*, 14:507–513, 1984

- Die Arbeit zeigt, dass es ein **NP**-vollständiges Problem ist, zu überprüfen, ob es einen Weg gerader Länge zwischen zwei Knoten eines gerichteten Graphen gibt