



Komplexitätstheorie

Wie schwierig ist es bestimmte Probleme zu lösen?

Erlaubt ein Maschinenmodelle schnellere Lösungen als ein anderes?

Beispiel: Geg. NEA A . Ist $L(A) = \Sigma^*$?



Terminologie und Konventionen

n : bezeichnet die „Eingabegröße“

Einheit ist noch festzulegen. Bits, Bandsymbole,
RAM-Maschinenworte.

(Entscheidungs)„Problem“: Eine zu erkennende Sprache.

Wir reden hier nur von entscheidbaren Problemen

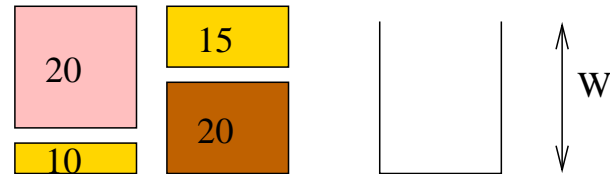


Komplexitätsmaße

- Zeit.** Hier die Hauptsache
- Platz
- Energieverbrauch
- Kommunikationsvolumen
- Plattenzugriffe
- Chipfläche für Hardwareimplementierung
- ...
- auch ziemlich abstrakte Sachen:
Zufallsbits, Richtungsänderungen einer TM, ...



Beispiel Rucksackproblem

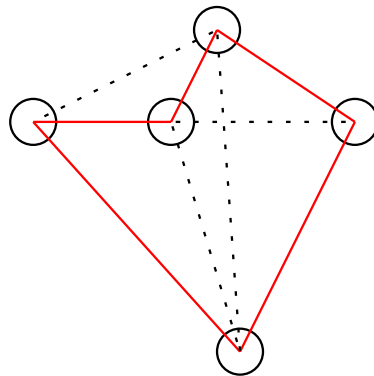


- n Gegenstände mit **Gewicht** $w_i \in \mathbb{N}$ und **profit** p_i
- Wähle eine Teilmenge **x** von Gegenständen
- so dass $\sum_{i \in \mathbf{x}} w_i \leq W$ und
- maximiere den Profit** $\sum_{i \in \mathbf{x}} p_i$



Beispiel: Handlungsreisendenproblem

[Der Handlungsreisende - wie er sein soll und was er zu thun hat, um Auftraege zu erhalten und eines gluecklichen Erfolgs in seinen Geschaeften gewiss zu sein - Von einem alten Commis-Voyageur, 1832].



Gegeben ein Graph $G = (V, V \times V)$, finde einen einfachen Kreis $C = (v_1, v_2, \dots, v_n, v_1)$ so dass $n = |V|$ und $\sum_{(u,v) \in C} d(u, v)$ minimiert wird.

Formulierung als Entscheidungsproblem: wie gehabt



Hamiltonkreisproblem

[Hamilton, William Rowan, Memorandum respecting a new system of roots of unity. Philosophical Magazine, 12 1856]

$$M := \{G = (V, E) : \exists C \subseteq E : |C| = |V|, C \text{ ist einfacher Kreis}\}$$

Codierung eines Graphen $G = (V, E)$ als Wort aus $\{0, 1, \#\}^*$:

OBdA, $V \subseteq \{0, 1\}^{\lceil \log |V| \rceil}$

Codierung $w_G := \prod_{(u,v) \in E} u\#v\#$



Steinerbäume

[C. F. Gauss 18??]

Gegeben graph $G = (V, E)$, mit positiven Kantengewichten

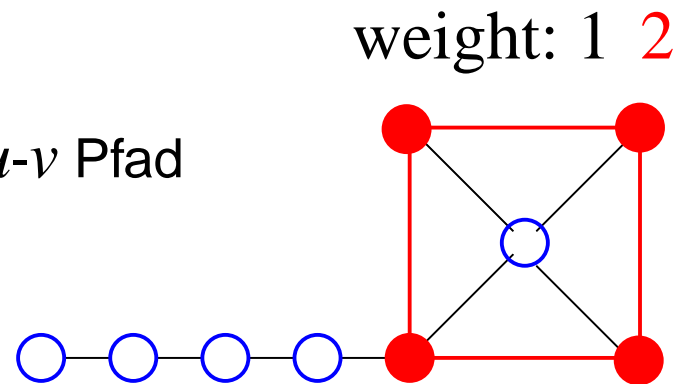
$$c : E \rightarrow \mathbb{R}_+$$

$V = R \cup F$, i.e., Pflichtknoten and Steinerknoten

finde einen Baum $T \subseteq E$ der mit minimalen Kosten alle Pflichtknoten verbindet.

$$\forall u, v \in R : T \text{ enthält } u\text{-}v \text{ Pfad}$$

DAS Netzwerkentwurfsproblem





Obere Schranken

Algorithmus angeben und analysieren.

Probleme:

- Ist die Analyse genau genug?
- Gibt es **bessere** Algorithmen?



Untere Schranken

kein Algorithmus kann eine bessere Lösung erreichen.

Triviale untere Schranke:

$$T = O(\text{inputSize} + \text{outputSize})$$

Problem: Kaum bessere Schranken bekannt!

Wir müssen Aussagen über **alle** Algorithmen machen!

Ausnahmen:

zusätzliche Annahmen: z.B. $\Omega(n \log n)$ für **vergleichsbasiertes**

Sortieren von n Elementen

sehr eingeschränkte Modelle: z.B. $\Omega(n^2)$ für EinbandTM-Akzeptor von

$L_P := \{w : w = w^R\}$. **Kommunikationskomplexitätsargument**

kollabiert bereits bei 2 Bändern



Untere Schranken: Lösungsansätze

- Vergrößerung: ignoriere kleinere Unterschiede
- Klassifiziere: Eine Menge von Problemen ist „ungefähr gleich schwierig“.

Wenn man für keins eine schnelle Lösung kennt, sind sie wahrscheinlich alle schwierig.



Eine Komplexitätsklasse

$\text{time}_M(w)$ = Anzahl der Rechenschritte einer TM M bei Eingabe von w

$\text{TIME}(f(n)) =$

$$\{L : \exists \text{TM } M : L(M) = L \wedge \forall w \in \Sigma_M^* : \text{time}_M(w) \leq f(|w|)\}.$$

Hier i.allg. **Mehrband**turingmaschinen.



Polynom

Eine Funktion $p : \mathbb{N} \rightarrow \mathbb{N}$ der Form

$$p(n) = a_k n^k + a_{k-1} n^{k-1} + \dots + a_1 n + a_0$$

mit $a_i, k \in \mathbb{N}$



Komplexitätsklasse P

$$\mathbf{P} := \bigcup_{\text{Polynom } p} \text{TIME}(p(n))$$

Analog definieren wir in polynomialer Zeit berechenbare **Funktionen**.



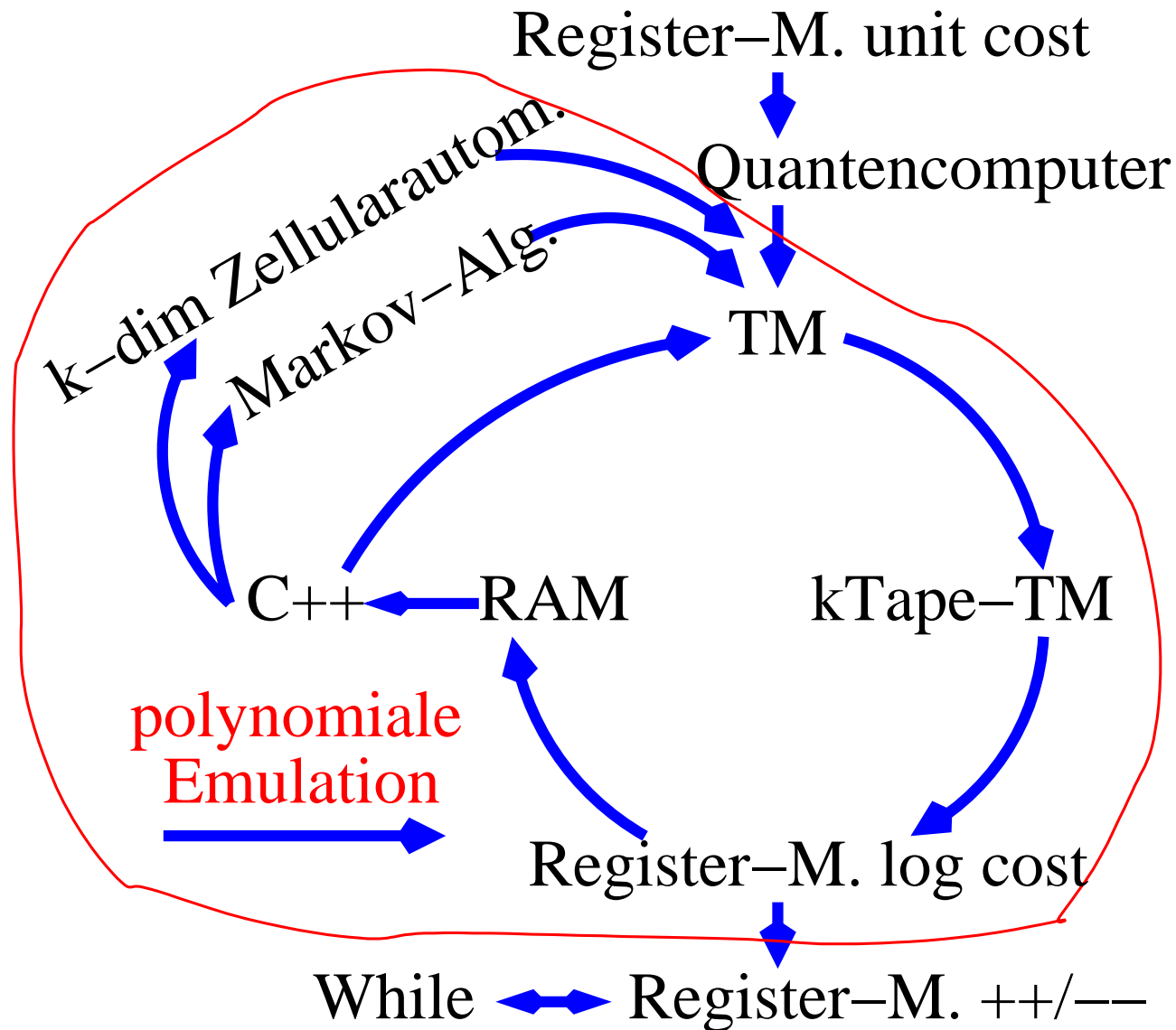
Komplexitätsklasse \mathbf{P}

$$\mathbf{P} := \bigcup_{\text{Polynom } p} \text{TIME}(p(n))$$

Interpretation/Vereinbarung: Probleme in \mathbf{P} sind **effizient** lösbar



P für verschiedene Maschinenmodelle





Transformation

Optimierungsproblem \rightarrow Entscheidungsproblem

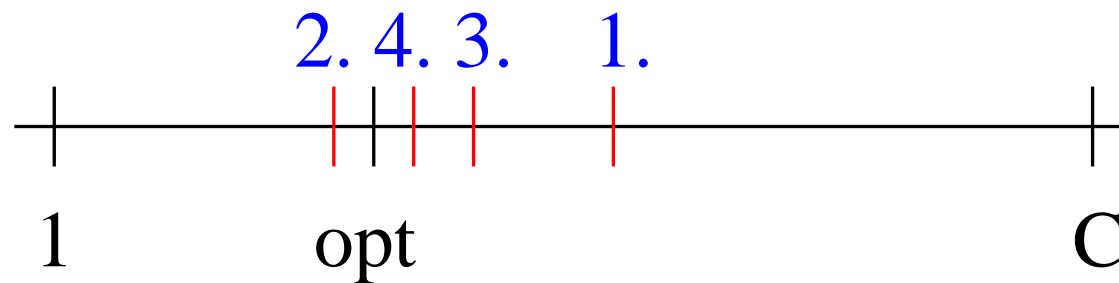
Annahme: Zielfunktionswert $\in 1..C$, ganzzahlig.

Binäre Suche: $\lceil \log C \rceil$ Entscheidungsprobleme lösen.

Polynomiell in n , wenn $\log C$ polynomiell in n .

Also wenn C polynomiell viele Bits hat.

Das ist aber bereits die **Ausgabekomplexität!**

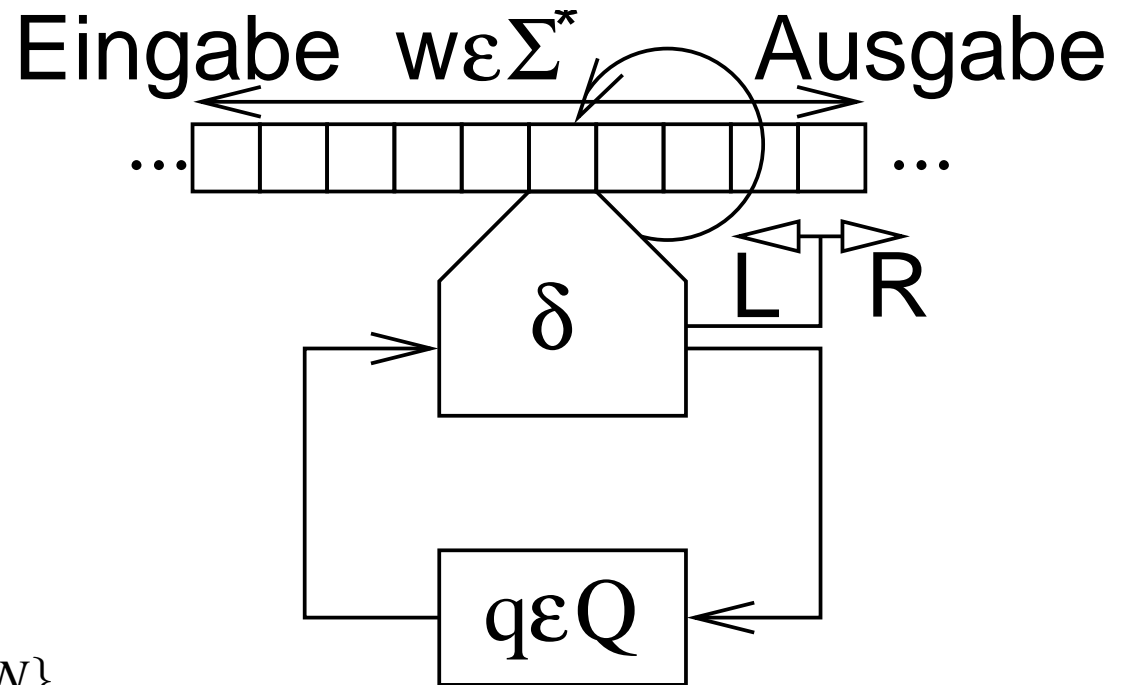




Nichtdeterministische Turingmaschinen (NTM)

$T = (Q, \Sigma, \Gamma, \delta, s, F)$:

- Q , Zustände
- Σ , Eingabealphabet
- Γ Bandalphabet,
□ $\sqcup \notin \Sigma$: Leersymbol,
 $\Sigma \cup \{\sqcup\} \subseteq \Gamma$
- $\delta : Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L, R, N\}}$,
Übergangsfunktion;
- $s \in Q$, Startzustand
- $F \subseteq Q$, Endzustände



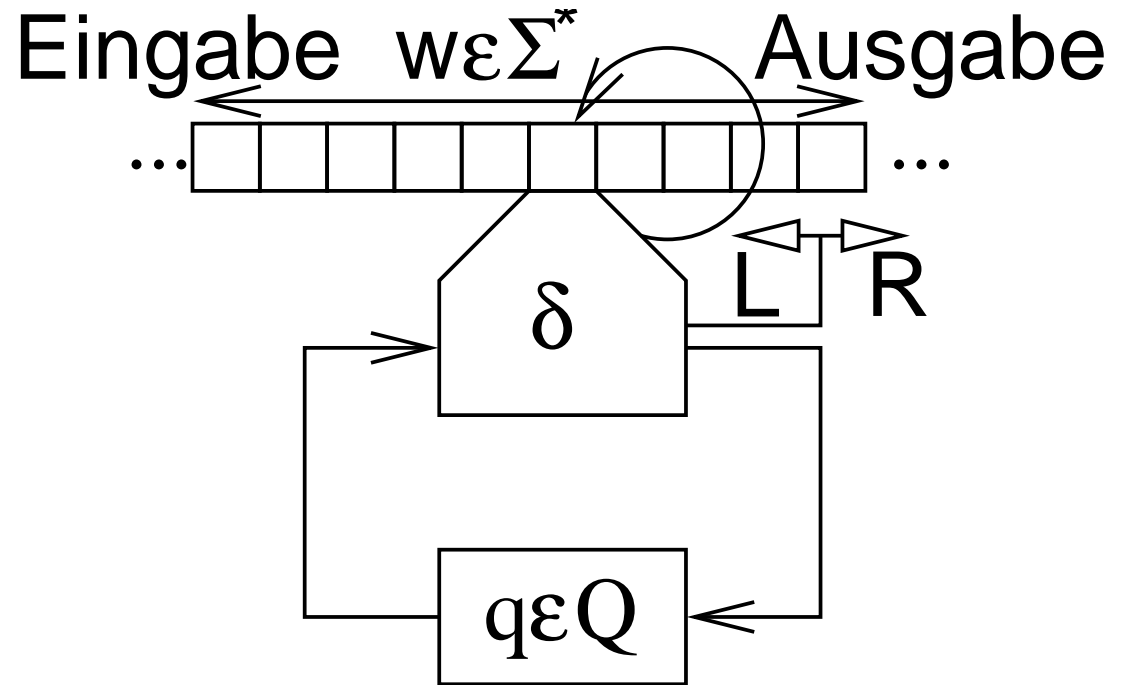
Nichtdeterministische Mehrbandturingmaschinen: analog



Nichtdeterministische Turingmaschinen (NTM)

$T = (Q, \Sigma, \Gamma, \delta, s, F)$:

- Q , Zustände
- Σ , Eingabealphabet
- Γ Bandalphabet,
□ $\sqcup \notin \Sigma$: Leersymbol,
□ $\Sigma \cup \{\sqcup\} \subseteq \Gamma$
- $\delta \subseteq Q \times \Gamma \times Q \times \Gamma \times \{L, R, N\}$,
Übergangsrelation;
- $s \in Q$, Startzustand
- $F \subseteq Q$, Endzustände





Funktionsweise NTM

mögliche Übergänge zwischen Konfigurationen

$$wa(q)bcv \xrightarrow{(q',b',N) \in \delta(q,b)} wa(q')b'cv$$

$$wa(q)bcv \xrightarrow{(q',b',L) \in \delta(q,b)} w(q')ab'cv$$

$$wa(q)bcv \xrightarrow{(q',b',R) \in \delta(q,b)} wab'(q')cv$$



Wann **hält** ein **NTM**?

T hält in Konfiguration $w(q)av$ gdw

$$\delta(q, a) = \{(q, a, N)\}.$$

Konvention:

$$\forall q \in F : \forall a \in \Gamma : \delta(q, a) = \{(q, a, N)\}$$



Graphinterpretation

$T = (Q, \Sigma, \Gamma, \delta, s, F)$ definiert
unendlichen Multigraphen

Knoten: **Konfigurationen** von T .

Kanten: von δ **zugelassene** Konfigurationsübergänge.

$w \in L(A) \Leftrightarrow \exists$ Pfad $P = (s)w \rightarrow \dots \rightarrow u(f)v : f \in F$

Unterschied **DTM** versus **NTM**:

δ **bestimmt** Konfigurationsübergänge versus

δ **läßt** Konfigurationsübergänge **zu**.



NTM als Akzeptor

$$T = (Q, \Sigma, \Gamma, \delta, s, F).$$

$L(T)$?

Definition:

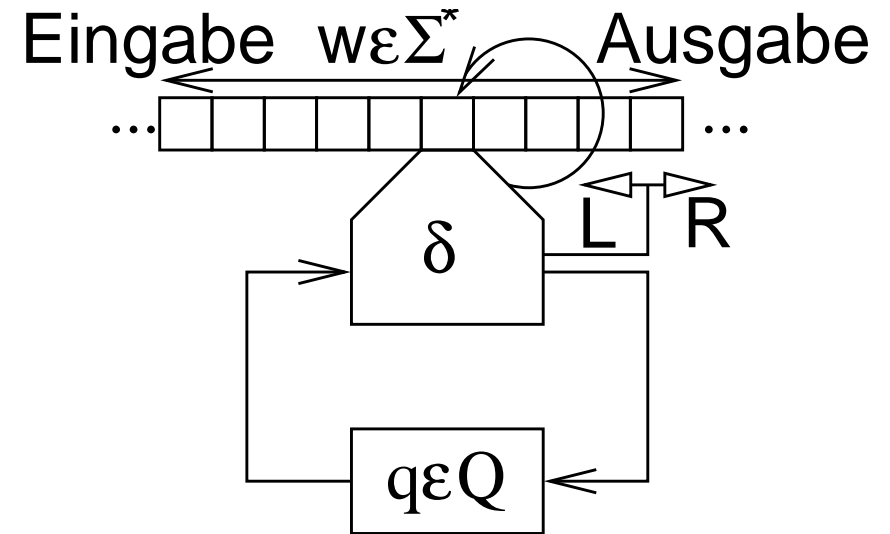
T akzeptiert $w \in \Sigma^*$ gdw

\exists Folge von (durch δ zugelassenen)

Konfigurationsübergangen

$(s)w \rightarrow \dots \rightarrow x(f)y$ mit $f \in F$.

$L(T) := \{w \in \Sigma^* : T \text{ akzeptiert } w\}$.





Noch eine Komplexitätsklasse

Sei M eine nichtdeterministische Turingmaschine

$$\text{ntime}_M(w) := \begin{cases} \min \{ |P| : P = (s)w \Rightarrow u(f)v, f \in F \} & \text{falls } w \in L(M) \\ 0 & \text{sonst} \end{cases}$$

$$\text{NTIME}(f(n)) := \{ L : \exists \text{NTM } M : L(M) = L \wedge \forall w \in \Sigma_M^* : \text{ntime}_M(w) \leq f(|w|) \}.$$



Komplexitätsklasse NP

$$\mathbf{NP} := \bigcup_{\text{Polynom } p} \mathbf{NTIME}(p(n))$$



Beispiel: Rucksackproblem

//Is there $x_1 \cdots x_n \in \{0, 1\}^n : \sum_i x_i w_i \leq W \wedge \sum_i x_i p_i \geq P$?

Procedure knapsack($\langle w_1, \dots, w_n \rangle, \langle p_1, \dots, p_n \rangle, W, P$)

for $i := 1$ **to** n **do** nondeterministically **guess** $x_i \in \{0, 1\}$

if $\sum_i x_i w_i > W$ **then** reject

if $\sum_i x_i p_i < P$ **then** reject

accept

Knapsack $\in NP$



Alternative Definition von NTIME: Orakel

Eine **DTM** M **Orakel-akzeptiert** $w \in \Sigma^*$ in Zeit $t = \text{otime}_M(w)$ gdw

$\exists o \in \Gamma^*$: M angesetzt auf $o(s)w$

hält nach t Zustandsübergängen

in einer Konfiguration $x(f)y$ mit $f \in F$.

Falls $w \in \Sigma^*$ von M **nicht** Orakel-akzeptiert wird, gilt $\text{otime}_M(w) := 0$.

$\text{OTIME}(f(n)) := \{L : \exists \text{DTM } M : L(M) = L \wedge \forall w \in \Sigma_M^* : \text{otime}_M(w) \leq f(|w|)\}$.



Äquivalenz von NTIME und OTIME

NTM emuliert DTM mit Orakel:

Nichtdet. Vorberechnung „rät“ Orakel \emptyset

Orakel-TM emuliert NTM:

Orakel gibt die nichtdet. Entscheidungen vor.



Die 1 000 000 \$ Frage

$$P = NP?$$

Eines von 7 mathematischen Problemen für die das
Clay Mathematics Institute
einen Preis von
1 000 000 US\$
ausgelobt hat.

Beobachtung: $P \subseteq NP$



Fragen wir das Publikum

100 Forscher wurden nach $\mathbf{P} = \mathbf{NP}$ gefragt

61: Nein

09: Ja

22: weiss nicht

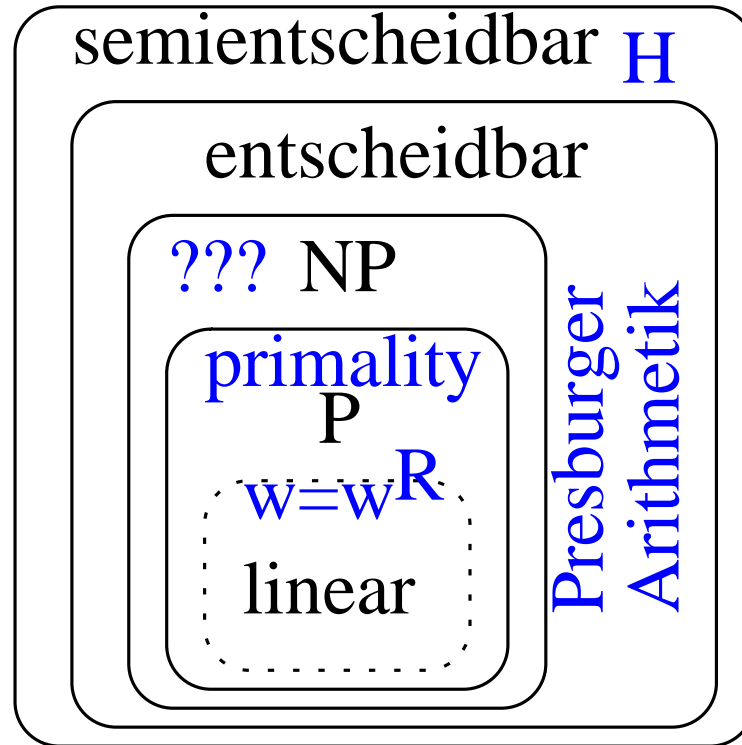
08: nicht beantwortbar

(unabhängig von gegenwärtig akzeptierten Axiomen.)

Warum ist diese Frage so wichtig?



Eine Komplexitätshierarchie





Presburger Arithmetik

Entscheidbarkeit von prädikatenlogischen Formeln erster Stufe mit folgenden Einschränkungen:

- Konstanten $0, 1$
- Variablen $x_i \in \mathbb{Z}$
- Funktionen $+, -$
- Relationen $<, =$
- logische Verknüpfungen \wedge, \vee, \neg
- Quantoren \exists, \forall



Polynomiale Reduzierbarkeit

Seien $A \subseteq \Sigma^*$ und $B \subseteq \Gamma^*$ Sprachen.

$A \leq_p B$ (A ist auf B polynomial reduzierbar)

\Leftrightarrow

$\exists f : \Sigma^* \rightarrow \Gamma^* : \forall w \in \Sigma^* : w \in A \Leftrightarrow f(w) \in B$

wobei f in **polynomialer** Zeit berechenbar ist.



Beispiel

Satz: $\text{HamiltonCycle} \leq_p \text{TSP}$

Beweis:

Sei $G = (V, E)$ beliebiger ungerichteter Graph.

Definiere $d(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E \\ 1 + \alpha & \text{else} \end{cases}$

Dann und nur dann, wenn G einen Hamiltonkreis hat gilt

\exists TSP Tour mit **Kosten n**

(sonst optimale **Kosten $\geq n + \alpha$**)



Lemma: $A \leq_p B, B \in \mathbf{P} \rightarrow A \in \mathbf{P}$

Beweis:

Es gelte $A \leq_p B$ mittels Funktion f .

Sei M_f eine TM, die f berechnet mit polynomialer Zeitschranke p .

Ferner sei $B \in \mathbf{P}$ mit polynomialer Zeitschranke q mittels TM M_B .

Betrachte **Hintereinanderausführung**sTM $M_A := (M_f; M_B)$.

M_A entscheidet A .

Rechenzeit bei Eingabe von w :

$$p(|w|) + q(|f(w)|) \leq p(|w|) + q(p(|w|))$$

Das ist polynomial in $|w| = n$



Lemma: $A \leq_p B, B \in \mathbf{NP} \rightarrow A \in \mathbf{NP}$

Beweis: analog



NP-harte und NP-vollständige Probleme

A ist **NP-hart**: $\Leftrightarrow \forall L \in \mathbf{NP} : L \leq_p A$

A ist **NP-vollständig**: $\Leftrightarrow A$ ist **NP-hart** und $A \in \mathbf{NP}$.



Ein einfacher Weg zu Ruhm und Reichtum?

Satz: Sei A **NP-vollständig**. Dann gilt: $A \in \mathbf{P} \Leftrightarrow \mathbf{P} = \mathbf{NP}$

Beweis:

Fall $\mathbf{P} = \mathbf{NP}$:

$A \in \mathbf{NP} = \mathbf{P}$ also insbesondere $A \in \mathbf{P}$

Fall $A \in \mathbf{P}$:

Sei $L \in \mathbf{NP}$ beliebig.

Da A **NP-hart** ist gilt $L \leq_p A$

und wegen $A \in \mathbf{P}$ folgt $L \in \mathbf{P}$

Also $\mathbf{P} = \mathbf{NP}$



SAT: Das Erfüllbarkeitsproblem

gegeben: Formel F der **Aussagenlogik**

($\wedge \vee \neg()$, Variablen)

gefragt: ist F erfüllbar?, d.h.

\exists Belegung der Var. mit $\{0, 1\}$ so dass $\text{Wahrheitswert}(F) = 1$?

Formaler:

$\text{SAT} := \{\text{code}(F) \in \Sigma^* : F \text{ ist erfüllbare Formel der Aussagenlogik}\}$,

code ist eine geeignete Codierung von Formeln als Zeichenkette.



Satz: [Cook 1971] und [Levin 1971] SAT ist **NP**-vollständig.



Beweis von $\text{SAT} \in \text{NP}$

Procedure satisfiable(F)

guess values $\in \{0, 1\}$ for all variables in F

substitute variables by their values

evaluate the resulting truth value v of F

if $v = 1$ **then** accept



Beweis dass SAT NP-hart ist.

Sei

$L \in \mathbf{NP}$ beliebig,

$M = (\{1, \dots, k\}, \Sigma, \{1, \dots, \ell\}, \delta, 1, H)$ NTM mit $L(M) = L$,

($\sqcup = \ell$)

$p(n)$ poly. Zeitschranke für Akzeption von L durch M ,

$w = w_1 w_2 \cdots w_n \in \Sigma^*$ beliebige Eingabe.

Ansatz: Wir geben Formel F **polynomieller Größe** an, so dass

$$w \in L \Leftrightarrow F \text{ ist erfüllbar}$$



Variablen für F

- $\text{zust}_{tz} = 1 \Leftrightarrow$ nach t Schritten ist M in Zustand z
- $\text{pos}_{ti} = 1 \Leftrightarrow$ nach t Schritten ist M an Bandposition i
- $\text{band}_{tia} = 1 \Leftrightarrow$ nach t Schritten ist Bandposition i mit a beschriftet

Man beachte, dass

$$0 \leq t \leq p(n) \text{ sowie}$$
$$-p(n) \leq i \leq p(n)$$

Es gibt also nur $O(p(n)^2)$ Variablen



Es kann nur einen geben

$$G(v_1, \dots, v_m) = 1 \Leftrightarrow \text{Genau ein } v_i = 1.$$

Implementierung:

$$G(v_1, \dots, v_m) =$$

$$v_1 \wedge \neg v_2 \wedge \neg v_3 \wedge \dots \wedge \neg v_m \vee$$

$$\neg v_1 \wedge v_2 \wedge \neg v_3 \wedge \dots \wedge \neg v_m \vee$$

$$\neg v_1 \wedge \neg v_2 \wedge v_3 \wedge \dots \wedge \neg v_m \vee$$

...

$$\neg v_1 \wedge \neg v_2 \wedge \neg v_3 \wedge \dots \wedge v_m$$

Größe: $O(m^2)$



Die Architektur von $F =$

$R \wedge$ „Die Var. beschreiben **stets** eine **Konfiguration**“

$A \wedge$ „Die Var. beschreiben **anfangs** die Konfiguration $(1)w$ “

$\ddot{U}_1 \wedge$ „An der **Kopfposition** entsprechen **Änderungen** δ “

$\ddot{U}_2 \wedge$ „An **Nichtkopfpositionen** ändert sich nie etwas“

E „ M **akzeptiert** w nach $\leq p(n)$ Schritten“



Randbedingung

„Die Variablen beschreiben stets eine Konfiguration“

$$R = \bigwedge_t G(\text{zust}_{t1}, \dots, \text{zust}_{tk}) \wedge \\ G(\text{pos}_{t,-p(n)}, \dots, \text{pos}_{tp(n)}) \wedge \\ \bigwedge_i G(\text{band}_{ti1}, \dots, \text{band}_{til})$$

Größe $O(p(n)k^2 + p(n)p(n)^2 + p(n)p(n)\ell^2) = O(p(n)^3)$



Anfangsbedingung

„Die Var. beschreiben **anfangs** die Konfiguration **(1)w**“

$$A = \text{zust}_{01} \wedge \text{pos}_{01} \wedge$$

$$\bigwedge_{j=1}^n \text{band}_{0jw_j} \wedge$$

$$\bigwedge_{j=-p(n)}^0 \text{band}_{0j\sqcup} \wedge$$

$$\bigwedge_{j=n+1}^{p(n)} \text{band}_{0j\sqcup}$$

$$\text{Gesamt } O(1 + 1 + n + p(n) + p(n) - n) = O(p(n))$$



Übergangsbedingung $\ddot{U}_1, t \rightarrow t + 1$

„An der **Kopfposition** entsprechen Änderungen δ “

$$\ddot{U}_1 = \bigwedge_{t,z,i,a} \left(\text{zust}_{tz} \wedge \text{pos}_{ti} \wedge \text{band}_{tia} \right) \\ \rightarrow \bigvee_{\{(z',a',y) \in \delta(z,a)\}} \left(\text{zust}_{t+1,z'} \wedge \text{pos}_{t+1,i+y} \wedge \text{band}_{t+1,ia'} \right)$$

wobei Kopfbewegungen als Zahlen interpretiert werden,

$$L = -1, N = 0, R = +1.$$

$$\text{Größe } O((p(n) \cdot k \cdot p(n) \cdot \ell) \cdot (k \cdot \ell \cdot 3)) = O(p(n)^2)$$



Übergangsbedingung $\ddot{U}_2, t \rightarrow t + 1$

„An **Nichtkopfpositionen** ändert sich nie etwas“

$$\ddot{U}_2 = \bigwedge_{t,i,a} ((\neg \text{pos}_{ti} \wedge \text{band}_{tia}) \rightarrow (\text{band}_{t+1,i,a}))$$

Größe $O((p(n) \cdot p(n) \cdot \ell)) = O(p(n)^2) = O(p(n)^2)$



Endebedingung E

„ M akzeptiert w nach $\leq p(n)$ Schritten“

$$E = \bigvee_{z \in H} \text{zust}_{p(n), z}$$

Gesamtgröße $O(1)$



Beweis $w \in L \rightarrow F$ ist erfüllbar

$w \in L$

$\rightarrow \exists$ Rechenpfad $P = (1)w \Rightarrow u(h)v$ mit $h \in H$ von M , $|P| = p(n)$

P definiert Variablenbelegungen für die F den Wahrheitswert 1 erhält.

$\rightarrow F$ ist erfüllbar.



Beweis F erfüllbar $\rightarrow w \in L$

F erfüllbar durch bestimmte Belegung der Variablen \rightarrow

R wird erfüllt \rightarrow

Die Var. beschreiben zu jedem Zeitpunkt eine Konfiguration

A wird erfüllt \rightarrow

Die Var. beschreiben anfangs die Konfiguration $(1)w$

\ddot{U}_1, \ddot{U}_2 erfüllt \rightarrow

von t nach $t + 1$ geschieht jeweils ein von δ erlaubter Konfigurationsübergang

E wird erfüllt \rightarrow

die Rechnung endet in einem akzeptierenden Zustand.

$\rightarrow w \in L$



Gesamtgröße von F

R Randbedingung

$$O(p(n)^3)$$

A Anfangsbedingung

$$O(p(n))$$

\ddot{U}_1 Übergangsbedingung 1

$$O(p(n)^2)$$

\ddot{U}_2 Übergangsbedingung 2

$$O(p(n)^2)$$

E Endebedingung „ M akzeptiert w nach $\leq p(n)$ Schritten“

$$O(1)$$

Insgesamt $O(p(n)^3)$.

Das ist wieder **polynomiell**

und kann in dieser Zeit automatisch „**hingeschrieben**“ werden.



Satz: $\mathbf{NP} \subseteq \bigcup_p \text{PolynomTIME}(2^{p(n)})$

Beweis: Sei $p(n)$ die nichtdet. Zeitschranke für eine Sprache $L \in \mathbf{NP}$.

Man nehme einen **deterministischen Algorithmus**, der alle **Kombinationen** der $\leq p(n)$ nichtdet. Entscheidungen **systematisch durchprobiert**.

Es gibt höchstens $2^{p(n)}$ solche Kombinationen.



Weitere NP-vollständige Probleme

Beobachtung: \leq_p ist transitiv, d.h.,

$$\forall L, L', L'' : L \leq_p L' \wedge L' \leq_p L'' \rightarrow L \leq_p L''.$$

Lemma: $L \in \mathbf{NP} \wedge \mathbf{SAT} \leq_p L \rightarrow L$ ist **NP**-vollständig.

Beweis: z.Z. $\forall L' \in \mathbf{NP} : L' \leq_p L$.

SAT ist **NP**-vollständig also $L' \leq_p \mathbf{SAT}$.

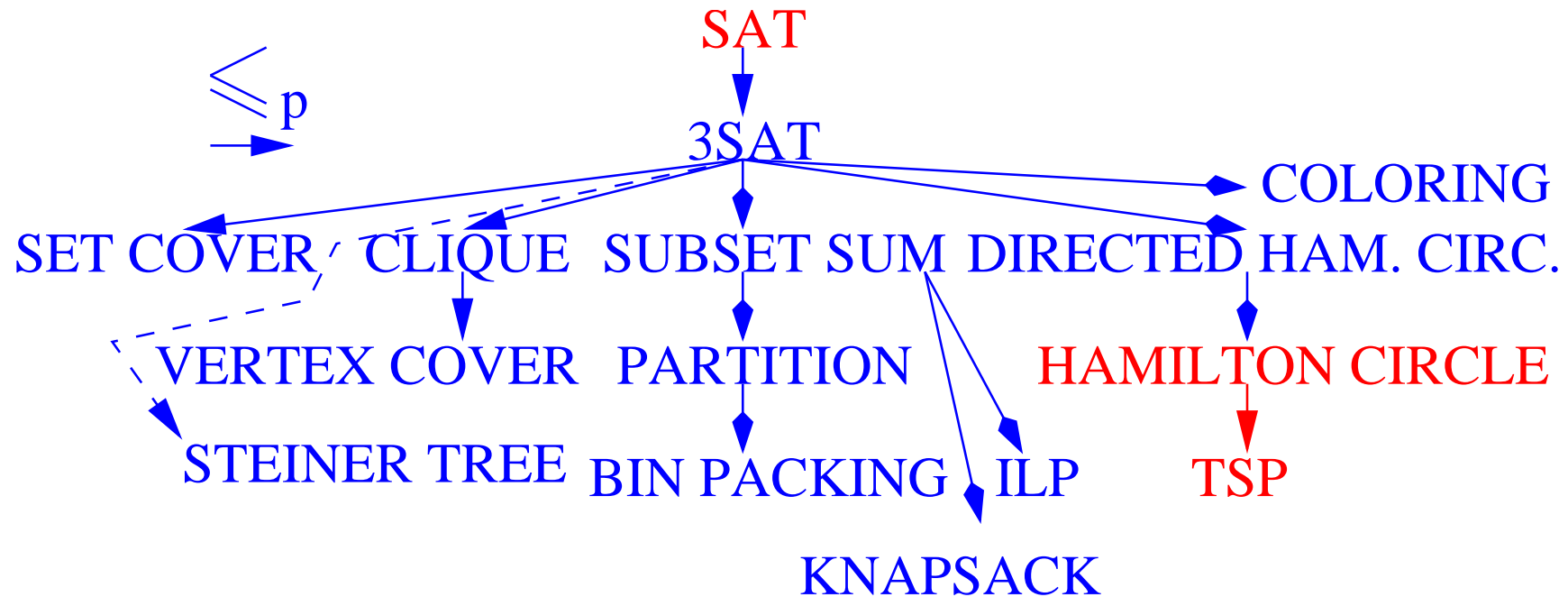
Also folgt wegen der Assoziativität von \leq_p ,

$$L' \leq_p \mathbf{SAT} \leq_p L.$$

qed



Weitere NP-vollständige Probleme





3SAT (KNF)

Gegeben: Aussagenlogische Formel in **konjunktiver Normalform**,
max. **3 Literale pro Klausel**

Gesucht: Ist F erfüllbar?

Beispiel: $(x \vee \neg y \vee z)(\neg u \vee x \vee z)(\neg x)$

Variable: x, y, z, \dots

Literal: Variable, \neg Variable

Klausel: **Literal** $\vee \dots \vee$ **Literal**

KNF Formel: (Klausel) $\cdot \dots \cdot$ (Klausel)



Satz: 3SAT ist NP-vollständig

Beweis:

Da SAT in **NP** ist, genügt es zu zeigen, dass 3SAT **NP-hart** ist.

Falle: In KNF bringen hilft überhaupt nicht.

- KNF kann exponentiell größer sein
- Man kriegt nicht notwendig 3KNF



Beweis „3SAT ist NP-hart“

Wir entwickeln einen polynomialen Alg.,
der 3KNF-Formel F in erfüllbarkeitsäquivalente Formel F' umformt,
d.h.,

$$F \text{ erfüllbar} \Leftrightarrow F' \text{ erfüllbar}$$

Ansatz: Folge von erfüllbarkeitsäquivalenten Transformationen.

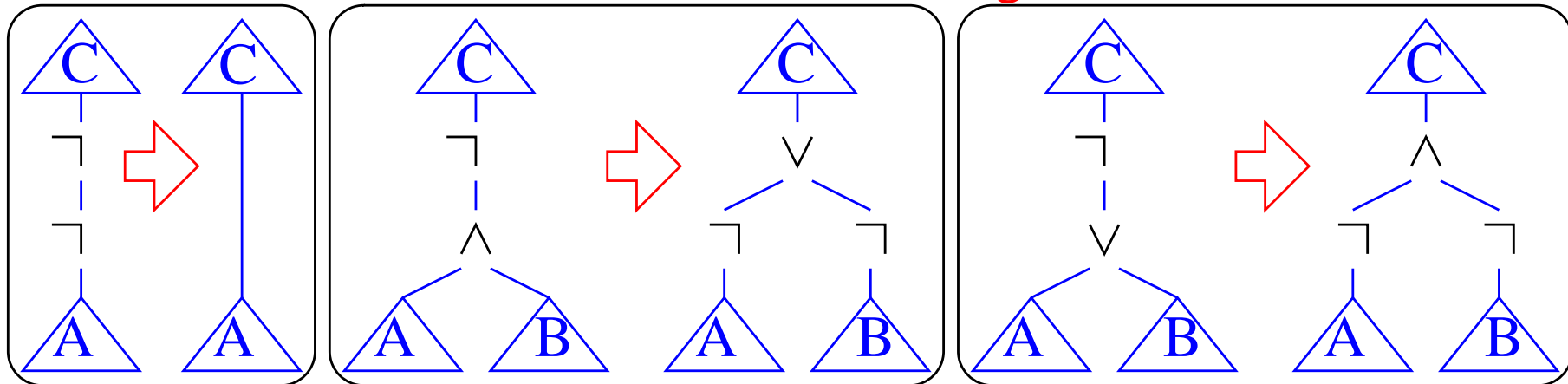
Wir betrachten Formeln als Bäume mit max. Grad zwei.

Blätter: Variablen x (oder Literale $\bar{x} = \neg x$) Weitere Knoten: \wedge, \vee, \neg



Negationen in die Blätter drücken

De Morgan

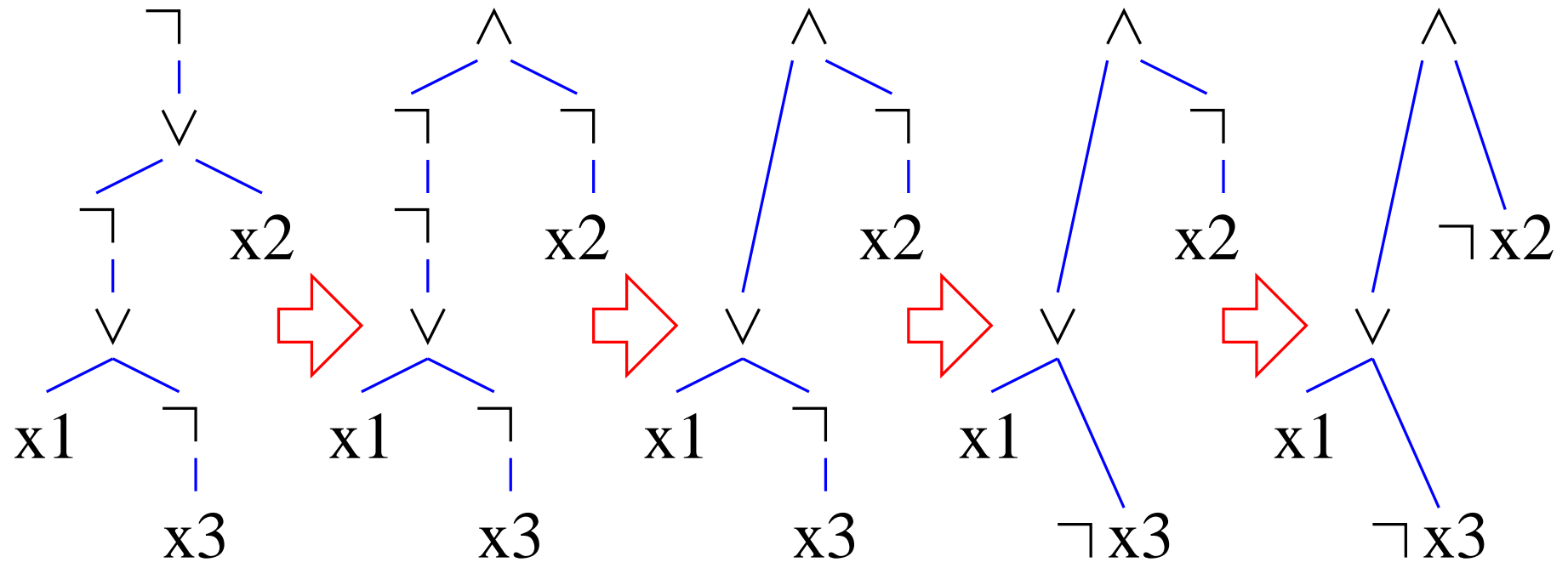


Mit Traversierung von oben nach unten geht das in linearer Zeit (RAM).

Nachbedingung: Formeln lassen sich als **binäre Bäume** mit inneren Knoten \vee , \wedge und **Literalen als Blättern** auffassen.



Beispiel





Negationen in die Blätter drücken

$\text{normalize}(\neg x)$ **return** $\text{negNormalize}(x)$

$\text{normalize}(x \wedge y)$ **return** $\text{normalize}(x) \wedge \text{normalize}(y)$

$\text{normalize}(x \vee y)$ **return** $\text{normalize}(x) \vee \text{normalize}(y)$

$\text{normalize}(v)$ **return** v

$\text{negNormalize}(\neg x)$ **return** $\text{normalize}(x)$

$\text{negNormalize}(x \wedge y)$ **return** $\text{negNormalize}(x) \vee \text{negNormalize}(y)$

$\text{negNormalize}(x \vee y)$ **return** $\text{negNormalize}(x) \wedge \text{negNormalize}(y)$

$\text{negNormalize}(v)$ **return** $\neg v$



Nichtblattknoten \rightarrow Neue Variablen

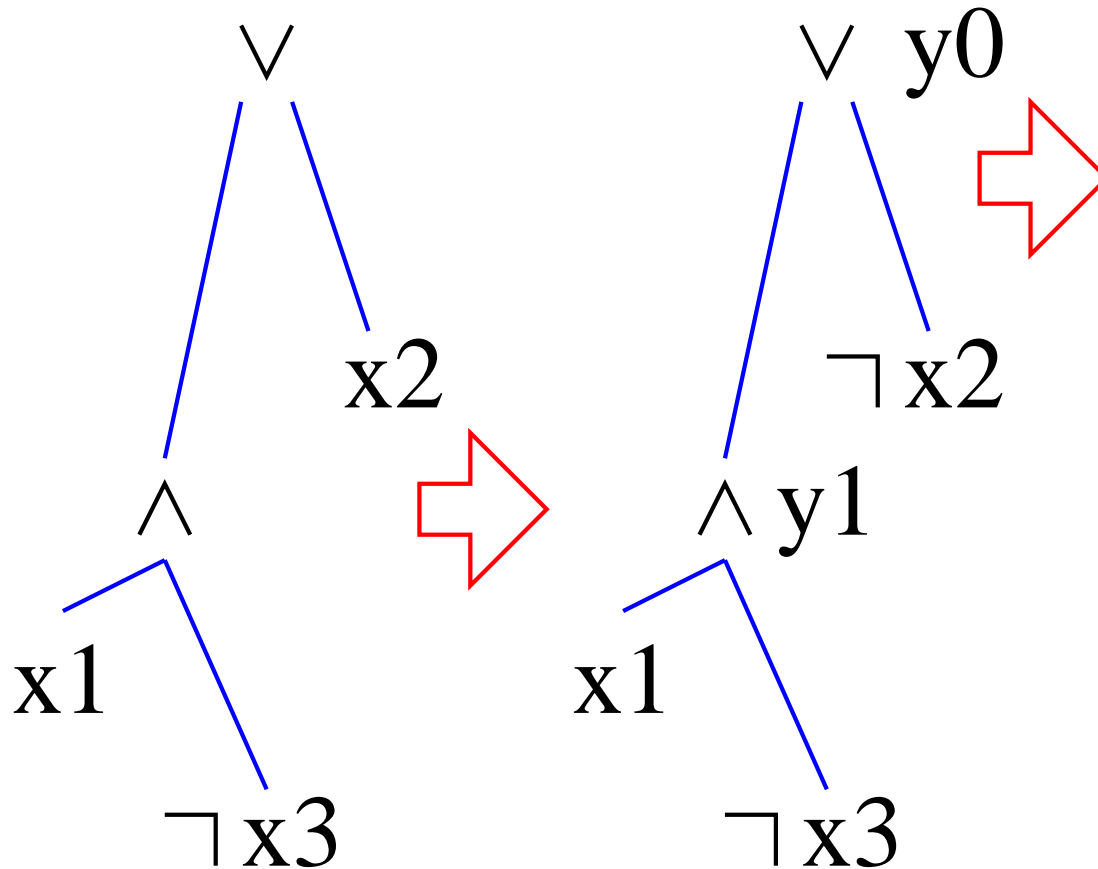
Ordne jedem \wedge, \vee eine neue Variable zu.

Sei y_0 das Literal der Wurzel.

$$F_1 := (y_0) \wedge \left(\bigwedge_{\substack{v \\ y \wedge z}} v \leftrightarrow (y \wedge z) \right) \wedge \left(\bigwedge_{\substack{v \\ y \vee z}} v \leftrightarrow (y \vee z) \right)$$



Beispiel



$$(y0)$$
$$(y0 \leftrightarrow (y1 \vee \neg x2))$$
$$(y1 \leftrightarrow (x1 \wedge \neg x3))$$



Erfüllbarkeitsäquivalenz von F_1

Beweis F erfüllbar $\rightarrow F_1$ erfüllbar:

Betrachte **erfüllende Variablenbelegung** für F .

Übernehme diese als **partielle Belegung** für F_1 .

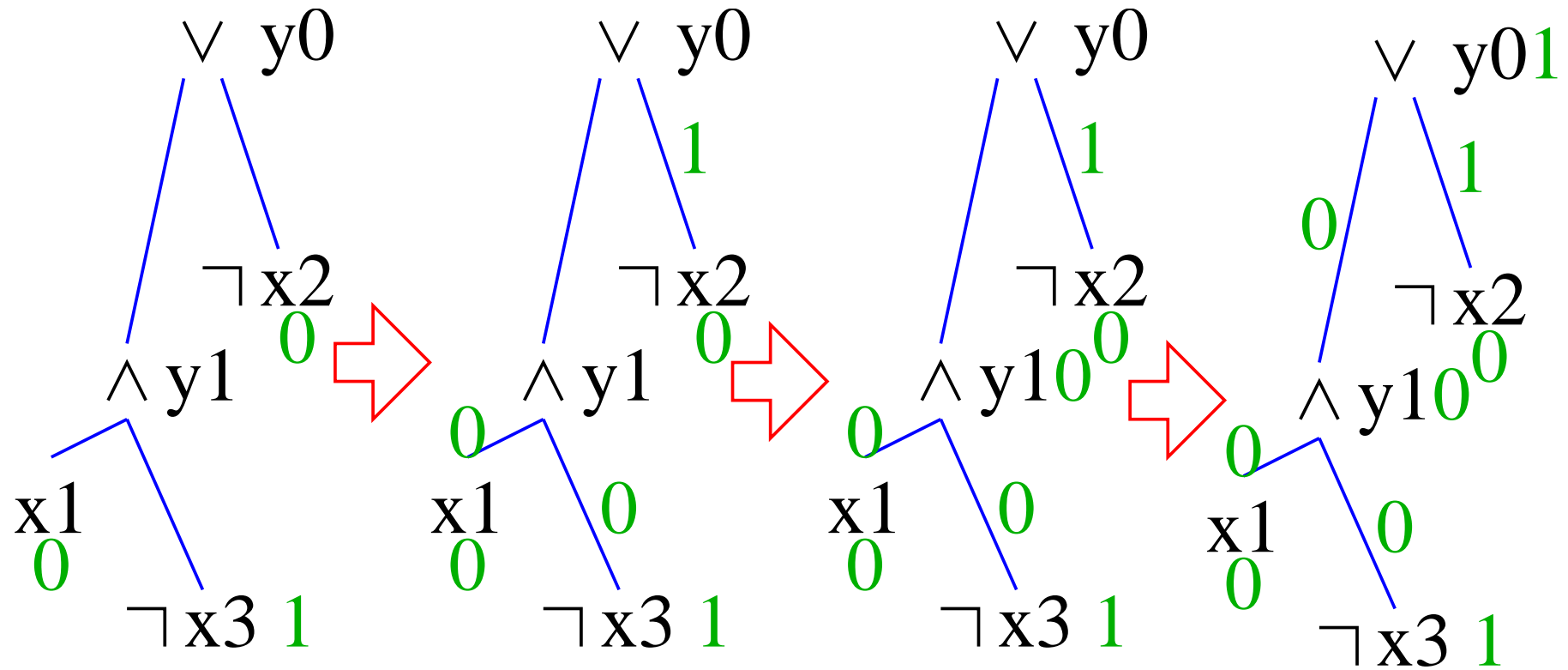
Werte die Formel nun bottom up aus.

Übernehme die sich ergebenden **Wahrheitswerte** des Teilbaum mit

Wurzelvariable ν für die **Belegung von ν** .



Beispiel





Erfüllbarkeitsäquivalenz von F_1

Beweis F_1 erfüllbar $\rightarrow F$ erfüllbar:

Betrachte **erfüllende Variablenbelegung** für F_1 .

Übernehme die **Belegungen der Blätter** für F .

Nichtblattknoten von F haben die gleichen Wahrheitswerte wie die entsprechenden **Variablen in F_1** .

y_0 ist mit 1 belegt.

Also ist F insgesamt **wahr**.



$F_1 \rightsquigarrow \mathbf{3KNF}$

Baue KNF für jede Teilformel:

$$a \leftrightarrow (b \vee c) \rightsquigarrow (a \vee \neg b)(\neg a \vee b \vee c)(a \vee \neg c)$$

$$a \leftrightarrow (b \wedge c) \rightsquigarrow (\neg a \vee b)(\neg a \vee c)(a \vee \neg b \vee \neg c)$$

a	b	c	$b \vee c$	$b \wedge c$	$a \leftrightarrow (b \vee c)$	$a \leftrightarrow (b \wedge c)$
0	0	0	0	0	1	1
0	0	1	1	0	0	1
0	1	0	1	0	0	1
0	1	1	1	1	0	0
1	0	0	0	0	0	0
1	0	1	1	0	1	0
1	1	0	1	0	1	0
1	1	1	1	1	1	1



Exkurs: $2SAT \in P$

Gegeben: Aussagenlogische Formel in konjunktiver Normalform,
max. 2 Literale pro Klausel.

Gesucht: Ist F erfüllbar?

Definiere $\bar{x} := \neg x$, $\neg \bar{x} := x$.

Graph $G = (V, E)$, $V := \{x, \bar{x} : x \in F\}$,
 $E := \{(\bar{A}, B), (\bar{B}, A) : (A \vee B) \in F\}$

Beobachtung: F erfüllbar gdw.

$\neg \exists$ Kreis C in G , Variable $x : x \in C \wedge \bar{x} \in C$.

Diese Bedingung kann in linearer Zeit überprüft werden.

(Starke Zusammenhangskomponenten.)



SET COVER

Gegeben:

Grundmenge M

Mengensystem $\mathcal{T} \subseteq 2^M$

Parameter n

Frage:

Gibt es Auswahl von $T_1 \in \mathcal{T}, \dots, T_n \in \mathcal{T}$ mit?

$T_1 \cup \dots \cup T_n = M$?

Beispiel:

$\{1, 2, 3, 4, 5\}$

$\{\{1, 3\}, \{1, 2\}, \{2, 4, 5\}\}$

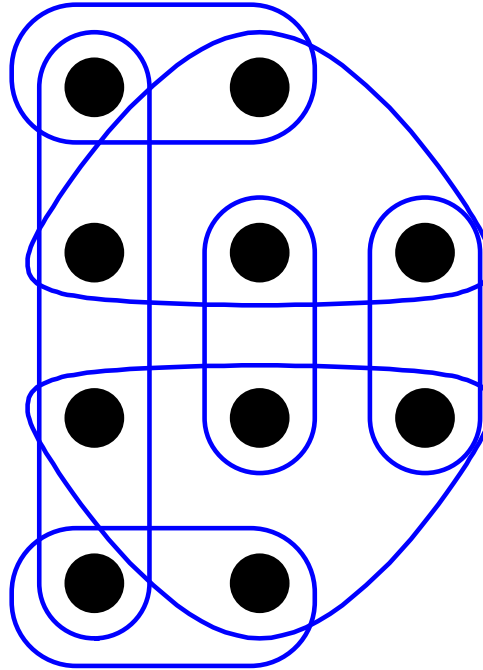
2

Ja:

$\{1, 3\}, \{2, 4, 5\}$



Set Cover Beispiel



Kleinstes n : 3



Satz: SET COVER ist NP-vollständig

Beweis von SET COVER \in NP:

Wähle nichtdet. n Mengen $T_1 \in \mathcal{T}, \dots, T_n \in \mathcal{T}$

Bilde Vereinigungsmenge $M' = T_1 \cup \dots \cup T_n$

Prüfe ob $M = M'$



Beweis „SET COVER ist NP-hart“

Wir zeigen $3SAT \leq_p SET\ COVER$.

Sei $F = K_1 \wedge \dots \wedge K_m$ eine KNF-Formel mit Variablen x_1, \dots, x_n .

Wähle $M = \{1, \dots, m + n\}$.

$T_i := \{j : x_i \text{ kommt in Klausel } K_j \text{ vor}\} \cup \{m + i\}$

$T'_i := \{j : \neg x_i \text{ kommt in Klausel } K_j \text{ vor}\} \cup \{m + i\}$

Setze $\mathcal{T} := \{T_1, \dots, T_n, T'_1, \dots, T'_n\}$.

Parameter n .

Zu zeigen: Die Set Cover Instanz (M, \mathcal{T}, n) ist lösbar gdw. F erfüllbar.



Beispiel

$$F = (x_1 \vee x_2 \vee x_3)(x_2 \vee \neg x_3 \vee x_4)(\neg x_3 \vee \neg x_4 \vee \neg x_1)$$

$$n = 4, m = 3.$$

$$T_1 = \{1, 4\},$$

$$\{3, 4\} = T'_1$$

$$T_2 = \{1, 2, 5\},$$

$$\{5\} = T'_2$$

$$T_3 = \{1, 6\},$$

$$\{2, 3, 6\} = T'_3$$

$$T_4 = \{2, 7\},$$

$$\{3, 7\} = T'_4$$

$$T_1 \cup T_2 \cup T'_3 \cup T_4 = \{1, 2, 3, 4, 5, 6, 7\}$$

$$\text{Also } x_1 = x_2 = x_4 = 1, x_3 = 0$$



Beweis F erfüllbar $\rightarrow (M, \mathcal{T}, n)$ lösbar.

Wähle $M = \{1, \dots, m + n\}$.

$T_i := \{j : x_i \text{ kommt in Klausel } K_j \text{ vor}\} \cup \{m + i\}$

$T'_i := \{j : \neg x_i \text{ kommt in Klausel } K_j \text{ vor}\} \cup \{m + i\}$

für $i = 1..n$

Wähle T_i falls $x_i = 1$ für $i = 1..n$.

Wähle T'_i falls $x_i = 0$.

$\{1, \dots, m\}$ wird abgedeckt, weil in jeder Klausel mindestens ein Literal erfüllt ist.

$\{m + 1, \dots, m + n\}$ wird abgedeckt, weil für jede Variable T_i oder T'_i ausgewählt wurde.



Beweis (M, \mathcal{T}, n) lösbar $\rightarrow F$ erfüllbar

$$M = \{1, \dots, m + n\}.$$

$$T_i := \{j : x_i \text{ kommt in Klausel } K_j \text{ vor}\} \cup \{m + i\}$$

$$T'_i := \{j : \neg x_i \text{ kommt in Klausel } K_j \text{ vor}\} \cup \{m + i\}$$

$\{m + 1, \dots, m + n\}$ wird abgedeckt \rightarrow

für jede Variable x_i wird entweder T_i oder T'_i ausgewählt.

$T_i \rightsquigarrow$ setze $x_i = 1$.

$T'_i \rightsquigarrow$ setze $x_i = 0$.

$\{1, \dots, m\}$ wird abgedeckt \rightarrow

Zu jeder Klausel wird zu \geq einem Literal die entsprechende Menge ausgewählt.



CLIQUE

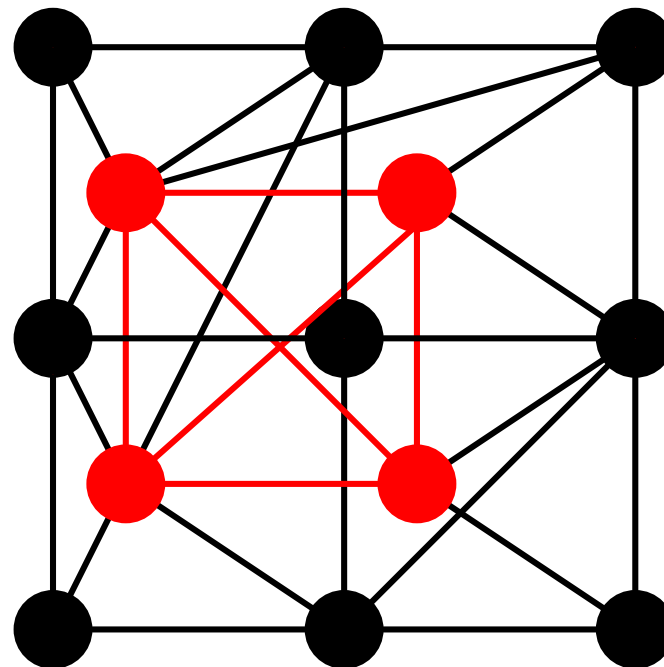
Gegeben: ungerichteter Graph $G = (V, E)$,

Parameter $k \in \mathbb{N}$.

Frage:

Enthält G eine **Clique der Größe k** ?

D.h. $\exists V' \subseteq V : |V'| = k \wedge \forall u \neq v \in V' : \{u, v\} \in E$





Beweis $\text{Clique} \in \text{NP}$

Rate die k Knoten von V'
prüfe ob sie eine Clique bilden.

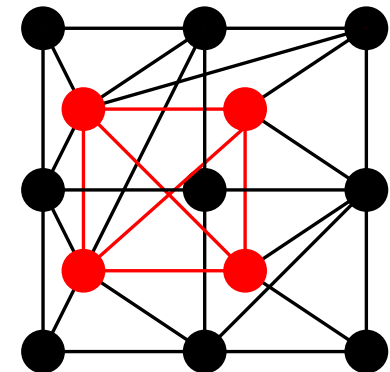
Gegeben: ungerichteter Graph $G = (V, E)$,

Parameter $k \in \mathbb{N}$.

Frage:

Enthält G eine **Clique der Größe k** ?

D.h. $\exists V' \subseteq V : |V'| = k \wedge \forall u \neq v \in V' : \{u, v\} \in E$





Beweis von „CLIQUE ist NP-hart“

Wir zeigen $3SAT \leq_p CLIQUE$.

OBdA sei $F = (z_{11} \vee z_{12} \vee z_{13}) \wedge \dots \wedge (z_{m1} \vee z_{m2} \vee z_{m3})$ mit

Literalen $z_{ij} \in \{x_1, x_2, \dots\} \cup \{\neg x_1, \neg x_2, \dots\}$.

Abbildung auf Graph $G = (V, E)$:

$$V = \{(1, 1), (1, 2), (1, 3), \dots, (m, 1), (m, 2), (m, 3)\},$$

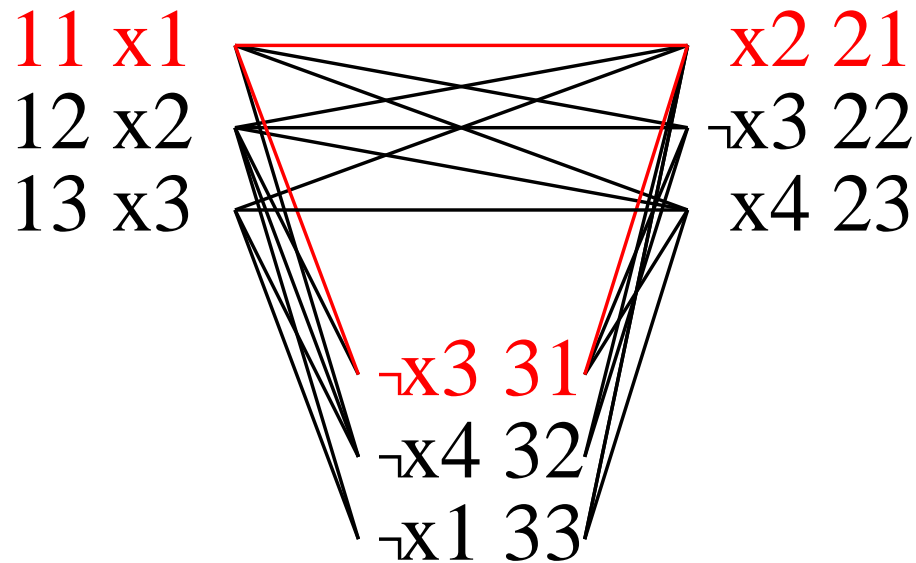
$$E = \{\{(i, j), (p, q)\} : i \neq p, z_{ij} \neq \bar{z}_{pq}\}.$$

Parameter $k = m$.



Beispiel:

$$F = (x_1 \vee x_2 \vee x_3)(x_2 \vee \neg x_3 \vee x_4)(\neg x_3 \vee \neg x_4 \vee \neg x_1)$$





$$F = (z_{11} \vee z_{12} \vee z_{13}) \wedge \cdots \wedge (z_{m1} \vee z_{m2} \vee z_{m3}).$$

$$V = \{(1, 1), (1, 2), (1, 3), \dots, (m, 1), (m, 2), (m, 3)\},$$

$$E = \{\{(i, j), (p, q)\} : i \neq p, z_{ij} \neq z_{pq}^-\}.$$

F erfüllbar $\rightarrow G = (V, E)$ hat m -Clique

F erfüllbar durch B

$$\rightarrow \exists z_{1,j_1}, \dots, z_{m,j_m} : z_{ij_i}[B] = 1$$

Diese Literale sind paarweise **nicht komplementär**

und stammen aus **verschiedenen Klauseln**.

$\rightarrow (1, j_1), \dots, (m, j_m)$ sind paarweise **verbunden**.

Sie bilden also eine **m -Clique**

qed



$$F = (z_{11} \vee z_{12} \vee z_{13}) \wedge \cdots \wedge (z_{m1} \vee z_{m2} \vee z_{m3}).$$

$$V = \{(1, 1), (1, 2), (1, 3), \dots, (m, 1), (m, 2), (m, 3)\},$$

$$E = \{\{(i, j), (p, q)\} : i \neq p, z_{ij} \neq z_{pq}^-\}.$$

$G = (V, E)$ hat m -Clique $\rightarrow F$ erfüllbar

$(k_1, j_1), \dots, (k_m, j_m)$ sei m -Clique mit zug. Literalen z_{k_i, j_i} .

Nur Kanten zwischen **verschiedenen** k_i .

\rightarrow die k_i sind paarweise verschieden.

\rightarrow OBda $k_1, \dots, k_m = 1, \dots, m$

Nur Kanten bei nichtkomplementären Literalen.

\rightarrow die **zug. Literale sind paarweise nichtkomplementär.**

$\rightarrow \exists$ Belegung $B : \forall i \in 1..m : z_{i j_i}[B] = 1$

$\rightarrow F$ **erfüllbar** durch B

qed

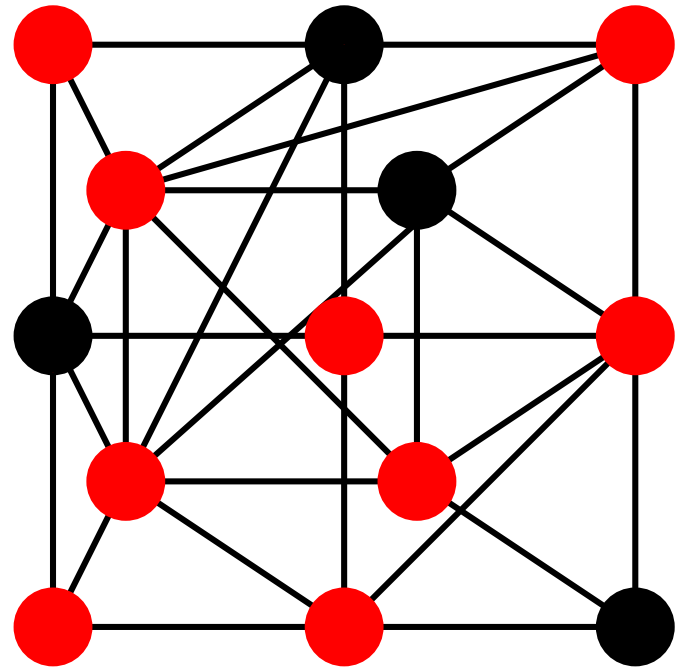


VERTEX COVER (Knotenüberdeckung)

Gegeben: ungerichteter Graph $G = (V, E)$,

Parameter $k \in \mathbb{N}$.

Frage: $\exists V' \subseteq V : |V'| = k \wedge \forall \{u, v\} \in E : u \in V' \vee v \in V'$





Beweis VERTEX COVER \in NP

... wie gehabt...



Beweis von „VERTEX COVER ist NP-hart“

Wir zeigen $\text{CLIQUE} \leq_p \text{VERTEX COVER}$.

Betrachte **Komplementgraphen**

$$\bar{G} = (V, \bar{E}) \text{ mit } \bar{E} = \{\{u, v\} : \{u, v\} \notin E\}.$$

G hat **Vertex Cover** V' der Größe k \Leftrightarrow

$$\exists V' \subseteq V : |V'| = k \wedge \forall \{u, v\} \in E : u \in V' \vee v \in V' \quad \Leftrightarrow$$

$$\exists V' \subseteq V : |V'| = k \wedge \forall \{u, v\} \subseteq V \setminus V' : \{u, v\} \notin E \quad \Leftrightarrow$$

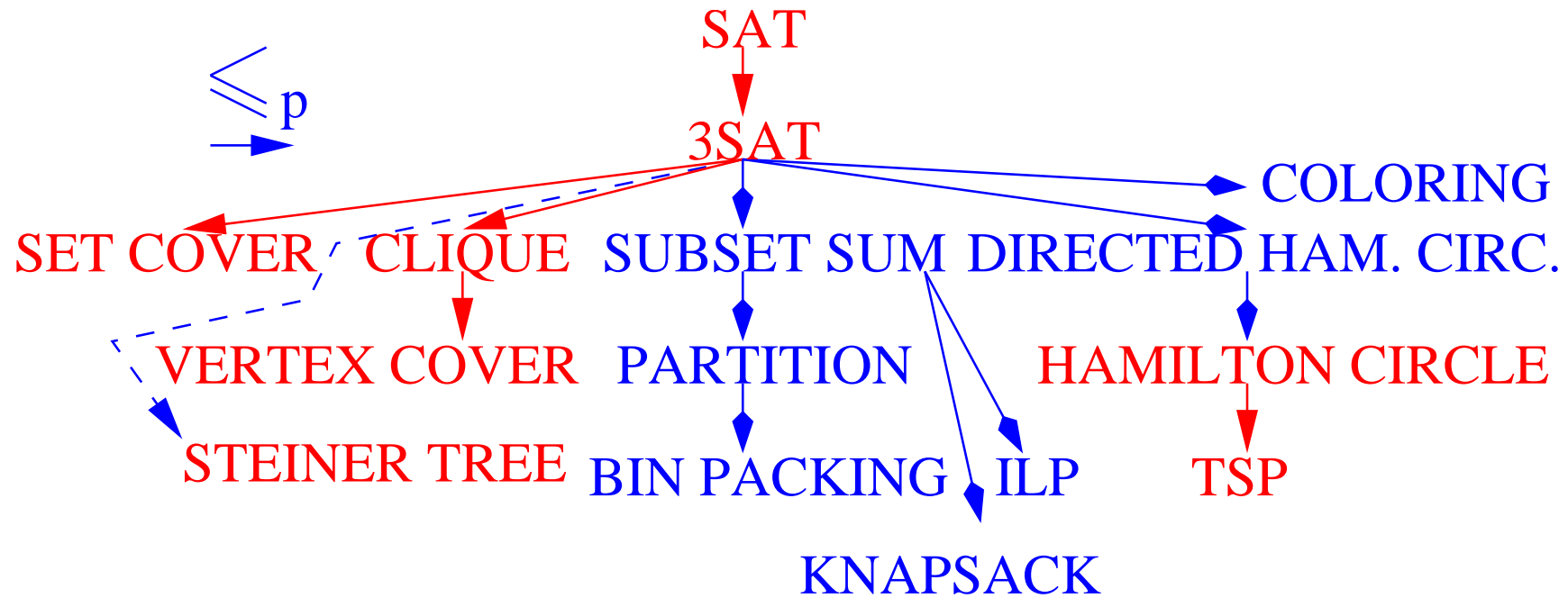
$$\exists V' \subseteq V : |V'| = k \wedge \forall u \neq v \in V \setminus V' : \{u, v\} \in \bar{E} \quad \Leftrightarrow$$

$$\exists \bar{V}' \subseteq V : |\bar{V}'| = |V| - k \wedge \forall u \neq v \in \bar{V}' : \{u, v\} \in \bar{E} \quad \Leftrightarrow$$

\bar{G} enthält eine **Clique** der Größe $|V| - k$



Gesehene Reduktionen





Beweistechniken für $A \leq_p B$:

Spezialfälle

Zeige, dass A ein Spezialfall von B ist.

Jedes elem. Objekt in der Problemformulierung von A entspricht einem elementaren Objekt in der Problemformulierung in B .

Beispiel: HAMILTON KREIS \leq_p TSP. Beispiel: SUBSET

SUM \leq_p KNAPSACK.



Beweistechniken für $A \leq_p B$:

Gadgets

Jedes elementare Objekt in der Problemformulierung von A wird auf **mehrere** zusammengehörige Objekte in B abgebildet.

Beispiel: $3\text{SAT} \leq_p \text{SET COVER}$; $x_i \rightsquigarrow T_i, T'_i$.

Beispiel: $\text{SAT} \leq_p 3\text{SAT}$; Knoten von $F \rightsquigarrow$ drei Klauseln.

Beispiel: $L \leq_p \text{SAT}$;

- Zustände, Kopfpositionen, Bandsymbole \rightsquigarrow viele Variablen.
- Anfangskonfiguration \rightsquigarrow Formel A
- Übergangsbedingung \ddot{U}_1
- Endebedingung E



Beweistechniken für $A \leq_p B$:

Randbedingungen

Erzwinge bestimmte Eigenschaften durch zusätzliche Konstrukte.

Beispiel: $L \leq_p \text{SAT}$; R, \ddot{U}_2 .

Beispiel: $\text{SAT} \leq_p \text{3SAT}$; (y_0) .

Beispiel: $\text{3SAT} \leq_p \text{COLORING}$



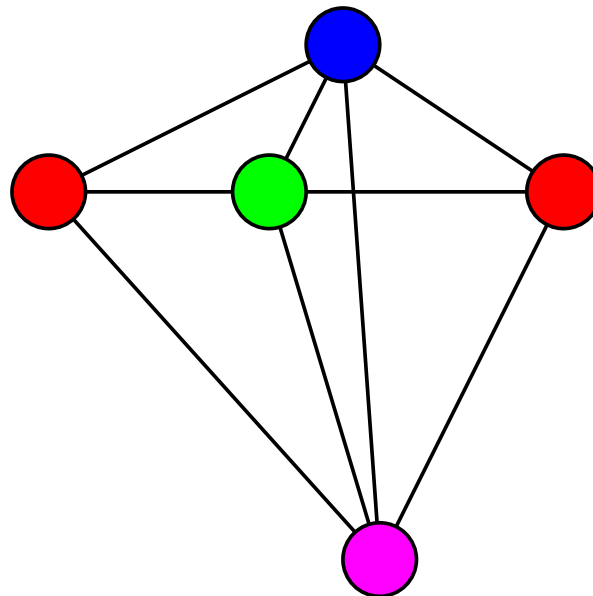
COLORING (Färbbarkeit)

Gegeben: ungerichteter Graph $G = (V, E)$,

Parameter $k \in \mathbb{N}$.

Frage:

Gibt es eine Knotenfärbung $c : V \rightarrow \{1, \dots, k\}$
mit $\forall \{u, v\} \in E : c(u) \neq c(v)$.





Anwendungen von Färbbarkeit

- Landkarten färben (4 sind genug !)
- Knotenmenge in **independent sets** faktorisieren.
Zum Beispiel für **Parallelisierung**.
- Registerallokation.
Knoten=In einer Prozedur berechnete Werte,
Kanten=Werte können nicht im gleichen Register liegen.
- ...

Verwandtes Problem: Frequenzzuteilung an Radiosender



Beweis COLORING \in NP

... wie gehabt...



Beweis von „COLORING ist NP-hart“

Wir zeigen $3\text{SAT} \leq_p 3\text{-COLORING}$ (d.h. $k = 3$).

Sei $F = K_1 \wedge \dots \wedge K_m$ mit Literalen aus

$\{x_1, \dots, x_n\} \cup \{\neg x_1, \dots, \neg x_n\}$.

OBdA: Genau 3 Literale pro Klausel.

Idee:

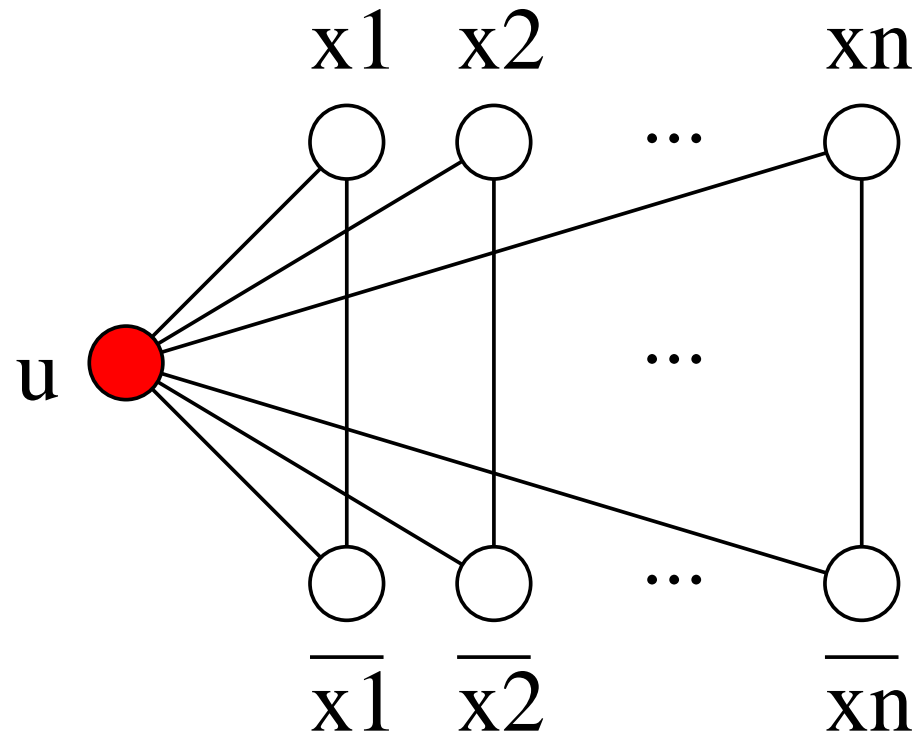
Baue Graphen $G = (V, E)$ mit $2n + 5m + 2$ Knoten, so dass

G 3-färbbar $\Leftrightarrow F$ erfüllbar.



Randbedingung erzwingt konsistente Wahrheitswerte für Literale

Ein Teil von G :



OBda Farben $0 = \text{wahr}$, $1 = \text{falsch}$, $2 = \text{rot} = c(u)$.



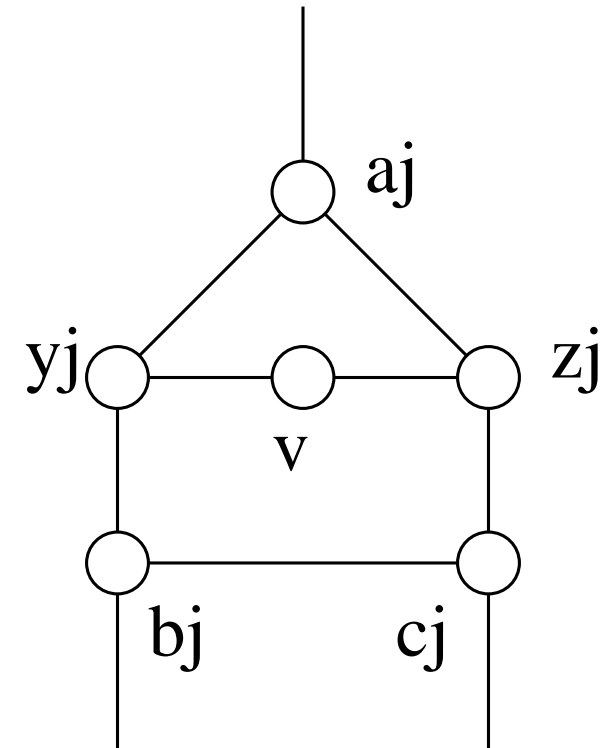
Gadget für Klausel $K_j = (a_j \vee b_j \vee c_j)$

v : Überall gleich. $(u, v) \in E$

a_j, b_j, c_j : Eigene Knoten für jede Klausel.

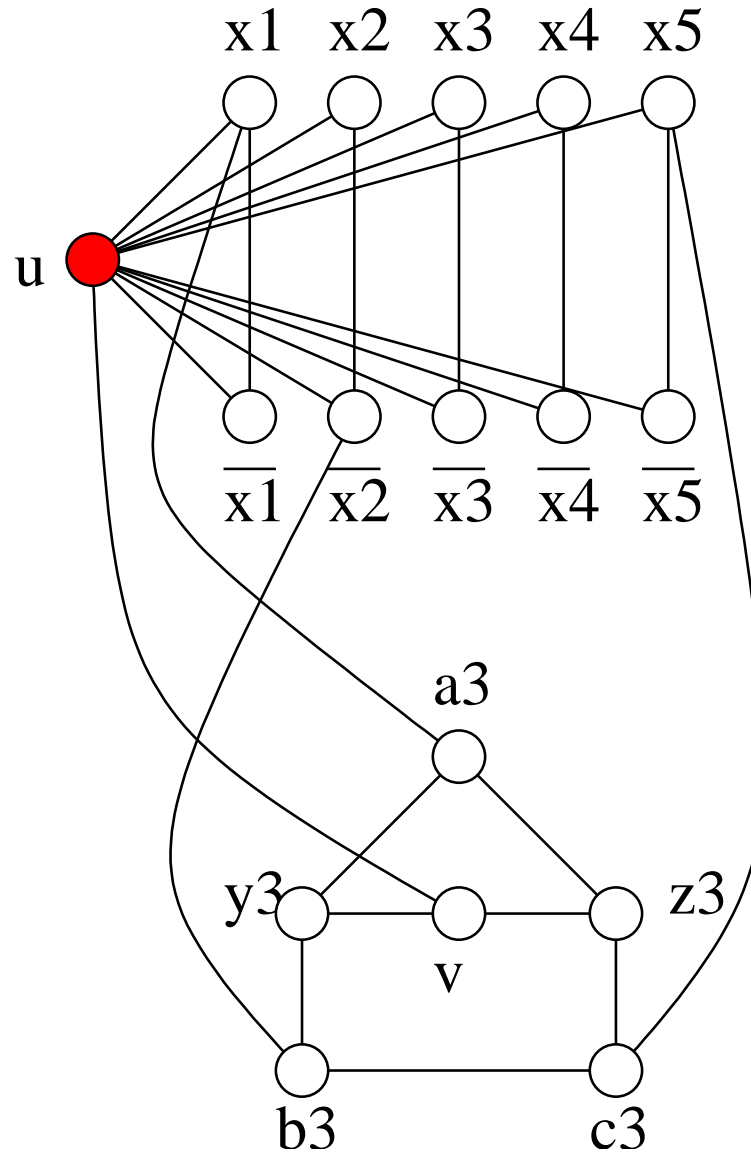
Kanten zu entsprechenden Literalknoten.

Vorsicht: Farbe $a_j \neq$ Wahrheitswert $a_j, \dots!$





Beispiel: Gadget für Klausel $K_3 = (x_1 \vee \bar{x}_2 \vee x_5)$



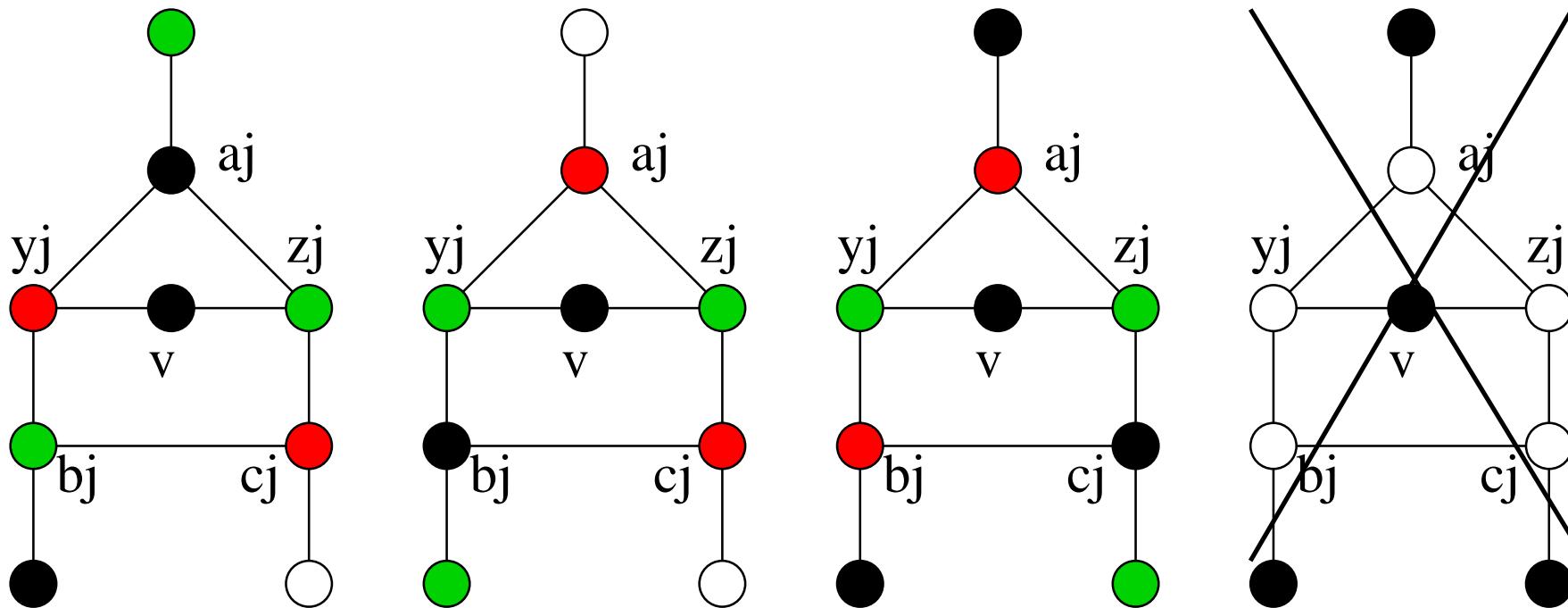


Beweis: F erfüllbar $\rightarrow G$ 3-färbbar

$c(u) := \text{rot}$, $c(v) := \text{falsch}$,

wähle $c(x_i) = \neg c(\bar{x}_i)$ entsprechend einer erfüllbaren Belegung.

Einfache Fallunterscheidung für Klausel-Gadget j :





Beweis: G 3-färbbar $\rightarrow F$ erfüllbar

Nenne die Farbe von u

rot.

Nenne die Farbe $c(v) \neq c(u)$

falsch.

Nenne die 3. Farbe

wahr.

Der Randbedingungsteilgraph erzwingt

konsistente Wahrheitswerte für Literalknoten $x_i, \bar{x}_i \in \{\text{wahr}, \text{falsch}\}$.

Zu zeigen:

Die entsprechende Belegung B macht F wahr—

$$F[B] = \text{wahr}$$

Beweis durch Widerspruch.

Annahme

$$F[B] = \text{falsch?}$$

$\rightarrow \exists j :$

$$K_j[B] = \text{falsch.}$$

\rightarrow Alle Literalnachbarn von K_j sind falsch.



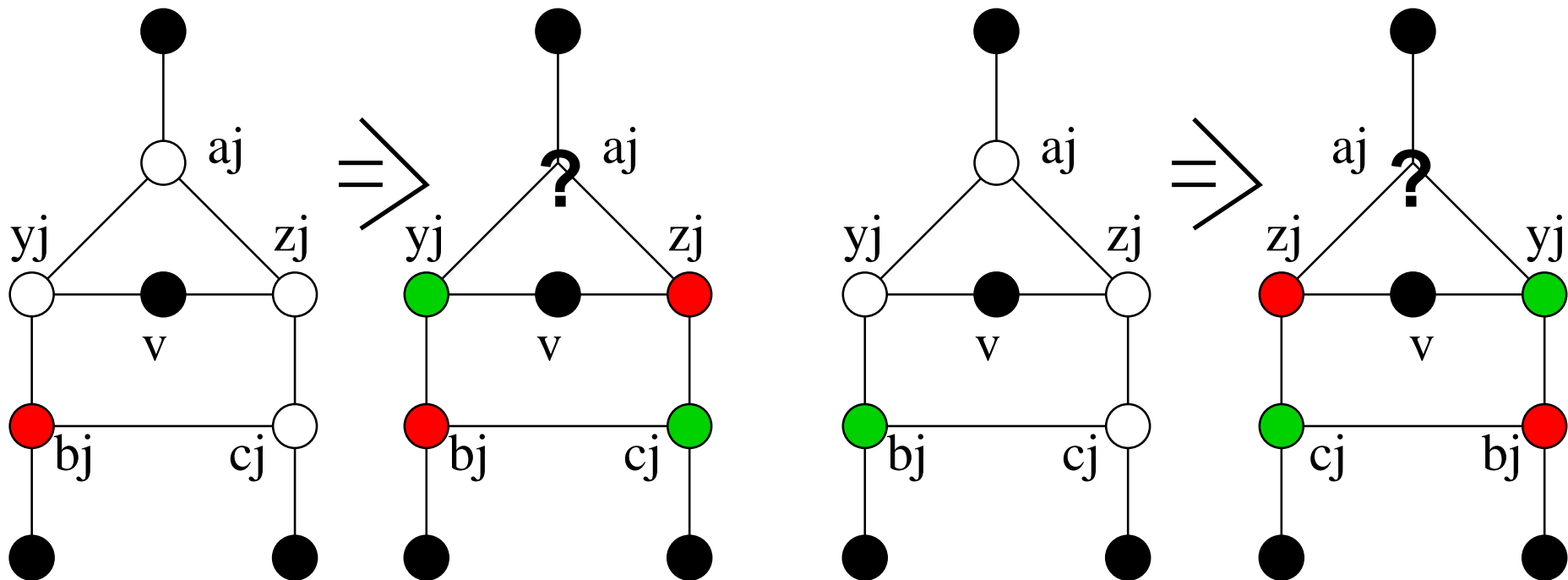
Beweis: G 3-färbbar $\rightarrow F$ erfüllbar

[...] Alle Literalnachbarn von K_j sind falsch

$\rightarrow c(b_j) = \text{rot}$

oder

$c(b_j) = \text{wahr}$



Offenbar ist also $F[B]$ wahr.

qed



Exkurs: 2-Färbbarkeit \in P

2-färbbare Graphen heißen **bipartit**.

Beobachtung: **Bäume** sind 2-färbbar.

Beobachtung: **Ungerade Kreise** sind nicht 2-färbbar. Das gleiche gilt für Graphen, die ungerade Kreise als Teilgraph enthalten.



Exkurs: 2-Färbbarkeit \in P

Betrachte eine beliebige **Zusammenhangskomponente** $C \subseteq V$.

Betrachte einen beliebigen **spannenden Baum** T von C .

Betrachte **2-Färbung** c von T .

$\exists \{u, v\} \in E : c(u) = c(v)$?

→ $\{u, v\} \cup \{ \text{Pfad von } u \text{ nach } v \text{ in } T \}$ bilden **ungeraden Kreis** (*).

→ G ist **nicht 2-färbbar**.

Sonst: c ist **legale 2-Färbung** von G .

(*):**Übung:** zeige durch Induktion über die Pfadlänge, dass zwei gleichgefärbte Knoten in T geraden Abstand haben.



SUBSET SUM

Gegeben: n Gegenstände mit Gewicht $w_i \in \mathbb{N}$

Parameter $W \in \mathbb{N}$.

Frage:

Gibt es eine Teilmenge M von $\{1, \dots, n\}$,

so dass $\sum_{i \in M} w_i = W$

\approx Spezialfall von Rucksack mit identischen Profiten.

Vorsicht! [\[Schöning\]](#) nennt SUBSET SUM RUCKSACK.



Schöning	Sanders
3KNF-SAT	3SAT
MENGENÜBERDECKUNG	SET COVERING
KNOTENÜBERDECKUNG	VERTEX COVER
RUCKSACK	SUBSET SUM
—	KNAPSACK (RUCKSACK)
FÄRBBARKEIT	COLORING



Beweis SUBSET SUM \in NP

... wie gehabt...



Beweis „SUBSET SUM ist NP-hart“

Wir zeigen $3SAT \leq_p SUBSET\ SUM$.

OBdA sei

$F = K_1 \wedge \dots \wedge K_m = (z_{11} \vee z_{12} \vee z_{13}) \wedge \dots \wedge (z_{m1} \vee z_{m2} \vee z_{m3})$ mit

Literalen $z_{ij} \in \{x_1, \dots, x_n\} \cup \{\neg x_1, \dots, \neg x_n\}$.

Wir definieren eine SUBSET SUM Instanz mit Gewichten

$(v_1, \dots, v_n, v'_1, \dots, v'_n, c_1, \dots, c_m, d_1, \dots, d_m)$

und Summe W



Die SUBSET SUM Instanz

$(v_1, \dots, v_n, v'_1, \dots, v'_n, c_1, \dots, c_m, d_1, \dots, d_m), W$

$m + n$ stellige Dezimalzahlen (Wörter über $\{0, \dots, 9\}$)

Sei $v_{ij} :=$ Anzahl Vorkommen von x_i in Klausel j .

Sei $v'_{ij} :=$ Anzahl Vorkommen von $\neg x_i$ in Klausel j .

$$v_i = v_{i1}v_{i2} \cdots v_{im} 0^{i-1} 10^{n-i},$$

$$v'_i = v'_{i1}v'_{i2} \cdots v'_{im} 0^{i-1} 10^{n-i}$$

$$c_j := 0^{j-1} 10^{m-j} 0^n$$

$$d_j := 0^{j-1} 20^{m-j} 0^n$$

$$W := 4^m 1^n$$



Beispiel

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

$$m = 3, n = 5$$

$v_1 = 100$	10000	$v'_1 = 010$	10000
$v_2 = 000$	01000	$v'_2 = 002$	01000
$v_3 = 000$	00100	$v'_3 = 100$	00100
$v_4 = 010$	00010	$v'_4 = 000$	00010
$v_5 = 110$	00001	$v'_5 = 001$	00001
$c_1 = 100$	00000	$d_1 = 200$	00000
$c_2 = 010$	00000	$d_1 = 020$	00000
$c_3 = 001$	00000	$d_1 = 002$	00000

$$W = 444 \ 11111$$



Beweisidee

Beobachtung:

bei Dezimaladdition der Gewichte treten **nie Überläufe** auf.

$$W = 444\ 11111$$

Der rechte **Einerblock** kann nur entstehen, indem für jede Variable x_i entweder $x_i \equiv v_i$ oder $\neg x_i \equiv v'_i$ ausgewählt wird.

Position j im linken Block steht für **Klausel j** .

Die ausgewählten $v_i, v'_i =$ zählen die **erfüllten Literale**:
1,2 oder 3 für erfüllte Klauseln.

Die c_j, d_j sind **Slack-Variablen**, die Position j von erfüllten Klauseln auf den Wert **4 auffüllen** können.



F erfüllbar \rightarrow Instanz lösbar

Betrachte Belegung B mit $F[B] = 1$.

Wähle für M , v_i wenn $x_i[B] = 1$ und v'_i wenn $x_i[B] = 0$.

Bisher gilt $\sum_{i \in M} w_i = t_1 t_2 \cdots t_m 1^n$

mit $t_j \in \{1, 2, 3\}$ (jede Klausel hat 1–3 erfüllte Literale).

for $j := 1$ **to** m **do**

if $t_j = 1$ **then** wähle c_j und d_j // $1 + 1 + 2 = 4$

if $t_j = 2$ **then** wähle d_j // $2 + 2 = 4$

if $t_j = 3$ **then** wähle c_j // $3 + 1 = 4$

Nun ist $\sum_{i \in M} w_i = 4^m 1^n = W$.

qed.



Instanz lösbar $\rightarrow F$ erfüllbar

Betrachte M mit $\sum_{i \in M} w_i = W = 4^m 1^n$.

Für jede Variable i wurde entweder v_i oder v'_i ausgewählt.

sonst wäre Position i im **rechten** Block $\neq 1$.

Wähle Belegung B mit $x_i = 1$ falls v_i ausgewählt, $x_i = 0$ sonst.

Behauptung: $F[B] = 1$ (Widerspruchsbeweis)

Annahme: $F[B] = 0$ also $\exists j : K_j[B] = 0$.

$\rightarrow \sum$ d. ausgewählten v_i, v'_i ist 0 an Position j

$\rightarrow \sum$ d. ausgewählten c_i, d_i ist $\leq 3 \neq 4$ an Position j

Widerspruch.

Also ist $F[B] = 1$

qed.



Rucksackproblem

∈ **NP**: wissen wir schon.

NP-hart: wir zeigen $\text{SUBSET SUM}_{\leq p} \leq_p \text{RUCKSACK}$

Offensichtlich ist die SUBSET SUM Instanz

(w_1, \dots, w_n) mit Zielgewicht = W

äquivalent zur RUCKSACK-Instanz

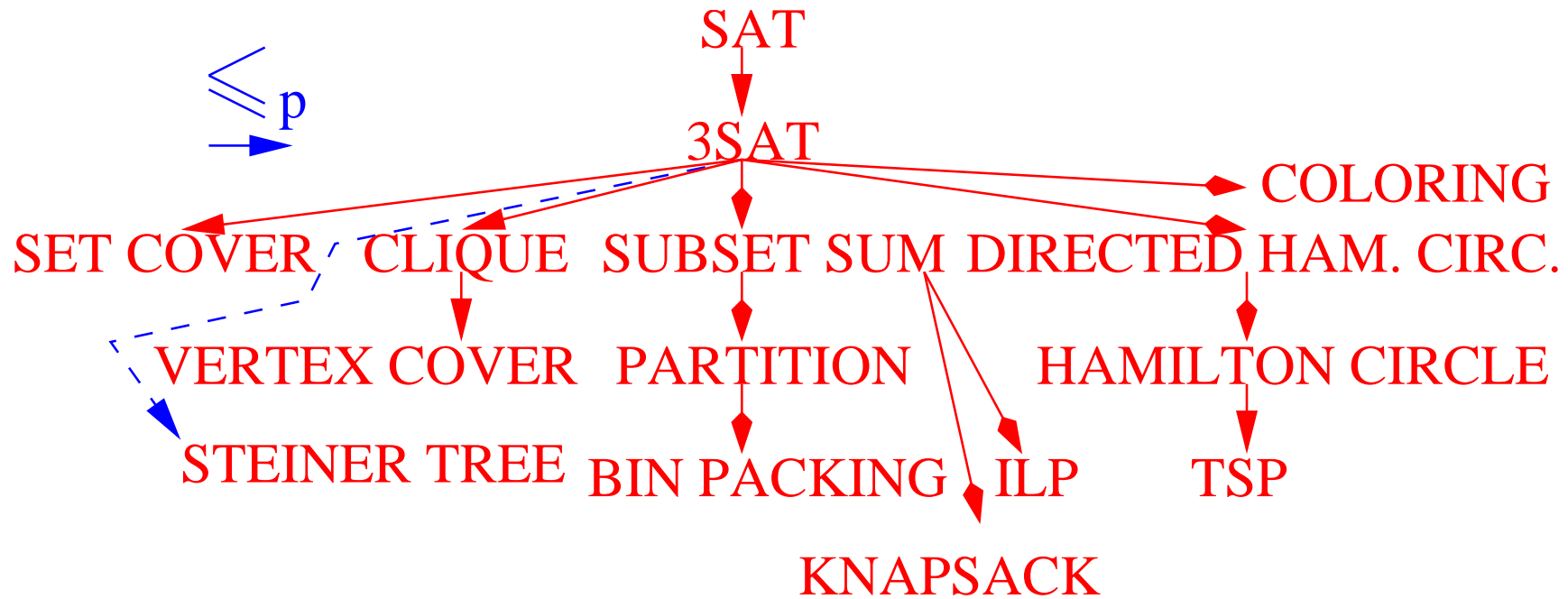
$\underbrace{(w_1, \dots, w_n)}_{\text{Gewichte}}, \underbrace{(w_1, \dots, w_n)}_{\text{Profite}}$ mit

Gewichtsschranke $\leq W$ und

Profitschranke $\geq W$.



Gesehene Reduktionen





PARTITION

Gegeben: n Gegenstände mit **Gewicht** $w_i \in \mathbb{N}$

Frage:

Gibt es eine Teilmenge M von $\{1, \dots, n\}$,

so dass $\sum_{i \in M} w_i = \sum_{i \notin M} w_i$?

Offensichtlich **in NP**, da Spezialfall von SUBSET SUM.

Anwendung: Können wir Geschenke für die Zwillinge aufteilen, ohne dass es Streit gibt?



Beweis $\text{SUBSET SUM}_{\leq p}$ PARTITION

Gegeben eine **SUBSET SUM Instanz** $S = (w_1, \dots, w_k), W$.

Sei $M := \sum_{i=1}^k w_i$.

Betrachte PARTITION Instanz $P = (w_1, \dots, w_k, M - W + 1, W + 1)$.

Gesamtgewicht: $M + M - W + 1 + W + 1 = 2(M + 1)$.

Zu Zeigen: P hat Lösung $\Leftrightarrow S$ hat Lösung.



SUBSET SUM Instanz $S = (w_1, \dots, w_k), W$.

$$M := \sum_{i=1}^k w_i.$$

PARTITION Instanz $P = (w_1, \dots, w_k, M - W + 1, W + 1)$.

Gesamtgewicht: $2(M + 1)$.

Fall P hat Lösung $\rightarrow S$ hat Lösung:

Sei J Lösung von P , d.h., $\sum_{j \in J} P[j] = M + 1$.

Entweder $M - W + 1$ oder $W + 1$ wird ausgewählt:

$$M - W + 1 + W + 1 = M + 2 > M + 1, \sum_{i=1}^k w_i = M < M + 1$$

OBdaA: $M - W + 1$ wird ausgewählt.

Dann ist $\sum_{j \in J \cap \{1, \dots, k\}} w_j = M + 1 - (M - W + 1) = W$.

Also ist $J \cap \{1, \dots, k\}$ Lösung von S .



SUBSET SUM Instanz $S = (w_1, \dots, w_k), W$.

$$M := \sum_{i=1}^k w_i.$$

PARTITION Instanz $P = (w_1, \dots, w_k, M - W + 1, W + 1)$.

Gesamtgewicht: $2(M + 1)$.

Fall S hat Lösung $\rightarrow P$ hat Lösung:

Sei I Lösung von S , d.h., $\sum_{i \in I} w_i = W$.

Dann ist $J := I \cup \{k + 1\}$ eine Lösung von P , denn

$$\sum_{j \in J} P[j] = \sum_{i \in I} w_i + M - W + 1 = W + M - W + 1 = M + 1$$



BIN PACKING

Gegeben:

Plätzchendosengröße $b \in \mathbb{N}$.

Plätzchen der Größe $w_1, \dots, w_n \in \mathbb{N}$

Anzahl Plätzchendosen k .

Frage:

Passen alle Plätzchen irgendwie in die Dosen?, d.h.,

$$\exists f : 1..n \rightarrow 1..k : \forall j \in 1..k : \sum \{w_i : f(i) = j\} \leq b$$





BIN PACKING ist NP-vollständig

Offensichtlich ist BIN PACKING in **NP**.

Andererseits ist $\text{PARTITION}_{\leq p}$ BIN PACKING (Spezialfall):

$$(w_1, \dots, w_k) \mapsto \begin{cases} \text{Dosengröße:} & b = \frac{1}{2} \sum_{i=1}^k a_i \\ \text{Anzahl:} & k = 2 \\ \text{Plätzchengrößen:} & w_1, \dots, w_k \end{cases}$$



Integer Linear Programming (ILP)

Gegeben: Variablenvektor $\mathbf{x} = (x_1, \dots, x_n)$,

Menge von Bedingungen (constraints) der Form $\mathbf{a} \cdot \mathbf{x} R b$ mit

$R \in \{\leq, \geq, =\}$, $b \in \mathbb{Z}$, $a \in \mathbb{Z}^n$.

Frage:

Gibt es Belegung für \mathbf{x} aus \mathbb{Z}^n , so dass alle Bedingungen erfüllt sind?



Beweis von „ILP ist NP-hart“

wir zeigen $\text{SUBSET SUM} \leq_p \text{ILP}$

Betrachte SUBSET SUM Instanz $(w_1, \dots, w_n), W$.

Diese ist offensichtlich äquivalent zur ILP Instanz

$$\{\mathbf{x} \cdot (w_1, \dots, w_n) = W\} \cup$$

$$\{x_1 \geq 0, \dots, x_n \geq 0\} \cup$$

$$\{x_1 \leq 1, \dots, x_n \leq 1\}$$



Beweis $ILP \in NP$?

Rate n Variablenwerte.

Prüfe ob Bedingungen erfüllt.

Problem: Genügen polynomiell viele bits für die erfüllenden Variablenwerte?

[Papadimitriou 1981] Ja.



DIRECTED HAMILTON CYCLE

(DHC) \in NP

$M :=$

$\{G = (V, E \subseteq V \times V) : \exists C \subseteq E : |C| = |V|, C \text{ ist einfacher Kreis}\}$

Rate C

überprüfe ob er jeder Knoten genau einmal besucht wird.



Beweis von „DHC ist NP-hart“

Wir zeigen $3\text{SAT} \leq_p \text{HAMILTON-KREIS}$.

OBdA sei $F = (z_{11} \vee z_{12} \vee z_{13}) \wedge \cdots \wedge (z_{m1} \vee z_{m2} \vee z_{m3})$ mit

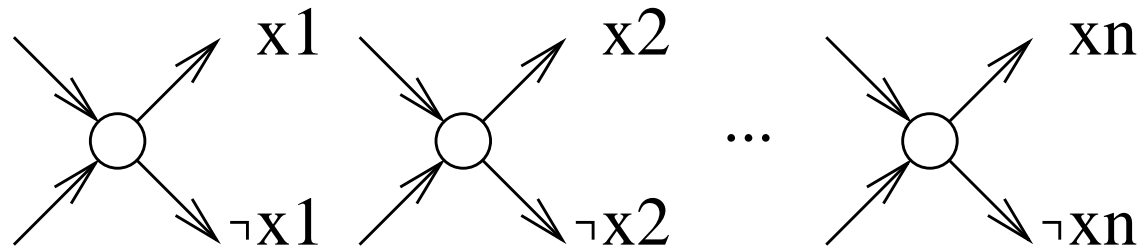
Literalen $z_{ij} \in \{x_1, x_2, \dots\} \cup \{\neg x_1, \neg x_2, \dots\}$.

Abbildung auf Graph $G = (V, E)$ mit $n + 6m$ Knoten.

- Ein Knoten pro Variable
- Gadget mit 6 Knoten je Klausel.

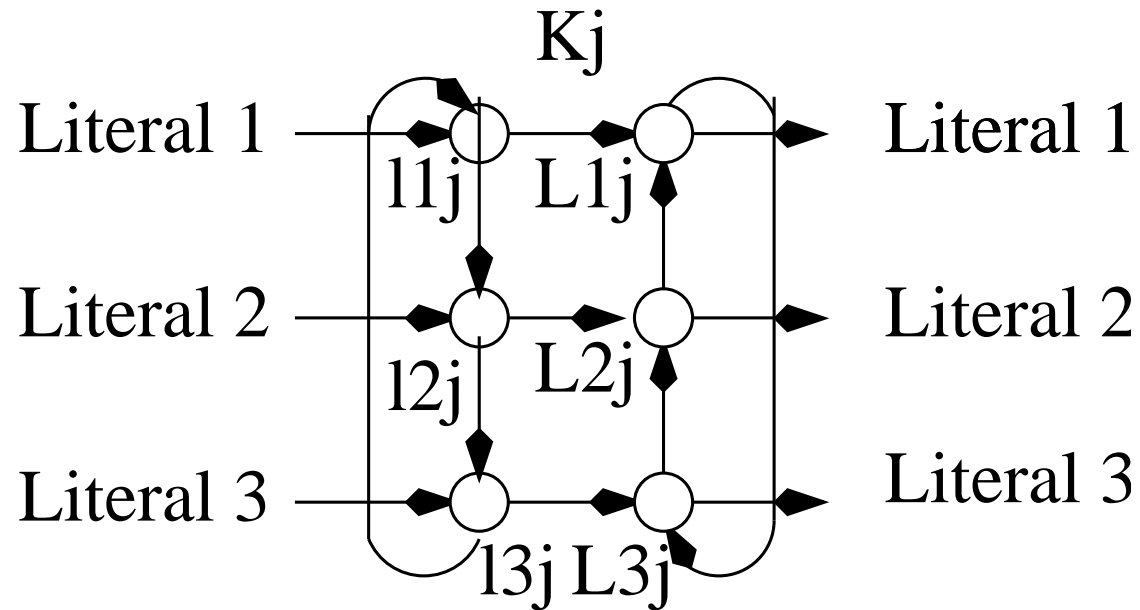


Ein Knoten pro Variable





Gadget K_j mit 6 Knoten je Klausel



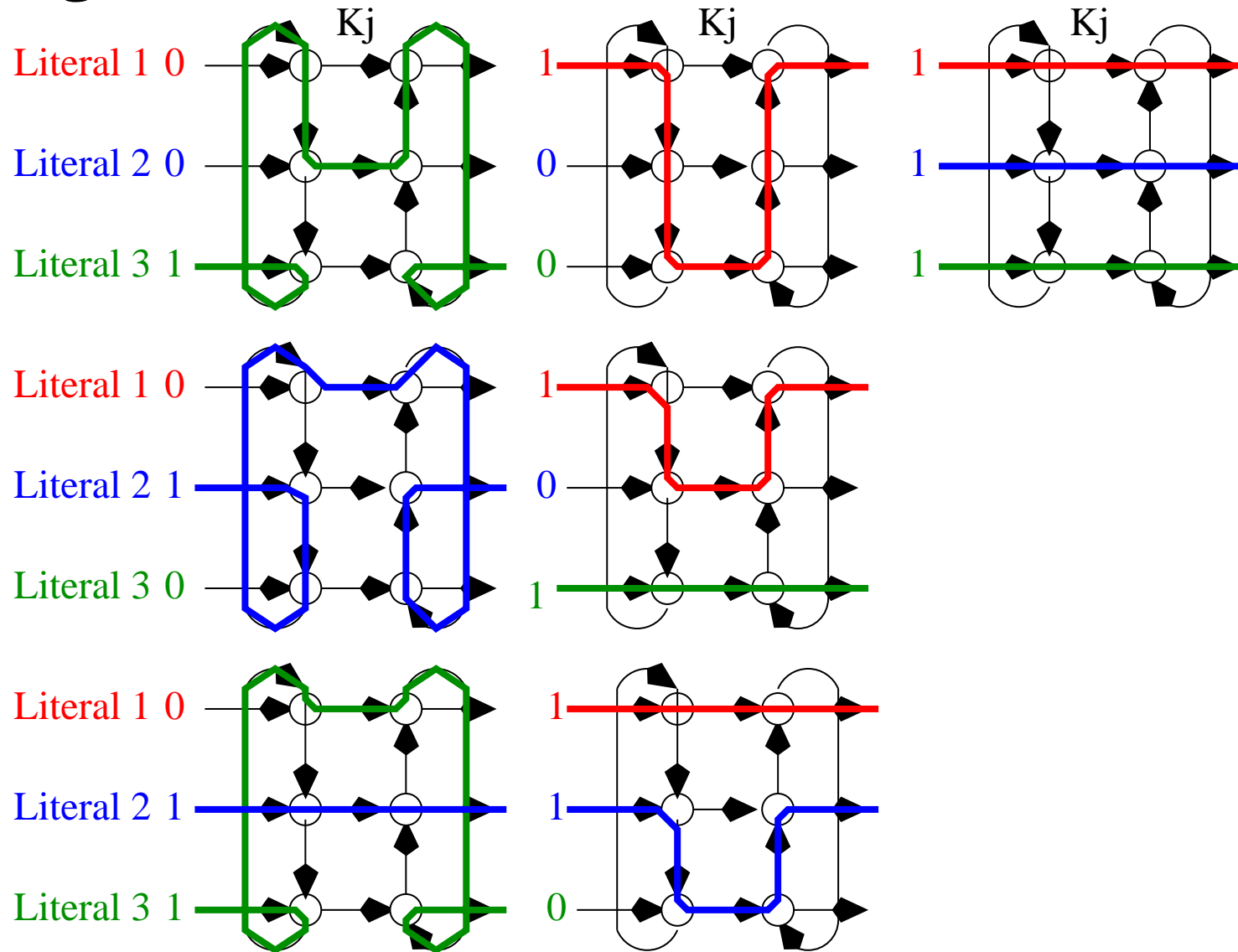
Idee: Hamiltonkreise betreten ein Gadget $1-3 \times$.

$1 \times$ für jedes erfüllte Literal.

Betreten und Verlassen jeweils in der gleichen Zeile.

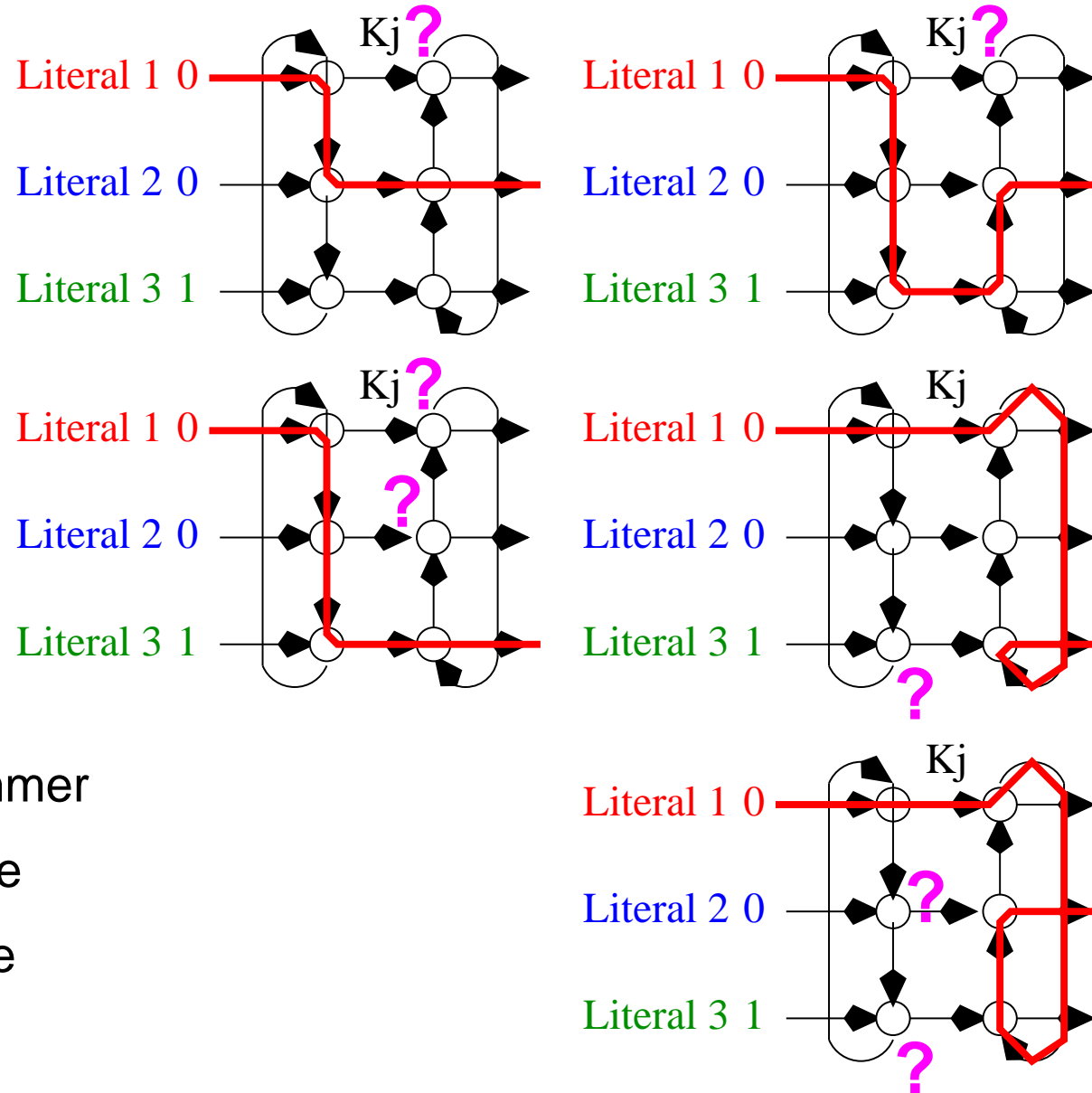


„Vorgesehene“ Hamiltonkreise





Unmögliche Hamiltonkreise



Gadgets werden immer
In der gleichen Zeile
verlassen, in der sie
betreten werden.



Verbindungen der Gadgets

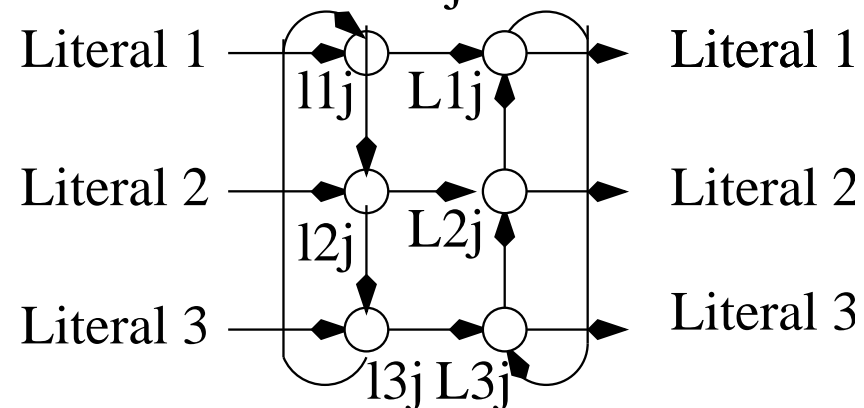
Sei $\{j_1, \dots, j_k\}$ die Menge der Klauselindizes in denen Literal z_i vorkommt ($z_i = x_i$ oder $z_i = \neg x_i$).

Seien $\{p_1, \dots, p_k\}$ die entsprechenden Positionen an denen z_i vorkommt.

Dann ist der Pfad

$$x_i \rightarrow \underbrace{\ell_{p_1 j_1} \rightarrow L_{p_1 j_1}}_{K_{j_1}} \rightarrow \underbrace{\ell_{p_2 j_2} \rightarrow L_{p_2 j_2}}_{K_{j_2}} \rightarrow \dots \rightarrow \underbrace{\ell_{p_k j_k} \rightarrow L_{p_k j_k}}_{K_{j_k}} \rightarrow x_{i+1 \bmod n}$$

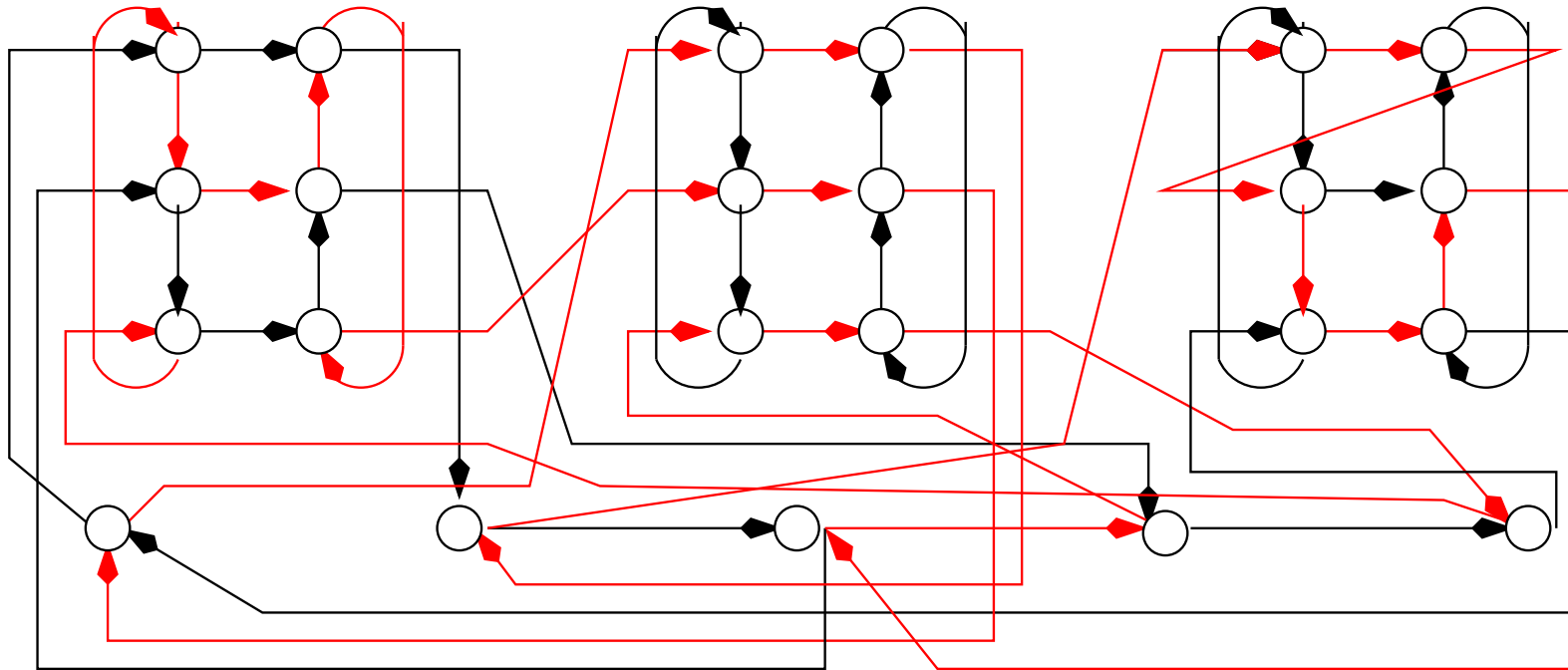
vorhanden.





Beispiel

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



erfüllende Belegung: $x_1 = x_2 = 0, x_3 = x_4 = x_5 = 1$



F erfüllbar \rightarrow Instanz lösbar

Betrachte Belegung B mit $F[B] = 1$.

$x_i = 1$: Zwischen Knoten x_i und $x_i + 1 \pmod n$ durchlaufe Gadgets $\{K_j : x_i \in \text{Klausel } j\}$.

$x_i = 0$: Zwischen Knoten x_i und $x_i + 1 \pmod n$ durchlaufe Gadgets $\{K_j : \neg x_i \in \text{Klausel } j\}$.

- Jedes **Gadget** wird mindestens einmal durchlaufen.
- Wir wissen wie jeder **Gadgetknoten** genau einmal durchlaufen wird.
- Jeder **Variablenknoten** wird genau einmal durchlaufen.



Instanz lösbar $\rightarrow F$ erfüllbar

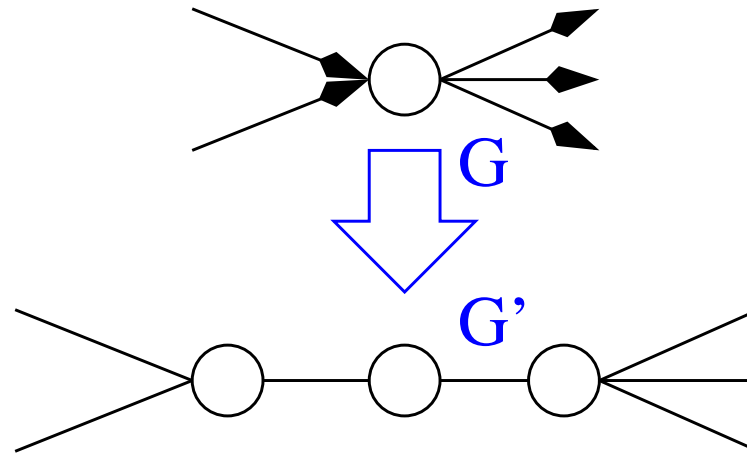
Betrachte Hamiltonkreis C .

- Zwischen x_i und x_{i+1} durchläuft C nur Gadgets so dass in den entsprechenden Klauseln entweder nur x_i oder nur $\neg x_i$ vorkommt.
- Wähle x_i so dass alle diese Klauseln erfüllt werden.
- Jedes Gadget wird mindestens einmal besucht.
- Alle Klauseln werden also erfüllt.



HAMILTON CIRCLE \leq_p DHC

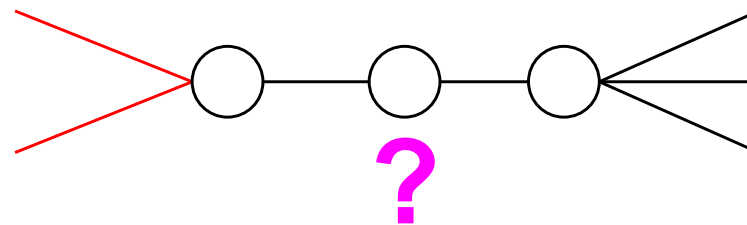
Knotengadget:



G hat HC $\rightarrow G'$ hat HC (klar).

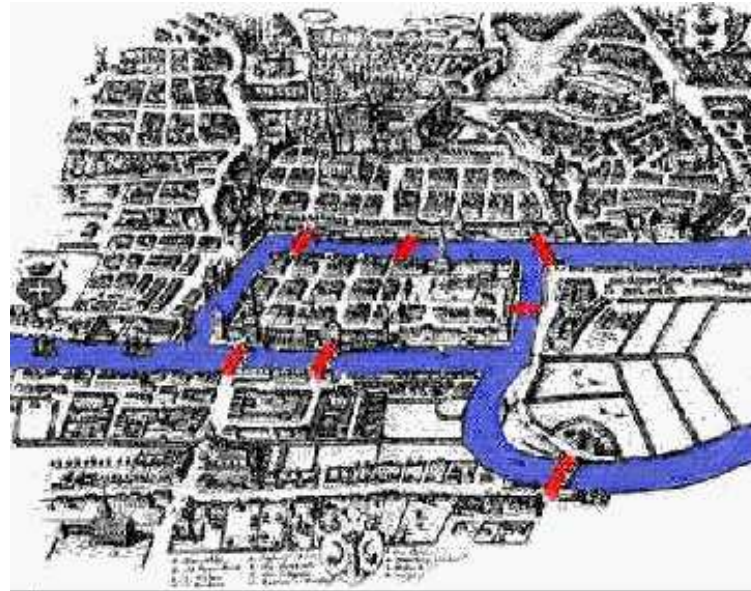
G' hat HC $\rightarrow G$ hat HC:

Angenommen HC durch Gadget ist nicht übersetzbar:





Exkurs: Euler-Tour $\in P$



$M :=$

$\{G = (V, E) : \exists C \subseteq E : C \text{ ist Kreis der jede Kante genau einmal besucht}\}$

Satz: [Euler 1736] G hat Euler-Tour \Leftrightarrow

G ist zusammenhängend und jeder Knoten hat geraden Grad.



Gesehene Reduktionen

