Theoretische Informatik II

Einheit 5.3

$\mathcal{NP}$-vollständige Probleme

1. Der Satz von Cook
2. Logische Probleme
3. Graphbasierte Probleme
4. Jenseits von $\mathcal{NP}$-vollständigkeit

Wie zeigt man $\mathcal{NP}$-Vollständigkeit?

$\mathcal{NP}$-Vollständigkeit muß einmal explizit bewiesen werden

- **Codiere Berechnungen beliebiger NTM's**
  - Codierung soll zu Sprache L gehören, wenn Maschine M akzeptiert
  - Codierung soll nicht zu L gehören, wenn M nicht akzeptiert
  - Polynomielle NTMs müssen in polynomieller Zeit codiert werden
  - Ergibt $L(M) \leq_p L$ für jede NTM M, d.h. L ist $\mathcal{NP}$-hart
- **Sprache L muß selbst in $\mathcal{NP}$ liegen**
  - Ergibt $\mathcal{NP}$-Vollständigkeit von L
- **Welches Sprache ist ausdrucksstark genug?**
  - Codiere mögliche Zustandsübergänge durch logische Formeln
  - Problem: Können Zustandsübergänge so kombiniert werden,
    daß terminierende Berechnung codiert wird?
  - Erfüllbarkeitsproblem der (Aussagen-)logik ist Kandidat für $\mathcal{NPC}$

Das Erfüllbarkeitsproblem

Ist eine aussagenlogische Formel in KNF erfüllbar?

Gegeben m Klauseln $k_1, .., k_m$ über n Variablen $x_1, .., x_n$. Gibt es eine Belegung $a_1, .., a_n \in \{0, 1\}$ der Variablen x_i, welche alle Klauseln erfüllt?

- **Klausel über den Variablen $x_1, ..x_n$**
  - Disjunktion einiger Literale der Form x_i bzw. $\overline{x_i}$
- **Belegung $a_1, ..., a_n \in \{0, 1\}$ erfüllt Klausel k_j**
  - Auswertung von k_j unter $a_1, ..., a_n$ ergibt den Booleschen Wert 1
- $$SAT = \{k_1, ..k_m \mid k_i \text{ Klausel über } x_1, ..x_n \wedge (\exists a_1, ..a_n \in \{0,1\}. \forall j \leq m.\ a_1, ..a_n \text{ erfüllt } k_j)\}$$

Codierbar als Teilmenge der Sprache der Aussagenlogik

Beispiele von Formeln in KNF

$(\overline{x_1} \vee x_2) \;\wedge\; (x_1 \vee \overline{x_2} \vee \overline{x_3}) \;\wedge\; \overline{x_3}$ *erfüllbar*

– Setze $x_3{=}0$, $x_2{=}1$, x_1 beliebig, z.B. $x_1{=}0$

– Auswertung: $(\overline{0}{+}1) * (0{+}\overline{1}{+}\overline{0}) * \overline{0} = (1{+}1) * (0{+}0{+}1) * 1 = 1 * 1 * 1 = 1$

$x_1 \;\wedge\; \overline{x_1}$ *nicht erfüllbar*

– Jede Belegung ergibt den Wert 0

$(x_1 \vee x_2) \;\wedge\; (\overline{x_1} \vee \overline{x_2})$ *erfüllbar, Belegung: (1,0)*

$(x_1 \vee x_2) \;\wedge\; (x_1 \vee \overline{x_2}) \;\wedge\; (\overline{x_1} \vee x_2) \;\wedge\; (\overline{x_1} \vee \overline{x_2})$

nicht erfüllbar

$(x_1 \vee \overline{x_2} \vee x_3) \;\wedge\; (\overline{x_1} \vee x_2 \vee \overline{x_4}) \;\wedge\; (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})$

erfüllbar, Belegung: (1,1,0,0)

Lösungen für das Erfüllbarkeitsproblem

$SAT = \{k_1..k_m \mid k_i \text{ Klausel über } x_1..x_n \wedge \exists a_1..a_n \in \{0{,}1\}.a_1..a_n \text{ erfüllt } k_1..k_m\}$

- **Deterministische Lösung**
  - Werte Klauseln für alle möglichen Belegungen der Variablen aus bis erfüllende Belegung gefunden ist
  - Es gibt 2^n möglichen Belegungen von $x_1, ..x_n$
  - Auswertung linear in Größe der Formel $\mathcal{O}(m * n)$
  - **Laufzeit ist in $\mathcal{O}(2^n)$**
- **Nichtdeterministisch: Raten und Verifizieren**
  - Erzeuge eine erfüllende Belegung der Variablen (falls es eine gibt)
  - Prüfe Belegung durch Auswertung der Formel in polynomieller Zeit

$$\Downarrow$$

$$SAT \in \mathcal{NP}$$

$\mathcal{SAT}$ ist $\mathcal{NP}$-vollständig (Satz von Cook)

- **Gegeben: NTM M, die in polynomieller Zeit terminiert**
- **Ziel: Codiere Berechnung von M bei Eingabe w durch KNF-Formel, die erfüllbar ist, g.d.w. $w \in L(M)$**
  - Codierung muß in polynomieller Zeit (relativ zu $|w|$) berechenbar sein
- **Vorgehen: Beschreibe mögliche Konfigurationsübergänge von M durch Klauseln**
  - Codiere Zustand, Kopfposition und Bandzellen durch Literale
  - Es werden nur polynomiell viele Literale und Klauseln benötigt
  - Formel ist erfüllbar, wenn Konfigurationsübergänge zu akzeptierender Berechnung zusammengesetzt werden können

Aufwendiger Beweis mit sehr vielen Details

Satz von Cook: Grundannahmen

Zeige $L \leq_p SAT$ für jede Sprache $L \in \mathcal{NP}$

- **L wird von NTM M akzeptiert**
  - $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ mit $Q{=}\{q_0, .., q_k\}$, $\Gamma{=}\{X_1, .., X_m\}$
- **M zeitbeschränkt durch Polynom $p(n)$**
  - $t_M(w) \leq p(n)$ für jedes Wort $w \in \Sigma^*$ mit $|w|{=}n$
  - Es sind genau $p(n)$ Berechnungsschritte als Formel zu codieren

    o.B.d.A.: M 'verharrt' in den Endzuständen anstatt abzubrechen

    d.h. $(u,q,v) \vdash (u,q,v)$ für $q \in F$
- **M ist auch platzbeschränkt durch $p(n)$**
  - M kann während der Berechnung maximal $p(n)$ Bandzellen aufsuchen

    o.B.d.A.: M arbeitet mit halbseitig unendlichem Band
  - Es reicht, genau die Bandzellen von 1 bis $p(n)$ zu modellieren
  - Schreibe Konfiguration (u,q,v) als String $\boldsymbol{uqvB^j}$ der Länge $p(n){+}1$

SATZ VON COOK: ZU CODIERENDE AUSSAGEN

- **Anfangsbedingungen bei Eingabe w**
  - M startet im Zustand q_0 und der Kopf ist über Bandzelle 0
  - Anfangskonfiguration ist $\boldsymbol{q_0 w_1..w_n B^{p(n)-n+1}}$
- **Übergangsbedingungen**
  - Zu jedem Zeitpunkt t steht der Kopf an einer Stelle j und verändert Bandinhalt und Zustand entsprechend der Tabelle von δ
- **Endbedingung**
  - Nach $p(n)$ Schritten befindet sich M in einem Endzustand $q_f \in F$
  - Endkonfiguration hat die Form $\boldsymbol{X_0..X_{j-1} q_f X_{j+1}..X_{p(n)}}$ für ein j
- **Randbedingungen für eindeutiges Verhalten**
  - Zu jedem Zeitpunkt t befindet sich M in genau einer Konfiguration $X_0..X_{j-1} q X_{j+1}..X_{p(n)}$

Beschreibe Berechnung als KNF-Formel $\alpha(M,w)$

Symbole der Codierung

- **Konfigurationsvariablen $y_{t,i,A}$,** $t, i \in \{0..p(n)\}$
  - Nach t Schritten steht an der i-ten Stelle der Konfiguration ein A
  - A ist ein Symbol aus Γ oder ein Zustand $(A \in \{X_1, .., X_m, q_0, .., q_k\})$
- **Codiere Randbedingungen als Formel:**
  - Zu jedem Zeitpunkt steht an jeder Stelle genau ein Symbol
  - Zu jedem Zeitpunkt steht nur an einer Stelle ein Zustand

$$R \equiv \exists_1(y_{0,0,X_1}, .., y_{0,0,q_k}) \wedge \ldots \wedge \exists_1(y_{p(n),p(n),X_1}, .., y_{p(n),p(n),q_k})$$
$$\wedge\ \exists_1(y_{0,0,q_1}, ..y_{0,p(n),q_k}) \wedge \ldots \wedge \exists_1(y_{p(n),0,q_1}, ..y_{p(n),p(n),q_k})$$

Dabei steht $\exists_1(x_1, .., x_m)$ für "genau eines der x_i gilt'

$$\exists_1(x_1, .., x_j) \equiv (x_1 \vee .. \vee x_j) \wedge (\bar{x}_1 \vee \bar{x}_2) \wedge .. \wedge (\bar{x}_1 \vee \bar{x}_j)$$
$$\wedge (\bar{x}_2 \vee \bar{x}_3) \wedge .. \wedge (\bar{x}_2 \vee \bar{x}_j) \wedge .. \wedge (\bar{x}_{j-1} \vee \bar{x}_j)$$

- **R ist in KNF** Konjunktion von $\exists_1$-Formeln
- **Größe: $\mathcal{O}(p(n)^3)$** $p(n)^2 * (m+k)^2 + p(n) * (k * p(n))^2$ Variablen
- **Berechnungsaufwand: $\mathcal{O}(p(n)^3)$** Bestimme $p(n)$,...

Satz von Cook: Codierung der Anfangsbedingungen

Anfangskonfiguration ist $q_0 w_1..w_n B^{p(n)-n+1}$

Codiere Anfangsbedingungen als Formel A mit

$$A \equiv y_{0,0,q_0} \wedge y_{0,1,w_1} \wedge .. \wedge y_{0,n,w_n} \wedge y_{0,n+1,B} \wedge .. \wedge y_{0,p(n),B}$$

- **A ist in KNF** — Rein konjunktive Formel
- **Größe: $\mathcal{O}(p(n))$** — $p(n)+1$ Variablen
- **Berechnungsaufwand: $\mathcal{O}(p(n))$** — Bestimmung von $p(n)$

Satz von Cook: Codiere Übergangsbedingungen

Konfigurationsübergänge sind verträglich mit δ

Formeln $\ddot{\mathrm{U}}(t,i)$ betrachten Zeit t und Stelle i

– Falls M an Stelle i steht, kann sich der Bereich $i-1..i+1$ ändern

$$\begin{aligned}
&\left(y_{t,i-1,Z} \wedge y_{t,i,q} \wedge y_{t,i+1,X}\right)\\
&\Rightarrow \left(y_{t+1,i-1,p_1} \wedge y_{t+1,i,Z} \wedge y_{t+1,i+1,Y_1}\right)\\
&\qquad \vee .. \vee \left(y_{t+1,i-1,p_l} \wedge y_{t+1,i,Z} \wedge y_{t+1,i+1,Y_l}\right)\\
&\qquad \vee \left(y_{t+1,i-1,Z} \wedge y_{t+1,i,Y'_1} \wedge y_{t+1,i+1,p'_1}\right)\\
&\qquad \vee .. \vee \left(y_{t+1,i-1,Z} \wedge y_{t+1,i,Y'_r} \wedge y_{t+1,i+1,p'_r}\right)
\end{aligned}$$

für jedes $Z \in \Gamma$, falls $\delta(q,X) = \{(p_1,Y_1,L),..,(p_l,Y_l,L),\ (p'_1,Y'_1,R),..,(p'_r,Y'_r,R)\}$

– Falls M nicht im Bereich $i-1..i+1$ steht, bleibt Stelle i unverändert

$$\begin{aligned}
&\left(\overline{y_{t,i-1,q_0}} \wedge .. \wedge \overline{y_{t,i-1,q_k}} \wedge \overline{y_{t,i,q_0}} \wedge .. \wedge \overline{y_{t,i,q_k}} \wedge \overline{y_{t,i+1,q_0}} \wedge .. \wedge \overline{y_{t,i+1,q_k}}\right)\\
&\Rightarrow \left(y_{t,i,X_1} \wedge y_{t+1,i,X_1}\right) \vee .. \vee \left(y_{t,i,X_m} \wedge y_{t+1,i,X_m}\right)
\end{aligned}$$

Formeln müssen normalisiert und mit $\wedge$ kombiniert werden

Satz von Cook: Codiere Übergangsbedingungen II

Konfigurationsübergänge sind verträglich mit δ

Kombiniere Übergangsbedingungen zu Formel Ü

$$\text{Ü} \equiv \text{Ü}(0,0) \wedge .. \wedge \text{Ü}(0,p(n)) \\ \wedge \text{Ü}(p(n),0) \wedge .. \wedge \text{Ü}(p(n),p(n))$$

- **Ü ist in KNF** — Alle Ü(t,i) wurden normalisiert
- **Größe: $\mathcal{O}(p(n)^2)$** — $p(n)^2$ Komponentenformeln

  Je Komponente nach Normalisierung maximal $k * m * 3^{2m*k} + 3k * 2^m$ Symbole
- **Berechnungsaufwand: $\mathcal{O}(p(n)^2)$**

Satz von Cook: Codierung der Endbedingung

Endkonfiguration hat Form $X_0..X_{j-1}q_fX_{j+1}..X_{p(n)}$

Sei $F=\{q_r, .., q_e\}$

Codiere Endbedingungen als Formel E mit

$$\begin{array}{rl} E = & (y_{p(n),0,q_r} \vee .. \vee y_{p(n),0,q_r}) \\ \vee & (y_{p(n),1,q_r} \vee .. \vee y_{p(n),1,q_r}) \\ \vee & \vdots \\ \vee & (y_{p(n),p(n),q_r} \vee .. \vee y_{p(n),p(n),q_r}) \end{array}$$

- **E ist in KNF** — Einfache Klausel
- **Größe: $\mathcal{O}(p(n))$** — $p(n) * (e{-}r)$ Variablen
- **Berechnungsaufwand: $\mathcal{O}(p(n))$**

Korrektheit der Codierung $\alpha(M,w) \equiv A \wedge \ddot{U} \wedge E \wedge R$

- **$\alpha(M,w)$ ist in KNF**
  – Jede der Teilformeln ist in KNF
- **$\alpha(M,w)$ ist in Zeit $\mathcal{O}(p(n)^3)$ konstruierbar**
  – Summe des (deterministischen) Berechnungsaufwands für Teilformeln
- **$w \in L \;\Rightarrow\; \alpha(M,w) \in SAT$**
  – Ist $w \in L$, dann gibt es eine akzeptierende Berechnung $K_0,..,K_{p(n)}$ für w
  – Setze: $y_{t,i,A}$:=1, falls A das i-te Symbol von K_t ist, und sonst $y_{t,i,A}$:=0
  – Per Konstruktion erfüllt dies die Formel $\alpha(M,w)$, also $\alpha(M,w) \in SAT$
- **$\alpha(M,w) \in SAT \;\Rightarrow\; w \in L$**
  – Ist $\alpha(M,w)$ erfüllbar, so kann mit Formel Ü die Belegung der Variablen in eine Konfigurationsfolge $K_0,..,K_{p(n)}$ umgerechnet werden
  – Wegen der Formel R gibt es genau eine solche Konfigurationsfolge
  – Wegen A und E repräsentiert diese Konfigurationsfolge eine akzeptierende Berechnung für w also $w \in L$

Satz von Cook: Zusammenfassung

- **Aufwendige Codierung von Berechnungen**
  - Formel $\alpha(M,w)$ codiert Berechnung der NTM M bei Eingabe w
  - $\alpha(M,w)$ ist in polynomieller Zeit berechenbar
  - Es gilt $w \in L(M) \;\Leftrightarrow\; \alpha(M,w) \in SAT$
  - Es folgt $L(M) \leq_p SAT$
- **Konstruktion ist uniform für polynomielle NTMs**
  - Es folgt **$L \leq_p SAT$ für jedes $L \in \mathcal{NP}$**
  - Lösungen für SAT sind in Lösungen für jedes $L \in \mathcal{NP}$ transformierbar
- **SAT ist selbst in $\mathcal{NP}$**
  - Belegungen können leicht als erfüllend überprüft werden

$$\Downarrow$$

SAT ist $\mathcal{NP}$-vollständig

Methodik für Nachweis von $\mathcal{NP}$-Vollständigkeit

- **Direkter Beweis ist zu aufwendig**
  - Würde explizite Codierung beliebiger NTMs erfordern
  - Verwende: $L \in \mathcal{NPC} \;\Leftrightarrow\; L \in \mathcal{NP} \wedge \exists L' \in \mathcal{NPC}.\ L' \leq_p L$

1. Zeige $L \in \mathcal{NP}$:

a) Beschreibe, welchen Lösungsvorschlag die NTM generiert

b) Beschreibe, wie Lösungsvorschlag deterministisch überprüft wird

c) Zeige, daß das Prüfverfahren polynomiell ist

2. Zeige $\exists L' \in \mathcal{NPC}.\ L' \leq_p L$:

d) Wähle ein ähnliches, bekanntes $\mathcal{NP}$-vollständiges Problem L'

e) Beschreibe Transformationsfunktion f, welche Eingaben über dem Alphabet Σ' für L' in Wörter über dem Alphabet für L umwandelt

f) Zeige für alle $x \in \Sigma'^*$: $x \in L' \Leftrightarrow f(x) \in L$ (also $L' = f^{-1}(L)$)

g) Zeige, daß f in polynomieller Zeit berechnet werden kann

Erfüllbarkeit mit 3 Literalen pro Klausel

$$3SAT = \{k_1,..k_m \mid k_i{=}z_{i1} \vee z_{i2} \vee z_{i3} \quad \text{mit} \quad z_{ij} \in \{x_1, \overline{x_1}, ..x_n, \overline{x_n}\} \wedge \exists a_1,..a_n \in \{0{,}1\}.\ \forall j{\leq}m.\ a_1,..a_n \text{ erfüllt } k_j\}$$

1. Zeige $3SAT \in \mathcal{NP}$:

– Wie $SAT \in \mathcal{NP}$: Rate Belegung der Variablen und werte Klauseln aus

2. Zeige $SAT \leq_p 3SAT$:

Satz 10.15

e) Normalisierung der Klauseln $k_1,..k_m$ über $x_1,..x_n$.

Ersetze Klausel k_i durche äquivalente Menge von Dreierklauseln

- Ersetze einelementige Klauseln $k_i = z$ durch $z \vee z \vee z$
- Ersetze zweielementige Klauseln $k_i = z \vee z'$ durch $z \vee z \vee z'$
- Übernehme dreielementige Klauseln unverändert
- Ersetze Klauseln $k_i = z_1 \vee z_2 \vee .. \vee z_j$ durch j-2 neue Klauseln mit neuen Variablen $y_{i,l}$: $(z_1 \vee z_2 \vee y_{i,1}) \wedge (\overline{y_{i,1}} \vee z_3 \vee y_{i,2}) \wedge .. \wedge (\overline{y_{i,j-3}} \vee z_{j-1} \vee z_j)$

f) k_i erfüllbar genau dann wenn normalisierte Klauselmenge erfüllbar

Für die Transformation f gilt: $\forall F.\ F \in SAT \Leftrightarrow f(F) \in 3SAT$

g) Normalisierung der Klauseln möglich in **polynomieller Zeit**

DAS CLIQUEN PROBLEM IST $\mathcal{NP}$-VOLLSTÄNDIG

$$\boldsymbol{CLIQUE} = \{\, (G,k) \mid G{=}(V,E) \text{ Graph} \wedge (\exists V_c{\subseteq}V.\, |V_c|{\geq}k \wedge V_c \text{ Clique in } G) \,\}$$

1. Zeige $\boldsymbol{CLIQUE \in \mathcal{NP}}$:

a) Rate eine Kantenmenge $V_c{\subseteq}V$

b/c) Prüfe $|V_c|{\geq}k$ — *maximal* $|V_c|$ *Schritte*

Prüfe: $\forall v{\neq}v' \in V_c.\ \{v,v'\} \in E$ — *maximal* $|V_c|^2 * |E|{\leq}|V|^4$ *Schritte*

2. Zeige $\boldsymbol{3SAT \leq_p CLIQUE}$:

– Gegeben $F = (k_1, ..., k_m)$ mit $k_i = z_{i1} \vee z_{i2} \vee z_{i3}$ und $z_{ij} \in \{x_1, ..., \overline{x_n}\}$

– Konstruiere Graphen $G_F := (V, E)$ mit

$V := \{v_{ij} \mid 1{\leq}i{\leq}m, 1{\leq}j{\leq}3\}$ und $E := \{\, \{v_{ij}, v_{i'j'}\} \mid i{\neq}i' \wedge z_{ij}{\neq}\overline{z_{i'j'}}\}$

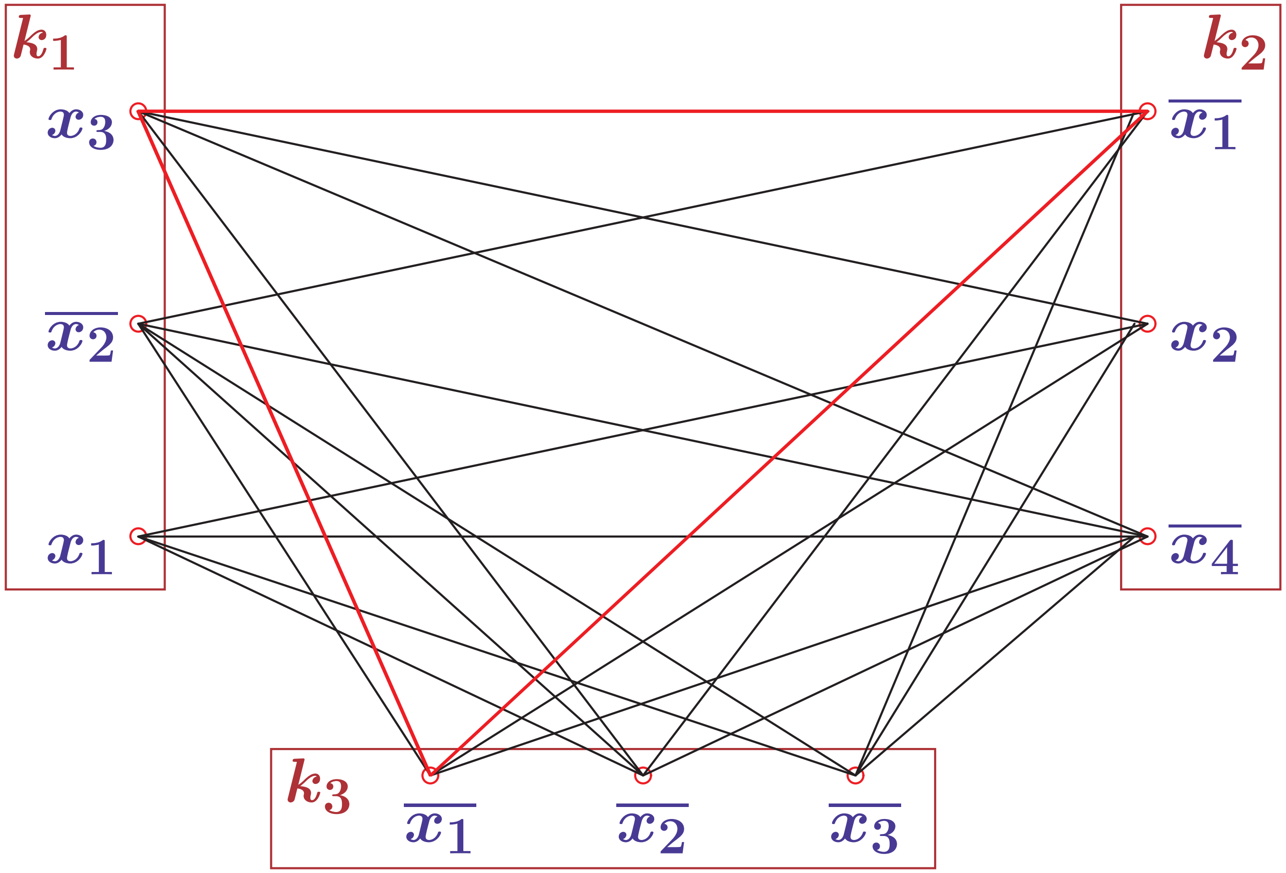
e) Setze $f(F) := (G_F, m)$

f) Dann gilt $F \in 3SAT \Leftrightarrow f(F) \in CLIQUE$ (Beweis folgt)

g) f ist in polynomieller Zeit berechenbar

Codierung einer Formel als Cliquenproblem

$F = (k_1, k_2, k_3)$ mit $k_1 = x_1 \vee \overline{x_2} \vee x_3$ $\quad k_2 = \overline{x_1} \vee x_2 \vee \overline{x_4}$ $\quad k_3 = \overline{x_1} \vee \overline{x_2} \vee \overline{x_3}$



Gibt es in dem Graphen eine 3-Clique?

Korrektheit der Transformation $3SAT \mapsto CLIQUE$

Gegeben $F = (k_1, ..., k_m)$ mit $k_i = z_{i1} \vee z_{i2} \vee z_{i3}$ und $z_{ij} \in \{x_1, ..., \overline{x_n}\}$

Setze $f(F) := (G_F, m)$ mit $G_F := (V, E)$, wobei

$V := \{v_{ij} \mid 1 \leq i \leq m, 1 \leq j \leq 3\}$ und $E := \{ \{v_{ij}, v_{i'j'}\} \mid i \neq i' \wedge z_{ij} \neq \overline{z_{i'j'}}\}$

Es sei $f(F) \in CLIQUE$

– Dann hat G_F eine m-Clique V_c, d.h.$\{v_{ij}, v_{i'j'}\} \in E$ für alle $v_{ij}, v_{i'j'} \in V_c$

– Per Konstruktion von E enthält V_c für jedes i genau einen Knoten v_{ij} und für je zwei Knoten $v_{ij}, v_{i'j'} \in V_c$ gilt $z_{ij} \neq \overline{z_{i'j'}}$

– Belegen aller zu $v_{ij} \in V_c$ gehörigen z_{ij} mit 1 ist widerspruchsfrei möglich und erfüllt alle Klauseln k_i

Also gilt $F \in 3SAT$

Ist umgekehrt $F \in 3SAT$, so gibt es eine erfüllende Belegung der z_{ij}

– Wähle aus jeder Klausel k_i ein Literal mit dem Wert 1

– Dann bilden die zugehörigen Knoten eine m-Clique in G_F

Also gilt $f(F) \in CLIQUE$

$$\Downarrow$$

$$\boldsymbol{3SAT \leq_p CLIQUE}$$

Das Vertex Cover Problem ist $\mathcal{NP}$-vollständig

$VC = \{(G,k) \mid G \text{ Graph} \wedge \exists V' \subseteq V.\, |V'| \leq k \wedge V' \text{ Knotenüberdeckung von } G\}$

1. Zeige $VC \in \mathcal{NP}$:

a) Rate eine Kantenmenge $V' \subseteq V$

b) Prüfe $|V'| \leq k$ *maximal* $|V'|$ *Schritte*

Prüfe: $\forall \{v,v'\} \in E.\, v \in V' \vee v' \in V'$ *maximal* $|V'| * |E| \leq |V|^3$ *Schritte*

c) Gesamte Anzahl der Schritte ist in $\mathcal{O}(|V|^3)$

2. Zeige $CLIQUE \leq_p VC$: (vgl. Einheit 5.2)

e) Es ist V_c eine Clique in $G = (V, E)$

$\Leftrightarrow \forall v, v' \in V_c.\, v \neq v' \Rightarrow \{v, v'\} \in E$ (Definition)

$\Leftrightarrow \forall \{v, v'\} \notin E.\, v \notin V_c \vee v' \notin V_c$ (Kontraposition)

$\Leftrightarrow \forall \{v, v'\} \in E^c.\, v \in V - V_c \vee v' \in V - V_c$ (Positive Formulierung)

$\Leftrightarrow V - V_c$ Knotenüberdeckung des Komplementgraphen $G^c = (V, E^c)$

Setze $f(G,k) := (G^c, |V| - k)$

f) Es folgt $(G,k) \in CLIQUE \Leftrightarrow f(G,k) \in VC$

g) f ist in polynomieller Zeit $\mathcal{O}(|V|^2)$ berechenbar

Gerichteter Hamiltonscher Kreis (DHC)

Gegeben ein gerichteter Graph $G = (V, E)$

Gibt es in G einen Hamiltonschen Kreis?

(Liste disjunkter Knoten $(v_{i_1}, .., v_{i_{|V|}})$ mit $(v_{i_j}, v_{i_{j+1}}) \in E$ und $(v_{i_{|V|}}, v_{i_1}) \in E$)

$$DHC = \{\, G \mid G = (V, E) \text{ gerichteter Graph} \\ \wedge\ \exists c{=}(v_{i_1}, ..v_{i_{|V|}}).\ c \text{ Hamiltonscher Kreis in } G\}$$

(Directed Hamiltonian Circuit)

1. Zeige $DHC \in \mathcal{NP}$:

a) Rate Zyklus $(v_{i_1}, .., v_{i_n})$

b) Prüfe ob alle $(v_{i_j}, v_{i_{j+1}})$ und (v_{i_n}, v_{i_1}) Kanten aus E sind

c) Anzahl der Schritte ist maximal $n * |E| \ \in \mathcal{O}(|n|^3)$

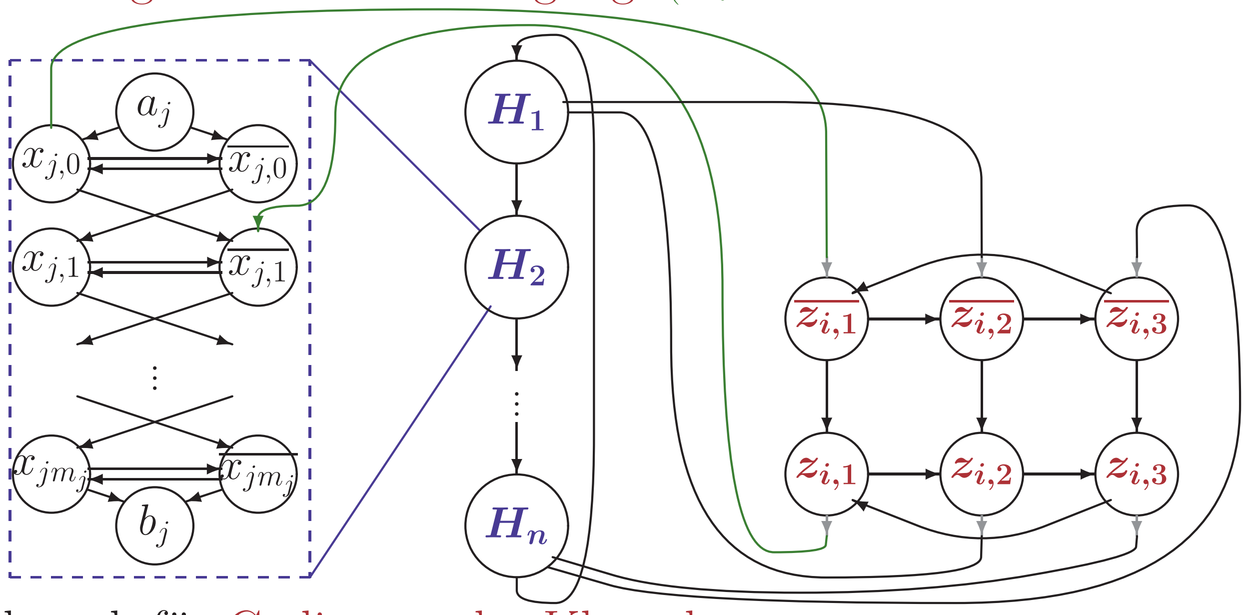
Ausgangspunkt für Beweis der $\mathcal{NP}$-Vollständigkeit von TSP

2. Zeige $3SAT \leq_p DHC$:

– Gegeben $F = (k_1, .., k_m)$ mit $k_i = z_{i1} \vee z_{i2} \vee z_{i3}$ und $z_{ij} \in \{x_1, .., \overline{x_n}\}$

e) Konstruiere Graphen $f(F) \equiv (G)$ wie folgt

- Codierung der Variablenbelegung (m_i: maximales Vorkommen von $x_j/\bar{x}_j$)



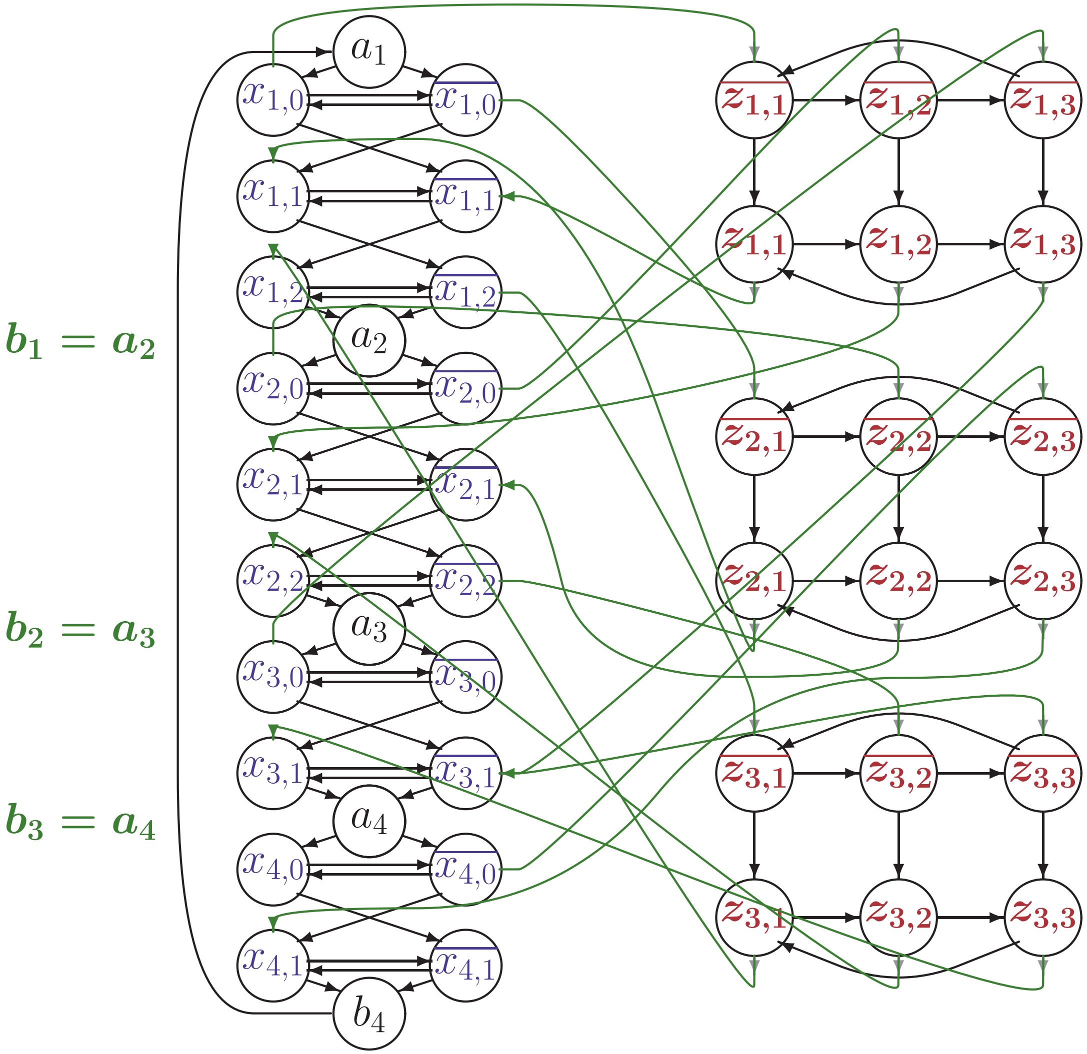
- Teilgraph für Codierung der Klauseln
- Verbindungskanten zur Zwischen Variablen und Klauselliteralen

f) Dann gilt $F \in 3SAT \Leftrightarrow f(F) \in DHC$ (Beweis aufwendig HMU p. 465)

g) f ist in polynomieller Zeit berechenbar

Codierung einer Formel als DHC Problem

$\boldsymbol{F=(k_1, k_2, k_3)}$ mit $\boldsymbol{k_1=x_1 \vee \overline{x_2} \vee x_3}$ $\boldsymbol{k_2=\overline{x_1} \vee x_2 \vee \overline{x_4}}$ $\boldsymbol{k_3=\overline{x_1} \vee \overline{x_2} \vee \overline{x_3}}$



Hamiltonscher Kreis (Hamiltonian Circuit)

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

Gibt es in G einen Hamiltonschen Kreis?

(Liste disjunkter Knoten $(v_{i_1}, .., v_{i_{|V|}})$ mit $\{v_{i_j}, v_{i_{j+1}}\} \in E$ und $\{v_{i_{|V|}}, v_{i_1}\} \in E$)

$$HC = \{ G \mid G \text{ Graph } \wedge G \text{ hat Hamiltonschen Kreis}\}$$

1. Zeige $HC \in \mathcal{NP}$: (Beweis wie bei DHC)

2. Zeige $DHC \leq_p HC$

– Gegeben gerichteter Graph $G_d = (V_d, E_d)$

e) Konstruiere ungerichteten Graphen $f(G_d) \equiv G = (V, E)$

mit $v^0, v^1, v^2 \in V$ für $v \in V_d$

und $\{v^0, v^1\}, \{v^1, v^2\} \in E$ für $v \in V_d$ und $\{v^2, v'^0\} \in E$ für $(v, v') \in E_d$

f) Dann gilt $G_d \in DHC \Leftrightarrow f(G_d) \in HC$ (Beweis folgt)

g) f ist in polynomieller Zeit berechenbar

Korrektheit der Transformation: $G_d \in DHC \Leftrightarrow f(G_d) \in HC$

$v^0, v^1, v^2 \in V$ für $v \in V_d$, $\{v^0, v^1\}, \{v^1, v^2\} \in E$ für $v \in V_d$ und $\{v^2, v'^0\} \in E$ für $(v, v') \in E_d$

Ist $G_d \in DHC$, so gibt es einen DHC $(v_{i_1}, .., v_{i_n})$ in G_d

– In diesem Fall ist $(v_{i_1}^0, v_{i_1}^1, v_{i_1}^2, .., v_{i_n}^2)$ ein Hamiltonscher Kreis in G

Also $\boldsymbol{f(G_d) \in HC}$

Gilt $f(G_d) \in HC$, dann gibt es einen Hamiltonkreis $(u_{j_1}, .., u_{j_{3n}})$ in G

– Da jeder Knoten v_i^1 nur v_i^0 und v_i^2 als Nachbarn hat und sonst nur Kanten zwischen v_i^2 und einem v_j^0 verlaufen muß der Kreis die Gestalt $(v_{i_1}^0, v_{i_1}^1, v_{i_1}^2, .., v_{i_n}^0, v_{i_n}^1, v_{i_n}^2)$ oder $(v_{i_1}^2, v_{i_1}^1, v_{i_1}^0, .., v_{i_n}^2, v_{i_n}^1, v_{i_n}^0)$ haben

– Da G ungerichtet ist, sind beide Kreise identisch, und wir können einen gerichteten Hamiltonschen Kreis $(v_{i_1}, .., v_{i_n})$ für G_d extrahieren

Also gilt $\boldsymbol{G_d \in DHC}$

Das Travelling Salesman Problem ist $\mathcal{NP}$-vollständig

Gegeben n Städte, Reisekostentabelle $c_{i,j}$, Kostenlimit B

Gibt es eine Rundreise durch alle Städte mit Gesamtkosten B?

$$TSP = \{ (c_{1,2}, ..c_{n-1,n}), B \mid \exists\pi{:}\{1..n\}\to\{1..n\}.\ \pi \text{ bijektiv} \wedge \sum_{i=1}^{n-1} c_{\pi(i),\pi(i+1)} + c_{\pi(n),\pi(1)} \leq B\}$$

1. Zeige $TSP \in \mathcal{NP}$:

a) Rate eine Rundreise $\pi{:}\{1..n\}\to\{1..n\}$ — *darstellbar als Liste $(\pi(1)..\pi(n))$*

b/c) Prüfe $\sum_{i=1}^{n-1} c_{\pi(i)\pi(i+1)} + c_{\pi(n)\pi(1)} \leq B$ — *maximal n Schritte*

2. Zeige $HC \leq_p TSP$:

e) Es ist $(v_{i_1}, .., v_{i_n})$ ein Hamiltonscher Kreis in $G = (V, E)$ mit $n = |V|$

$\Leftrightarrow$ $(i_1, .., i_n)$ Rundreise durch n Städte mit Kosten n,
wobei $c_{ij} = 1$ genau dann, wenn $\{v_i, v_j\} \in E$ (sonst größer)

$\Leftrightarrow$ $(i_1, .., i_n)$ Lösung des entsprechenden TSP $((c_{1,2}, ..c_{n-1,n}), n)$

Setze $f(G) := ((c_{1,2}, ..c_{n-1,n}), |V|)$ mit $c_{i,j} = \begin{cases} 1 & \text{falls } \{v_i, v_j\} \in E \\ 2 & \text{sonst} \end{cases}$

f) Es folgt $G \in HC \Leftrightarrow f(G) \in TSP$

g) f ist in polynomieller Zeit $\mathcal{O}(|V|^2)$ berechenbar

Das Rucksackproblem (Knapsack)

Gegeben n Objekte mit Gewichten g_i und Nutzenwerten a_i
Rucksack mit Gewichtslimit G, Minimalnutzwert A

Gibt es eine Bepackung des Rucksatz mit Mindestnutzwert A und Maximalgewicht G?

$$KP = \{(g_1..g_n,\ a_1..a_n,\ G,\ A) \mid \exists J \subseteq \{1..n\}.\ \sum_{i\in J} g_i \leq G \wedge \sum_{i\in J} a_i \geq A\}$$

1. Zeige $KP \in \mathcal{NP}$:

a) Rate Menge von Gegenständen $J\subseteq\{1..n\}$

b) Prüfe $\sum_{i\in J} g_i \leq G$ und $\sum_{i\in J} a_i \geq A$

c) Anzahl der Schritte ist maximal $2|J| \in \mathcal{O}(n)$

Das Rucksackproblem ist $\mathcal{NP}$-vollständig

2. Zeige $3SAT \leq_p KP$:

– Gegeben $F = (k_1, ..., k_m)$ mit $k_i = z_{i1} \vee z_{i2} \vee z_{i3}$ und $z_{ij} \in \{x_1, ..., \overline{x_n}\}$

– Codiere Anzahl der Vorkommen von Literalen in den Klauseln in Dezimalstellen der Gewichte und Nutzenwerte

e) Konstruiere Rucksackproblem $f(F) \equiv (g_1, ..g_{2m+2n}, a_1, ..a_{2m+2n}, G, A)$
wobei alle a_j und g_j $m+n$-stellige Zahlen sind mit $g_j = a_j$ und

- a_j: Anzahl der x_j in k_i an Stelle $i \leq m$, 1 an Stelle $m+j$, sonst 0 $(j \leq n)$
- $a_{n+j} \equiv b_j$: Anzahl der $\bar{x}_j$ in k_i an Stelle $i \leq m$, 1 an Stelle $m+j$, sonst 0 $(j \leq n)$
- $a_{2n+i} \equiv c_i$: 1 an Stelle $m+i$, sonst 0 $(i \leq m)$
- $a_{2n+m+i} \equiv d_i$: 2 an Stelle $m+i$, sonst 0 $(i \leq m)$

Setze $A \equiv G = \underbrace{4...4}_{m-mal}\underbrace{1...1}_{n-mal}$

f) Dann gilt $F \in 3SAT \Leftrightarrow f(F) \in KP$ (Beweis folgt)

g) f ist in polynomieller Zeit berechenbar

Codierung einer Formel als Rucksackproblem

$F=(k_1,k_2,k_3)$ mit $k_1=x_1\vee\overline{x_2}\vee x_3$ $k_2=\overline{x_1}\vee x_2\vee\overline{x_4}$ $k_3=\overline{x_1}\vee\overline{x_2}\vee\overline{x_3}$

$A = 444\,1111$

$a_1 = 100\,1000$	$b_1 = 011\,1000$	$c_1 = 100\,0000$	$d_1 = 200\,0000$
$a_2 = 010\,0100$	$b_2 = 101\,0100$	$c_2 = 010\,0000$	$d_2 = 020\,0000$
$a_3 = 100\,0010$	$b_3 = 001\,0010$	$c_3 = 001\,0000$	$d_3 = 002\,0000$
$a_4 = 000\,0001$	$b_4 = 010\,0001$		

$(1,1,0,0)$ ist erfüllende Belegung

$$a_1+a_2+b_3+b_4+c_1+c_3+d_1+d_2+d_3=A$$

Korrektheit der Transformation: $F \in 3SAT \Leftrightarrow f(F) \in KP$

- a_j: Anzahl der x_j in k_i an Stelle $i \leq m$, 1 an Stelle $m+j$, sonst 0 $\qquad (j \leq n)$
- b_j: Anzahl der $\bar{x}_j$ in k_i an Stelle $i \leq m$, 1 an Stelle $m+j$, sonst 0 $\qquad (j \leq n)$
- c_i: 1 an Stelle $m+i$, sonst 0 $\qquad d_i$: 2 an Stelle $m+i$, sonst 0 $\qquad (i \leq m)$
- $g_j = a_j$ für alle j $\qquad A \equiv G = \underbrace{4\ldots4}_{m-mal}\underbrace{1\ldots1}_{n-mal}$

Ist $F \in 3SAT$, so gibt es eine erfüllende Belegung der x_j

– Für $j \leq n$ wähle a_j falls $x_j = 1$ und b_j sonst

Dann haben in der Summe alle Stellen $m+j$ den Wert 1

und die Stellen $i \leq n$ einen Wert aus $\{1..3\}$, da jedes k_i erfüllt wird

– Ergänzung der Stellen $i \leq n$ mit c_i und d_i zu 4 liefert A und G

Also $f(F) \in KP$

Gilt $f(F) \in KP$, so gibt es eine Bepackung die genau den Wert A ergibt

– Diese enthält für Stelle $m+j$ entweder a_j (setze $x_j := 1$) oder b_j ($x_j := 0$)

– Wegen $c_i + d_i = 3$ muß die Summe der a_j und b_j in der Bepackung an jeder Stelle $i \leq m$ mindestens den Wert 1 haben

– Also kommt in Klausel k_i mindestens ein Literal mit dem Wert 1 vor

Damit erfüllt die gewählte Belegung die Formel F, d.h. $F \in 3SAT$

Das Färbbarkeitsproblem (Graph Coloring)

Gegeben ein Graph $G = (V, E)$ und eine Zahl $k \leq |n|$

Gibt es eine Färbung von V mit k verschiedenen Farben, so daß verbundene Knoten verschiedene Farben haben?

$$GC = \{ (G, k) \mid G{=}(V, E) \text{ Graph } \wedge \exists f_V{:}V \rightarrow \{1..k\}.\, \forall \{u, v\} \in E.\, f_V(u) \neq f_V(v) \}$$

1. Zeige $KP \in \mathcal{NP}$:

a) Rate Färbung $f_V{:}V \rightarrow \{1..k\}$ des Graphen

b) Prüfe $\forall \{u, v\} \in E. f_V(u) \neq f_V(v)$

c) Anzahl der Schritte ist maximal $|E| \in \mathcal{O}(|V|^2)$

Das Färbbarkeitsproblems ist $\mathcal{NP}$-vollständig

2. Zeige $3SAT \leq_p GC$:

– Gegeben $F = (k_1, ..., k_m)$ mit $k_i = z_{i1} \vee z_{i2} \vee z_{i3}$ und $z_{ij} \in \{x_1, ..., \overline{x_n}\}$

e) Konstruiere Färbungsproblem $f(F) \equiv (G, 3)$ wie folgt

· Teilgraph für Codierung der Variablenbelegung

Wähle $V_{var} = \{u,\ x_1, ..., \overline{x_n}\}$

und $E_{var} = \{\{u, x_1\}, \{u, \overline{x_1}\}, \{x_1, \overline{x_1}\}, ..\{u, x_n\}, \{u, \overline{x_n}\}, \{x_n, \overline{x_n}\}\}$

Bei 3-Färbbarkeit erhalten x_i und $\overline{x_i}$ verschiedene Farben aus 0 oder 1

· Teilgraph für Codierung der Klauseln

Wähle $V_k = \{v,\ a_1, b_1, c_1, y_1, z_1, .., a_m, b_m, c_m, y_m, z_m\}$

und $E_k = \{\{v, y_1\}, \{v, z_1\}, \{a_1, y_1\}, \{a_1, z_1\}, \{b_1, y_1\}, \{c_1, z_1\}, \{b_1, c_1\},$
$...\{v, y_m\}, ..\{b_m, c_m\},\ \{u, v\}\}$

Knoten v erhält Farbe 0 oder 1

· Verbindungskanten zur Codierung der Klauselliterale

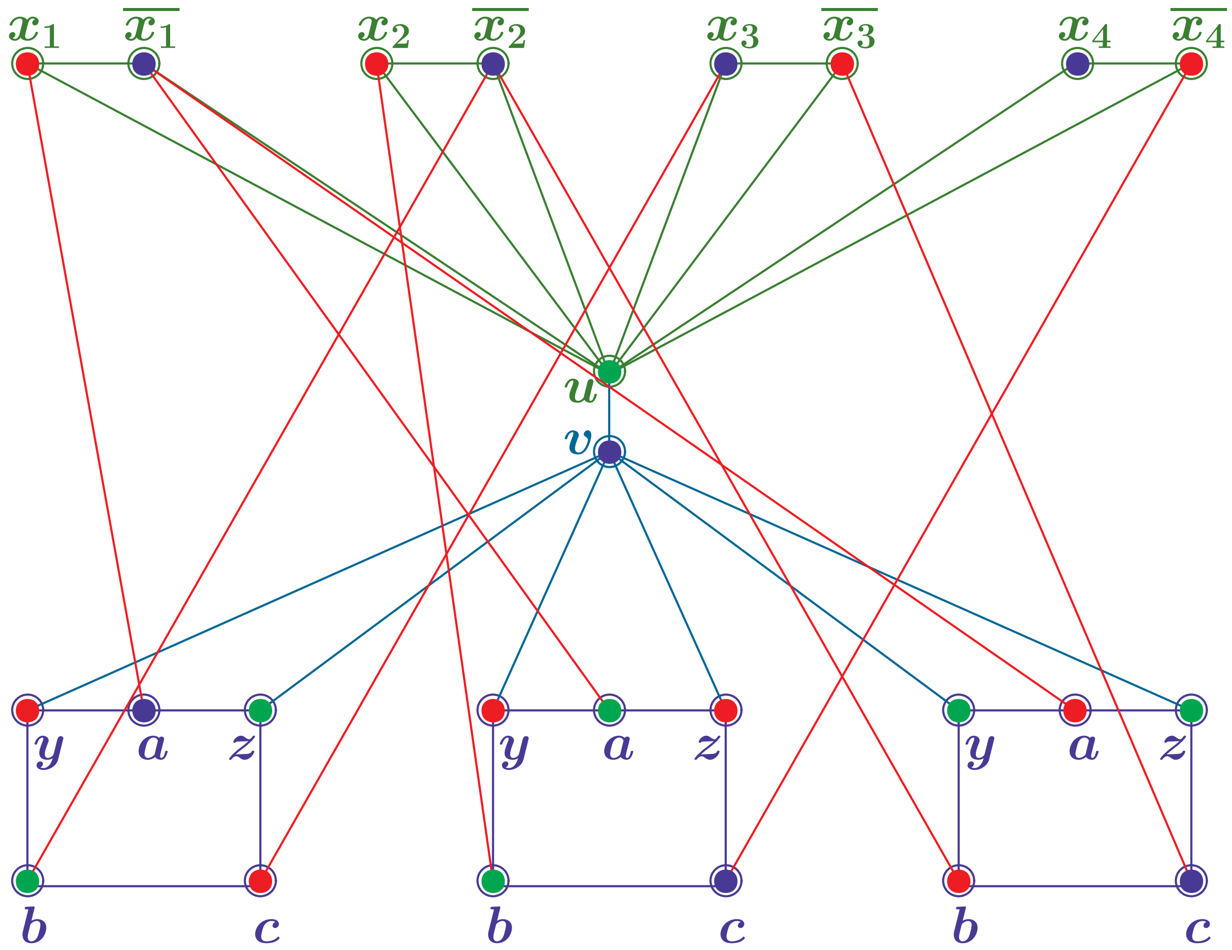
$E_{lit} = \{\{a_1, z_{11}\}, \{b_1, z_{12}\}, \{c_1, z_{13}\}, ..\{a_m, z_{m1}\}, \{b_m, z_{m2}\}, \{c_m, z_{m3}\}\}$

g) $G{:=}(V_{var} \cup V_k,\ E_{var} \cup E_k \cup E_{lit})$ ist in polynomieller Zeit berechenbar

f) Dann gilt $F \in 3SAT \Leftrightarrow f(F) \in GC$ (Beweis folgt)

Codierung einer Formel als Färbungsproblem

$F=(k_1, k_2, k_3)$ mit $k_1 = x_1 \vee \overline{x_2} \vee x_3$ $\quad k_2 = \overline{x_1} \vee x_2 \vee \overline{x_4}$ $\quad k_3 = \overline{x_1} \vee \overline{x_2} \vee \overline{x_3}$



Gibt es für den Graphen eine 3-Färbung?

Korrektheit der Transformation: $F \in 3SAT \Leftrightarrow f(F) \in GC$

- Ist $F \in 3SAT$, so gibt es eine erfüllende Belegung der x_j
  - Wähle $f_v(x_i), f_v(\overline{x_i}) \in \{\mathbf{0},\mathbf{1}\}$ entsprechend, $f_v(u)=\mathbf{2}$ und $f_v(v)=\mathbf{0}$.
  - Da jedes k_i erfüllbar ist, kann einer der a_i, b_i, c_i die Farbe **0** erhalten
  - Die anderen 4 Knoten bilden eine Kette und werden abwechselnd gefärbt
  - Also gibt es eine 3-Färbung des Graphen und somit $f(F) \in GC$

- Ist $f(F) \in GC$ dann ist o.B.d.A. $f_v(u)=\mathbf{2}$ und $f_v(v)=\mathbf{0}$
  - Wähle Belegung der x_i entsprechend der Färbung von x_i
  - Wäre Klausel k_i nicht erfüllt, so wäre die Farbe der a_i, b_i, c_i **1** oder **2**
  - Wegen $f_v(b_i) \neq f_v(c_i)$ und $f_v(v)=\mathbf{0}$ wäre dann $f_v(y_i) \neq f_v(z_i) \in \{\mathbf{1},\mathbf{2}\}$
  - Dies widerspricht der Färbbarkeit, da a_i ebenfalls mit **1** oder **2** gefärbt ist.
  - Also $F \in 3SAT$

$$\Downarrow$$

$$\boldsymbol{3SAT \leq_p GC}$$

WEITERE $\mathcal{NP}$-VOLLSTÄNDIGE GRAPHENPROBLEME

- **Independent Set** (HMU §10.4.2) $\boldsymbol{3SAT \leq_p IS}$
  - Gegeben ein Graph $G = (V, E)$ der Größe n und eine Zahl $k \leq |V|$.
  - Gibt es in G eine unabhängige Knotenmenge der Größe k?

  $\boldsymbol{IS} = \{ (G, k) \mid G{=}(V, E) \text{ Graph } \wedge \exists V_i{\subseteq}V.\ |V_i| \geq k \ \wedge \forall u, v \in V_i.\ \{u, v\} \notin E \}$

- **Subgraph Isomorphism** $\boldsymbol{CLIQUE \leq_p SGI}$
  - Gegeben zwei Graphen $G_1 = (V_1, E_1)$ und $G_2 = (V_2, E_2)$.
  - Gibt es einen Subgraphen H von G_1, der isomorph zu G_2 ist?

  $\boldsymbol{SGI} = \{ (G_1, G_2) \mid G_1, G_2 \text{ Graphen } \wedge \exists H \text{ Graph. } H{\sqsubseteq}G_1 \ \wedge H \cong G_2 \}$

- **Largest Common Subgraph** $\boldsymbol{SGI \leq_p LCS}$
  - Gegeben Graphen $G_1{=}(V_1, E_1)$ und $G_2{=}(V_2, E_2)$ und eine Zahl $k \leq |G_1|$
  - Gibt es isomorphe Subgraphen H_1 von G_1 und H_2 von G_2 der Größe k?

  $\boldsymbol{LCS} = \{ (G_1, G_2, k) \mid G_1, G_2 \text{ Graphen } \wedge k \leq |G_1| \ \wedge \exists H_1, H_2 \text{ Graphen. } H_1{\sqsubseteq}G_1 \ \wedge H_2{\sqsubseteq}G_2 \ \wedge H_1 \cong H_2 \ \wedge |H_1| \geq k \}$

WEITERE $\mathcal{NP}$-VOLLSTÄNDIGE PROBLEME

- **Partitionsproblem** $\hfill$ ***KP*** $\boldsymbol{\leq_p}$ ***PART***
  - Gegeben n Objekte mit Wert $b_1, ..., b_n$.
  - Gibt es eine Aufteilung der Objekte in zwei gleichwertige Stapel?

  $\boldsymbol{PART} = \{ b_1, ..b_n \mid b_i \in \mathbb{N} \wedge \exists I \subseteq \{1..n\}.\ \sum_{i \in I} b_i = \sum_{i \in \overline{I}} b_i \}$

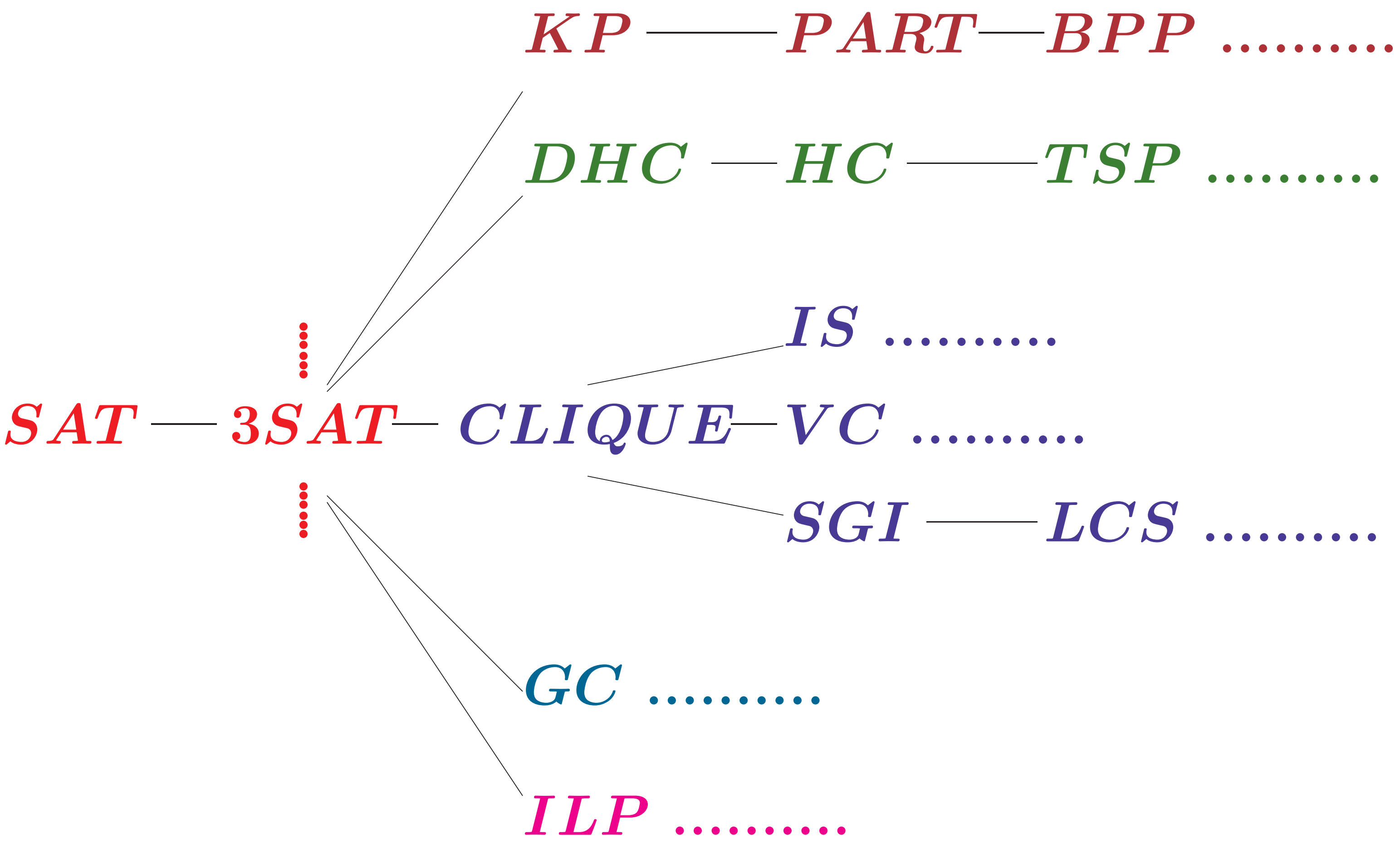
- **Binpacking** $\hfill$ ***PART*** $\boldsymbol{\leq_p}$ ***BPP***
  - Gegeben n Objekte der Größe $a_1, ...a_n$ und k Behälter der Größe b
  - Kann man alle Objekte in den Behältern unterbringen?

  $\boldsymbol{BPP} = \{(a_1, ..a_n, b, k) \mid \exists f{:}\{1..n\} \to \{1..k\}.\ \forall j \leq k.\ \sum_{i \in \{i \mid f(i)=j\}} a_i \leq b\}$

- **Multiprozessor-Scheduling** $\hfill$ ***MPS*** $\boldsymbol{\hat{=}}$ ***BPP***
  - Gegeben n Prozesse j_i mit Laufzeit $t(j_i)$, m Prozessoren, Deadline t_D.
  - Gibt es eine Verteilung der Prozesse auf die Prozessoren, so daß bei Startzeit t_0 alle Prozesse vor der Zeit t_D beendet sind?

- **Integer Linear Programming** $\hfill$ ***3SAT*** $\boldsymbol{\leq_p}$ ***ILP***
  - Gegeben eine $k \times k$ Matrix A und einen Vektor $\vec{b} \in \mathbb{Z}^k$
  - Gibt es ein $\vec{x} \in \mathbb{Z}^k$, welches das lineare Ungleichungssystem $A * \vec{x} \geq \vec{b}$ löst?

$\mathcal{NP}$-vollständige Probleme – Reduktionsstruktur



Es gibt weitere wichtige Schwierigkeitsgrade

- **$co\text{–}\mathcal{NP}$: Probleme mit Komplement in $\mathcal{NP}$**
  - Menge der allgemeingültigen Formeln (Komplement von SAT)
  - (Rationale) lineare Programmierung (analog zu ILP)

  Wenn $\mathcal{P}\neq\mathcal{NP}$, dann gibt es i.a. keinen effizienten Erfolgstest

- **Es gibt Probleme in $\mathcal{NP} \cap co\text{–}\mathcal{NP}$**
  - Lineare Programmierung liegt auch in $\mathcal{NP}$

- **Sehr wahrscheinlich gilt $co\text{–}\mathcal{NP}\neq\mathcal{NP}$**

  $\mathcal{NP} = co\text{–}\mathcal{NP} \;\Leftrightarrow\; \mathcal{NPC} \cap co\text{–}\mathcal{NP}\neq\emptyset$

  $\Rightarrow$: offensichtlich, da $\mathcal{NPC}\neq\emptyset$

  $\Leftarrow$: Ist $L \in \mathcal{NPC} \cap co\text{–}\mathcal{NP}$ so gilt $\overline{L'} \in \mathcal{NP}$ für jedes $L' \in co\text{–}\mathcal{NP}$

  also $\overline{L'} \leq_p L$ und damit $L' \leq_p \overline{L} \in \mathcal{NP}$

Zeit- und Platzkomplexität

- $PSPACE \subseteq EXPTIME$
  – Eine terminierende Maschine, die maximal $f(n)$ Bandzellen aufsucht, terminiert nach maximal $c^{1+\mathrm{f(n)}}$ Schritten
  – Für $|\Gamma| + |Q| = c$ sind maximal $c^{1+\mathrm{f(n)}}$ Konfigurationen möglich
  $NPSPACE \subseteq NEXPTIME$ gilt aus dem gleichen Grund

- $PSPACE = NPSPACE$ HMU §11.2.3
  – Eine NTM mit Platzverbrauch $p(n)$ kann deterministisch mit $\mathcal{O}(p(n)^2)$ Bandzellen simuliert werden (Satz von Savitch)
  – Der Zeitaufwand ist allerdings exponentiell höher als bei der NTM

- **Es gibt $PSPACE$-vollständige Probleme**
  – QBF: Ist eine (geschlossene) quantifizierte boolesche Formel wahr?
  – **In-Place Acceptance**: Kann eine gegebene DTM jedes Wort w ihrer Sprache mit Platzbedarf $|w|$ akzeptieren? Asteroth/Baier §4.5

Quantifizierte Boolesche Formeln

- **Aussagenlogik mit booleschen Quantoren**
  - Ausagenlogische Variablen $P, Q, R, ...,$ Konstante t und f
  - Formeln $\neg F,\ E \wedge F,\ E \vee F,\ E \Rightarrow F,\ (\forall P)F,\ (\exists P)F$
  - Wert von $(\forall P)F$ entspricht dem von $F[t/P] \wedge F[f/P]$
  - Wert von $(\exists P)F$ entspricht dem von $F[t/P] \vee F[f/P]$
  - Wert anderer Formeln wie in gewöhnlicher Aussagenlogik

  QBF ist die Menge der geschlossenen QB-Formeln mit Wert 1

- $\boldsymbol{QBF \in PSPACE}$
  - Auswerten aussagenlogischer Formeln braucht linearen Platz
  - $(\forall P)F = F[t/P] \wedge F[f/P]$ wird kaskadisch ausgewertet
  - Gesamtbedarf, einschließlich Zwischenspeicherung ist quadratisch

- **Für alle $\boldsymbol{L \in PSPACE}$ gilt $\boldsymbol{L \leq_p QBF}$** HMU §11.3.4
  - Codierung deterministischer Maschinen ähnlich wie im Satz von Cook

Wichtige Vertreter verschiedener Klassen

- **Isomorphie ungerichteter Graphen** $\mathcal{NP}$-unvollständig
- **Zuverlässigkeit von Netzwerken** $\mathcal{NP}$-hart, vermutlich nicht in $\mathcal{NP}$
  – Wahrscheinlichkeit für fehlerfreie Verbindung zwischen zwei Knoten
- **Minimale äquivalente Schaltkreise** $\mathcal{NP}$-hart, nicht in $\mathcal{NP}$ (“$\mathbf{\Sigma_2}$”)
  – Bestimme optimale Größe einer Schaltung
- **Sprache regulärer Ausdrücke** $PSPACE$-vollständig
  – Ist $L(E) = \Sigma^*$ für einen beliebigen regulären Ausdruck über Σ?
- **Strategische Spiele** $PSPACE$-vollständig
  – Details in Garey/Johnson Seite 254ff
- **TSP*: Bestimmung aller Rundreisen mit gegebenen Kosten**
  – Unrealistische Problemstellung: zu viele Lösungen $EXPSPACE$

Komplexitätsklassenhierarchie