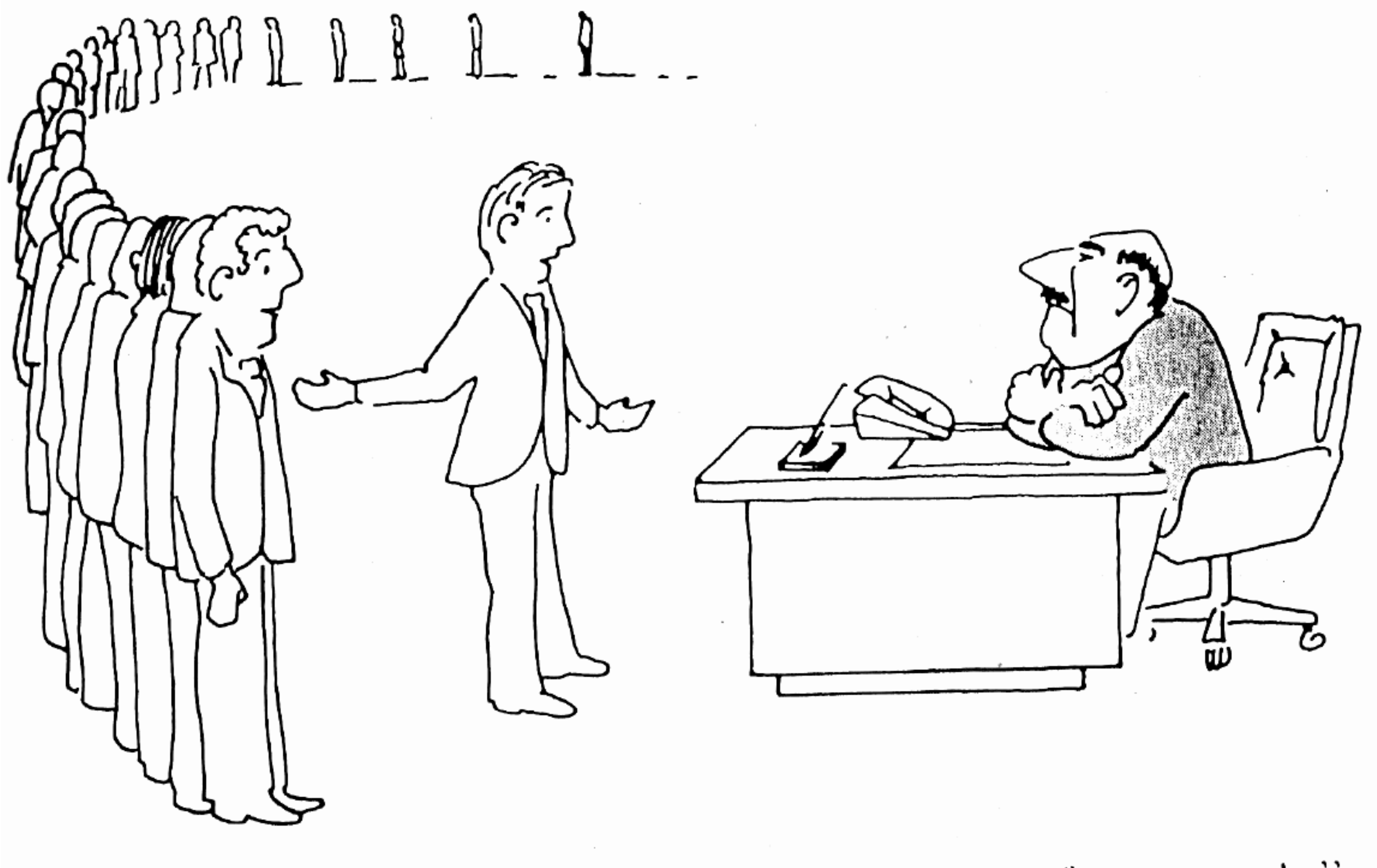
Theoretische Informatik II

Einheit 8.4

NP-Vollständigkeit

1. Reduzierbarkeit und Vollständigkeit von Klassen
2. Der Satz von Cook
3. NP-vollständige Probleme

Das $\mathcal{P}$–$\mathcal{NP}$ Problem
Was tun, wenn ein Problem nicht effektiv lösbar zu sein scheint?



"I can't find an efficient algorithm, but neither can all these famous people."

Vielleicht der einzig mögliche Weg

Das $\mathcal{P}$–$\mathcal{NP}$ Problem

Gilt $\mathcal{P}=\mathcal{NP}$ oder $\mathcal{P}\neq\mathcal{NP}$?

- **Eines der wichtigsten offenen Probleme der TI**
  - Sind nichtdeterministisch lösbare Probleme effizient lösbar?
  - Seit mehr als 30 Jahren ungeklärt, möglicherweise unlösbar
- **Mehr als 1000 algorithmische Probleme betroffen**
  - Suchprobleme (Travelling Salesman, ...)
  - Reihenfolgenprobleme (Scheduling, Binpacking, ...)
  - Graphenprobleme (Clique, Vertex cover, ...) $\mapsto$ Operations Research
  - Logische Probleme (Erfüllbarkeit, ...) $\mapsto$ Model Checking, Hardwareverifikation
  - Zahlenprobleme (Primzahltest, ...) $\mapsto$ Kryptographie, IT Sicherheit
- **Indizien sprechen gegen $\mathcal{P}=\mathcal{NP}$**
  - Zu viele $\mathcal{NP}$-Probleme ohne bekannte polynomielle Lösung
  - Mehr als 1000 äquivalente Probleme in der 'schwersten Teilklasse' von $\mathcal{NP}$

Wie analysiert man die Frage "$\mathcal{P}=\mathcal{NP}$ oder $\mathcal{P}\neq\mathcal{NP}$"?

- **Untersuche die "schwierigsten" $\mathcal{NP}$-Probleme**
  - Kann man eines davon effizient lösen?
  - Wenn ja, dann gilt $\mathcal{P}=\mathcal{NP}$
  - Wenn nein, dann gibt es ein Beispiel für $\mathcal{P}\neq\mathcal{NP}$
- **Was heißt "M ist schwierigstes $\mathcal{NP}$-Problem"?**
  - Jedes andere $\mathcal{NP}$-Problem M' ist leichter als M
  - Lösungen für M' können in Lösungen für M' umgewandelt werden
  - Transformation der Lösung ist effizient

$$\Downarrow$$

Polynomielle Reduktion

Polynomielle Reduzierbarkeit

- **$M\subseteq X^*$ polynomiell reduzierbar auf $M'\subseteq Y^*$** Definition D
  - Es gibt eine in polynomieller Zeit berechenbare totale Funktion $f{:}X^*\to Y^*$ mit $M{=}f^{-1}(M)$
  - Schreibweise: $M{\leq_p}M'$
- **Reduzierbarkeit $\equiv$ geringere Komplexität** Lemma E
  - $M{\leq_p}M' \wedge M'\in\mathcal{P} \;\Rightarrow\; M\in\mathcal{P}$
  - $M{\leq_p}M' \wedge M'\in\mathcal{NP} \;\Rightarrow\; M\in\mathcal{NP}$

Beweis:

- $\chi_M(x){=}1 \Leftrightarrow x\in M \Leftrightarrow f(x)\in M' \Leftrightarrow \chi_{M'}(f(x)){=}1 \Leftrightarrow (\chi_{M'}\circ f)(x){=}1$
- $\chi_{M'}\circ f$ ist in polynomieller Zeit berechenbar, wenn dies für $\chi_{M'}$ gilt

Wichtige Graphentheoretische Definitionen

- Ein (ungerichteter) *Graph* ist ein Paar $G = (V, E)$, wobei V endliche Menge und $E \subseteq \{\, \{v, v'\} \mid v, v' \in V \wedge v \neq v' \,\}$.
  Ein Graph ist darstellbar als Liste $v_1, \ldots, v_n,\ \{v_{i_1}, v'_{i_1}\}, \ldots, \{v_{i_m}, v'_{i_m}\}$.
- Ein Graph $H = (V_H, E_H)$ ist genau dann *Subgraph* des Graphen $G = (V, E)$ ($H \sqsubseteq G$), wenn alle Ecken und Kanten von H auch Ecken bzw. Kanten in G sind:
  $$(V_H, E_H) \sqsubseteq (V, E)\ :\Leftrightarrow\ V_H \subseteq V \wedge E_H \subseteq E$$
- $H = (V_H, E_H)$ ist *isomorph* zu $G = (V, E)$ (kurz: $H \cong G$), wenn die Graphen durch Umbenennung (bijektive Abbildung $h : V_2 \to V$) ineinander überführt werden können:
  $$(V_H, , E_H,) \cong (V, E)\ :\Leftrightarrow\ \exists h : V \to V_H, .(h \text{ bijektiv} \wedge E_H = \{\{h(u), h(v)\} \mid \{u, v\} \in E\})$$
- Die *Größe* $|G|$ eines Graphen $G = (V, E)$ ist die Anzahl $|E|$ seiner Kanten.
- Der *Komplementärgraph* des Graphen $G = (V, E)$ ist der Graph $G^c = (V, E^c)$ mit $E^c = \{\, \{v, v'\} \mid v, v' \in V \,\} - E$.
- Eine *Clique* der Größe k im Graphen $G = (V, E)$ ist eine vollständig verbundene Knotenmenge $V' \subseteq V$ mit $|V| = k$. (Dabei heißt *vollständig verbunden*: $\forall v, v' \in V'.\, v \neq v' \Rightarrow \{v, v'\} \in E$)
- Eine *Knotenüberdeckung* (Vertex cover) des Graphen $G = (V, E)$ ist eine Knotenmenge $V' \subseteq V$ mit der Eigenschaft $\forall \{v, v'\} \in E.\ v \in V' \vee v' \in V'$
- Ein *Hamilton'scher Kreis* im Graphen $G = (V, E)$ ist ein Kreis, der nur aus Kanten aus E besteht und jeden Knoten genau einmal berührt.
  (D.h. eine Permutation $\pi : \{1..n\} \to \{1..n\}$ mit $\forall i < n.\ \{v_{\pi(i)}, v_{\pi(i+1)}\} \in E \wedge \{v_{\pi(n)}, v_{\pi(1)}\} \in E$)
- Ein *gerichteter Graph* ist ein Paar $G = (V, E)$, wobei V endliche Menge und $E \subseteq V \times V$.
  Ein *Hamilton'scher Kreis* im gerichteten Graphen $G = (V, E)$ ist ein gerichteter Kreis, der nur aus Kanten aus E besteht und jeden Knoten genau einmal berührt.

Polynomielle Reduktion auf Graphenproblemen

- **Cliquen Problem**
  - Gegeben ein Graph $G = (V, E)$ der Größe n und eine Zahl $k \leq n$.
  - Gibt es in G eine Clique der Größe k?

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

- **Vertex Cover Problem**
  - Gegeben ein Graph $G = (V, E)$ der Größe n und eine Zahl $k \leq n$.
  - Gibt es eine Teilmenge $V' \subseteq V$ mit höchstens k Elementen, so daß aus jeder Kante in G mindestens eine Ecke in V' liegt?

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

Reduzierbarkeit: Clique $\leq_p$ Vertex Cover

- **Analyse der Eigenschaften**

  V' Knotenüberdeckung von G

  $\Leftrightarrow\ \forall\{v,v'\}\in E.\ v\in V' \vee v'\in V'$

  $\Leftrightarrow\ \forall\{v,v'\}\in E.\ \{v,v'\}\cap V'\neq\emptyset$

  $\Leftrightarrow\ \forall v,v'\in V-V'.\ \{v,v'\}\notin E$

  $\Leftrightarrow\ \forall v,v'\in V-V'.\ v\neq v' \Rightarrow \{v,v'\}\in E^c := \{\ \{v,v'\}\subseteq V \mid v\neq v'\}-E$

  $\Leftrightarrow$ $V-V'$ ist Clique im Komplementgraphen $G^c=(V,E^c)$

- **Transformation der Probleme** (Vertausche G und G^c)

  $(G,k)\in CLIQUE$

  $\Leftrightarrow$ G hat Clique V_c der Mindestgröße k

  $\Leftrightarrow$ G^c hat Knotenüberdeckung $V'=V-V_c$ der Maximalgröße $|V|-k$

  $\Leftrightarrow\ (G^c,|V|-k)\in VC$

  Wähle $f(G,k) := (G^c,|V|-k)$

  f in polynomieller Zeit berechenbar und $CLIQUE = f^{-1}(VC)$ $\surd$

$\mathcal{NP}$-Vollständigkeit

- **$\mathcal{NP}$-hart: nicht leichter als $\mathcal{NP}$**
  - $M' \subseteq X^*$ ist $\mathcal{NP}$-hart, wenn $M \leq_p M'$ für alle $M \in \mathcal{NP}$
- **$\mathcal{NP}$-vollständig: das Schwierigste in $\mathcal{NP}$**
  - $M \subseteq X^*$ ist $\mathcal{NP}$-vollständig, wenn $M \in \mathcal{NP}$ und M $\mathcal{NP}$-hart
  - Schreibweise: $\boldsymbol{M \in \mathcal{NPC}}$

Konsequenzen von $\mathcal{NP}$-Vollständigkeit

- **Alle $\mathcal{NP}$-vollständigen Probleme sind äquivalent**
  - $M, M' \in \mathcal{NPC} \;\Rightarrow\; M' \leq_p M \wedge M \leq_p M'$

- **$\mathcal{NP}$-Vollständigkeit ist leicht nachweisbar, wenn ein $\mathcal{NP}$-vollständiges Problem bekannt ist**
  - $M \in \mathcal{NPC} \;\Leftrightarrow\; M \in \mathcal{NP} \wedge \exists M' \in \mathcal{NPC}.\, M' \leq_p M$
  - $M \in \mathcal{NPC} \;\Leftrightarrow\; \exists M' \in \mathcal{NPC}.\, M' \leq_p M \wedge M \leq_p M'$

- **$\mathcal{NP}$-vollständige Probleme entscheiden "$\mathcal{P} = \mathcal{NP}$"**
  - $\mathcal{P} = \mathcal{NP} \;\Leftrightarrow\; \exists M \in \mathcal{NPC}.\, M \in \mathcal{P}$ Satz G
  - Ist $\mathcal{P} = \mathcal{NP}$ dann sind alle $\mathcal{NP}$-vollständigen Probleme in $\mathcal{P}$
  - $\mathcal{P} \neq \mathcal{NP} \;\Leftrightarrow\; \exists M \in \mathcal{NPC}.\, M \notin \mathcal{P}$
  - Ist $\mathcal{P} \neq \mathcal{NP}$ dann sind alle $\mathcal{NP}$-vollständigen Probleme nicht in $\mathcal{P}$

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

- **Codiere Berechnung einer NTM in einem Problem L**
  - Wenn Berechnung positiv ausfällt, soll Problem lösbar sein
  - Wenn Berechnung negativ ausfällt, soll Problem unlösbar sein
  - Ergibt $M \leq_p L$ für das von der NTM entschiedene Problem M
- **Problem L muß alle polynomiellen NTMs codieren**
  - Ergibt $M \leq_p L$ für jedes $M \in \mathcal{NP}$
- **Welches Problem ist ausdrucksstark genug?**
  - Codiere mögliche Zustandsübergänge durch logische Formeln
  - Problem: Können Zustandsübergänge so kombiniert werden,
    daß Berechnung mit Ergebnis 1 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 $x_1, ..x_n$, 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 Boole'schen 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)

Lösungsalgorithmen für das Erfüllbarkeitsproblem

$SAT = \{k_1..k_m |\ k_i \text{ Klausel über } x_1..x_n \ \wedge\ \exists a_1..a_n \in \{0{,}1\}.\forall j{\le}m.a_1..a_n \text{ erfüllt } k_j\}$

- **Deterministisch**
  - 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**
  - Rate eine erfüllende Belegung der Variablen (falls es eine gibt)
  - Prüfe Belegung durch Auswertung der Formel
  - Polynomielle Laufzeit

$$\Downarrow$$

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

$\mathcal{SAT}$ ist $\mathcal{NP}$-vollständig

- **Gegeben: NTM τ, die ein Problem in polynomieller Zeit löst**
- **Ziel: Codiere Berechnung von τ bei Eingabe w durch Formel in KNF, die genau dann erfüllbar ist, wenn $h_\tau(w) = 1$**
  - Codierung muß in polynomieller Zeit (relativ zu $|w|$) berechenbar sein
- **Vorgehen: Beschreibe mögliche Konfigurationsübergänge von τ durch Klauseln**
  - Codiere Zustand, Kopfposition und Bandzellen durch Literale
  - Es werden nur polynomiell viele Literale und Klauseln benötigt
  - Formel ist erfüllbar, wenn Konfigurationsübergänge zu Berechnung zusammengesetzt werden können

Satz von Cook: Grundannahmen

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

- **L wird von NTM τ entschieden**
  – $\tau = (S, X, \Gamma, \delta, s_0, b)$ mit $S{=}\{s_0, .., s_e\}$, $\Gamma{=}\{x_1, .., x_v\}$, $b{=}x_1$
  – o.B.d.A.: τ hat Menge F von "Endzuständen" in denen τ "verharrt"
  (Ersetze hierzu $\delta(s, a){=}(s', a', h)$
  durch $\delta(s, a){=}(\bar{s}, a', L)$, $\delta(\bar{s}, a){=}(\bar{\bar{s}}, a, R)$, $\delta(\bar{\bar{s}}, a){=}(\bar{s}, a, L)$)
  – o.B.d.A.: Die akzeptierende Ausgabe 1 steht in Bandzelle 0

- **τ zeitbeschränkt durch Polynom $p(n)$**
  – $t_\tau(w){\leq}p(n)$ für jedes Wort $w \in X^*$ mit $|w|{=}n$
  – Es sind genau $p(n)$ Berechnungsschritte als Formel zu codieren

- **τ platzbeschränkt durch $p(n)$**
  – τ kann während der Berechnung maximal $p(n)$ Bandzellen aufsuchen
  – Es reicht die Bandzellen von $-p(n)$ bis $+p(n)$ zu modellieren

Satz von Cook: zu codierende Aussagen

- **Anfangsbedingungen bei Eingabe w**
  - τ startet im Zustand s_0 und der Kopf ist über Bandzelle 0
  - Bandinhalt der Zellen $-p(n)...0...p(n)$ ist $b^{p(n)}w_1..w_n b^{p(n)-n+1}$
- **Randbedingungen für eindeutiges Verhalten**

  Zu jedem Zeitpunkt t der Berechnung gilt
  - τ befindet sich in genau einem Zustand und liest genau eine Bandzelle
  - Jede Bandzelle enthält genau ein Symbol
  - τ wendet genau eine Zeile der Zustandsüberführungstabelle δ an
- **Übergangsbedingungen**
  - Zu jedem Zeitpunkt t sind Zustand, Kopfposition und Bandinhalt mit der Zustandsüberführungstabelle δ verträglich
- **Endbedingung**
  - Nach $p(n)$ Schritten befindet sich τ in einem Endzustand aus F
  - Bandinhalt der Zellen $-p(n)...0...p(n)$ ist $b^{p(n)}1b^{p(n)}$

Symbole in der Codierung einer Berechnung

- $\boldsymbol{z_{t,k}}$, $t\in\{0..p(n)\}$, $k\in\{0..e\}$
  – τ ist nach t Schritten im Zustand s_k

- $\boldsymbol{a_{t,i,j}}$, $t\in\{0..p(n)\}$, $i\in\{-p(n)..p(n)\}$, $j\in\{1..v\}$
  – Der Bandinhalt von Zelle i nach t Schritten ist das Symbol x_j

- $\boldsymbol{s_{t,i}}$, $t\in\{0..p(n)\}$, $i\in\{-p(n)..p(n)\}$
  – τ liest im Schritt t den Bandinhalt von Zelle i

- $\boldsymbol{b_{t,l}}$, $t\in\{0..p(n)\}$, $l\in\{1..v*(e+1)\}$
  – τ verwendet beim Übergang von t nach $t+1$ die Zeile l der Tabelle von δ

- **Insgesamt $\mathcal{O}(\,(p(n))^2 * |S| * |\Gamma|)$ Variablen**

Beschreibe Berechnung von $h_\tau(w)$
als KNF-Formel $\alpha(\tau,w)$ über diesen Variablen

Satz von Cook: Codierung der Anfangsbedingungen

Start im Zustand s_0, Kopf über Bandzelle 0,
Bandinhalt $b^{p(n)} w_1..w_n b^{p(n)-n+1}$

Sei $w_1 = x_{j_1}, \ldots, w_n = x_{j_n}$

Codiere Anfangsbedingungen als Formel A mit

$$
\begin{aligned}
A \;\equiv\;\; & z_{0,0} \wedge s_{0,0} \\
& \wedge\; a_{0,-p(n),1} \wedge \ldots \wedge a_{0,-1,1} \\
& \wedge\; a_{0,0,j_1} \wedge \ldots \wedge a_{0,n-1,j_n} \\
& \wedge\; a_{0,n,1} \wedge \ldots \wedge a_{0,p(n),1}
\end{aligned}
$$

- **A ist in KNF** — Rein konjunktive Formel
- **Größe: $\mathcal{O}(p(n))$** — $2{*}p(n){+}3$ Variablen
- **Berechnungsaufwand: $\mathcal{O}(p(n))$** — Bestimmung von $p(n)$

Satz von Cook: Codierung der Randbedingungen

In jedem Schritt genau ein Zustand, eine Bandzelle ein Symbol pro Bandzelle, eine verwendete Zeile von δ

Codiere Randbedingungen als Formel R mit

$$\begin{aligned} R \;\equiv\; & \exists_1(z_{0,0}, .., z_{0,e}) \wedge \ldots \wedge \exists_1(z_{p(n),0}, .., z_{p(n),e}) \\ & \wedge \exists_1(s_{0,-p(n)}, .., s_{0,p(n)}) \wedge \ldots \wedge \exists_1(s_{p(n),-p(n)}, .., s_{p(n),p(n)}) \\ & \wedge \exists_1(a_{0,-p(n),1}, .., a_{0,-p(n),v}) \wedge \ldots \wedge \exists_1(a_{p(n),p(n),1}, .., a_{p(n),p(n),v}) \\ & \wedge \exists_1(b_{0,,1}, .., b_{0,v*(e+1)}) \wedge \ldots \wedge \exists_1(b_{p(n),1}, .., b_{p(n),v*(e+1)}) \end{aligned}$$

Dabei ist $\exists_1(y_1, .., y_m)$ Abkürzung für “genau eines der y_i gilt’

– $\exists_1(y_1, .., y_m) \equiv (y_1 \vee .. \vee y_m) \wedge (\bar{y}_1 \vee \bar{y}_2) \wedge .. \wedge (\bar{y}_1 \vee \bar{y}_m)$
$\qquad \wedge (\bar{y}_2 \vee \bar{y}_3) \wedge .. \wedge (\bar{y}_2 \vee \bar{y}_m) \wedge .. \wedge (\bar{y}_{m-1} \vee \bar{y}_m)$

- **R ist in KNF** — Konjunktion von $\exists_1$-Formeln
- **Größe: $\mathcal{O}((p(n))^3)$** — $(p(n)+1) * ((e+1) + (2*p(n)+1)^2 + \ldots)$ Variablen
- **Berechnungsaufwand: $\mathcal{O}((p(n))^3)$** — Bestimmung von $p(n) + \ldots$

Satz von Cook: Codierung der Übergangsbedingungen

Zustand, Kopfposition und Bandinhalt verträglich mit δ

- **Betrachte Zeit t und Bandzelle i einzeln**
  - Zeile l von δ sei $\delta(s_{k_l},x_{j_l})=(s_{k'_l},x_{j'_l},p)$ mit $p\in\{+1,-1\}$
  - Falls τ zur Zeit t Zelle i nicht liest, bleibt sie unverändert

    Formulierung: $((\bar{s}_{t,i} \wedge a_{t,i,1}) \Rightarrow a_{t+1,i,1}) \ \wedge\ ..$

    $\wedge\ ((\bar{s}_{t,i} \wedge a_{t,i,v}) \Rightarrow a_{t+1,i,v})$
  - Falls τ zur Zeit t Zelle i liest und Zeile l benutzt
    - Zur Zeit t: Zustand s_{k_l}, Zelle i ist x_{j_l}

      Formulierung: $(s_{t,i} \wedge b_{t,l}) \Rightarrow (z_{t,k_l} \wedge a_{t,i,j_l})$
    - Zur Zeit $t{+}1$: Zustand $s_{k'_l}$, Zelle i ist $x_{j'_l}$, neue Zelle $i{+}p$

      Formulierung: $(s_{t,i} \wedge b_{t,l}) \Rightarrow (z_{t+1,k'_l} \wedge a_{t+1,k'_l} \wedge s_{t+1,i+p})$

Formeln müssen normalisiert und kombiniert werden

Satz von Cook: Codierung der Übergangsbedingungen

- **Definiere Formeln $\ddot{U}(t,i)$** $t\in\{0..p(n)\},\ i\in\{-p(n)..p(n)\}$

$$
\begin{aligned}
\ddot{U}(t,i) \equiv\ & (s_{t,i} \vee \bar{a}_{t,i,1} \vee a_{t+1,i,q}) \wedge .. \wedge (s_{t,i} \vee \bar{a}_{t,i,v} \vee a_{t+1,i,v}) \\
& \wedge (\bar{s}_{t,i} \vee \bar{b}_{t,1} \vee z_{t,k_1}) \wedge (\bar{s}_{t,i} \vee \bar{b}_{t,1} \vee a_{t,i,j_1}) \\
& \wedge (\bar{s}_{t,i} \vee \bar{b}_{t,1} \vee z_{t+1,k'_1}) \wedge (\bar{s}_{t,i} \vee \bar{b}_{t,1} \vee a_{t+1,k'_1}) \\
& \qquad \wedge (\bar{s}_{t,i} \vee \bar{b}_{t,1} \vee s_{t+1,i+p}) \\
& \vdots \\
& \wedge (\bar{s}_{t,i} \vee \bar{b}_{t,m} \vee z_{t,k_m}) \wedge (\bar{s}_{t,i} \vee \bar{b}_{t,m} \vee a_{t,i,j_m}) \\
& \wedge (\bar{s}_{t,i} \vee \bar{b}_{t,m} \vee z_{t+1,k'_m}) \wedge (\bar{s}_{t,i} \vee \bar{b}_{t,m} \vee a_{t+1,k'_m}) \\
& \qquad \wedge (\bar{s}_{t,i} \vee \bar{b}_{t,m} \vee s_{t+1,i+p})
\end{aligned}
$$

$$
\ddot{U} \equiv \ddot{U}(0,-p(n)) \wedge .. \wedge \ddot{U}(0,p(n)) \wedge .. \wedge \ddot{U}(p(n),p(n))
$$

- **$\ddot{U}$ ist in KNF**
- **Größe: $\mathcal{O}((p(n))^3)$**
- **Berechnungsaufwand: $\mathcal{O}((p(n))^3)$**

Satz von Cook: Codierung der Endbedingung

Nach $p(n)$ Schritten: Endzustand aus F, Bandinhalt $b^{p(n)}1b^{p(n)}$

Sei $F{=}\{s_r, .., s_e\}$ und $x_2 = 1$

Codiere Endbedingungen als Formel E mit

$$
\begin{aligned}
E = \; & (z_{p(n),r} \vee .. \vee z_{p(n),e}) \\
& \wedge \; a_{p(n),-p(n),1} \wedge \ldots \wedge a_{p(n),-1,1} \\
& \wedge \; a_{p(n),0,2} \; \wedge \\
& \wedge \; a_{p(n),1,1} \wedge \ldots \wedge a_{p(n),p(n),1}
\end{aligned}
$$

- **E ist in KNF**
- **Größe: $\mathcal{O}(p(n))$**
- **Berechnungsaufwand: $\mathcal{O}(p(n))$**

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

- **$\alpha(\tau,w)$ ist in KNF**
  - Jede der Teilformeln ist in KNF
- **$\alpha(\tau,w)$ ist in Zeit $\mathcal{O}((p(n))^3)$ konstruierbar**
  - Summe des (deterministischen) Berechnungsaufwands für Teilformeln
- **$w \in L \;\Rightarrow\; \alpha(\tau,w) \in SAT$**
  - Ist $w \in L$, dann gibt es eine akzeptierende Berechnung $\kappa_0,..,\kappa_{p(n)}$ für w
  - Setze: $z_{t,k}{=}1$, falls s_k Zustand von κ_t ist (sonst 0)
    $a_{t,i,j}{=}1$, falls x_j Inhalt der i-ten Bandzelle in κ_t ist (sonst 0)
    $s_{t,i}{=}1$, falls i die Kopfposition in κ_t ist (sonst 0)
    $b_{t,l}{=}1$, falls in κ_t die Zeile l der Tabelle von δ benutzt wird (sonst 0)
  - Per Konstruktion erfüllt dies die Formel $\alpha(\tau,w)$, also $\alpha(\tau,w) \in SAT$
- **$\alpha(\tau,w) \in SAT \;\Rightarrow\; w \in L$**
  - Ist $\alpha(\tau,w)$ erfüllbar, so kann wegen Formel Ü die Belegung der Variablen in eine Konfigurationsfolge $\kappa_0,..,\kappa_{p(n)}$ umgerechnet werden
  - Wegen der Formel R gibt es genau eine solche Konfigurationsfolge
  - Wegen A und E repräsentiert diese die Berechnung $h_\tau(w) = 1$, also $w \in L$

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

- **Zeige $L \in \mathcal{NP}$:**
  - Beschreibe, welchen Lösungsvorschlag das Orakel generiert
  - Beschreibe, wie Lösungsvorschlag überprüft wird
  - Zeige, daß das Prüfverfahren polynomiell ist
- **Zeige $\exists L' \in \mathcal{NPC}.\ L' \leq_p L$** (anstatt $\forall L' \in \mathcal{NP}.\ L' \leq_p L$)
  - Wähle ein ähnliches, bekanntes Problem $L' \in \mathcal{NPC}$
  - Beschreibe Transformationsfunktion f, welche Eingaben aus der Sprache für L' in Worte der Sprache für L umwandelt
  - Zeige für alle x: $x \in L' \Leftrightarrow f(x) \in L$ (also $L' = f^{-1}(L)$)
  - Zeige, daß f in polynomieller Zeit berechnet werden kann

Erfüllbarkeitsproblem mit 3 Literalen pro Klausel

$\boldsymbol{3SAT}$

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

- $\boldsymbol{3SAT \in \mathcal{NP}}$
  - Wie $SAT \in \mathcal{NP}$: Rate Belegung der Variablen und werte Klauseln aus
- $\boldsymbol{SAT \leq_p 3SAT}$**:** Satz J
  - 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 $k_i = z_1 \vee z_2 ... \vee z_j$ durch j-2 neue Klauseln mit neuen Variablen $y_{i,l}$: $(z_1 \vee z_2 \vee y_{i,1})\ \wedge\ (\overline{y_{i,1}} \vee z_3 \vee y_{i,2})\ \wedge\ \ldots (\overline{y_{i,j-3}} \vee z_{j-1} \vee z_j)$
  - Normalisierung der Klauseln möglich in **polynomieller Zeit**
  - k_i erfüllbar genau dann wenn normalisierte Klauselmenge erfüllbar
  - Für die Transformation f gilt: $\forall F.\ F \in SAT\ \Leftrightarrow\ f(F) \in 3SAT$

Das Cliquen Problem ist $\mathcal{NP}$-vollständig

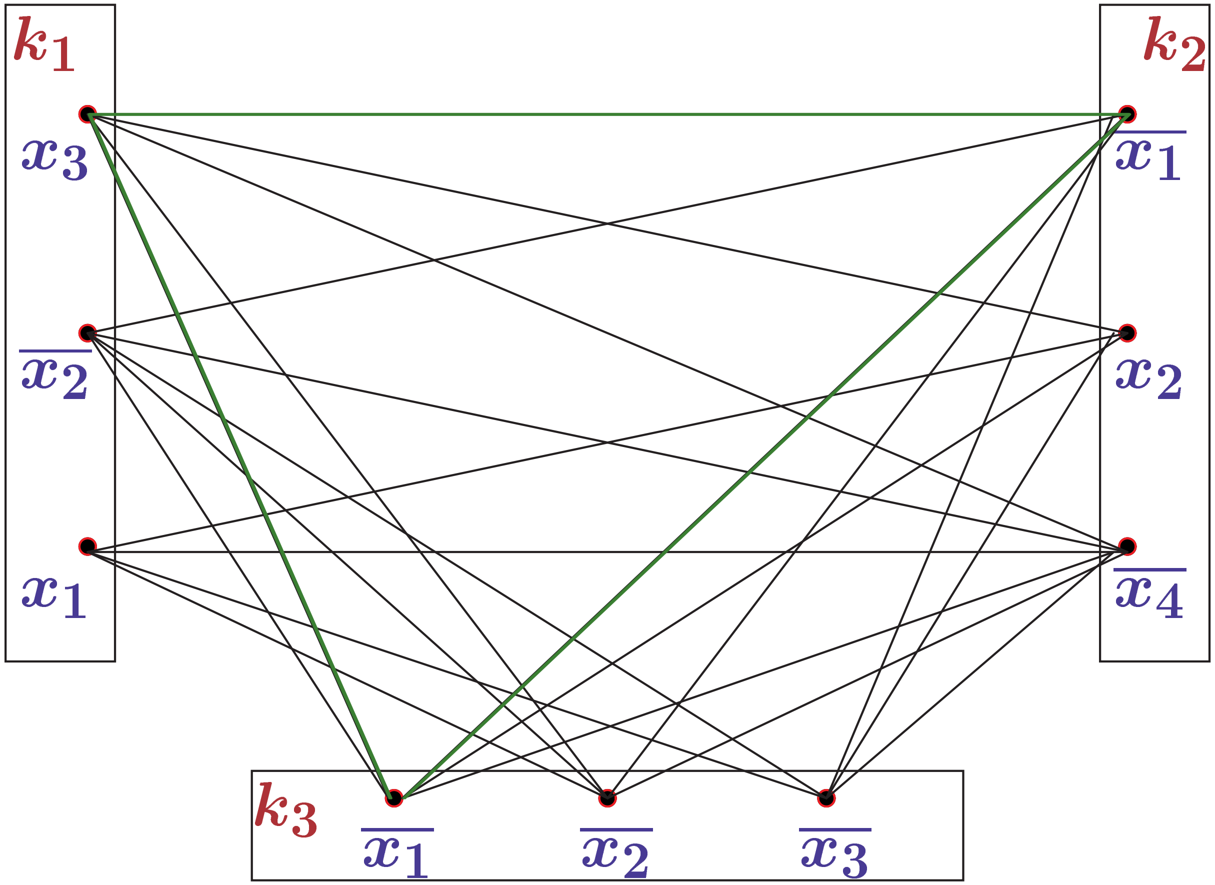
Def K/ Satz L

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

- **$\boldsymbol{CLIQUE \in \mathcal{NP}}$:**
  - Rate eine Kantenmenge $V_c \subseteq V$
  - 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*
- **$\boldsymbol{3SAT \leq_p CLIQUE}$**
  - 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 $E := \{ \{v_{ij}, v_{i'j'}\} \mid i \neq i' \wedge z_{ij} \neq \overline{z_{i'j'}} \}$
  - Setze $f(F) := (G_F, m)$
  - f ist in polynomieller Zeit berechenbar

Codierung einer Formel als Cliquenproblem

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



Gibt es in dem Graphen eine 3-Clique?

Korrektheit der Transformation

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
Per Konstruktion von E enthält V_c aus jedem der Blöcke $b_i := \{v_{ij} \mid 1 \leq j \leq 3\}$
genau einen Knotenund keine zwei Knoten in V_c sind komplementär $(z_{ij} \neq \overline{z_{i'j'}})$
Eine Belegung der zugehörigen z_{ij} mit 1 erfüllt alle Klauseln k_i von F
Also gilt $F \in 3SAT$

Gilt 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
Die zugehörigen Knoten bilden eine m-Clique in G_F
Also gilt $f(F) \in CLIQUE$

$$\Downarrow$$

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

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

Def/Satz M

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

- $\boldsymbol{VC \in \mathcal{NP}}$:
  - Rate eine Kantenmenge $V'{\subseteq}V$
  - 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*

- $\boldsymbol{CLIQUE \leq_p VC}$
  - Bereits beweisen — Folie 6

WEITERE $\mathcal{NP}$-VOLLSTÄNDIGE GRAPHENPROBLEME

- **Independent Set** ***CLIQUE*** $\leq_p$ ***IS***
  - Gegeben ein Graph $G = (V, E)$ der Größe n und eine Zahl $k \leq n$.
  - Gibt es in G eine unabhängige Knotenmenge der Größe k?

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

- **Subgraph Isomorphism** ***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?

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

- **Largest Common Subgraph** ***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?

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

WEITERE $\mathcal{NP}$-VOLLSTÄNDIGE GRAPHENPROBLEME

- **Directed Hamiltonian Circuit** $\quad 3SAT \leq_p DHC$
  - Gegeben ein gerichteter Graph G
  - Gibt es in G einen Hamilton'schen Kreis?

$$DHC = \{\, G \mid G=(V,E) \text{ gerichteter Graph} \wedge \exists \pi{:}\{1..n\}{\to}\{1..n\}.\ \pi \text{ Hamilton'scher Kreis in } G \,\}$$

- **Hamiltonian Circuit** $\quad DHC \leq_p HC$
  - Gegeben ein ungerichteter Graph $G=(V,E)$.
  - Gibt es in G einen Hamilton'schen Kreis?

$$HC = \{\, G \mid G=(V,E) \text{ Graph} \wedge \exists \pi{:}\{1..n\}{\to}\{1..n\}.\ \pi \text{ Hamilton'scher Kreis in } G \,\}$$

- **Travelling Salesman Problem** $\quad HC \leq_p TSP$
  - Gegeben n Städte, Reisekostentabelle c_{ij}, Kostenbeschränkung B
  - Gibt es eine Rundreise durch alle n Städte, deren Kosten unter B liegt?

$$TSP = \{\, c_{12}, ..., c_{n-1,n},\ B \mid B, c_{ij} \in \mathbb{N} \wedge \exists \pi{:}\{1..n\}{\to}\{1..n\}.\ \pi \text{ bijektiv} \wedge \sum_{i=1}^{n-1} c_{\pi(i)\pi(i+1)} + c_{\pi(n)\pi(1)} \leq B \,\}$$

Weitere $\mathcal{NP}$-vollständige Probleme

- **Knapsack: (Rucksack-Bepackung)** $\quad$ $\boldsymbol{3SAT \leq_p KP}$
  - Gegeben n Objekte mit Gewichten $g_1, ..., g_n$ und Nutzwerten $a_1, ...a_n$
  - Rucksack mit Gewichtsschranke G, Minimalnutzwert A.
  - Gibt es eine Bepackung mit Mindestnutzen A und Maximalgewicht G?

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

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

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

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

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

Weitere $\mathcal{NP}$-vollständige Probleme

- **Graph Coloring** $\quad$ ***3SAT*** $\le_p$ ***GC***
  - Gegeben ein Graph $G = (V, E)$ der Größe n und eine Zahl $k \le n$.
  - Gibt es eine Färbung von V mit k verschiedenen Farben, so daß verbundene Konten verschieden Farben haben?

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

- **Multiprozessor-Scheduling** $\quad$ ***MSP*** $\hat{=}$ ***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** $\quad$ ***3SAT*** $\le_p$ ***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 $A * \vec{x} \ge \vec{b}$ löst?

- **Zusammengesetztheit** (vermutlich nicht $\mathcal{NP}$-vollständig)
  - Gegeben eine n-stellige Zahl $x \in \mathbb{N}$
  - Gibt es zwei natürliche Zahlen p und q mit $x = p * q$?

$\mathcal{NP}$-vollständigkeit des Rucksackproblems

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

- **$KP \in \mathcal{NP}$:**
  - Rate Menge von Gegenständen Kantenmenge $J{\subseteq}\{1..n\}$
  - Prüfe $\Sigma_{i\in J} g_i \leq G$ und $\Sigma_{i\in J} a_i \geq A$ *maximal $2|J|$ Schritte*
- **Zeige $3SAT \leq_p KP$**
  - Gegeben $F = (k_1, ..., k_m)$ mit $k_i = z_{i1} \vee z_{i2} \vee z_{i3}$ und $z_{ij} \in \{x_1, ..., \overline{x_n}\}$
  - Konstruiere Rucksackproblem $f(F) \equiv (g_1, ..g_{2m+2n}, a_1, ..a_{2m+2n}, G, A)$ wobei die a_j und g_j $m+n$-stellige Zahlen sind, welche die Anzahl der Vorkommen von Literalen in den Klauseln codieren
    - a_j, $j{\leq}n$: Stelle $i{\leq}m$ ist Anzahl der x_j in k_i, Stelle $m{+}j$ ist 1, sonst 0
    - $b_j \equiv a_{n+j}, j{\leq}n$: Stelle $i{\leq}m$ ist Anzahl der $\bar{x}_j$ in k_i, Stelle $m{+}j$ ist 1, sonst 0
    - $c_i \equiv a_{2n+i}, i{\leq}m$: Stelle $m{+}i$ ist 1, sonst 0
    - $d_i \equiv a_{2n+m+i}, i{\leq}m$: Stelle $m{+}i$ ist 2, sonst 0
    - $g_j{=}a_j$ für alle j $\quad A \equiv G = \underbrace{4\ldots4}_{m-mal}\underbrace{1\ldots1}_{n-mal}$
  - f ist in polynomieller Zeit berechenbar

Codierung einer Formel als Rucksackproblem

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

$A = 444\,1111$

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

$(1, 1, 0, 0)$ 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

· a_j: Stelle $i{\leq}m$ ist Anzahl der x_j in k_i, Stelle $m{+}j$ ist 1, sonst 0

· b_j : Stelle $i{\leq}m$ ist Anzahl der $\bar{x}_j$ in k_i, Stelle $m{+}j$ ist 1, sonst 0

· c_i: Stelle $m{+}i$ ist 1, sonst 0 $\qquad$ d_i: Stelle $m{+}i$ ist 2, sonst 0

· $g_j{=}a_j$ 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

$\mapsto$ In der Summe haben alle Stellen $m{+}j$ den Wert 1

$\mapsto$ Da k_i erfüllt wird, haben die Stellen $i{\leq}n$ einen Wert aus $\{1..3\}$

Die Stellen $i{\leq}n$ können mit c_i und d_i zu 4 ergänzt werden Also $f(F) \in KP$

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

Die Bepackung enthält für $j{\leq}n$ entweder a_j (wähle $x_j{:=}1$) oder b_j ($x_j{:=}0$)

Wegen $c_i{+}d_i{=}3$ ist jede Stelle $i{\leq}m$ der Summe der a_j und b_j mindestens 1

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

Damit erfüllt die Belegung die Formel F, also $F \in 3SAT$

$$\Downarrow$$

$$\mathbf{3SAT \leq_p KP}$$

$\mathcal{NP}$-vollständigkeit des Färbbarkeitsproblems

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

Zeige $3SAT \leq_p GC$

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

– 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

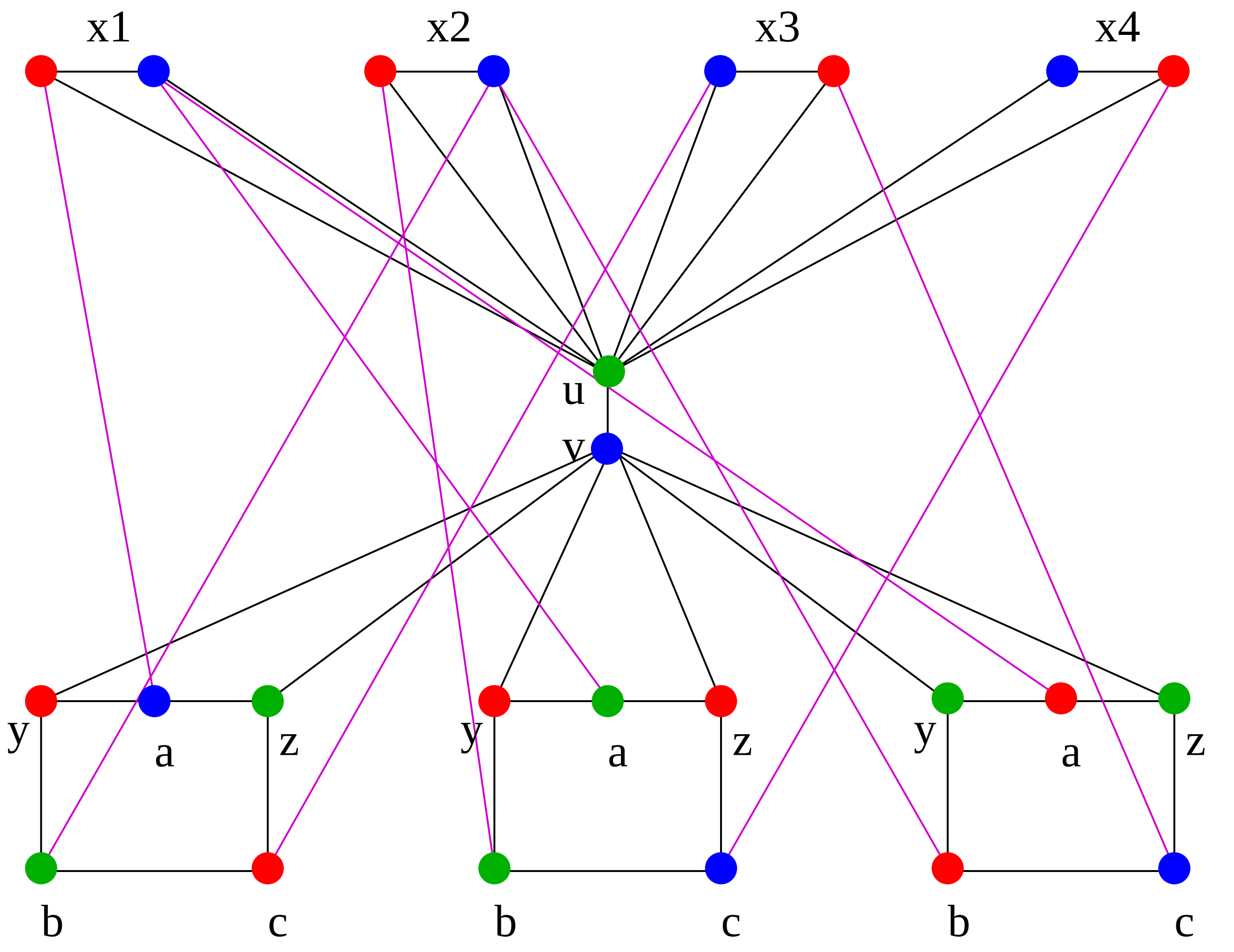
· Kanten zur Codierung der Klauselliterale

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

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

Codierung einer Formel als Färbungsproblem

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



Korrektheit der Transformation

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

  Wähle $f_v(x_i), f_v(\overline{x_i}) \in \{0{,}1\}$ entsprechend, $f_v(u) = 2$ und $f_v(v) = 0$.

  Da jedes k_i erfüllbar ist, kann einer der a_i, b_i, c_i die Farbe 0 erhalten

  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 $f(F) \in GC$

- Ist $f(F) \in GC$ dann ist o.B.d.A. $f_v(u) = 2$ und $f_v(v) = 0$

  Wähle Belegung der x_i entsprechend der Färbung von x_i

  Wäre Klausel k_i nicht erfüllt, so müßte die Farbe der a_i, b_i, c_i 1 oder 2 sein

  Wegen $f_v(b_i) \neq f_v(c_i)$ und $f_v(v) = 0$ wäre dann $\{f_v(y_i), f_v(z_i)\} = \{1, 2\}$

  Dies widerspricht der Färbbarkeit, da a_i ebenfalls mit 1 oder 2 gefärbt ist.

  Also $F \in 3SAT$

$$\Downarrow$$

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

Jenseits von $\mathcal{NP}$-Vollständigkeit

- ***co*–$\mathcal{NP}$: Probleme mit Komplement in $\mathcal{NP}$**
  - Die Menge der gültigen Formeln ist in $co\text{–}\mathcal{NP}$ (Komplement von SAT)
  - Das Primzahlproblem liegt in $co\text{–}\mathcal{NP}$
  - Das Primzahlproblem liegt auch in $\mathcal{NP}$ — Reischuk, 313–315
  - Ist ein $co\text{–}\mathcal{NP}$ Problem L $\mathcal{NP}$-vollständig, so gilt $\mathcal{NP} = co\text{–}\mathcal{NP}$

    Es würde folgen: $L' \leq_p \overline{L} \in \mathcal{NP}$ für jedes $L' \in co\text{–}\mathcal{NP}$
  - Das Zusammengesetztheitsproblem ist vermutlich nicht $\mathcal{NP}$-vollständig sondern liegt zwischen $\mathcal{P}$ und $\mathcal{NPC}$ (sofern

- **Es gibt *PSPACE*-vollständige Probleme**

  In-Place Acceptance — Asteroth/Baier §4.5
  - Gegeben DTM τ und $w \in X^*$: Gilt $h_\tau(w) = 1$ und $s_\tau(w) \leq |w|$?

  QBF: Ist eine gegebene quantifizierte boole'sche Formel wahr? — Reischuk §6.4.4

Wie kann man unhandhabbare Probleme angehen?

Komplexitätsklassenhierarchie

vgl. Kap 8.3

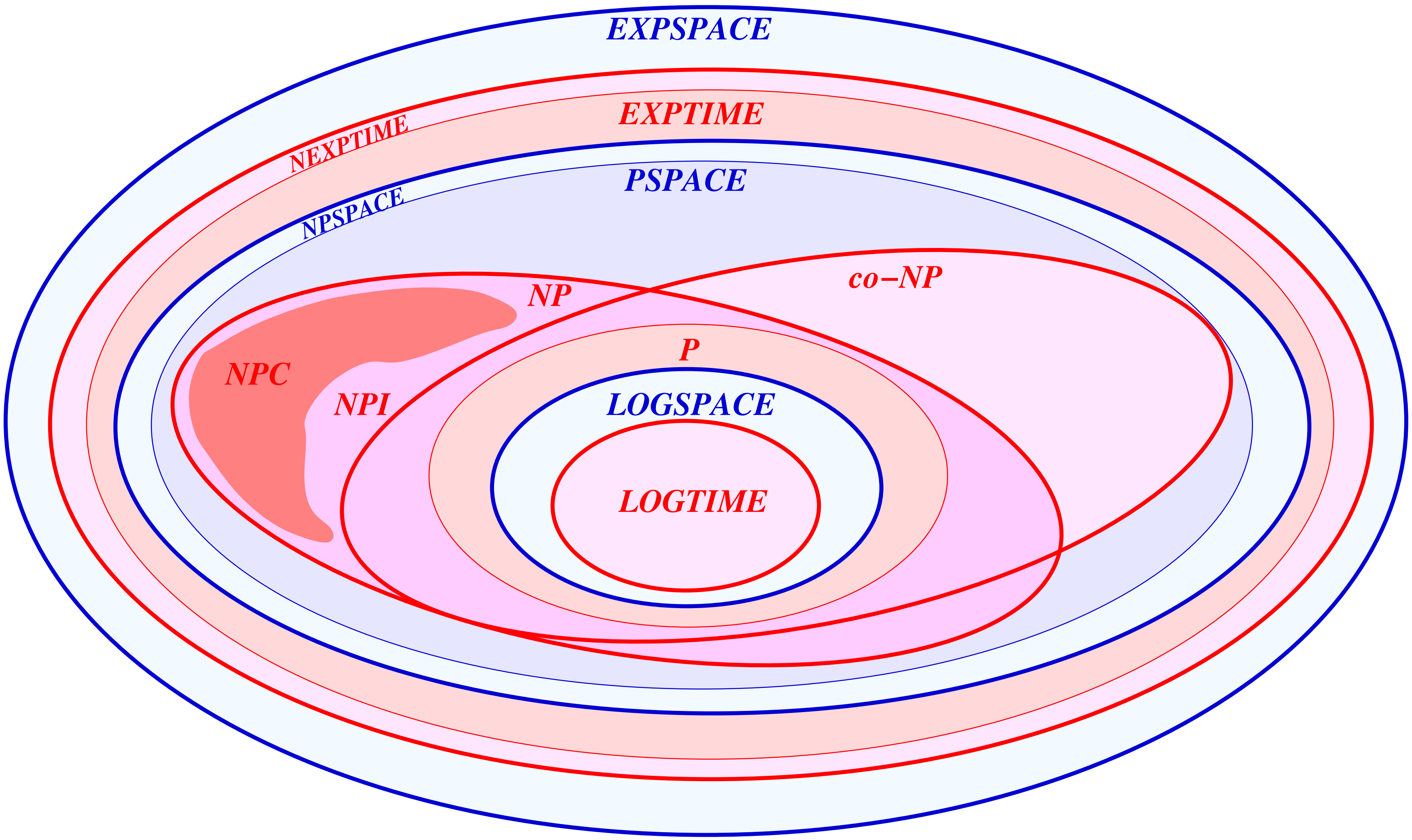
- **Zeitkomplexitätsklassen**

$LOGTIME$	in logarithmischer Zeit lösbar
$NLOGTIME$	nichtdeterministisch in logarithmischer Zeit lösbar
$\mathcal{P}$	in polynomieller Zeit lösbar
$\mathcal{NP}$	nichtdeterministisch in polynomieller Zeit lösbar
$\mathcal{NPC}$	$\mathcal{NP}$-vollständige Probleme
$\mathcal{NPI}$	$\mathcal{NP}$-unvollständige Probleme: $\mathcal{NP}$ - $\mathcal{NPC}$ - $\mathcal{P}$
$co\text{–}\mathcal{NP}$	Komplement in $\mathcal{NP}$
$EXPTIME$	in exponentieller Zeit lösbar
$NEXPTIME$	nichtdeterministisch in exponentieller Zeit lösbar

- **Platzkomplexitätsklassen**

$LOGSPACE$	mit logarithmischem Platzverbrauch lösbar
$NLOGSPACE$	nichtdeterministisch mit logarithmischem Platzverbrauch lösbar
$PSPACE$	mit polynomiellem Platzverbrauch lösbar
$NPSPACE$	nichtdeterministisch mit polynomiellem Platzverbrauch lösbar
$EXPSPACE$	mit exponentiellem Platzverbrauch lösbar

Sprachklassenhierarchie



Wichtige Vertreter verschiedener Klassen

- **Isomorphie ungerichteter Graphen** $\mathcal{NPI}$
- **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 $\mathcal{NP}$ ("Σ_2")
  – Bestimme optimale Größe einer Schaltung
- **Quantifizierte boole'sche Formeln** $PSPACE$-vollständig
  – Gültigkeit aussagenlogischer Formeln mit boole'schen Quantoren
- **Strategische Spiele** $PSPACE$-vollständig
  – Details in Garey/Johnson Seite 254ff
- **TSP*: Bestimmung aller Rundreisen mit gegebenen Kosten**
  – Unrealistische Problemstellung: zu viele Lösungen $EXPSPACE$