Theoretische Informatik II

Einheit 6.3

$\mathcal{NP}$-vollständige Probleme

1. Logische Probleme
2. Graphbasierte Probleme
3. Scheduling Probleme

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 OTM 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, bekannt $\mathcal{NP}$-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 $x \in \Sigma'^*$: $x \in L' \;\Leftrightarrow\; f(x) \in L$ (d.h. $L' = f^{-1}(L)$)

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

3SAT IST NP-VOLLSTÄNDIG

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

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

Begründung analog zu $SAT \in \mathcal{NP}$

a) Generiere Belegung der Variablen als Lösungsvorschlag $\surd$

b) Werte Klauseln aus $\surd$

c) Auswertung benötigt lineare Zeit $\surd$

2. Es gilt $SAT \leq_p 3SAT$:

§6.2 Folie 10

d) Wähle SAT als bekanntes $\mathcal{NP}$-vollständiges Ausgangsproblem $\surd$

e) Reduktionsfunktion f normalisiert jede der Klauseln $k_1, ..k_m$ $\surd$

f) Es gilt $\forall F.\ F \in SAT \ \Leftrightarrow\ f(F) \in 3SAT$ $\surd$

g) Aufwand für Normalisierung der Klauseln ist linear $\surd$

Wegen $SAT \in \mathcal{NPC}$ ist damit auch $3SAT \in \mathcal{NPC}$

Graphbasierte Probleme

- **Viele Probleme lassen sich gut mit Graphen formalisieren**
  - Travelling Salesman: Suche Kreis in gewichtetem Graphen
  - Soziale Netze: Suche vollständig verbundenen Teilgraphen
  - Netzwerküberwachung: Suche Überdeckung aller Kanten eines Graphen

- **Graphen haben Knoten und Kanten (“$G = (V, E)$”)**
  - Eine Kante $e \in E$ zwischen zwei Knoten $v \neq v' \in V$ kann gerichtet ($e = (v, v')$) oder ungerichtet ($e = \{v, v'\}$) sein
  - E ist beschreibbar als Liste $v_1, ..., v_n, \{v_{i_1}, v'_{i_1}\}, ..., \{v_{i_m}, v'_{i_m}\}$
  - Die Größe des Graphen ist $|V|+|E|$ (meist überwiegt $|E|$)
  - In gewichteten Graphen ist jede Kante mit einer Zahl markiert
  - Bäume sind zyklenfreie ungerichtete Graphen

Mehr im Handout Graphentheorie

Das Cliquen Problem

Gibt es in einem Graphen $G{=}(V,E)$ eine Clique der Mindestgröße k?

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

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

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

a) Generiere eine Knotenmenge $V_c{\subseteq}V$

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

Prüfe ob V_c vollständig verbunden ist

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

2. Zeige $3SAT \leq_p CLIQUE$:

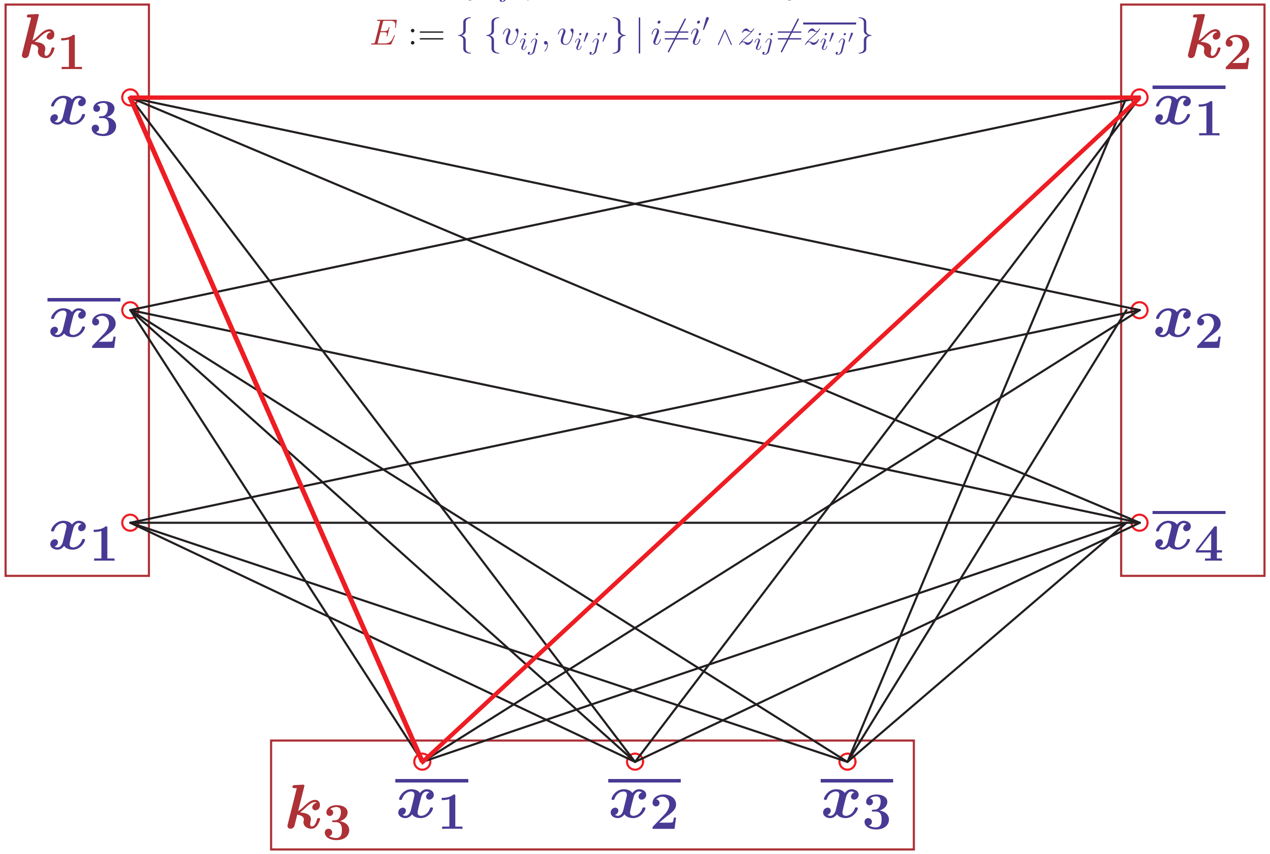
– Transformiere Klauseln in Menge von Knoten eines Graphen

– Verbinde Knoten verschiedener Klauseln, die sich nicht widersprechen

Codierung einer Formel als Cliquenproblem

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

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



Gibt es in dem Graphen eine 3-Clique?

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 $E := \{ \{v_{ij}, v_{i'j'}\} \mid i \neq i' \wedge z_{ij} \neq \overline{z_{i'j'}}\}$

Zeige $\boldsymbol{F \in 3SAT \Leftrightarrow f(F) \in CLIQUE}$

Es sei $F \in 3SAT$. Dann 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 $f(F) \in CLIQUE$

Sei umgekehrt $f(F) \in CLIQUE$
– Dann hat G_F eine m-Clique V_c, d.h. $\{v_{ij}, v_{i'j'}\} \in E$ für alle $v_{ij} \neq v_{i'j'} \in V_c$
– 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 $F \in 3SAT$

$\mathcal{CLIQUE}$ IST $\mathcal{NP}$-VOLLSTÄNDIG (SUMMARISCHER BEWEIS)

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

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

a) Generiere eine Knotenmenge $V_c{\subseteq}V$

b) Prüfe $|V_c|{\geq}k$ und $\forall v{\neq}v' \in V_c.\ \{v, v'\} \in E$

c) Die Tests sind in maximal $|V_c|$ bzw. $|V_c|^2{*}|E|{\leq}|V|^4$ Schritten, also in polynomieller Zeit, durchführbar

2. $3SAT \leq_p CLIQUE$:

e) Gegeben $F = (k_1, ..., k_m)$ mit $k_i = z_{i1} \lor z_{i2} \lor z_{i3}$ 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 $E := \{\ \{v_{ij}, v_{i'j'}\} \mid i{\neq}i' \land z_{ij}{\neq}\overline{z_{i'j'}}\}$. Setze $f(F) := (G_F, m)$

f) Dann gilt $F \in 3SAT \Leftrightarrow f(F) \in CLIQUE$ (siehe vorige Folie)

g) V ist mit linearem und E mit quadratischem Aufwand konstruierbar
also ist f in polynomieller Zeit berechenbar

Wegen $3SAT \in \mathcal{NPC}$ ist damit auch $CLIQUE \in \mathcal{NPC}$

DAS VERTEX COVER PROBLEM

Gibt es im Graphen G eine Knotenüberdeckung maximaler Größe k?

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

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

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

a) Generiere eine Knotenmenge $V' \subseteq V$

b/c) Prüfe $|V_c| \leq k$ — *maximal $|V'|$ Schritte*

Prüfe ob aus jeder Kante in G mindestens ein Knoten in V' liegt

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

2. Zeige $CLIQUE \leq_p VC$:

Wenn V_c eine Clique in G ist, dann sind alle Knoten in V_c durch eine Kante aus E verbunden. Also haben alle Nicht-Kanten (aus E^c) mindestens einen Knoten außerhalb von V_c bzw. innerhalb von $V - V_c$. Damit bildet $V - V_c$ eine Knotenüberdeckung im Komplementärgraphen.

Transformiere (G,k) in $(G^c, |V| - k)$

Reduzierbarkeit: Clique $\leq_p$ Vertex Cover

e) Definiere Reduktionsfunktion f

– Wähle $f(G,k) := (G^c, |V|-k)$

f) Korrektheit der Transformation

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

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

$\Leftrightarrow \forall \{v, v'\} \notin E.\ v \neq v' \Rightarrow\ v \notin V_c \vee v' \notin V_c$ (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 $G^c = (V, E^c)$

Damit folgt $(G,k) \in CLIQUE$

$\Leftrightarrow$ G hat Clique V_c der Mindestgröße k

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

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

g) Laufzeitverhalten von f

– Der Komplementärgraph kann in $\mathcal{O}(|V|^2)$ Schritten generiert werden

– $|V|-k$ kann in linearer Zeit berechnet werden

– Also ist f in polynomieller Zeit berechenbar

VC IST $\mathcal{NP}$-VOLLSTÄNDIG (SUMMARISCHER BEWEIS)

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

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

a) Generiere eine Knotenmenge $V'\subseteq V$

b) Prüfe $|V_c|\leq k$ und $\forall\{v,v'\}\in E.\ v\in V' \lor v'\in V'$

c) Die Tests sind in maximal $|V'|$ bzw. $|V'|*|E|\leq|V|^3$ Schritten, also in polynomieller Zeit, durchführbar

2. $\boldsymbol{CLIQUE \leq_p VC}$: (vgl. Einheit 6.2)

e) Gegeben ein Graph $G{=}(V,E)$ und eine Zahl $k\leq|V|$
Setze $f(F) := (G^c, |V|-k)$

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

g) G^c ist mit quadratischem und $|V|-k$ mit linearem Aufwand konstruierbar, also ist f in polynomieller Zeit berechenbar

Wegen $\boldsymbol{CLIQUE \in \mathcal{NPC}}$ ist damit auch $\boldsymbol{VC \in \mathcal{NPC}}$

Gerichteter Hamiltonscher Kreis (DHC)

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

Gibt es in einem gerichteten Graphen einen Hamiltonschen Kreis?

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

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

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

a) Generiere Liste von n verschiedenen Knoten $c = (v_{i_1}, .., v_{i_n})$

b) Prüfe ob c einen Kreis bildet, der jeden Knoten von G berührt

d.h. ob $(v_{i_j}, v_{i_{j+1}}) \in E$ für alle $j{<}n$ und $(v_{i_n}, v_{i_1}) \in E$ gilt

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

2. Zeige $3SAT \leq_p DHC$

$\mapsto$ Anhang (aufwendig)

– Konstruiere Graphen, der Variablen und Klauseln der Formel codiert

Wegen $3SAT \in \mathcal{NPC}$ ist damit auch $DHC \in \mathcal{NPC}$

Hamiltonscher Kreis (Hamiltonian Circuit)

Gibt es in einem ungerichteten Graphen einen hamiltonschen Kreis?

$HC = \{\, G \mid G \textit{ ungerichteter Graph} \wedge G \textit{ hat Hamiltonschen Kreis}\}$

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

(Beweis wie bei DHC)

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

a) Generiere Liste von n verschiedenen Knoten $c = (v_{i_1}, .., v_{i_n})$

b) Prüfe ob c einen Kreis bildet, der jeden Knoten von G berührt

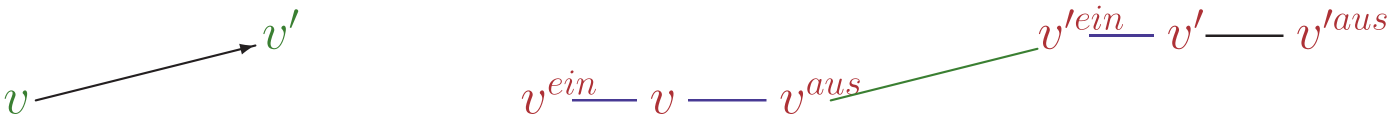
d.h. ob $\{v_{i_j}, v_{i_{j+1}}\} \in E$ für alle $j<n$ und $\{v_{i_n}, v_{i_1}\} \in E$ gilt

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

2. Zeige $DHC \leq_p HC$

– Transformiere jeden Knoten in drei Knoten mit Richtungsmarkierung

– Transformiere gerichtete Kante (v, v') in Kante zwischen “Ausgang” von v und “Eingang” von v'



– Um v' zu erreichen, muß man über v'^{ein} hinein und über v'^{aus} hinaus

TRANSFORMATION $DHC \mapsto HC$

Konstruiere aus $G_d = (V_d, E_d)$ einen ungerichteten Graphen $G = (V, E)$ mit $v^{ein}, v, v^{aus} \in V$ und $\{v^{ein}, v\}, \{v, v^{aus}\} \in E$ für alle $v \in V_d$ und $\{v_i^{aus}, v_j^{ein}\} \in E$ für alle $(v_i, v_j) \in E_d$. Setze $f(G_d) = G$.

Zeige $\boldsymbol{G_d \in DHC \Leftrightarrow f(G_d) \in HC}$

Es sei $G_d \in DHC$. Dann gibt es einen DHC $(v_{i_1}, .., v_{i_n})$ in G_d.
In diesem Fall ist $(v_{i_1}^{ein}, v_{i_1}, v_{i_1}^{aus}, .., v_{i_n}^{aus})$ ein Hamiltonscher Kreis in G
Also $\boldsymbol{f(G_d) \in HC}$

Es sei $f(G_d) \in HC$. Dann gibt es einen Hamiltonkreis $(u_{j_1}, .., u_{j_{3n}})$ in G
– Da jeder Knoten v_i nur v_i^{ein} und v_i^{aus} als Nachbarn hat und sonst nur Kanten zwischen v_i^{aus} und einem v_j^{ein} verlaufen muß der Kreis die Gestalt $(v_{i_1}^{ein}, v_{i_1}, v_{i_1}^{aus}, .., v_{i_n}^{ein}, v_{i_n}, v_{i_n}^{aus})$ oder $(v_{i_1}^{aus}, v_{i_1}, v_{i_1}^{ein}, .., v_{i_n}^{aus}, v_{i_n}, v_{i_n}^{ein})$ 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 $\boldsymbol{G_d \in DHC}$

HC IST $\mathcal{NP}$-VOLLSTÄNDIG (SUMMARISCHER BEWEIS)

$\boldsymbol{HC} = \{ G \mid G = (V, E)$ *ungerichteter Graph* $\wedge$ G *hat Hamiltonschen Kreis*$\}$

1. $HC \in \mathcal{NP}$:

(Beweis wie bei DHC)

a) Generiere Liste von n verschiedenen Knoten $c = (v_{i_1}, .., v_{i_n})$

b) Prüfe ob $\{v_{i_j}, v_{i_{j+1}}\} \in E$ für alle $j<n$ und $\{v_{i_n}, v_{i_1}\} \in E$ gilt

c) Der Test ist in $n*|E|$ Schritten, also in polynomieller Zeit, durchführbar

2. $DHC \leq_p HC$

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

e) Konstruiere ungerichteten Graphen $f(G_d) \equiv G = (V, E)$
mit $v^{ein}, v, v^{aus} \in V$ und $\{v^{ein}, v\}, \{v, v^{aus}\} \in E$ für alle $v \in V_d$
und $\{v_i^{aus}, v_j^{ein}\} \in E$ für alle $(v_i, v_j) \in E_d$.

f) Dann gilt $G_d \in DHC \Leftrightarrow f(G_d) \in HC$ (siehe vorige Folie)

g) V und E sind mit linearem Aufwand konstruierbar
also ist f in polynomieller Zeit berechenbar

Wegen $DHC \in \mathcal{NPC}$ ist damit auch $HC \in \mathcal{NPC}$

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?

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

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

a) Generiere eine Rundreise $\pi{:}\{1..n\}\rightarrow\{1..n\}$ *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) $|V|$ ist mit linearem und die Kostentabelle mit maximal quadratischem Aufwand konstruierbar, also ist f in polynomieller Zeit berechenbar

Wegen $HC \in \mathcal{NPC}$ ist damit auch $TSP \in \mathcal{NPC}$

Das Färbbarkeitsproblem (Graph Coloring)

Gibt es zu einem Graphen $G{=}(V,E)$ eine k-Färbung von V?

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

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

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

a) Generiere eine Färbung $f_V{:}V\to\{1..k\}$ der Knoten des Graphen

b) Prüfe ob alle verbundenen Knoten in V verschiedene Farben haben

d.h. $\forall\{u,v\}\in E. f_V(u)\neq f_V(v)$

c) Anzahl der Schritte ist maximal $|E|*|V|^2 \in \mathcal{O}(|G|^3)$

2. Zeige $3SAT\leq_p GC$:

Konstruiere Färbungsproblem aus einer Klauselmenge (Details folgen)

– Ein Teilgraph zur Codierung der Variablenbelegung

· Knoten für x_i und $\overline{x_i}$ können nur Farben aus 0 oder 1 bekommen

– Ein Teilgraph zur Codierung der Klauseln

· Knoten für Literale sind mit zugehörigen Variablen verbunden

· Bei Erfüllbarkeit erzwingt Anordnung Farbe 1 für ein Literal

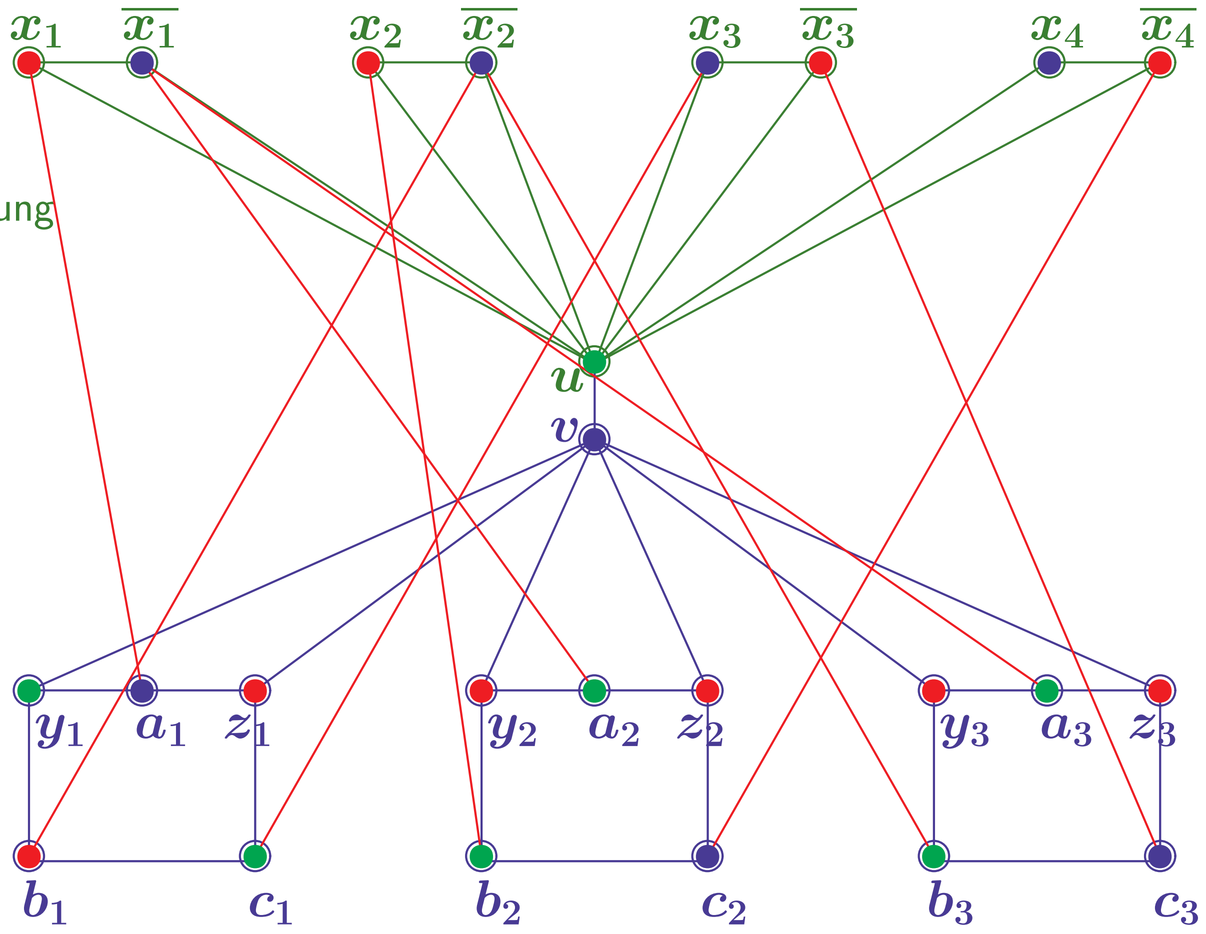
Codierung einer Formel als Färbungsproblem

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

Codierung der Variablenbelegung

Zuordnung der Klauselliterale

Codierung der Klauseln



Zwei Farben an Variablen

Zwei Farben an u und v

Drei Farben für Klausel nötig

Kein Blau an y_i,z_i

Blau an a_i, wenn y_i Grün

Sonst y_i Rot und Blau an b_i oder c_i

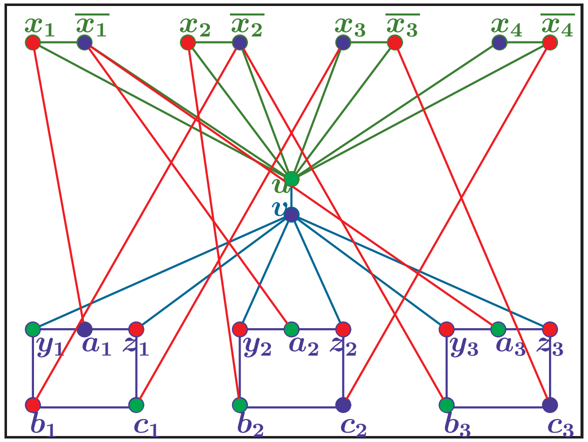
Ein Klausselliteral muß Rot sein

Gibt es für den Graphen eine 3-Färbung?

TRANSFORMATION $3SAT \mapsto GC$

- **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 $f(F) \equiv (G, 3)$
- **Teilgraph für Codierung der Variablenbelegung:**
  – $V_{var} = \{u,\ x_1, ..., \overline{x_n}\}$
  – $E_{var} = \{\{u, x_1\}, \{u, \overline{x_1}\}, \{x_1, \overline{x_1}\}, ..\{u, x_n\}, \{u, \overline{x_n}\}, \{x_n, \overline{x_n}\}\}$
  Bei 3-Färbbarkeit erhalten x_i und $\overline{x_i}$ verschiedene Farben aus 0 oder 1
- **Teilgraph für Codierung der Klauseln:**
  – $V_k = \{v,\ a_1, b_1, c_1, y_1, z_1, .., a_m, b_m, c_m, y_m, z_m\}$
  – $E_k = \{\{v, y_1\}, \{v, z_1\}, \{a_1, y_1\}, \{a_1, z_1\}, \{b_1, y_1\}, \{c_1, z_1\}, \{b_1, c_1\},$
  $...\{v, y_m\}, ..\{b_m, c_m\},\ \{u, v\}\}$
  Knoten u erhält Farbe 2, Knoten v erhält Farbe 0 oder 1
- **Verbindungskanten zur Codierung der Klauselliterale**
  – $E_{lit} = \{\{a_1, z_{11}\}, \{b_1, z_{12}\}, \{c_1, z_{13}\}, ..\{a_m, z_{m1}\}, \{b_m, z_{m2}\}, \{c_m, z_{m3}\}\}$
- **Gesamtgraph** $\boldsymbol{G := (V_{var} \cup V_k, E_{var} \cup E_k \cup E_{lit})}$

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



- Sei $F \in 3SAT$

  Dann gibt es eine erfüllende Belegung der x_j
  - Wähle $f_v(x_i), f_v(\overline{x_i}) \in \{\mathbf{0},\mathbf{1}\}$ entsprechend, sowie $f_v(u)$=**2** und $f_v(v)$=**0**.
  - Da jedes k_i erfüllbar ist, kann eines 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. $f_v(u)$=**2** und $f_v(v)$=**0**
  - Wegen 3-Färbbarkeit ist $f_v(x_i), f_v(\overline{x_i}) \in \{\mathbf{0},\mathbf{1}\}$ und $f_v(y_i), f_v(z_i) \in \{\mathbf{1},\mathbf{2}\}$
  - Ist $f_v(y_i) \neq f_v(z_i)$, dann folgt $f_v(a_i)=0$, sonst $f_v(b_i)=0$ oder $f_v(c_i)=0$
  - Damit ist für jede Klausel die Farbe eines der zugehörigen Literale **1**
  - Wähle Belegung der x_i entsprechend der Färbung von x_i
  - Dann wird jede der Klauseln k_i erfüllt ud es folgt $F \in 3SAT$

$\mathcal{GC}$ IST $\mathcal{NP}$-VOLLSTÄNDIG (SUMMARISCHER BEWEIS)

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

1. $GC \in \mathcal{NP}$:

a) Generiere eine Färbung $f_V{:}V\rightarrow\{1..k\}$ (z.B. als Funktionstabelle)

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

c) Der Test ist in $|E|*|V|^2$ Schritten, d.h. polynomieller Zeit, durchführbar

2. $3SAT \leq_p GC$:

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

Setze $f(F) := (G,3)$, wobei $G := (V_{var}\cup V_k, E_{var}\cup E_k\cup E_{lit})$ und

$V_{var} = \{u,\ x_1, \ldots, \overline{x_n}\}$, $\quad V_k = \{v,\ a_1, b_1, c_1, y_1, z_1, .., a_m, b_m, c_m, y_m, z_m\}$

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

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

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

f) Dann gilt $F\in 3SAT \Leftrightarrow f(F)\in GC$ (siehe vorige Folie)

g) V und E sind mit linearem Aufwand konstruierbar

also ist f in polynomieller Zeit berechenbar

Wegen $3SAT \in \mathcal{NPC}$ ist damit auch $GC \in \mathcal{NPC}$

DAS RUCKSACKPROBLEM (KNAPSACK)

Gibt es eine Bepackung eines Rucksacks mit Gegenständen, die einen Mindestwert haben und ein Maximalgewicht nicht überschreiten?

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

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

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

a) Generiere eine Auswahlliste von Gegenständen $J\subseteq\{1..n\}$

b) Prüfe ob die gewählten Gegenstände mindestens den Nutzen A und maximal Gewicht G haben, d.h. $\sum_{i\in J} g_i \leq G$ und $\sum_{i\in J} a_i \geq A$

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

2. Zeige $3SAT \leq_p KP$:

(Details folgen)

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

– Wähle Nutzen A und Gewicht G so, daß jede Variable eine Belegung erhalten muß und jede Klausel ein Literal mit Wert 1 haben muß

Codierung einer Formel als Rucksackproblem

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

$G = A = 444\,1111$

Vier Arten von Gegenständen, Nutzen und Gewichte sind jeweils gleich

$a_1 = 100\,1000$	$b_1 = 011\,1000$	$c_1 = 100\,0000$	$d_1 = 200\,0000$
$a_2 = 010\,0100$	$b_2 = 101\,0100$	$c_2 = 010\,0000$	$d_2 = 020\,0000$
$a_3 = 100\,0010$	$b_3 = 001\,0010$	$c_3 = 001\,0000$	$d_3 = 002\,0000$
$a_4 = 000\,0001$	$b_4 = 010\,0001$	*Ausgleich um Summe 4 zu erreichen*	

$(1, 1, 0, 0)$ ist erfüllende Belegung

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

Nutzen A ist erreicht, Gewichtslimit G wird eingehalten

Beweis für $3SAT \leq_p KP$

- **Reduktion nutzt Spezialfall des Rucksackproblems**
  - Gewichte g_i und Nutzen a_i sind jeweils gleich
  - Gesamtnutzen G und Gesamtgewicht A sind gleich
  - Wegen $\sum_{i \in J} g_i \leq G$ und $\sum_{i \in J} a_i \geq A$ gilt also $\sum_{i \in J} g_i = G$

  $$KP^* = \{(g_1..g_n,\ G) \mid \exists J \subseteq \{1..n\}.\ \sum_{i \in J} g_i = G\}$$

  Manche Lehrbücher bezeichnen dies als "das Rucksackproblem"

- **Reduktion $KP^* \leq_p KP$ ist einfach**

  e) Gegeben n Objekte mit Gewichten g_i und ein Gewichtslimit G

  Setze $f(g_1..g_n,\ G) = (g_1..g_n, g_1..g_n,\ G, G)$

  f) Dann gilt $(g_1..g_n,\ G) \in KP^* \Leftrightarrow f(g_1..g_n,\ G) \in KP$ (siehe oben)

  g) Der Aufwand ist linear, also ist f in polynomieller Zeit berechenbar

- **Reduktion $3SAT \leq_p KP^*$ folgt dem Beispiel**
  - Codiere Anzahl der Vorkommen von Literalen in den Klauseln in Dezimalstellen der Gewichte (Details folgen)

- **Damit sind KP^* und KP $\mathcal{NP}$-vollständig**

TRANSFORMATION $3SAT \mapsto KP^*$

$\boldsymbol{KP^*} = \{(g_1..g_n,\ G) \mid \exists J \subseteq \{1..n\}.\ \sum_{i \in J} g_i = G\}$

- **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 $f(F) \equiv (a_1..a_n, b_1..b_n, c_1..c_m, d_1..d_m, G)$
- **Gewichte sind $m{+}n$-stellige Dezimalzahlen:**
  – Die Stellen $1..m$ codieren das Vorkommen der Literale in den Klauseln
  – Die Stellen $m{+}1..m{+}n$ codieren die jeweilige Variable
- **Gewichte a_j ($j{\leq}n$) für Vorkommen von x_j in den Klauseln:**
  An Stelle $i{\leq}m$ steht die Anzahl der x_j in k_i, 1 an Stelle $m{+}j$, sonst 0
- **Gewichte b_j ($j{\leq}n$) für Vorkommen von $\overline{x_j}$ in den Klauseln:**
  An Stelle $i{\leq}m$ steht Anzahl der $\overline{x_j}$ in k_i, 1 an Stelle $m{+}j$ ($j{\leq}n$), sonst 0
- **Ausgleichsgewichte c_i und d_i ($i{\leq}m$):**
  c_i hat eine 1 an Stelle i, sonst 0, d_i hat eine 2 an Stelle i, sonst 0
- **Gesamtgewicht:** $\boldsymbol{G} = \underbrace{4\ldots4}_{m-mal}\underbrace{1\ldots1}_{n-mal}$

KORREKTHEIT: $F \in 3SAT \Leftrightarrow f(F) \in KP$

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

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

– Für $j \leq n$ wähle Gegenstand 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 $i \leq m$ einen Wert aus $\{1..3\}$, da jedes k_i erfüllt wird

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

Also $f(F) \in KP$

Gilt $f(F) \in KP$, so gibt es eine Bepackung die genau den Wert G 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. $F \in 3SAT$

KP^* IST $\mathcal{NP}$-VOLLSTÄNDIG (SUMMARISCHER BEWEIS)

$$KP^* = \{(g_1..g_n, G) \mid \exists J \subseteq \{1..n\}.\ \sum_{i \in J} g_i = G\}$$

1. $KP^* \in \mathcal{NP}$:

a) Generiere eine Auswahlliste von Gegenständen $J \subseteq \{1..n\}$

b) Prüfe $\sum_{i \in J} g_i = G$

c) Der Test ist in n Schritten, also in polynomieller Zeit, durchführbar

2. $3SAT \leq_p KP^*$:

e) 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 $f(F) \equiv (a_1..a_n, b_1..b_n, c_1..c_m, d_1..d_m, G)$, wobei

a_j: Anzahl der x_j in k_i an Stelle $i \leq m$, 1 an Stelle $m+j$, sonst 0 $(j \leq n)$

b_j: Anzahl der $\bar{x}_j$ in k_i an Stelle $i \leq m$, 1 an Stelle $m+j$, sonst 0 $(j \leq n)$

c_i: 1 an Stelle i, sonst 0 d_i: 2 an Stelle i, sonst 0 $(i \leq m)$

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

f) Dann gilt $F \in 3SAT \Leftrightarrow f(F) \in KP^*$ (siehe vorige Folie)

g) Die Gewichte sind jeweils mit linearem Aufwand konstruierbar

also ist f in polynomieller Zeit berechenbar

Wegen $3SAT \in \mathcal{NPC}$ ist damit auch $KP^* \in \mathcal{NPC}$

WEITERE $\mathcal{NP}$-VOLLSTÄNDIGE GRAPHENPROBLEME

- **Independent Set**
  - 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?

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

- **Subgraph Isomorphism**
  - 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?

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

- **Largest Common Subgraph**
  - 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?

  LCS $= \{ (G_1, G_2, k) \mid G_1, G_2$ Graphen $\wedge\ k{\leq}|G_1|\ \wedge\ \exists H_1, H_2$ 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**
  - Gegeben n Objekte mit Wert $b_1, ..., b_n$.
  - Gibt es eine Aufteilung der Objekte in zwei gleichwertige Stapel?

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

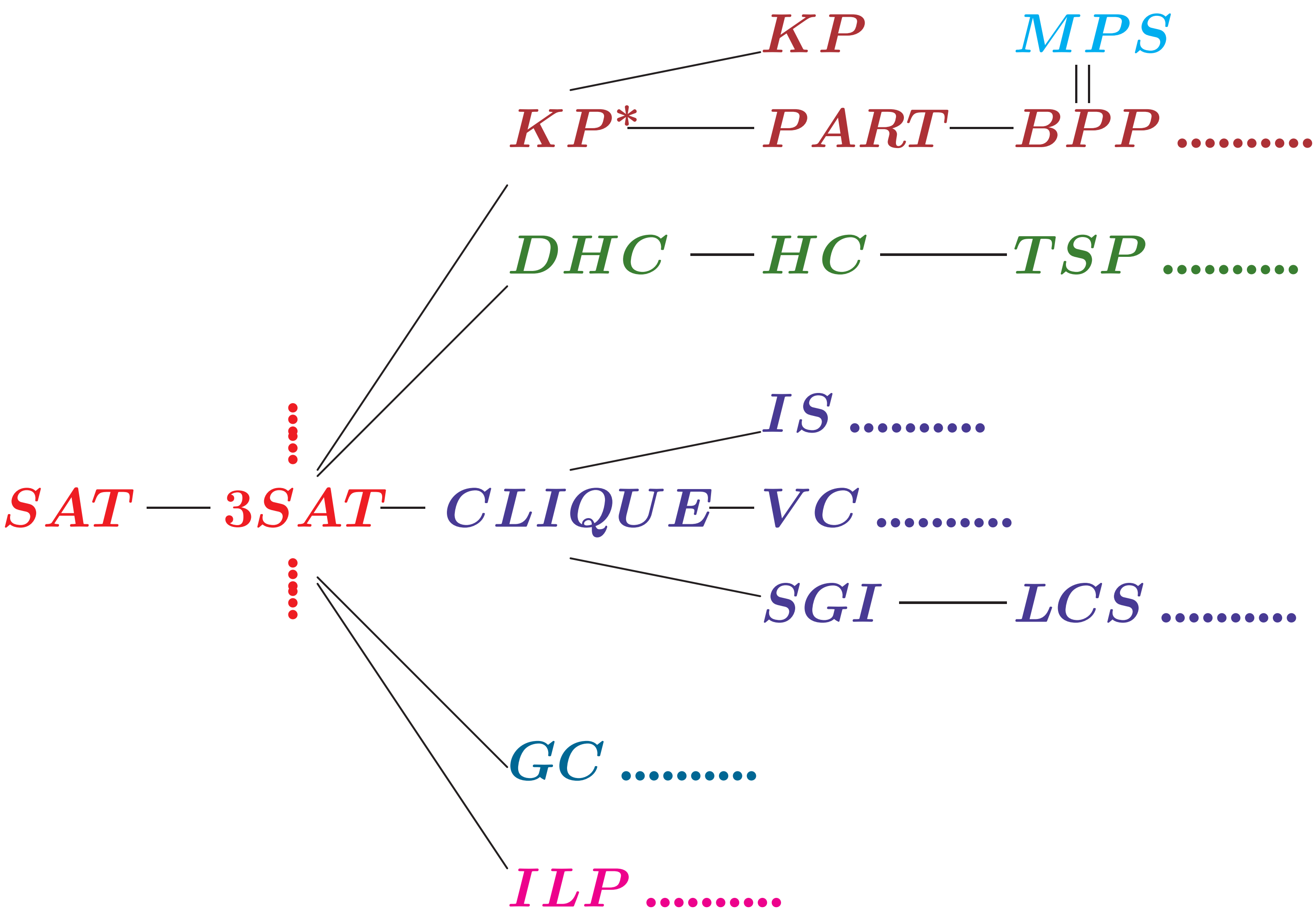
- **Binpacking**
  - Gegeben n Objekte der Größe $a_1, ...a_n$ und k Behälter der Größe b
  - Kann man alle Objekte in den Behältern unterbringen?

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

- **Multiprozessor-Scheduling** ($\boldsymbol{MPS}$)
  - 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** ($\boldsymbol{ILP}$)
  - 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



ANHANG

DHC IST $\mathcal{NP}$-VOLLSTÄNDIG: $\boldsymbol{3SAT \leq_p DHC}$

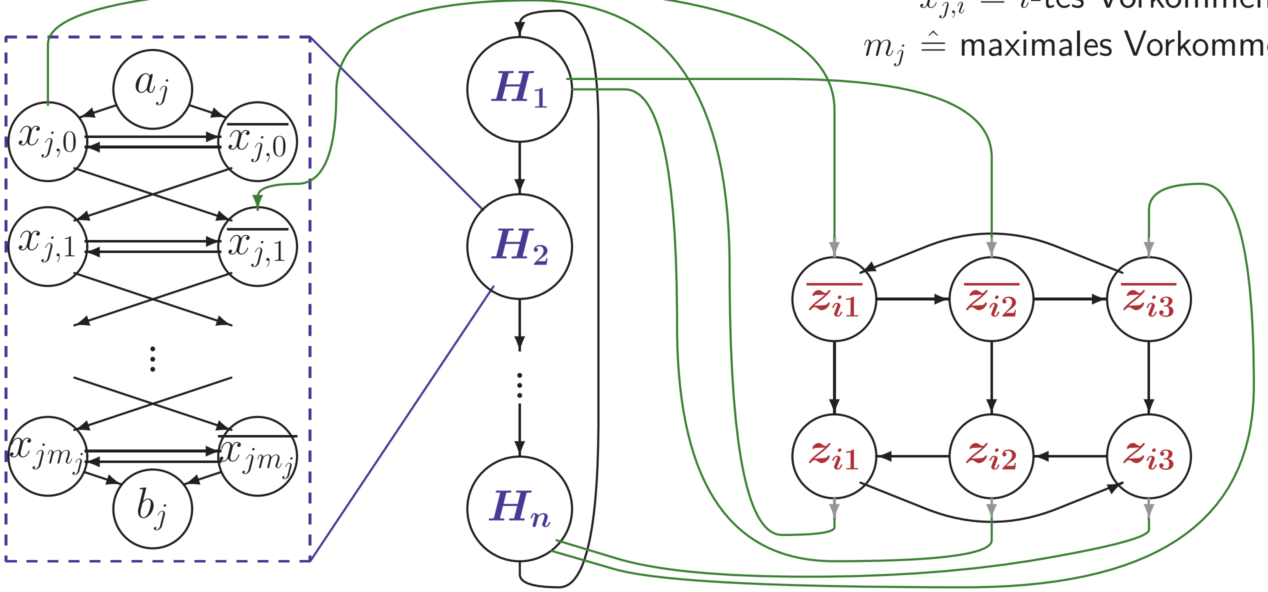
Gegeben $F = (k_1, .., k_m)$ mit $k_i = z_{i1} \vee z_{i2} \vee z_{i3}$ und $z_{ij} \in \{x_1, .., \overline{x_n}\}$

e) **Konstruiere Graphen $\boldsymbol{G_F \equiv f(F)}$ aus drei Komponenten**

· Teilgraphen H_j für Codierung der Variablenvorkommen von $x_j/\bar{x}_j$

$x_{j,i} \hat{=}$ i-tes Vorkommen von x_j in F

$m_j \hat{=}$ maximales Vorkommen von $x_j/\bar{x}_j$



· Teilgraph für Codierung der Klauseln

· Verbindungskanten zwischen Variablen und Klauselliteralen

Für z_{ik}=x_j verbinde erstes ungenutzte Vorkommen $x_{j,p}$ mit $\overline{z_{ik}}$ und $z_{i,k}$ mit $\overline{x_{j,p+1}}$

Für z_{ik}=$\overline{x_j}$ verbinde erstes ungenutzte Vorkommen $\overline{x_{j,p}}$ mit $\overline{z_{ik}}$ und z_{ik} mit $x_{j,p+1}$

g) $\boldsymbol{f}$ **ist in polynomieller Zeit berechenbar**

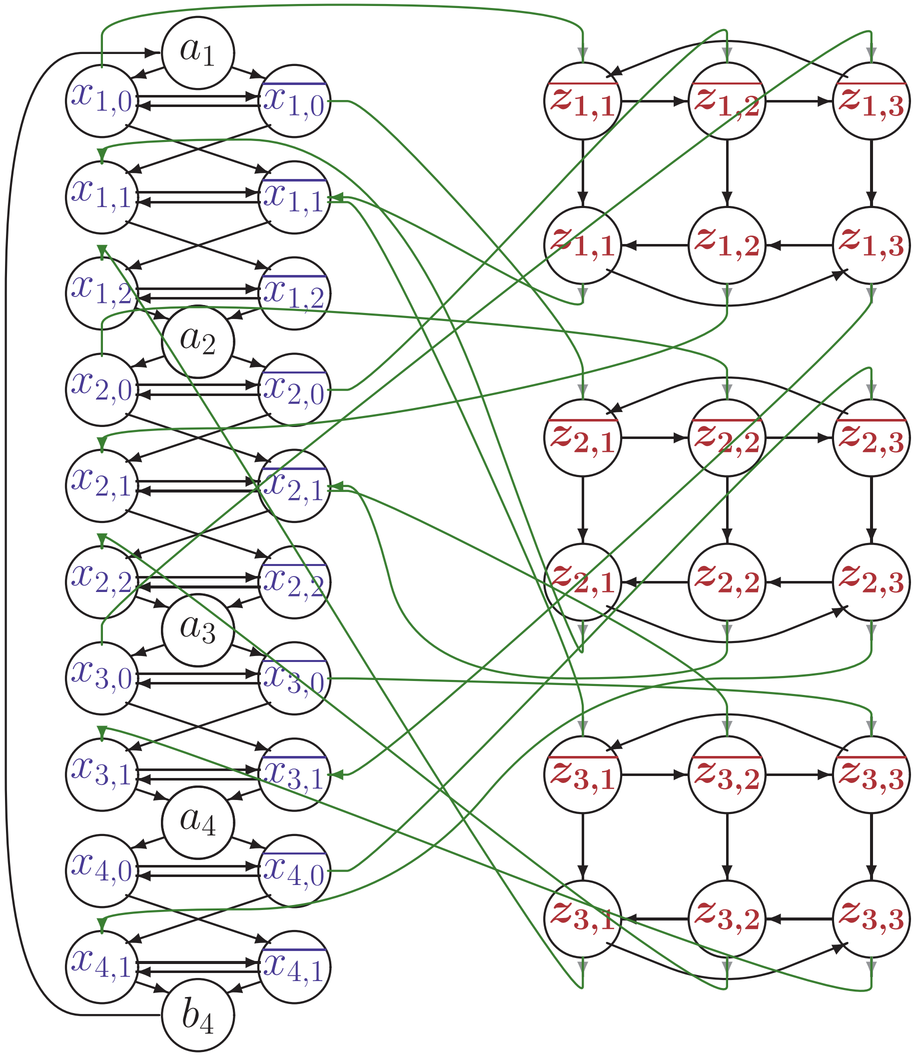
Codierung einer Formel als DHC Problem

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

$b_1 = a_2$

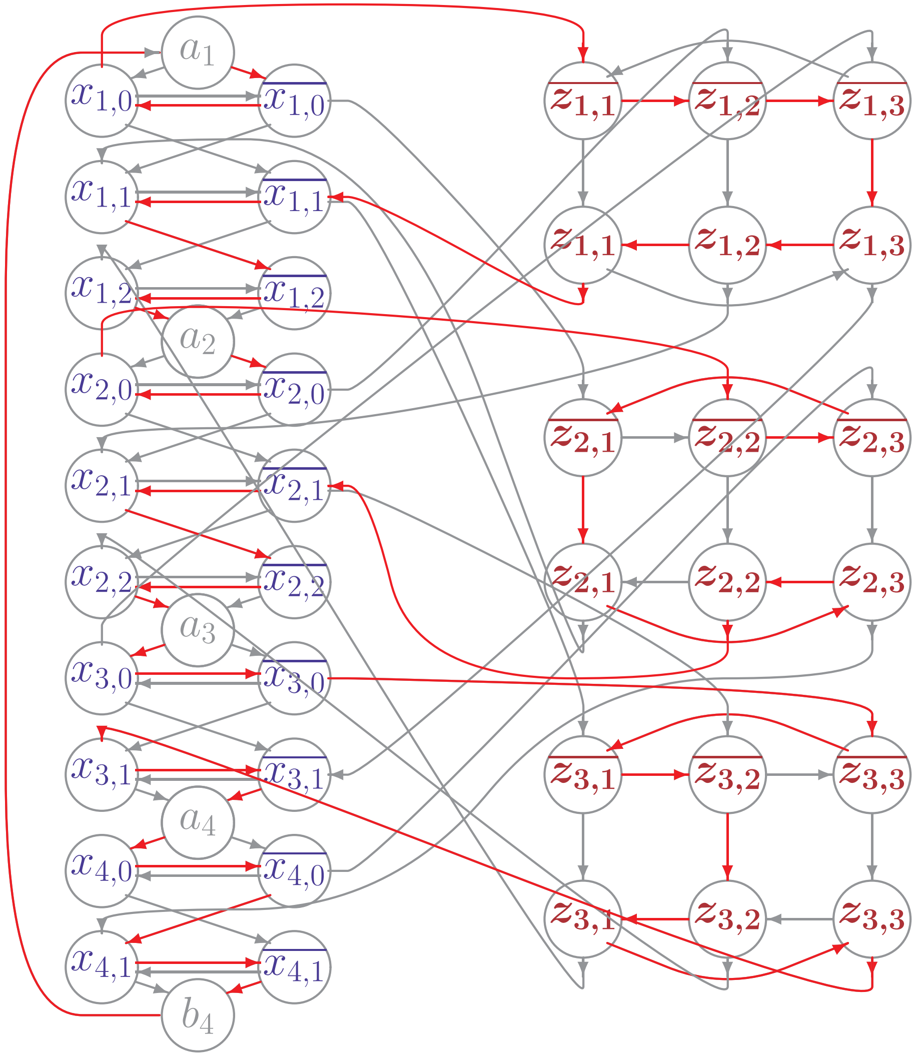
$b_2 = a_3$

$b_3 = a_4$



Erfüllende Belegung als DHC

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



Erfüllende Belegung: (1, 1, 0, 0)

Beginne mit Teilpfad $a_1 \to \overline{x_{1,0}} \to x_{1,0}$

Verbinde $x_{1,0}$ mit Klausel 1

Laufe von $\overline{x_{1,1}}$ durch H_1 bis a_2

Weiter mit Teilpfad $a_2 \to \overline{x_{2,0}} \to x_{2,0}$

Verbinde $x_{2,0}$ mit Klausel 2

Laufe von $\overline{x_{2,1}}$ durch H_2 bis a_3

Weiter mit Teilpfad $a_3 \to x_{3,0} \to \overline{x_{3,0}}$

Verbinde $\overline{x_{3,0}}$ mit Klausel 3

Laufe von $x_{3,1}$ durch H_3 bis a_4

Beliebig weiter durch H_3 bis b_4 und a_1

KORREKTHEIT (SKIZZE): $F \in 3SAT \Leftrightarrow f(F) \in DHC$

- **Es gelte $F \in 3SAT$**
  Sei $c_1, ..c_n$ eine erfüllende Belegung von F.
  Konstruiere einen Hamiltonschen Kreis wie folgt
  – In H_j beginne mit a_j und $\overline{x_{j,0}}$ falls $c_j = 1$, sonst mit a_j und $x_{j,0}$
  – Verbinde $\overline{x_{j,p}}$ mit $x_{j,p}$ und dann mit $\overline{x_{j,p+1}}$.
  Wenn möglich, gehe dabei über die Knoten einer verbundenen Klausel z_i
  – Analog verbinde $x_{j,p}$ mit $\overline{x_{j,p}}$ und dann mit $x_{j,p+1}$, evtl. mit Umweg
  – Verlasse H_j in b_j und verbinde mit H_{j+1}
  Da $c_1, ..c_n$ die Formel F erfüllt, **wird jedes H_j und z_i durchlaufen**.

- **Es gelte $G_F \in DHC$**
  Verbindet der Kreis a_j mit $x_{j,0}$ wähle $c_j = 0$, sonst $c_j = 1$
  – Betritt der Kreis Klausel z_i bei $\overline{z_{i,k}}$, so muß er sie bei $z_{i,k}$ verlassen
  – Damit verbindet der Kreis immer ein $\overline{x_{j,p}}$ mit $x_{j,p}$ mit $\overline{x_{j,p+1}}$ $(c_j = 1)$
  oder $x_{j,p}$ mit $\overline{x_{j,p}}$ mit $x_{j,p+1}$ $(c_j = 0)$, bis er H_j verläßt.
  – Bei Umweg über z_i muß das verbundene Literal $z_{ik} \in \{x_j, \overline{x_j}\}$ erfüllt sein
  – Da alle Klauseln durchlaufen werden, **sind alle Klauseln erfüllt**.