Komplexitätstheorie 5

Montag und Donnerstag
14:15 – 15:45 Uhr
in C-221

NP-Vollständigkeitsbeweise durch Reduktion

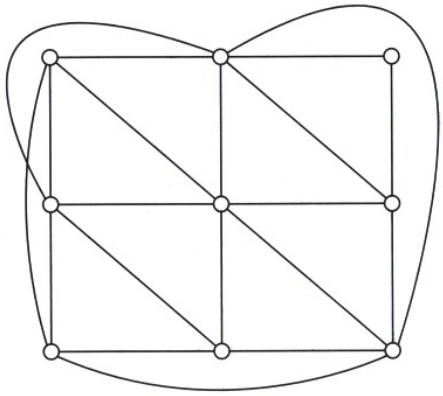
- $3{-}SAT <_{pol} VC$ (*vertex cover*)
- $VC <_{pol} Clique$
- $3{-}SAT <_{pol} IS$ (*Independent Set*)
- $VC <_{pol} HC$ (*hamiltonian circuit*)
- $3{-}SAT <_{pol} HP$ (*hamiltonian path*)
- $3{-}SAT <_{pol} 3{-}DM$ (*3-dimensional matching*)
- $3{-}DM <_{pol} Partition$
- ...

Vertex Cover

Definition:
VC (Knotenüberdeckung, *vertex cover*)

Eingabe:
Ein endlicher Graph $G = (V, E)$ und eine Zahl $k \leq |V|$.

Frage:
Gibt es $U \subset V$ so, dass $|U| \leq k$ und
$\forall (v_i, v_j) \in E : \{v_i, v_j\} \cap U \neq \emptyset$?



Clique und *Independent Set*

Definition:
Clique

Eingabe:
Ein endlicher Graph $G = (V, E)$, und eine Zahl $k \in I\!N$.

Frage:
Gibt es $U \subseteq V$ so, dass $|U| \geq k$ und $(U \times U) \setminus Id \subseteq E$?

Definition:
IS (*Independent Set*)

Eingabe:
Ein endlicher Graph $G = (V, E)$, und eine Zahl $k \in I\!N$.

Frage:
Gibt es $U \subseteq V$ so, dass $|U| \geq k$ und $(U \times U) \cap E = \emptyset$?

nützlicher Zusammenhang

Lemma:

Sei $G = (V, E)$ endl. Graph und $U \subseteq V$, dann sind folgende Aussagen äquivalent:

1. U ist Knotenüberdeckung (*vertex cover*),
2. $V \setminus U$ ist independent set für G,
3. $V \setminus U$ ist *Clique* im Komplement G^c von G, d.h., $G^c := (V, E^c)$, wobei $E^c := \{(v_1, v_2) \mid v_1, v_2 \in V : (v_1, v_2) \notin E\}$

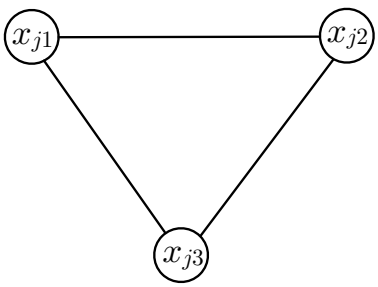
Alle drei Probleme sind verschiedene Ausprägungen der gleichen Fragestellung!
Zudem sind die Probleme polynomiell ineinander transformierbar und wenn eines davon *NP*-vollständig ist, sind es die anderen auch!

$3{-}SAT <_{pol} VC$ (1)

Beweis:

1. Der Nachweis von $VC \in NP$ ist leicht: Eine geratene Knotenüberdeckung $U \subseteq V$ wird daraufhin überprüft, ob jede Kante $e \in E$ tatsächlich einen Endpunkt in U hat. Dazu wird für jede Kante die gesamte Knotenmenge durchsucht, d.h. der Aufwand ist $|E| \cdot |V|$.
2. Zur Reduktion von $3 - SAT$ auf VC wird zu jeder Formel $F = c_1 \wedge c_2 \wedge \cdots \wedge c_q$ ein Graph $G(F)$ konstruiert, so dass für ein noch zu bestimmendes k gilt: $F \in 3{-}SAT \leftrightarrow (G(F), k) \in VC$.

   Zu jeder Klausel $c_j = (x_{j1} \vee x_{j2} \vee x_{j3})$ mit den Literalen x_{j1}, x_{j2}, x_{j3}, wird der Graph $G(c_j) := (V_{c_j}, V_{c_j} \times V_{c_j})$ mit $V_{c_j} := \{x_{j1}, x_{j2}, x_{j3}\}$ definiert.



Eine Knotenüberdeckung eines Dreiecks benötigt immer zwei Knoten!

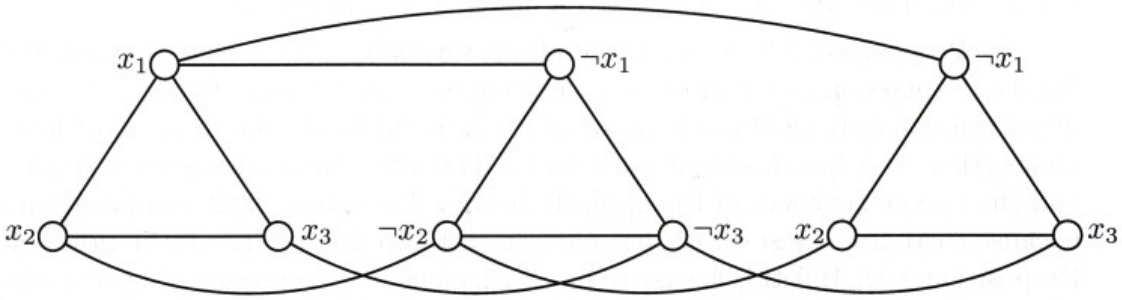
$3{-}SAT <_{pol} VC$ (2)

Nun werden alle Dreiecke $G(c_j)$, $c_j \in F$, disjunkt vereinigt zu $\bigcup_{j=1}^{q} G(c_j)$.

Hinzu kommen nun Kanten $(x, \overline{x})$, falls x Variable ist, deren Literale x und $\overline{x}$ in verschiedenen Teilgraphen $G(c_j)$ vorkommen.

Das Ergebnis ist der Graph $G(F)$.

$$(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$$



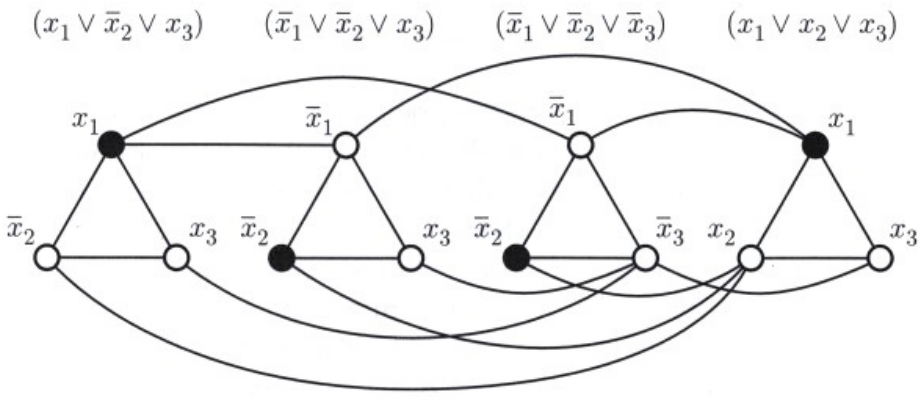
$3{-}SAT <_{pol} VC$ (3)

In $G(F)$ kann es keine Knotenüberdeckung mit weniger als $2q$ Knoten geben, denn jedes Dreieck benötigt schon zwei Knoten dafür.

Wir zeigen (sogar):

$$F \in 3{-}SAT \leftrightarrow (G(F), 2q) \in VC.$$

Sei σ eine erfüllende Belegung für F, d.h. $\sigma(F)$=1. In jeder Klausel wird also mindestens ein Literal x_{ij} wahr.



$3\text{-}SAT <_{pol} VC$ (4)

In den Teilgraphen $G(c_j)$ wählen wir genau 2 Knoten so aus: diejenigen Knoten in jedem Dreieck, die zu Literalen gehören, die durch die Belegung **nicht erfüllt** werden und in jedem Dreieck eventuell einen weiteren, der zu einer Variablen gehört, die in **keinem** Literal erfüllt wird. Damit sind das genau $2q$ viele und U mit $|U|=2q$ ist Knotenüberdeckung.
(Alle Dreiecke sind überdeckt und alle Kanten $(x_j,\neg x_j)$ auch, denn wenn x_j wahr ist, so wird Knoten zum Literal $\neg x_j$ ausgewählt (sofern er existiert, und nur dann gibt es diese Kante) und wenn die Variable x von der Belegung unberücksichtigt bleibt, wird ohnehin das zugehörige Literal ausgewählt.)

$3\text{-}SAT <_{pol} VC$ (5)

Wenn umgekehrt $U \subseteq V$ mit $|U| = 2q$ Knotenüberdeckung ist, so sind in jedem Dreieck $G(c_j)$ genau zwei Knoten markiert (d.h. in U). Wir setzen den Wahrheitswert des Literal, das zum nichtmarkierten Knoten eines Dreiecks gehört auf *wahr*. Damit wird die Formel erfüllt, denn jedes Dreieck entspricht einer Klausel. Dabei kann es vorkommen, dass manchen Variablen kein Wert zugewiesen wurde, was aber nicht schlimm ist, denn diese Wahrheitswerte können beliebig gewählt werden. Knoten mit Kanten zwischen negierten und nicht negierten Variablen kommen in der Knotenüberdeckung vor, sind also noch nicht belegt worden!

Die hier benutzten Knoten entsprechen also dem *independent set* und lösen dieses Problem.

Clique

Im *Cliquen-Problem* wir danach gefragt, ob in einem Graph G=(V,E) zu einer Zahl k eine Clique mit mindestens k Knoten existiert.

Beweis:

1. Der Nachweis von $Clique \in NP$ ist leicht: Eine geratene Knotenauswahl $U \subseteq V$ mit $k = |U|$ wird daraufhin überprüft, ob zu jedem Knotenpaar aus $U \times U$ eine Kante $e \in E$ existiert, d.h. ob $U \times U \subseteq E$ gilt. Dazu wird für jedes Element aus $U \times U$ die gesamte Kantenmenge durchsucht, d.h. der Aufwand ist $|E| \cdot |V|^2$.
2. Die Reduktion von VC auf $Clique$ ist einfach: Sei $(G = (V, E), k)$ Instanz des Knotenüberdeckungsproblems. Konstruiere den Komplementären Graph G^c und benutze die Konstante $|V| - k$. Es gibt in G^c eine Clique U mit mindestens $|V| - k$ Knoten genau dann, wenn $V \backslash U$ eine Knotenüberdeckung von G mit höchstens k Knoten ist.

$3{-}SAT <_{pol} 3{-}DM$ (1)

Definition:
3-DM (*3-dimensional matching*):

Eingabe:
Eine Menge $M \subseteq W \times X \times Y$, wobei W, X, Y disjunkte Mengen gleicher Kardinalität q sind.

Frage:
Enthält M ein *matching*, d.h. gibt es Teilmenge $M' \subseteq M$ mit $|M'| = q$ und keine zwei Elemente aus M stimmen in irgendeiner Komponente überein?

Als sogenanntes Heiratsproblem ist dies für Mengen $M \subseteq X \times Y$ bekannt, und (gut für Heiratsvermittler) in P !
Um zu zeigen, dass 3-DM NP-vollständig ist, muss ein bekanntes NP-vollständiges Problem darauf reduziert werden.
Wir wählen dazu $3{-}SAT$, führen den Beweis aber nicht vollständig durch! (wird zu langwierig!)

$3-SAT <_{pol} 3-DM$ (2)

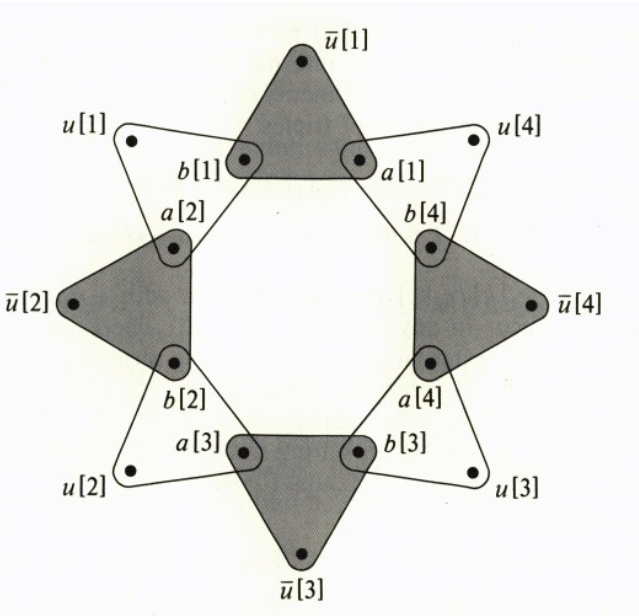
1.
Zunächst ist offensichtlich $3-DM$ in NP:
für eine geratene Menge M, muss nur verifiziert werden, dass in allen Tripeln aus M verschiedene Elemente vorkommen, d.h., wenn jedes Tripel als Menge mit 3 Elementen interpretiert werden, sind diese alle paarweise disjunkt!

2.
Nun muss eine beliebige Instanz von $3-SAT$ in ein $3-DM$ Problem umgeformt werden.
Dazu seien:
$U := \{u_1, u_2, ..., u_n\}$ die Variablen und $C := \{c_1, c_2, ..., c_m\}$ die Klauseln,
d.h. jedes $c_i = u_r \vee u_s \vee u_t$ (wird meist notiert als $c_i = \{u_r, u_s, u_t\}$).

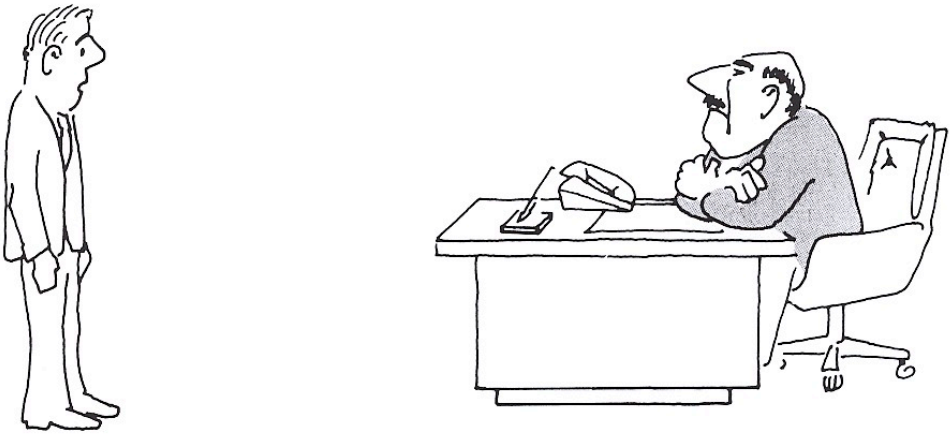
$3-SAT <_{pol} 3-DM$ (4)

Nun müssen Mengen W, X und Y konstruiert werden, mit $|W| = |X| = |Y|$ und eine Menge $M \subseteq W \times X \times Y$, die genau dann ein *matching* besitzt, wenn die Konjunktion der Klauseln erfüllbar ist.

Konstruktionen, ähnlich denen, die beim Cliquen-Problem oder Knotenüberdeckungen benutzt werden, führen auch hier zum Erfolg.
Die Standardliteratur ist immer noch
M.R. Garey und D.S. Johnson: Computers and Intractability: A Guide to the Theory of NP-Completenes, Freeman, San Francisco (1979).
Dort auch diese Bilder:

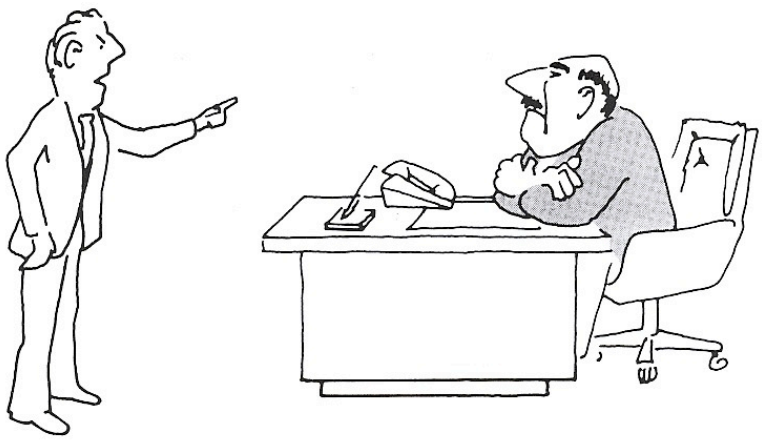


Warum Komplexitätstheorie wichtig ist:



"I can't find an efficient algorithm, I guess I'm just too dumb."

Zeichnungen aus Garey & Johnson



"I can't find an efficient algorithm, because no such algorithm is possible!"

Aber ist der Chef damit zufrieden?

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

Hamiltonkreis

Zur Erinnerung:

In einem Graphen (gerichtet oder ungerichtet) nennt man einen geschlossenen Pfad *Hamiltonkreis*, wenn jeder Knoten auf diesem Kreis genau einmal vorkommt.

Wir wollen *EC* (*Exakt Cover*) auf *dHC* (*Hamiltonkreis* in gerichteten Graphen, *directed hamiltonian cycle*) reduzieren und danach *dHC* auf *HC* (*Hamiltonkreis* in ungerichteten Graphen, *hamiltonian cycle*).

Zunächst Definition von *EC,* dann Redukionen, danach (viell.) NP-Vollständigkeit von *EC* durch $SAT <_{pol} EC$.

Exakte Mengenüberdeckung (*EC*)

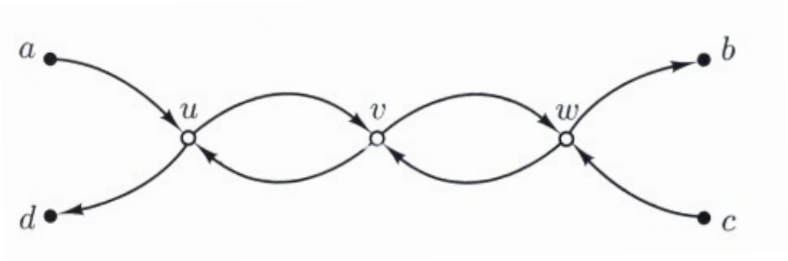
Definition:
EC (Exakte Mengenüberdeckung, *exact cover*)

Eingabe:
Eine endliche Familie von endlichen Mengen $\mathcal{F} = \{S_1, S_2, \ldots, S_m\}$.

Frage:
Gibt es Teilfamilie $\mathcal{D} \subseteq \mathcal{F}$ so, dass die Elemente aus $\mathcal{C}$ paarweise disjunkt sind und $\bigcup_{S_i \in \mathcal{D}} S_i = \bigcup_{S_j \in \mathcal{F}} S_j$?

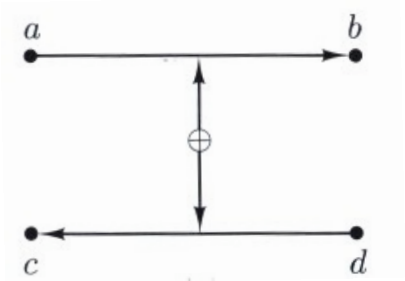
Wir erinnern ähnliche Probleme mit Zahlenwerten:
Partition oder *Knapsack* wie diese, ist auch *EC* *NP-vollständig!*

$EC <_{pol} dHC$ (*directed hamiltonian cycle*) (1)

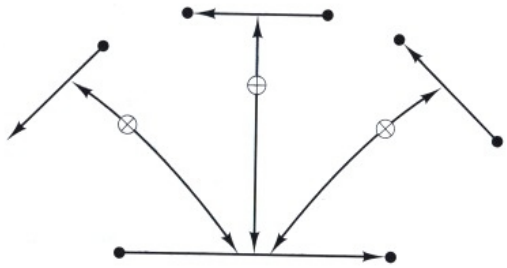


Gadgets (kleine Hilfsgeräte) wie der Teilgraph oben, sorgen dafür, dass entweder $a \to b$ oder $c \to d$ über $u{-}v{-}w$ gegangen wird.
Im obigen *gadget* sind mit u,v,w keine weiteren Kante verbunden, aber a,b,c und d sicherlich!
Das ist wie ein Schalter zwischen "Kanten" $a \to b$ oder $c \to d$.

Die Grafik unten kürzt obiges *gadet* zur Vereinfachung ab!



$EC <_{pol} dHC$ (*directed hamiltonian cycle*) (2)



(c)

In dieser Konstruktion bedeutet das:
entweder wird im Hamiltonkreis des Graphen, der diesen Teilgraphen enthält, die untere Kante (der dadurch abstrahierte Pfad) benutzt,
oder alle drei Kanten oben werden durchlaufen!

Für die Transformation $EC <_{pol} dHC$ werden solche *gadgets* verwendet!

$EC <_{pol} dHC$ (*directed hamiltonian cycle*) (3)

Definition:

Sei ein EC gegeben durch die Mengenfamilie $\mathcal{F} = \{S_1, \ldots, S_m\}$. und $U := \{u \mid \exists S_i \in \mathcal{F} : u \in S_i\}$.
Wir definieren endl. Graphen $G(\mathcal{F}) := (V, E)$ mit folgender Knotenmenge $V := V_1 \uplus V_2 \uplus V_3$:

1. $V_i := U$ (ein Knoten für jedes Element der Mengenvereinigung $\bigcup_{S_i} \in \mathcal{F} S_i$),
2. $V_2 := \{S_i \mid 1 \leq i \leq m\}$ (ein Knoten für jedes Element aus $\mathcal{F}$)
3. $\{u_0, S_0\}$

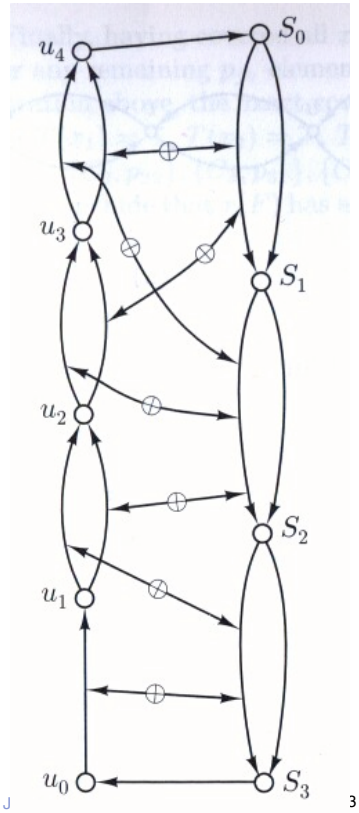
Die Kanten des Graphen $G(\mathcal{F})$ werden an einer Grafik erläutert:

$EC <_{pol} dHC$ (*directed hamiltonian cycle*) (4)

$S_1 = \{u_3, u_4\}$
$S_2 = \{u_2, u_3, u_4\}$
$S_3 = \{u_1, u_2\}$

Zwischen u_{j-1} und u_j sind soviele Kopien von Kanten, wie es Mengen S_i gibt, die u_j enthalten.
Jede Kopie einer Kante (u_{j-1}, u_j) wird *exclusiv-or* verbunden mit sog. langer Kante (S_{i-1}, S_i), falls $u_j \in S_i$ ist.
S_0 und u_0 schließen den Graph!

Dieser Graph hat genau dann gerichteten Hamiltonkreis, wenn es eine exakte Mengenüberdeckung für $\mathcal{F}$ gibt!



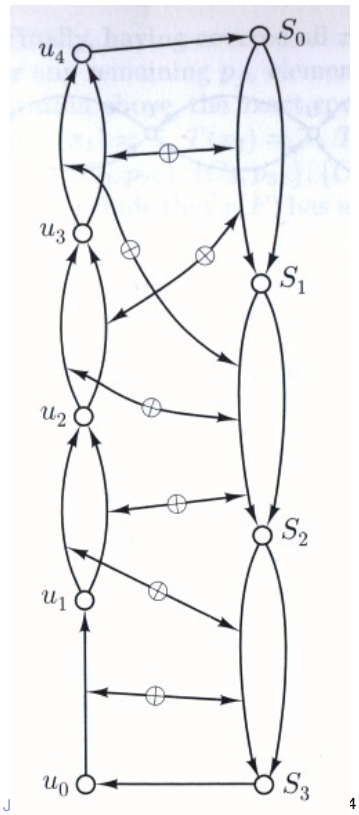
$EC <_{pol} dHC$ (*directed hamiltonian cycle*) (5)

$S_1 = \{u_3, u_4\},\ S_2 = \{u_2, u_3, u_4\},\ S_3 = \{u_1, u_2\}$

Wenn es Hamiltonkreis in G gibt, so geht der durch $S_0,\ S_1,\ \ldots,\ S_m,\ u_0,\ \ldots,\ u_n$.
Die "kurzen" Kanten, die der Pfad benutzt, definieren ein *exact cover*:

$\mathcal{D} := \{S_i \mid (S_{i-1}, S_i)$ ist kurze Kante$\}$.

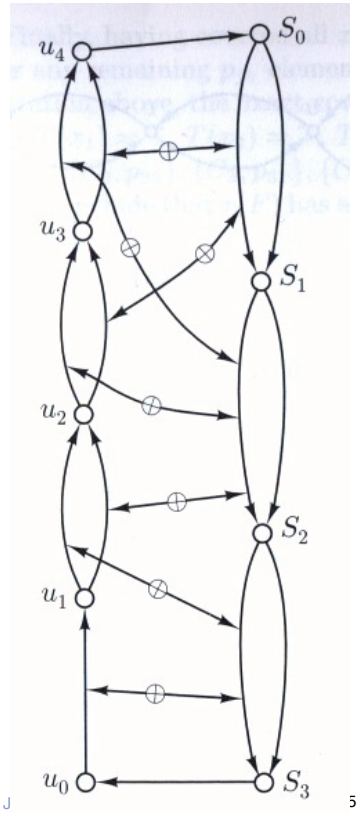
Die Kopien (u_{j-1}, u_j) [ohne $\oplus$ Verbindung] werden auf der anderen Seite durchlaufen, genau eine dieser Kanten kommt für jedes u_i vor und u_i kommt in genau einer Menge von $\mathcal{D}$ vor.



$EC <_{pol} dHC$ (*directed hamiltonian cycle*) (5)

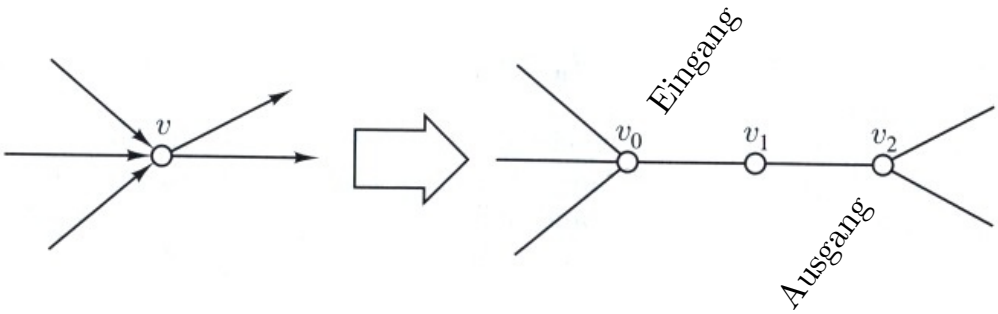
$S_1 = \{u_3, u_4\},\ S_2 = \{u_2, u_3, u_4\},\ S_3 = \{u_1, u_2\}$

Wenn es ein *exact cover* $\mathcal{D}$ gibt, so wird ein Hamiltonkreis so bestimmt: Wähle kurze Kanten mit Endpunkt in einem $S_j \in \mathcal{D}$, sonst die langen Kanten. Für jedes u_i nimm Kopie einer Kante (u_{i-1}, u_i) die zur (eindeutigen) Menge von $\mathcal{D}$ gehört, die u_i enthält!
Die Kanten (u_n, S_0) und (S_m, u_0) schließen den Kreis dann.



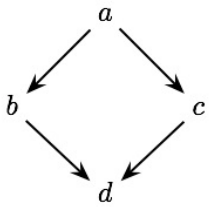
$dHC <_{pol} HC$ (*hamiltonian cycle*) (1)

Zu einem gerichten Graph wird ungerichter Graph konstruiert, der genau dann Hamiltonkreis hat, wenn es einen solchen im gerichten Graphen gibt:



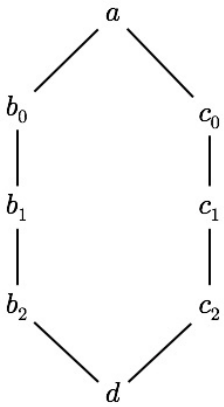
$dHC <_{pol} HC$ (*hamiltonian cycle*) (2)

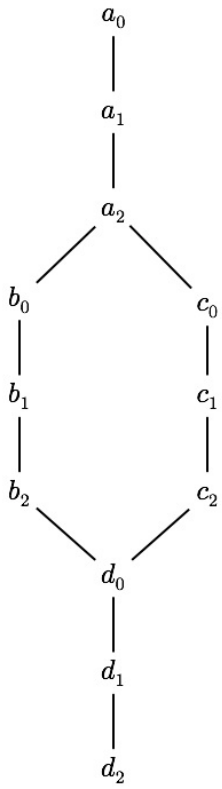
Wie sieht der ungerichtete Graph hierzu aus?



Nein So:

So?





$dHC <_{pol} HC$ (*hamiltonian cycle*) (3)

Satz:

Die Hamiltonkreis Probleme dHC und HC sind NP-vollständig.

Satz:

Das Mengenüberdeckungsproblem EC (*exact cover*) ist NP-vollständig.

Zum *EC-Problem*: Reduktion von SAT, siehe Lewis & Papadimitriou: Elements of the Theory of Computing, Prentice Hall (1998).

wie geht man mit NP-vollständigen Problemen um?

Heuristische Verfahren:

- Ziel: schnelle Laufzeit
- "hoffentlich" wird gute Lösung gefunden
- typisch dafür: greedy Verfahren

Randomisierte Verfahren:

- Ziel: finde (optimale) Lösung mit großer Wahrscheinlichkeit

wie geht man mit NP-vollständigen Problemen um?

parametrisierte Verfahren:

- finden stets (optimale) Lösung
- Beschränkung der nicht-polynomiellen Laufzeit durch Parameter

Näherungsverfahren:

- Heuristiken mit Leistungsgarantie
- Güte kann gemessen werden:
  a) absolut
  b) relatv zum Optimium

Approximation statt exakt

Literatur dazu:
G. Ausiello et al.: Complexity and Aproximation, Springer (2003)

(sehr umfangreiche Darstellung)

Kap. 13 in
Ch.H. Papadimitriou: Computational Complexity, Addison Wesley (1994)

(darin nur Kapitel 13)

Approximation *Definitionen*

Definition:
Ein Optimierungsproblem P wird charakterisiert durch $(I_P, SOL_P, m_P, goal_P)$, wobei:

1. I_P die Menge aller Instanzen von P ist,
2. SOL_P ist Funktion, die $x \in I_P$ die Menge aller durchführbaren Lösungen zu x zuordnet,
3. m_P ist Funktion (*measure function*)
   $m_P : I_P \times SOL_P \to \mathbb{Q}$ (oder $\mathbb{N}$),
   die Wert der Lösung ergibt,
4. $goal_P \in \{MIN, MAX\}$.

$SOL^*{}_P(x)$ bezeichnet die Menge der optimalen Lösungen zu $x \in I_P$ bzgl. $goal_P$

Für alle $y^* \in SOL_P(x)$ sei:

$m_P(x, y^*) := goal_P\{v \mid v = m_P(z) \wedge z \in SOL_P(x)\}$

$m^*_P(x, y)$ bezeichnet Wert der optimalen Lösung.

Optimierungsprobleme: *Instanzbeschreibung*

Definition:
Maximum Knapsack (MK) $(goal_{MK} = MAX)$

Eingabe:
Eine endliche Menge X von Objekten,
$\forall x_i \in X:$ Wert $p_i \in \mathbb{N}^+$
sowie Größe $a_i \in \mathbb{N}^+$ und eine Zahl $p \in \mathbb{N}^+$.

Gesucht:
$Y \subset X$ so, dass $\sum_{x_i \in Y} a_i \leq p$.

Measure:
$\sum_{x_i \in Y} p_i$

Hier (jetzt) keine Verfahren, sondern zunächst Definition der Güteberechnung!

Optimierungsklassen OP und NOP

Definition:

Ein Optimierungsproblem $P = (I_P, SOL_P, m_P, goal_P)$ gehört zur Klasse NPO genau dann, wenn:

1. $I_P \in P$,
2. $\exists p \in POLY \forall x \in I_P \forall y \in SOL_P(x) : |y| \leq p(|x|)$
   und $\forall y$ mit $|y| \leq p(|x|)$ ist die Frage „$y \in SOL_P(x)$?“ in Polynomzeit entscheidbar,
3. m_P ist in Polynomzeit berechenbar.

Definition:

Ein Optimierungsproblem $P = (I_P, SOL_P, m_P, goal_P)$ gehört zur Klasse PO genau dann, wenn:

1. $P \in NPO$,
2. Es gibt Algorithmus A, der in Polynomialzeit für jedes $x \in I_P$ die optimale Lösung $y \in SOL_P$ zusammen mit dem Wert $m^*(x)$ berechnet.

Approximation *Bewertung*

Definition:

Sei P ein Optimierungsproblem, x eine Instanz von P, dann ist der absolute Fehler $D(x, y)$ einer Lösung y definiert durch:

$$D(x, y) := |m^*(x) - m(x, y)|.$$

Definition:

Ist A ein Approximationsalgorithmus für P, d.h. es gilt

$$\forall x \in I_P : A(x) \in SOL_P,$$

dann heißt A **absoluter** Approximationsalgorithmus, wenn es Konstante $k \in \mathbb{N}$ gibt, so dass

$$\forall x \in I_P : A(x) \leq k.$$

Satz:

Falls $NP \neq P$, dann gibt es keinen polynomiellen absoluten Approximationsalgorithmus für $MaximumKnapsack$.

Beweis:

Sei $X = \{x_1, \ldots x_n\}$ eine Menge von Gegenständen mit den zugehörigen Profiten p_i und Größen a_i bei der Rucksackkapazität p. Wäre MK mit einem polynomiellen absoluten Approximationsalgorithmus A bei der Fehlerschranke k lösbar, so könnten wir $MK \in P$ wie folgt beweisen:

Wir generieren eine neue Instanz, indem wir alle Profite mit $(k+1)$ multiplizieren. Die Menge SOL_{MK} der zulässigen Lösungen wird dadurch nicht geändert. Der Wert der Lösung ist nun aber ein Vielfaches von $(k + 1)$, so dass eine Lösung mit absolutem Fehler nicht größer als k bereits optimal ist!

ε-Approximation

In der Literatur gibt es mindestens zwei verschiedene Maße um relative Approximation zu bewerten:

ε-Approximierbarkeit und r-Approximierbarkeit

Definition:

Ist A ein Approximationsalgorithmus für P,
so ist für jede Instanz $x \in I_P$ und jede Lösung $y \in SOL_P(x)$

$$E(x,y) := \frac{|m^*(x) - m(x,y)|}{\max(m^*(x), m(x,y))}$$

der relative Fehler von y bezüglich x mit $0 \leq E(x,y) \leq 1$.

A nennt man ϵ-Approximation, wenn

$$\forall x \in I_P : E(x, A(x)) \leq \epsilon.$$

r-Approximation

Definition:

Ist A ein Approximationsalgorithmus für P,
so ist für jede Instanz $x \in I_P$ und jede Lösung $y \in SOL_P(x)$

$$R(x,y) := \max\left(\frac{m^*(x)}{m(x,y)}, \frac{m(x,y)}{m^*(x)}\right)$$

der Leistungsgüte von y bezüglich x mit $1 \leq R(x,y)$.

A nennt man r-Approximation, wenn

$$\forall x \in I_P : R(x, A(x)) \leq r.$$

Bemerkung:

Die Leistungsgüte ist gleich Eins, wenn die Lösung optimal ist, und je schlechter sie ist, umso größer wird sie! Offenbar gilt:

$$E(x,y) = 1 - \frac{1}{R(x,y)}.$$

r-Approximation

Beweis:

Ist P ein Maximierungsproblem, so ist

$$E(x,y) = 1 - \frac{1}{R(x,y)} = 1 - \frac{1}{\frac{m^*(x)}{m(x,y)}} = 1 - \frac{m(x,y)}{m^*(x)} = \frac{m^*(x) - m(x,y)}{m^*(x)},$$

Ist P ein Minimierungsproblem, so ist

$$E(x,y) = 1 - \frac{1}{R(x,y)} = 1 - \frac{m^*(x)}{m(x,y)} = \frac{m(x,y) - m^*(x)}{m(x,y)}.$$

Algorithmen, die im Minimisierungsfall

$$m(x,y) \leq r \cdot m^*(x) + k$$

erfüllen, sind gem. Def. nur $(r-k)$-Approximationen, werden i.d. Literatur aber oft als r-Approximationen bezeichnet.

relative Approximation: die Klasse APX

Definition:

Ein Optimierungsproblem $P = (I_P, SOL_P, m_P, goal_P)$ gehört zur Klasse APX genau dann, wenn:

1. $P \in NPO$,
2. Es gibt Approximation A, die P für ein $r \geq 1$ r-approximiert.

Das folgende Problem $MaxSAT$ ist $2-$approximierbar, also in APX:

Ein Problem aus APX

Definition:
$MaxSAT$ $(goal_{MK} = MAX)$

Eingabe:
Eine endliche Menge C von Klauseln,
über einer Menge V von Variablen,
die eine KNF beschreiben,

Gesucht:
Eine Belegung $f : V \to \{true, false\}$.

Measure:
$|\{c \mid f(c) = true\}|$

$MaxSAT$

Beweis:

Wir analysieren folgenden Greedy-Algorithmus:

1. Für alle $v \in V$ setze $f(v) := true$;
2. Wiederhole, solange $C \neq \emptyset$:
   - 2a. Sei l ein Literal, das in einer maximalen Zahl von Klauseln aus C vorkommt und x die zugehörige Variable.
   - 2b. Bestimme die Menge C_x von Klauseln in denen x vorkommt.
   - 2c. Bestimme die Menge $C_{\overline{x}}$ von Klauseln in denen $\overline{x}$ vorkommt.
   - 2d. Ist $l = x$, so
     - 2d(i). $C := C \setminus C_x$
     - 2d(ii). Lösche $\overline{x}$ aus allen Klauseln in $C_x \setminus C_{\overline{x}}$
     - 2d(iii). Lösche alle leeren Klauseln aus C
   - 2e. sonst:

MaxSAT (Fortsetzung)

2e. $l \neq x$:

2e(0). $f(x) := \mathit{false}$;

2e(i). $C := C \setminus C_{\overline{x}}$;

2e(ii). Lösche x aus allen Klauseln in $C_{\overline{x}} \setminus C_x$;

2e(iii). Lösche alle leeren Klauseln aus C;

3. Liefere f zurück.

Beweis

Beweis:

Per Induktion über Zahl der Variablen zeigen wir:

Hat das *MaxSAT*-Problem c Klauseln, so liefert der Algorithmus in polynomieller Zeit stets eine Lösung, die mindestens $\frac{c}{2}$ der Klauseln erfüllt.

1. Falls $|V| = 1$, folgt die Behauptung trivialerweise.

2. Die Behauptung gelte nun für $|V| = n - 1$.

   Der Algorithmus betrachte als erstes Variable x.
   Seien c_x (bzw. $c_{\overline{x}}$) die Anzahlen der Klauseln, in denen x (bzw. $\overline{x}$) vorkommt. O.B.d.A. sei $c_x \geq c_{\overline{x}}$. Der Alg. setzt $f(x) = \mathit{true}$.
   Nach dieser Zuweisung müssen (nach Schritt 2d(i) bzw. 2d(ii)) noch $\overline{c} \geq (c - c_x - c_{\overline{x}})$ viele Klauseln (mit $n - 1$ Variablen) betrachtet werden.
   Nach Induktionsvoraussetzung werden von f bisher wenigstens $\frac{\overline{c}}{2} \geq \frac{(c - c_x - c_{\overline{x}})}{2}$ der $\overline{c}$ vielen Klauseln erfüllt. Insgesamt werden daher wenigstens $c_x + \frac{(c - c_x - c_{\overline{x}})}{2} = \frac{(c + c_x - c_{\overline{x}})}{2} \geq \frac{c}{2}$ der ursprünglichen Klauselmenge efüllt.

Beziehung zum *"P=NP ?"* Problem

Definition:
Minimum Travelling salesperson (MTS) $(goal_{MK} = MIN)$

Eingabe:
Ein vollständiger gerichteter Graph $G = (V, E), V = \{v_1, \ldots, v_n\}$,
mit positiven Kantengewichten $D : V \times V \to \mathbb{Q}^+ \cup \{0\}$ (d.h. $D \in \mathbb{Q}^{V \times V}$),

Gesucht:
Eine Rundreise $(v_{i_1}, \ldots, v_{i_n}) \in V^n$ mit $\forall 1 \leq j < n : (v_{i_j}, v_{i_{j+1}}) \in E$ und $(i_1, \ldots, i_n)$ ist Permutation von $(1, 2, \ldots, n)$.

Measure:
$$D(v_{i_1}, \ldots, v_{i_n}) := D(v_{i_n}, v_{i_1}) + \sum_{j=1}^{n-1} D(v_{i_j}, v_{i_{j+1}})$$

Ein *MTS*-Problem heißt *metrisch*, wenn G schleifenfrei und ungerichtet ist, sowie die Dreiecksungleichung $D(u,w) \leq D(u,v) + D(v,w)$ gilt

Ein Ergebnis von vielen ähnlichen:

Satz:

Wenn *MTS* zu *APX* gehört, so folgt *P=NP.*

Beweis:
Zu einem beliebigen Hamiltonkreisproblem mit G=(V,E) setze im dazu konstruierten *MTS*-Problem die Gewichte so: $D(u,v)$:=1 falls $(u,v) \in E$ ist und $D(u,v) := 1 + r \cdot |V|$, sonst.
MTS hat minimale Lösung vom Wert $|E|$ genau dann, wenn Hamiltonkreis existiert!
Wenn also für *MTS* die r-Approximation A einen Wert $\leq r \cdot |E|$ liefert, so hatte G einen Hamiltonkreis. Liefert A dagegen einen Wert $> r \cdot |E|$, so gibt es keinen Hamiltonkreis.

Folgerung und metrisches *MTS*

Korollar:

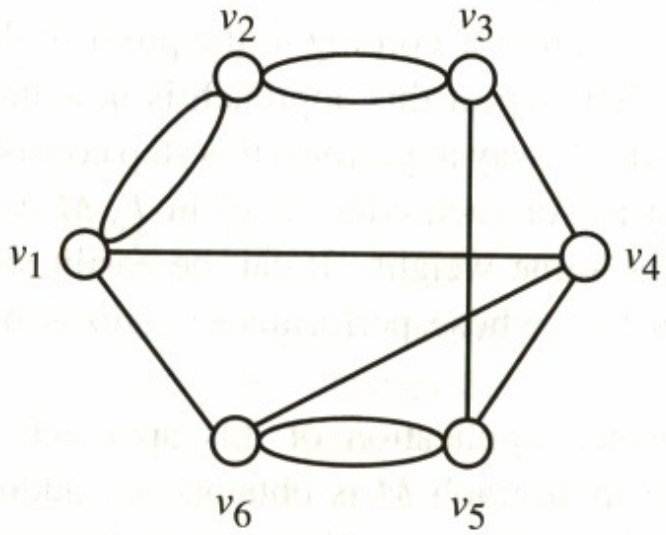
Wenn $P \neq NP$ dann gilt $APX \subset NPO$.

Satz [Christofides]**:**

Für metrische *MTS* gibt es einen Algorithmus, der in Polynomzeit eine Lösung mit Güte $< 3/2$ bestimmt.

Beweisidee: Man ergänzt den gegebenen *MTS*-Graphen durch weitere Kanten zu Multigraph mit gradem Knotengrad (bei Einhaltung der Δ-Ungleichung) und sucht in diesem Multigraphen einen Eulerkreis (was in Polynomzeit gelingt).

Eulersch?



Was gibt es Weiteres?

Sehr Vieles!

Polynomzeit Approximation-Schemata (*PTAS*)

APX versus *PTAS*: $P \neq NP \rightarrow PTAS \subset APX$

Pseudopolynomialität: NPO-Probleme, die durch Polynom in |x| und der größten vorkommenden Zahl beschränkt sind.

Zwischen *APX* und *NPO*

Approximationsklassen und Reduktion

Alles zur Approximation wäre eine 2 SWS-Vorlesung!