

Berechenbarkeits- und Komplexitätstheorie

Lerneinheit 6: Weitere NP-vollständige Probleme

Prof. Dr. Christoph Karg

Studiengang Informatik
Hochschule Aalen



Wintersemester 2015/2016



22.12.2015

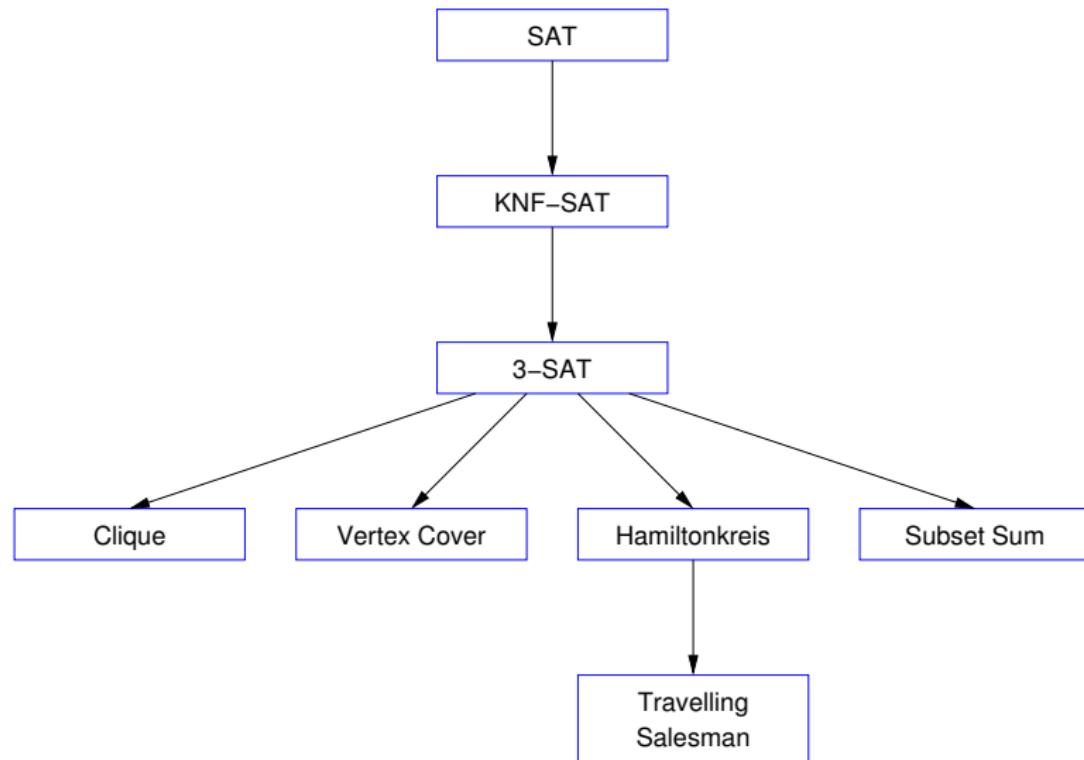
Einleitung

In dieser Lerneinheit werden weitere NP-vollständige Probleme vorgestellt.

Konkret wird nachgewiesen, dass die folgenden Sprachen NP-vollständig sind:

- Clique
- Vertex Cover
- Subset Sum
- Hamiltonkreis
- Travelling Salesman

Übersicht



Clique

Definition. Sei $G = (V, E)$ ein ungerichteter Graph. Sei $C \subseteq V$ eine Teilmenge der Knoten. C ist eine **Clique**, falls für jedes Paar von Knoten $u, v \in C$ gilt, dass $(u, v) \in E$ ist.

Clique

Gegeben:

- Ungerichteter Graph $G = (V, E)$
- Ganze Zahl $k \in \{1, \dots, n\}$

Gefragt: Besitzt G eine Clique C der Größe k ?

Verifikationsalgorithmus für Clique

CHECKCLIQUE($\langle G, k \rangle, C$)

Input: Ungerichteter Graph $G = (V, E)$, Zahl $k \in \{1, \dots, \|V\|\}$, Knotenmenge $C \subseteq V$

Output: true, falls C eine Clique der Größe k ist,
false, sonst.

```
1 if  $\|C\| \neq k$  then
2   return false
3 else
4   for jeden Knoten  $u \in C$  do
5     for jeden Knoten  $v \in C$  do
6       if  $(u, v) \notin E$  then
7         return false
8   return true
```

Clique ist NP-vollständig

Satz. Clique ist NP-vollständig

Beweis. Clique \in NP: ✓

Clique ist NP-hart: Betrachte die Formel:

$$F = (\ell_1^1 \vee \ell_2^1 \vee \ell_3^1) \wedge \dots \wedge (\ell_1^k \vee \ell_2^k \vee \ell_3^k),$$

wobei die ℓ_j^i 's Literale sind

Der Graph $G_F = (V, E)$ wird wie folgt berechnet:

Die Knotenmenge V enthält für jedes Literal einen Knoten:

$$V = \{v_1^1, v_2^1, v_3^1, \dots, v_1^k, v_2^k, v_3^k\}$$

Clique ist NP-vollständig (Forts.)

Die Knoten v_i^r und v_j^s sind durch eine Kante verbunden genau dann, wenn

- $r \neq s$, d.h., die Literale stammen aus verschiedenen Klauseln,
- $\ell_i^r \neq \neg \ell_j^s$, d.h., die Literale sind konsistent.

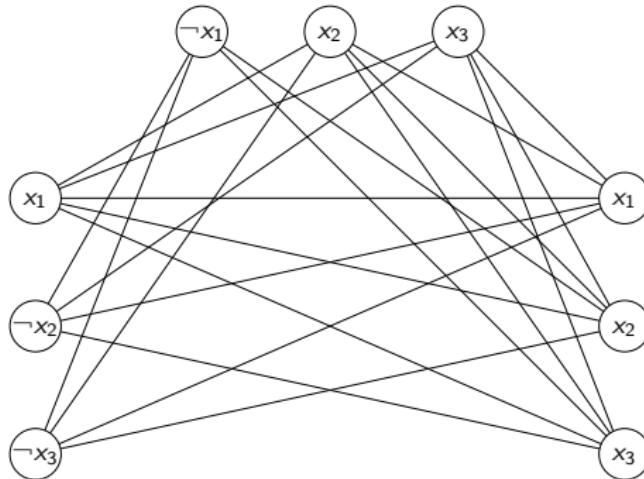
Aufwand zur Konstruktion des Graphen: $O(|F|^2)$

Clique ist NP-vollständig (Forts.)

Beispiel: Gegeben ist die Formel

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

Der zugehörige Graph ist:



Clique ist NP-vollständig (Forts.)

Behauptung: F ist erfüllbar genau dann, wenn G_F eine Clique der Größe k besitzt

“ \Rightarrow ”: Angenommen, F ist erfüllbar.

Dann existiert eine Belegung, unter der in jeder Klausel mindestens ein Literal wahr wird.

Angenommen, die Belegung macht die Literale $\ell_{j_1}^1, \dots, \ell_{j_k}^k$ wahr, wobei $j_i \in \{1, 2, 3\}$ für $i = 1, \dots, k$.

Für alle r, s mit $r \neq s$ gilt: $\ell_{j_r}^r$ und $\ell_{j_s}^s$ sind konsistent.

Also sind $v_{j_r}^r$ und $v_{j_s}^s$ durch eine Kante verbunden.

Folglich ist $C = \{v_{j_1}^1, \dots, v_{j_k}^k\}$ eine Clique der Größe k

Clique ist NP-vollständig (Forts.)

" \Leftarrow ": Angenommen, $v_{j_1}^{i_1}, \dots, v_{j_k}^{i_k}$ bilden eine Clique, wobei $i_p \in \{1, \dots, k\}$ und $j_p \in \{1, 2, 3\}$ für alle $p = 1, \dots, k$.

Wegen der Struktur von G_F können keine zwei Knoten aus derselben Klauselgruppe stammen, d.h.,

$$\{i_1, \dots, i_k\} = \{1, \dots, k\}.$$

Für jedes Paar $v_{j_r}^{i_r}, v_{j_s}^{i_s}$ von Knoten sind die entsprechenden Klauseln $\ell_{j_r}^{i_r}, \ell_{j_s}^{i_s}$ konsistent. Sonst gäbe es keine Kante zwischen $v_{j_r}^{i_r}$ und $v_{j_s}^{i_s}$.

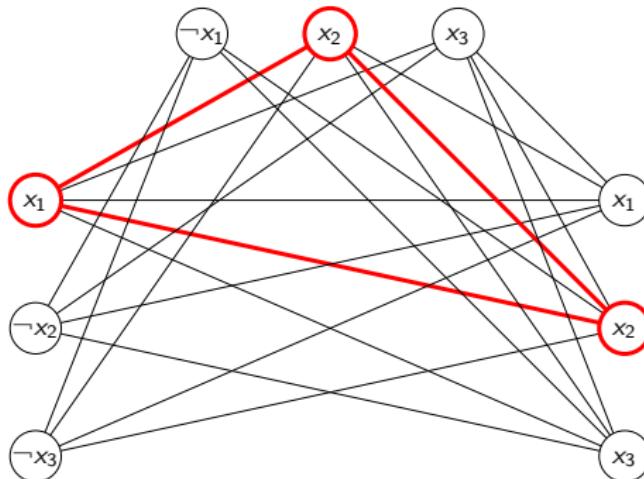
Folglich ist jede Belegung, die die zur Clique korrespondierenden Literale mit 1 belegt, eine erfüllende Belegung für F .

Clique ist NP-vollständig (Forts.)

Beispiel: Gegeben ist die Formel

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

Der zugehörige Graph ist:



$$\rightsquigarrow x_1 = 1, x_2 = 1$$

Vertex Cover

Definition. Sei $G = (V, E)$ ein ungerichteter Graph. Sei $C \subseteq V$ eine Teilmenge der Knoten. Falls für alle Kanten $(u, v) \in E$ $u \in C$ oder $v \in C$, dann nennt man C eine **Knotenabdeckung (Vertex Cover)** von G .

Vertex Cover (VC)

Gegeben:

- Ungerichteter Graph $G = (V, E)$
- Ganze Zahl $k \in \{1, \dots, n\}$

Gefragt: Besitzt G eine Knotenabdeckung C der Größe k ?

Verifikationsalgorithmus für Vertex Cover

CHECKVERTEXCOVER($\langle G, k \rangle, C$)

Input: Ungerichteter Graph $G = (V, E)$, Zahl $k \in \{1, \dots, \|V\|\}$, Knotenmenge $C \subseteq V$

Output: true, falls C eine Knotenabdeckung der Größe k ist, false, sonst.

```
1 if  $\|C\| \neq k$  then
2   return false
3 else
4   for jede Kante  $(u, v) \in E$  do
5     if  $u \notin C$  and  $v \notin C$  then
6       return false
7   return true
```

Vertex Cover ist NP-vollständig

Satz. VC ist NP-vollständig

Beweis. $\text{VC} \in \text{NP}$: ✓

VC ist NP-hart: Reduktion von 3-SAT auf VC

Betrachte die Formel

$$F = (a_1 \vee b_1 \vee c_1) \wedge \dots \wedge (a_\ell \vee b_\ell \vee c_\ell)$$

über den Variablen x_1, \dots, x_m

Vertex Cover ist NP-vollständig (Forts.)

Graph $G_F = (V_F, E_F)$:

Knoten:

- Variablenknoten $V[x_1], \dots, V[x_m], V[\neg x_1], \dots, V[\neg x_m]$
- Klauselknoten $C[a_1], C[b_1], C[c_1], \dots, C[a_\ell], C[b_\ell], C[c_\ell]$

Insgesamt: $2m + 3\ell$ Knoten

Kanten:

- $(V[x_i], V[\neg x_i])$ für alle $i = 1, \dots, m$
- $(C[a_i], C[b_i]), (C[b_i], C[c_i]), (C[c_i], C[a_i])$ für alle $i = 1, \dots, \ell$
- $(C[a_i], V[a_i]), (C[b_i], V[b_i]), (C[c_i], V[c_i])$ für alle $i = 1, \dots, \ell$

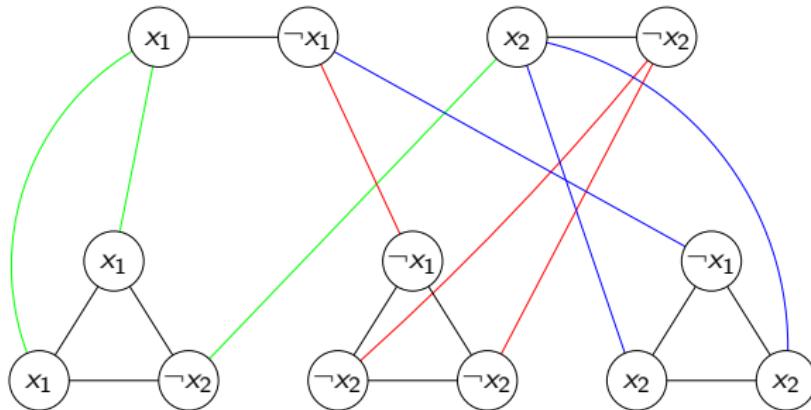
Insgesamt: $m + 3\ell$ Kanten

Vertex Cover ist NP-vollständig (Forts.)

Beispiel: Betrachte die Formel

$$F = (x_1 \vee x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_2) \wedge (\neg x_1 \vee x_2 \vee x_2)$$

Der zugehörige Graph G_F ist:



Vertex Cover ist NP-vollständig (Forts.)

Behauptung: F ist erfüllbar genau dann, wenn G_F eine Knotenabdeckung der Größe $k = m + 2\ell$ besitzt

“ \Rightarrow ”: Angenommen, F ist erfüllbar.

Die Menge C wird wie folgt gebildet:

- Für alle $i = 1, \dots, m$: Falls $x_i = 1$, dann füge x_i zu C hinzu. Ansonsten füge $\neg x_i$ zu C hinzu
- Für $j = 1, \dots, \ell$: Streiche aus der Menge $\{C[a_j], C[b_j], C[c_j]\}$ einen Knoten, dessen Literal wahr ist, und füge die restlichen beiden zu C hinzu

Es gilt: C ist eine Knotenüberdeckung mit $m + 2\ell$ Knoten

Vertex Cover ist NP-vollständig (Forts.)

“ \Leftarrow ”: Sei C eine Knotenüberdeckung mit $m + 2\ell$ Knoten

Eigenschaften:

- C enthält entweder $V[x_i]$ oder $V[\neg x_i]$ für alle $i = 1, \dots, m$
- C enthält genau zwei Knoten der Menge $\{C[a_j], C[b_j], C[c_j]\}$ für alle $j = 1, \dots, \ell$

Betrachte die Belegung

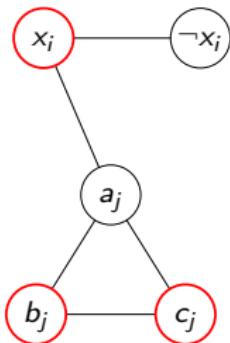
$$x_i = \begin{cases} 1 & \text{falls } V[x_i] \in C, \\ 0 & \text{sonst.} \end{cases}$$

für $i = 1, \dots, m$

Vertex Cover ist NP-vollständig (Forts.)

Betrachte den Teilgraph $\{C[a_j], C[b_j], C[c_j]\}$.

Da nur zwei der Knoten in C enthalten sind, muss eine der "Variablenkanten" durch einen Variablenknoten abgedeckt sein.



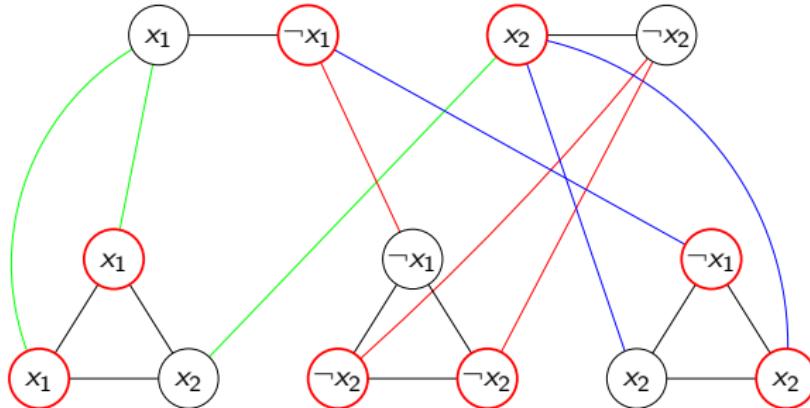
Das entsprechende Literal wurde mit 1 belegt und erfüllt die Klausel $(a_j \vee b_j \vee c_j) \rightsquigarrow F$ ist erfüllbar

Vertex Cover ist NP-vollständig (Forts.)

Zurück zum **Beispiel**:

$$F = (x_1 \vee x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_2) \wedge (\neg x_1 \vee x_2 \vee x_2)$$

Graph G_F :



Erfüllende Belegung $x_1 = 0, x_2 = 1$

Vertex Cover ist NP-vollständig (Forts.)

Die Reduktionsfunktion bildet eine Formel F mit m Variablen und ℓ Klauseln ab in den Graphen $G_F = (V_F, E_F)$ und die Zahl $k = m + 2\ell$

Die Reduktion ist in Zeitkomplexität $O(|F|^2)$ durchführbar

Ergebnis: 3-SAT \leq_m^p VC

Subset Sum

Gegeben:

- Folge von Zahlen a_1, \dots, a_n , wobei $a_i \in \mathbb{Z}$ für $i = 1, \dots, n$
- Zahl $b \in \mathbb{Z}$

Gefragt:

- Gibt es eine Teilmenge $I \subseteq \{1, \dots, n\}$ mit $\sum_{i \in I} a_i = b$?

Verifikationsalgorithmus für Subset Sum

CHECKSUBSETSUM($\langle a_1, \dots, a_n, b \rangle, I$)

Input: Folge von Zahlen a_1, \dots, a_n , Zahl $b \in \mathbb{Z}$,
Menge $I \subseteq \{1, \dots, n\}$

Output: true, falls $\sum_{i \in I} a_i = b$, false, sonst.

- 1 $s := 0$
- 2 **for** jeden Index $i \in I$ **do**
- 3 $s := s + a_i$
- 4 **if** ($s = b$) **then**
- 5 **return** true
- 6 **else**
- 7 **return** false

SubsetSum ist NP-vollständig

Satz. SubsetSum ist NP-vollständig

Beweis. $\text{SubsetSum} \in \text{NP}$: ✓

SubsetSum ist NP-hart: Reduktion von 3-SAT auf SubsetSum

SubsetSum ist NP-vollständig (Forts.)

Beispiel: Gegeben ist die Formel

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

Subset Sum Problem: Alle Zahlen haben $5 + 3 = 8$ Stellen

Zahl	Variablen					Klauseln		
	1	2	3	4	5	1	2	3
y_1	1	0	0	0	0	1	0	0
z_1	1	0	0	0	0	0	1	0
y_2	0	1	0	0	0	0	0	0
z_2	0	1	0	0	0	0	0	1
y_3	0	0	1	0	0	0	0	0
z_3	0	0	1	0	0	1	0	0

SubsetSum ist NP-vollständig (Forts.)

Zahl	Variablen					Klauseln		
	1	2	3	4	5	1	2	3
y_4	0	0	0	1	0	0	1	0
z_4	0	0	0	1	0	0	0	0
y_5	0	0	0	0	1	1	1	0
z_5	0	0	0	0	1	0	0	1
g_1	0	0	0	0	0	1	0	0
h_1	0	0	0	0	0	1	0	0
g_2	0	0	0	0	0	0	1	0
h_2	0	0	0	0	0	0	1	0
g_3	0	0	0	0	0	0	0	1
h_3	0	0	0	0	0	0	0	1
b	1	1	1	1	1	3	3	3

SubsetSum ist NP-vollständig (Forts.)

Sei $F = C_1 \wedge \dots \wedge C_k$ eine Formel in 3-KNF mit den Variablen x_1, \dots, x_n .

Berechnung der Eingabe für Subset Sum:

- Es werden ausschließlich Dezimalzahlen mit $k + n$ Stellen konstruiert
- Notation: $a[j]$ steht für die j -te Stelle von $a = a_1 \dots a_{k+n}$

Aufbau der Zahlen g_i und h_i für $i = 1, \dots, k$:

Für $1 \leq j \leq n$: $g_i[j] = 0$, $h_i[j] = 0$

Für $1 \leq \ell \leq k$:

$$g_i[n + \ell] = \begin{cases} 1 & i = \ell, \\ 0 & \text{sonst.} \end{cases} \quad h_i[n + \ell] = \begin{cases} 1 & i = \ell, \\ 0 & \text{sonst.} \end{cases}$$

SubsetSum ist NP-vollständig (Forts.)

Aufbau der Zahlen y_i und z_i für $i = 1, \dots, n$:

Für $1 \leq j \leq n$:

$$y_i[j] = \begin{cases} 1 & i = j, \\ 0 & \text{sonst.} \end{cases} \quad z_i[j] = \begin{cases} 1 & i = j, \\ 0 & \text{sonst.} \end{cases}$$

Für $1 \leq \ell \leq k$:

$$y_i[n + \ell] = \begin{cases} 1 & x_i \in C_\ell, \\ 0 & \text{sonst.} \end{cases} \quad z_i[n + \ell] = \begin{cases} 1 & \neg x_i \in C_\ell, \\ 0 & \text{sonst.} \end{cases}$$

Aufbau der Zahl b :

$$b = \underbrace{11\dots1}_{n\text{-mal}} \underbrace{33\dots3}_{k\text{-mal}}$$

SubsetSum ist NP-vollständig (Forts.)

Behauptung: F ist erfüllbar genau dann, wenn $y_1, z_1, \dots, y_n, z_n, g_1, h_1, \dots, g_\ell, h_\ell, b$ lösbar ist

“ \Rightarrow ”: Angenommen, F ist erfüllbar. Dann gibt es eine Belegung der Variablen, die in jeder Klausel mindestens ein Literal wahr macht

Auswahl der Zahlen:

- Für alle $1 \leq j \leq n$: Falls $x_j = 1$, dann füge y_j zu T hinzu.
Ansonsten füge z_j zu T hinzu.
- Für alle $1 \leq \ell \leq k$:
 - ▷ Falls die Anzahl der wahren Literale in C_ℓ gleich 1 ist, dann füge g_ℓ und h_ℓ zu T hinzu
 - ▷ Falls die Anzahl der wahren Literale in C_ℓ gleich 2 ist, dann füge g_ℓ zu T hinzu

Es gilt: $\sum_{a \in T} a = b$

SubsetSum ist NP-vollständig (Forts.)

“ \Leftarrow ”: Angenommen, es gibt eine Menge $T \subseteq S$, so dass $\sum_{a \in T} a = b$.

Es gilt:

- Bei der Addition der Zahlen in S gibt es keine Überträge
- Wegen dem Aufbau von b muss entweder y_i oder z_i in T enthalten sein, wobei $i = 1, \dots, n$
- Falls $y_i \in T$, dann setze $x_i = 1$. Ansonsten, setze x_i auf 0
- g_ℓ und h_ℓ reichen nicht aus, um in der Spalte $n + \ell$ den Wert 3 zu erzielen. Daher muss es für jede Klausel-Spalte mindestens eine y - oder z -Zahl in T geben, deren Ziffer in Spalte $n + \ell$ gleich 1 ist. Das entsprechende Literal erfüllt die Klausel C_ℓ

Fazit: F ist erfüllbar

SubsetSum ist NP-vollständig (Forts.)

Beispiel: Gegeben ist die Formel

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

Lösung für Subset Sum Problem:

Zahl	Variablen					Klauseln		
	1	2	3	4	5	1	2	3
y_1	1	0	0	0	0	1	0	0
z_2	0	1	0	0	0	0	0	1
y_3	0	0	1	0	0	0	0	0
z_4	0	0	0	1	0	0	0	0
y_5	0	0	0	0	1	1	1	0

SubsetSum ist NP-vollständig (Forts.)

Zahl	Variablen					Klauseln		
	1	2	3	4	5	1	2	3
g_1	0	0	0	0	0	1	0	0
g_2	0	0	0	0	0	0	1	0
h_2	0	0	0	0	0	0	1	0
g_3	0	0	0	0	0	0	0	1
h_3	0	0	0	0	0	0	0	1
Σ	1	1	1	1	1	3	3	3

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

Gerichteter Hamiltonkreis

Definition. Sei $G = (V, E)$ ein gerichteter Graph. Ein **Hamiltonkreis** in G ist ein Zyklus, auf dem jeder Knoten in V genau einmal vorkommt.

Gerichteter Hamiltonkreis (DHAM)

Gegeben:

- Gerichteter Graph $G = (V, E)$, wobei $\|V\| = n$

Gefragt:

- Besitzt G einen Hamiltonkreis? Formal: Gibt es eine Knotenreihenfolge $v_0, v_1, v_2, \dots, v_n$ mit:
 - ▷ $v_0 = v_n$
 - ▷ Für alle $1 \leq i, j \leq n$ gilt: Falls $i \neq j$, dann ist $v_i \neq v_j$
 - ▷ Für alle $i = 1, \dots, n$ gilt: $(v_{i-1}, v_i) \in E$

Verifikationsalgorithmus für DHAM

CHECKHAMILTONCIRCUIT(G, H)

Input: Gerichteter Graph $G = (V, E)$, wobei $n = \|V\|$

Knotenfolge $H = (v_0, v_1, \dots, v_n)$

Output: true, falls H ein Hamiltonkreis in G ist, false, sonst.

```
1 if  $v_0 \neq v_n$  then
2   return false
3   for  $i := 1$  to  $n - 1$  do
4     for  $j := 2$  to  $n$  do
5       if  $v_i = v_j$  then
6         return false
7   for  $i := 1$  to  $n$  do
8     if  $(v_{i-1}, v_i) \notin E$  then
9       return false
10  return true
```

DHAM ist NP-vollständig

Satz. DHAM ist NP-vollständig

Beweis. $\text{DHAM} \in \text{NP}$: ✓

DHAM ist NP-hart: Reduktion von 3-SAT auf DHAM

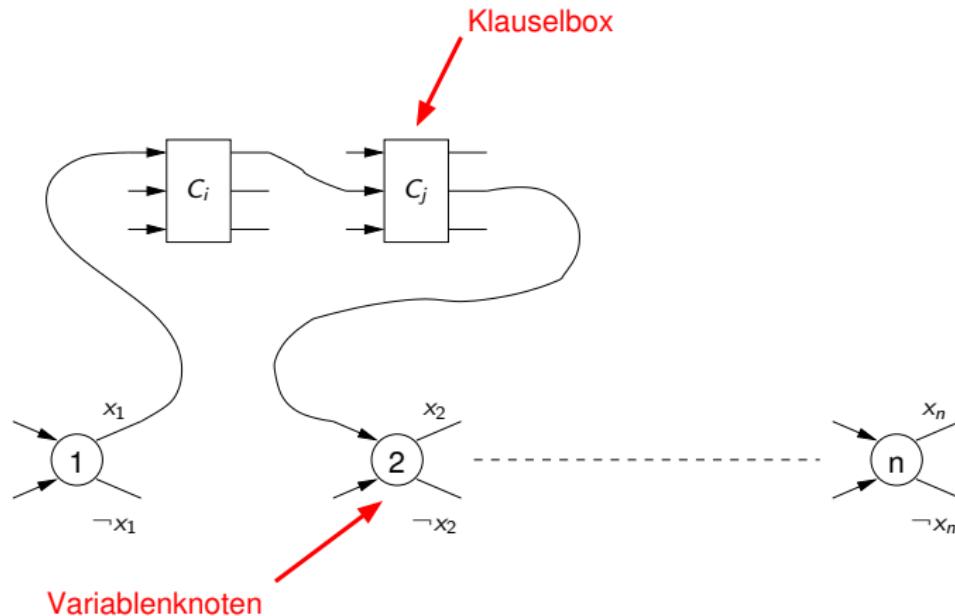
Gegeben ist die Formel

$$F = (\ell_1^1 \vee \ell_2^1 \vee \ell_3^1) \wedge \dots \wedge (\ell_1^k \vee \ell_2^k \vee \ell_3^k),$$

über den Variablen x_1, \dots, x_n

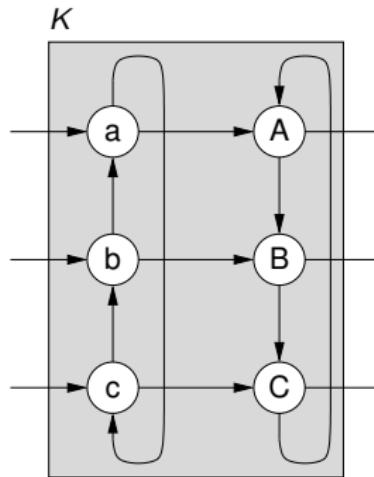
DHAM ist NP-vollständig (Forts.)

Aufbau des Graphen:



DHAM ist NP-vollständig (Forts.)

Aufbau einer Klauselbox:



Pro Klausel enthält G eine Klauselbox

DHAM ist NP-vollständig (Forts.)

Behauptung: Angenommen, der eine Klauselbox K umgebende Graph besitzt einen Hamiltonkreis. Dann durchläuft der Pfad den Teilgraph K folgendermaßen: wenn der Pfad bei a (bzw. b, c) in K hineinführt, dann verlässt er K bei A (bzw. B, C).

Beweis. Angenommen, der Hamiltonkreis beginnt bei a , verlässt K jedoch nicht beim Knoten A .

Fallunterscheidung:

- $a \rightarrow A \rightarrow B$: Sackgasse bei b
- $a \rightarrow A \rightarrow B \rightarrow C$: Sackgasse bei b
- $a \rightarrow c \rightarrow C$: A nicht mehr erreichbar

DHAM ist NP-vollständig (Forts.)

- $a \rightarrow c \rightarrow b \rightarrow B: A$ und C nicht mehr erreichbar
- $a \rightarrow c \rightarrow b \rightarrow B \rightarrow C: A$ nicht mehr erreichbar
- $a \rightarrow c \rightarrow C \rightarrow A \rightarrow B: Sackgasse bei b$

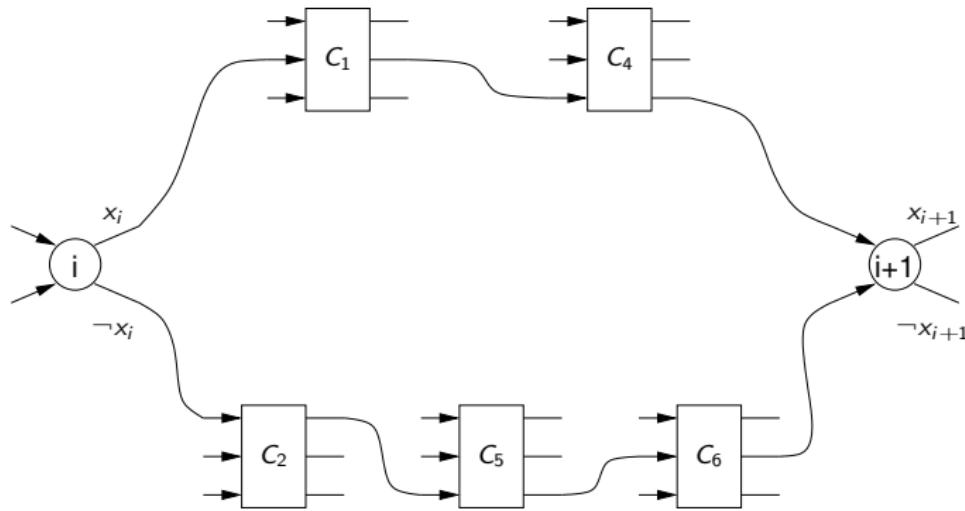
Analog für die Startknoten b und c

Bemerkung: Zulässige Pfade durch K mit Startknoten a :

- $a \rightarrow A$
- $a \rightarrow c \rightarrow C \rightarrow A$
- $a \rightarrow c \rightarrow b \rightarrow B \rightarrow C \rightarrow A$

Analog für die Startknoten b und c

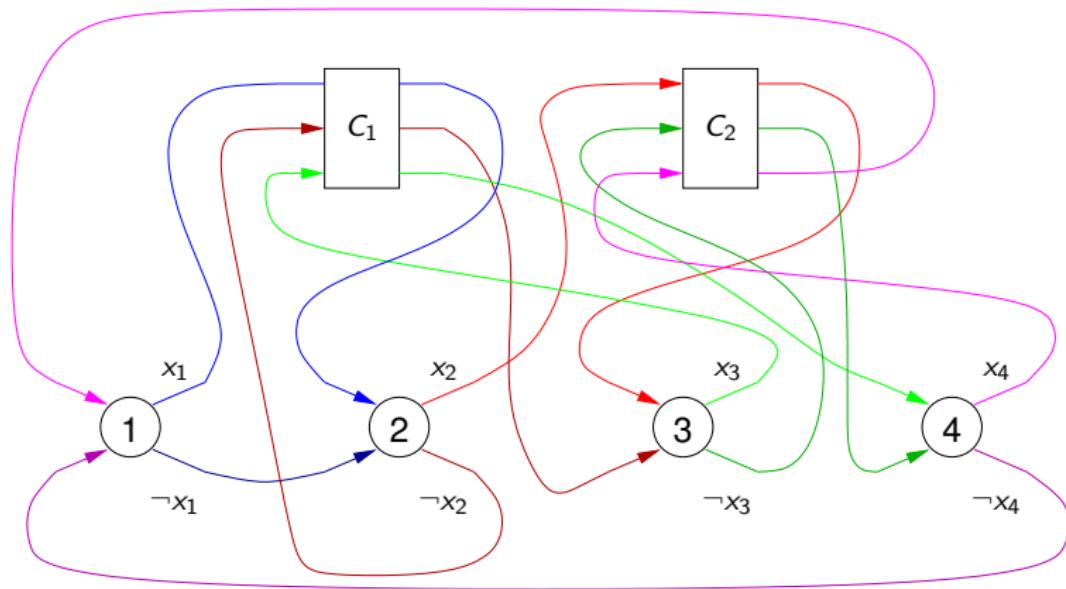
DHAM ist NP-vollständig (Forts.)



Der Ausgangspfad vom Knoten i für x_i durchläuft alle Klauselboxen, die x_i enthalten und endet im “oberen” Eingang von $i + 1$. Analog für den Pfad für $\neg x_i$;

DHAM ist NP-vollständig (Forts.)

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



DHAM ist NP-vollständig (Forts.)

Behauptung: F ist erfüllbar genau dann, wenn G einen Hamiltonkreis besitzt

Beweis. Angenommen, F ist erfüllbar.

Betrachte den Pfad, der anhand einer erfüllenden Belegung konstruiert wird:

Für $i = 1, \dots, n$ führe folgende Schritte durch:

- Wenn $x_i = 1$, dann folge ausgehend vom Knoten i dem “oberen” Pfad. Ansonsten, folge dem “unteren” Pfad
- Durchlaufe alle Klauseln C_j , die x_i (oberer Pfad) bzw. $\neg x_i$ (unterer Pfad) enthalten. Jede Klauselbox wird so durchlaufen, dass die Anzahl der Durchläufe gleich der Anzahl der wahren Literale ist

DHAM ist NP-vollständig (Forts.)

- Nach dem Durchlaufen der Klauselboxen führe den Pfad in den “oberen” (bzw. “unteren”) Eingang des Knotens $(i + 1) \bmod n$

Insgesamt werden in jeder Klauselbox alle Knoten sowie die Knoten $1, \dots, n$ durchlaufen. G besitzt also einen Hamiltonkreis

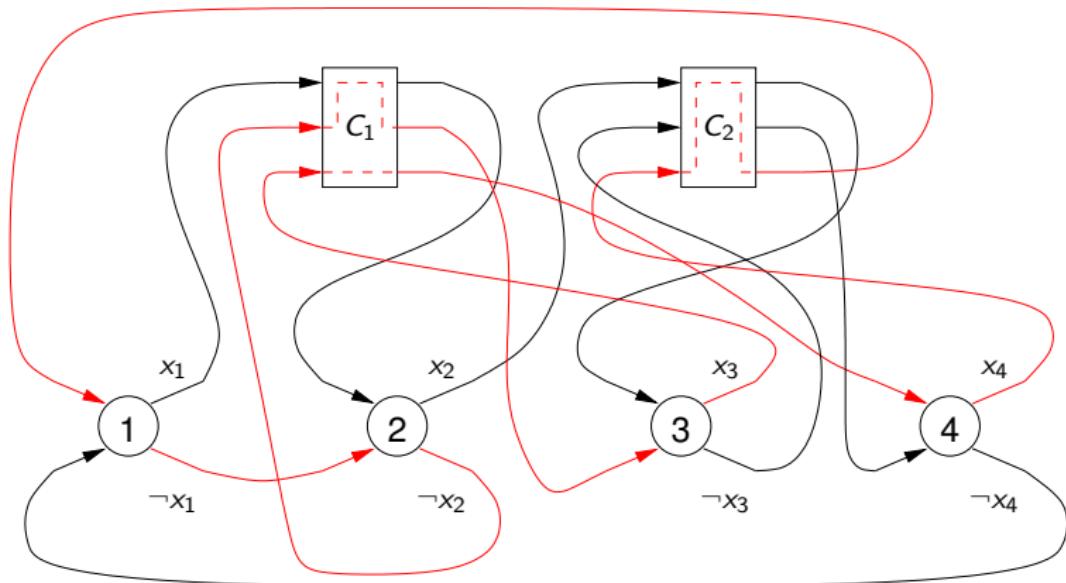
“ \Leftarrow ”: Angenommen, G besitzt einen Hamiltonkreis.

Die Variable x_i wird mit 1 belegt, wenn der Hamiltonkreis den Knoten i auf dem “oberen” Pfad verlässt. Ansonsten erhält x_i den Wert 0

Wegen des Aufbaus des Graphen ist die Belegung erfüllend für die Formel F .

DHAM ist NP-vollständig (Forts.)

Beispiel: Hamiltonkreis:



Erfüllende Belegung: $x_1 = 0, x_2 = 0, x_3 = 1, x_4 = 1$

Ungerichteter Hamiltonkreis

Ungerichteter Hamiltonkreis (UHAM)

Gegeben:

- Ungerichteter Graph $G = (V, E)$

Gefragt:

- Besitzt G einen Hamiltonkreis? Formal: Gibt es eine Knotenreihenfolge $v_0, v_1, v_2, \dots, v_n$ mit:
 - ▷ $v_0 = v_n$
 - ▷ Für alle $1 \leq i, j \leq n$ gilt: Falls $i \neq j$, dann ist $v_i \neq v_j$
 - ▷ Für alle $i = 1, \dots, n$ gilt: $(v_{i-1}, v_i) \in E$

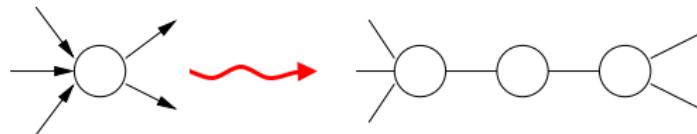
UHAM ist NP-vollständig

Satz. UHAM ist NP-vollständig

Beweis. $\text{UHAM} \in \text{NP}$: analog zu $\text{DHAM} \in \text{NP}$

UHAM ist NP-hart: $\text{DHAM} \leq_m^p \text{UHAM}$

Betrachte einen gerichteten Graphen G . Jeder Knoten in G wird im ungerichteten Graphen G' wie folgt dargestellt: durch:



Es gilt: G besitzt einen Hamiltonkreis genau dann, wenn G' einen Hamiltonkreis besitzt

Travelling Salesman Problem

Travelling Salesman Problem (TSP)

Gegeben:

- $n \times n$ -Matrix $M_{i,j}$ von Entfernungen zwischen n Städten
- Ganze Zahl k

Gefragt: Gibt es eine Rundreise der Länge $\leq k$?

Formal: Gibt es eine Permutation π so dass

$$\sum_{i=1}^{n-1} M_{\pi(i),\pi(i+1)} + M_{\pi(n),\pi(1)} \leq k?$$

TSP ist NP-vollständig

Satz. TSP ist NP-vollständig

Beweis. $\text{TSP} \in \text{NP}$: ✓

TSP ist NP-hart: $\text{UHAM} \leq_m^p \text{TSP}$

Betrachte den ungerichteten Graphen $G = (V, E)$, wobei

$$V = \{1, \dots, n\}$$

Die Matrix $M_{i,j}$ ist wie folgt definiert:

$$M_{i,j} = \begin{cases} 1 & (i,j) \in E \\ 2 & (i,j) \notin E \end{cases}$$

Es gilt: G besitzt einen Hamiltonkreis genau dann, wenn es für $M_{i,j}$ eine Rundreise der Länge $k = n$ gibt

Zusammenfassung

- Die NP-vollständigen Probleme sind die schwierigsten Probleme in NP
- Falls es für eines der NP-vollständigen Probleme einen Polynomialzeit Algorithmus gibt, dann ist $P = NP$
- NP-vollständige Probleme findet man in allen Gebieten der Informatik
- Für weitere Informationen siehe: Garey , Johnson: Computers And Intractability — A Guide to the Theory of NP-Completeness