Karlsruher Institut für Technologie

Theoretische Grundlagen der Informatik

Vorlesung am 25.11.2021

Torsten Ueckerdt (Vertretung: Laura Merker) | 25. November 2021

Letzte Vorlesung: $\mathcal{P}$ vs. $\mathcal{NP}$

Ist $\mathcal{P} = \mathcal{NP}$?

- Die Klasse $\mathcal{P}$ aller Probleme, die von deterministischen Turing-Maschinen in polynomialer Zeit entschieden werden.
- Die Klasse $\mathcal{NP}$ aller Probleme, die von nichtdeterministischen Turing-Maschinen in polynomialer Zeit entschieden werden.
- **Polynomiale Transformation** $\Pi' \propto \Pi$
  - Instanzen von $\Pi' \longrightarrow$ Instanzen von Π
  - in polynomialer Zeit berechenbar
  - Ja-Instanz von $\Pi' \longrightarrow$ Ja-Instanz von Π
    Nein-Instanz von $\Pi' \longrightarrow$ Nein-Instanz von Π
- Problem Π ist $\mathcal{NP}$-vollständig wenn
  - $\Pi \in \mathcal{NP}$ und
  - $\Pi' \propto \Pi$ für alle $\Pi' \in \mathcal{NP}$

Letzte Vorlesung: $\mathcal{P}$ vs. $\mathcal{NP}$

- Die Klasse $\mathcal{P}$ aller Probleme, die von deterministischen Turing-Maschinen in polynomialer Zeit entschieden werden.
- Die Klasse $\mathcal{NP}$ aller Probleme, die von nichtdeterministischen Turing-Maschinen in polynomialer Zeit entschieden werden.
- **Polynomiale Transformation** $\Pi' \propto \Pi$
  - Instanzen von $\Pi' \longrightarrow$ Instanzen von Π
  - in polynomialer Zeit berechenbar
  - Ja-Instanz von $\Pi' \longrightarrow$ Ja-Instanz von Π
    Nein-Instanz von $\Pi' \longrightarrow$ Nein-Instanz von Π
- Problem Π ist $\mathcal{NP}$-vollständig wenn
  - $\Pi \in \mathcal{NP}$ und
  - ~~$\Pi' \propto \Pi$ für alle $\Pi' \in \mathcal{NP}$~~
  - $\Pi' \propto \Pi$ für ein $\mathcal{NP}$-vollständiges Π'

$\boxed{\text{Ist } \mathcal{P} = \mathcal{NP}?}$

Letzte Vorlesung: $\mathcal{P}$ vs. $\mathcal{NP}$

Ist $\mathcal{P} = \mathcal{NP}$?

- Die Klasse $\mathcal{P}$ aller Probleme, die von deterministischen Turing-Maschinen in polynomialer Zeit entschieden werden.
- Die Klasse $\mathcal{NP}$ aller Probleme, die von nichtdeterministischen Turing-Maschinen in polynomialer Zeit entschieden werden.
- **Polynomiale Transformation** $\Pi' \propto \Pi$
  - Instanzen von $\Pi' \longrightarrow$ Instanzen von Π
  - in polynomialer Zeit berechenbar
  - Ja-Instanz von $\Pi' \longrightarrow$ Ja-Instanz von Π
    Nein-Instanz von $\Pi' \longrightarrow$ Nein-Instanz von Π
- Problem Π ist $\mathcal{NP}$-vollständig wenn
  - $\Pi \in \mathcal{NP}$ und
  - ~~$\Pi' \propto \Pi$ für alle $\Pi' \in \mathcal{NP}$~~
  - $\Pi' \propto \Pi$ für ein $\mathcal{NP}$-vollständiges Π'

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

Wiederholung: Das Problem SAT (satisfiability)

Sei $U = \{u_1, \ldots, u_m\}$ eine Menge von booleschen Variablen.
Es heißen $u_i, \overline{u_i}$ Literale.
Eine Wahrheitsbelegung für U ist eine Funktion $t\colon U \to \{\texttt{wahr}, \texttt{falsch}\}$.
Eine Klausel ist ein Boolescher Ausdruck der Form

$$y_1 \vee \ldots \vee y_s \quad \text{mit} \quad y_i \in \{u_1, \ldots, u_m\} \cup \{\overline{u_1}, \ldots, \overline{u_m}\} \text{ Literale}$$

Problem SAT

Gegeben: Menge U von Variablen, Menge C von Klauseln über U.
Frage: Existiert eine Wahrheitsbelegung von U, so dass jede Klausel in C erfüllt wird?

Beispiel:
$U = \{u_1, u_2\}$ mit $C = \{u_1 \vee \overline{u_2},\ \overline{u_1} \vee u_2\}$ ist Ja-Instanz von SAT.
Wahrheitsbelegung $t(u_1) = t(u_2) = \texttt{wahr}$ erfüllt alle Klauseln in C.

Der Satz von Cook (Steven Cook, 1971)

Satz von Cook.

SAT ist $\mathcal{NP}$-vollständig.

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

- SAT $\in \mathcal{NP}$ und
- für alle $\Pi \in \mathcal{NP}$ gilt $\Pi \propto$ SAT.

Der Satz von Cook (Steven Cook, 1971)

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

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

- SAT $\in \mathcal{NP}$ und
- für alle $\Pi \in \mathcal{NP}$ gilt $\Pi \propto$ SAT.

Beweis:

- SAT $\in \mathcal{NP}$ ist erfüllt:
  Für eine Instanz I von SAT (mit n Klauseln und m Variablen) und einer Wahrheitsbelegung t kann in $O(m \cdot n)$ überprüft werden, ob t alle Klauseln erfüllt, d.h. ob I eine Ja–Instanz ist.
- Wir müssen zeigen, dass für jede Sprache $L \in \mathcal{NP}$ gilt: $L \propto L_{\text{SAT}}$, wobei $L_{\text{SAT}} = L[\text{SAT}, s]$ für ein geeignetes Kodierungsschema s ist.

Beweis: das Setup

Wir müssen zeigen, dass für jede Sprache $L \in \mathcal{NP}$ gilt: $L \propto L_{\text{SAT}}$, wobei $L_{\text{SAT}} = L[\text{SAT}, s]$ für ein geeignetes Kodierungsschema s ist.

- Dazu muss für jede Sprache $L \in \mathcal{NP}$ eine polynomiale Transformation f_L angegeben werden, für die gilt, dass für alle $x \in \Sigma^*$ (Σ Alphabet zu L) gilt

$$x \in L \Longleftrightarrow f_L(x) \in L_{\text{SAT}}.$$

Beweis: das Setup

Wir müssen zeigen, dass für jede Sprache $L \in \mathcal{NP}$ gilt: $L \propto L_{\mathrm{SAT}}$, wobei $L_{\mathrm{SAT}} = L[\mathrm{SAT}, s]$ für ein geeignetes Kodierungsschema s ist.

- Dazu muss für jede Sprache $L \in \mathcal{NP}$ eine polynomiale Transformation f_L angegeben werden, für die gilt, dass für alle $x \in \Sigma^*$ (Σ Alphabet zu L) gilt

$$x \in L \Longleftrightarrow f_L(x) \in L_{\mathrm{SAT}}.$$

- Wir benutzen, dass es eine NTM $\mathcal{M}$ zu L gibt, die L in polynomialer Laufzeit entscheidet.
- $\mathcal{M}$ sei gegeben durch $(Q, \Sigma, \sqcup, \Gamma, q_0, \delta, q_J, q_N)$ und akzeptiere die Sprache $L = L_{\mathcal{M}}$ in der Laufzeit $T_{\mathcal{M}}(n) \leq p(n)$, wobei p ein Polynom ist mit (O.B.d.A.) $p(n) \geq n$.

Ziel

Gegeben: $x \in \Sigma^*$
Konstruiere SAT-Instanz, die genau dann erfüllbar ist, wenn $\mathcal{M}$ die Eingabe x akzeptiert

Beweis: das Setup

Ziel

Gegeben: $x \in \Sigma^*$

Konstruiere SAT-Instanz, die genau dann erfüllbar ist, wenn $\mathcal{M}$ die Eingabe x akzeptiert

- Sei $x \in \Sigma^*$ eine Eingabe mit $n := |x|$.
- Bei akzeptierender Berechnung von $\mathcal{M}$ für x ist die Anzahl der Berechnungsschritte $\leq p(n)$
- Dann sind höchstens die Zellen $-p(n)$ bis $p(n) + 1$ des Bandes beteiligt

Beweis: das Setup

Ziel

Gegeben: $x \in \Sigma^*$

Konstruiere SAT-Instanz, die genau dann erfüllbar ist, wenn $\mathcal{M}$ die Eingabe x akzeptiert

- Sei $x \in \Sigma^*$ eine Eingabe mit $n := |x|$.
- Bei akzeptierender Berechnung von $\mathcal{M}$ für x ist die Anzahl der Berechnungsschritte $\leq p(n)$
- Dann sind höchstens die Zellen $-p(n)$ bis $p(n) + 1$ des Bandes beteiligt

Die Berechnung der deterministischen Stufe ist zu jedem Zeitpunkt eindeutig festgelegt durch:

- den jeweiligen Bandinhalt dieser $-p(n)$ bis $p(n) + 1$ Plätze,
- den Zustand der endlichen Kontrolle
- und der Position des Lese-/Schreibkopfs.

Im Folgenden beschreiben wir Bandinhalt, Zustand der endlichen Kontrolle und Position des Lese-/Schreibkopfs vollständig durch Variablen einer Instanz von SAT.

Beweis: Konstruktion der Variablen

- Bezeichne die Zustände aus Q durch $q_0, q_1 = q_J, q_2 = q_N, q_3, \ldots, q_r$
- Bezeichne die Symbole aus Γ durch $s_0 = \sqcup, s_1, \ldots, s_\ell$ mit $|\Gamma| = \ell + 1$.

Es gibt drei Typen von Variablen in der zugehörigen SAT-Instanz.

Variable	Gültigkeitsbereich	Bedeutung
$Q[i,k]$	$0 \le i \le p(n)$ $0 \le k \le r$	zum Zeitpunkt i der Überprüfungsphase ist $\mathcal{M}$ in Zustand q_k
$H[i,j]$	$0 \le i \le p(n)$ $-p(n) \le j \le p(n)+1$	zum Zeitpunkt i der Überprüfungsphase ist der Lese-/Schreibkopf an Position j des Bandes
$S[i,j,k]$	$0 \le i \le p(n)$ $-p(n) \le j \le p(n)+1$ $0 \le k \le \ell$	zum Zeitpunkt i der Überprüfungsphase ist der Bandinhalt an Position j das Symbol s_k

Beweis: Konstruktion der Variablen

- Jede Berechnung von $\mathcal{M}$ induziert eine Wahrheitsbelegung dieser Variablen.
- Konvention: Falls $\mathcal{M}$ vor dem Zeitpunkt $p(n)$ stoppt, bleibt $\mathcal{M}$ in allen folgenden Zuständen in demselben Zustand und der Bandinhalt unverändert.

Variable	Gültigkeitsbereich	Bedeutung
$Q[i,k]$	$0 \leq i \leq p(n)$ $0 \leq k \leq r$	zum Zeitpunkt i der Überprüfungsphase ist $\mathcal{M}$ in Zustand q_k
$H[i,j]$	$0 \leq i \leq p(n)$ $-p(n) \leq j \leq p(n)+1$	zum Zeitpunkt i der Überprüfungsphase ist der Lese-/Schreibkopf an Position j des Bandes
$S[i,j,k]$	$0 \leq i \leq p(n)$ $-p(n) \leq j \leq p(n)+1$ $0 \leq k \leq \ell$	zum Zeitpunkt i der Überprüfungsphase ist der Bandinhalt an Position j das Symbol s_k

Beweis: Konstruktion der Variablen

Der Bandinhalt zum Zeitpunkt 0 der Überprüfungsphase sei (bis auf blanks):

- Eingabe x auf Platz 1 bis n
- Orakelwort w auf Platz -1 bis $-|w|$

Variable	Gültigkeitsbereich	Bedeutung
$Q[i,k]$	$0 \leq i \leq p(n)$ $0 \leq k \leq r$	zum Zeitpunkt i der Überprüfungsphase ist $\mathcal{M}$ in Zustand q_k
$H[i,j]$	$0 \leq i \leq p(n)$ $-p(n) \leq j \leq p(n)+1$	zum Zeitpunkt i der Überprüfungsphase ist der Lese-/Schreibkopf an Position j des Bandes
$S[i,j,k]$	$0 \leq i \leq p(n)$ $-p(n) \leq j \leq p(n)+1$ $0 \leq k \leq \ell$	zum Zeitpunkt i der Überprüfungsphase ist der Bandinhalt an Position j das Symbol s_k

Beweis: Zielsetzung

- Eine beliebige Wahrheitsbelegung muss nicht notwendigerweise eine Berechnung induzieren (zum Beispiel $Q[i, k] = Q[i, \ell]$ für $k \neq \ell$).

Also konstruiere Transformation f_L, die Klauseln einführt, sodass äquivalent ist:

- Für Eingabe x gibt es eine akzeptierende Berechnung, deren Überprüfungsphase höchstens $p(n)$ Zeit benötigt, und deren Orakel höchstens Länge $p(n)$ hat.
- Es gibt eine erfüllende Belegung für die SAT-Instanz $f_L(x)$.

Beweis: Zielsetzung

- Eine beliebige Wahrheitsbelegung muss nicht notwendigerweise eine Berechnung induzieren (zum Beispiel $Q[i,k] = Q[i,\ell]$ für $k \neq \ell$).

Also konstruiere Transformation f_L, die Klauseln einführt, sodass äquivalent ist:

- Für Eingabe x gibt es eine akzeptierende Berechnung, deren Überprüfungsphase höchstens $p(n)$ Zeit benötigt, und deren Orakel höchstens Länge $p(n)$ hat.
- Es gibt eine erfüllende Belegung für die SAT-Instanz $f_L(x)$.

Damit können wir dann schließen:

$x \in L \Leftrightarrow$ es existiert akzeptierende Berechnung von $\mathcal{M}$ bei Eingabe x

$\Leftrightarrow$ es existiert akzeptierende Berechnung von $\mathcal{M}$ bei Eingabe x mit höchstens $p(n)$ Schritten in der Überprüfungsphase und einem Orakel w der Länge $|w| \leq p(n)$

$\Leftrightarrow$ es existiert erfüllende Wahrheitsbelegung für Klauselmenge $f_L(x)$

Beweis: Konstruktion der Klauseln — Übersicht

Klauselgruppe	Einschränkung / Bedeutung
G_1	Zum Zeitpunkt i ist $\mathcal{M}$ in genau einem Zustand.
G_2	Zum Zeitpunkt i hat der Lese-/Schreibkopf genau eine Position.
G_3	Zum Zeitpunkt i enthält jede Bandstelle genau ein Symbol aus Γ.
G_4	Festlegung der Anfangskonfiguration zum Zeitpunkt 0: $\mathcal{M}$ ist im Zustand q_0, der Lese-/Schreibkopf steht an Position 1 des Bandes, in den Zellen 1 bis n steht das Wort $x = s_{k_1} \cdots s_{k_n}$
G_5	Zum Zeitpunkt $p(n)$ hat $\mathcal{M}$ den Zustand q_J erreicht.
G_6	Zu jedem Zeitpunkt i folgt die Konfiguration von $\mathcal{M}$ zum Zeitpunkt $i+1$ aus einer einzigen Anwendung von δ aus der Konfiguration von $\mathcal{M}$ zum Zeitpunkt i.

Klauselgruppe 1:

Zum Zeitpunkt i ist $\mathcal{M}$ in genau einem Zustand.

Erinnerung:
Variablen $Q[i,k]$ bedeuten: zum Zeitpunkt i ist $\mathcal{M}$ in Zustand q_k

Konstruktion:

- Zu jedem Zeitpunkt i ist $\mathcal{M}$ in mindestens einem Zustand

$$Q[i,0] \vee \ldots \vee Q[i,r] \quad \text{für } 0 \leq i \leq p(n)$$

- Zu jedem Zeitpunkt i ist $\mathcal{M}$ in nicht mehr als einem Zustand

$$\overline{Q[i,j]} \vee \overline{Q[i,j']} \quad \text{für } 0 \leq i \leq p(n),\ 0 \leq j < j' \leq r$$

Klauselgruppe 2:

Zum Zeitpunkt i hat der Lese-/Schreibkopf genau eine Position

Erinnerung:
Variablen $H[i, j]$ bedeuten: zum Zeitpunkt i ist der Kopf an Position j des Bandes

Konstruktion:

- Zu jedem Zeitpunkt i hat der Lese-/Schreibkopf mindestens eine Position
$$H[i, -p(n)] \vee \ldots \vee H[i, p(n) + 1] \quad \text{für } 0 \leq i \leq p(n)$$
- Zu jedem Zeitpunkt i hat der Lese-/Schreibkopf höchstens eine Position
$$\overline{H[i, j]} \vee \overline{H[i, j']} \quad \text{für } 0 \leq i \leq p(n),\ -p(n) \leq j < j' \leq p(n) + 1$$

Klauselgruppe 3:

Zum Zeitpunkt *i* enthält jede Bandstelle genau ein Symbol

Erinnerung:
Variablen $S[i, j, k]$ bedeuten: zum Zeitpunkt i steht an Position j des Bandes das Symbol s_k

Konstruktion:

- Zu jedem Zeitpunkt i enthält Bandstelle j mindestens ein Symbol
$$S[i, j, 0] \vee S[i, j, 1] \vee \ldots \vee S[i, j, \ell] \quad \text{für } 0 \leq i \leq p(n),\ -p(n) \leq j \leq p(n) + 1$$
- Zu jedem Zeitpunkt i enthält Bandstelle j höchstens ein Symbol
$$\overline{S[i, j, k]} \vee \overline{S[i, j, k']} \quad \text{für } 0 \leq i \leq p(n),\ -p(n) \leq j \leq p(n) + 1,\ 0 \leq k < k' \leq \ell$$

Klauselgruppe 4:

Festlegung der Anfangskonfiguration zum Zeitpunkt 0

Konstruktion:

- $\mathcal{M}$ ist im Zustand q_0

$$Q[0,0]$$

- der Lese-/Schreibkopf steht an Position 1 des Bandes

$$H[0,1]$$

- in den Zellen 1 bis n steht das Wort $x = s_{k_1} \dots s_{k_n}$

$$\begin{cases} S[0,0,0], S[0,1,k_1], \dots, S[0,n,k_n] & \text{für Eingabe } x = s_{k_1} \cdots s_{k_n} \\ S[0,n+1,0], \dots, S[0,p(n)+1,0] & \text{für rechts der Eingabe} \end{cases}$$

Klauselgruppe 5:

Zum Zeitpunkt $p(n)$ hat $\mathcal{M}$ den Zustand q_J erreicht.

Konstruktion:

$$Q[p(n), 1]$$

Klauselgruppe 6:

Zu jedem Zeitpunkt i folgt die Konfiguration von $\mathcal{M}$ zum Zeitpunkt $i+1$ aus einer einzigen Anwendung von δ aus der Konfiguration von $\mathcal{M}$ zum Zeitpunkt i.

Wir unterteilen Klauselgruppe G_6 in zwei Teilgruppen $G_{6,1}$, $G_{6,2}$.

- $G_{6,1}$:
  Falls $\mathcal{M}$ zum Zeitpunkt i an der Position j das Symbol s_k hat und
  der Lese-/Schreibkopf nicht an der Position j steht,
  dann hat $\mathcal{M}$ auch zum Zeitpunkt $i+1$ an Position j das Symbol s_k.
- $G_{6,2}$:
  Der Wechsel von einer Konfiguration zur nächsten entspricht tatsächlich δ.

Klauselgruppe 6,1:

Falls $\mathcal{M}$ zum Zeitpunkt i an der Position j das Symbol s_k hat und der Lese-/Schreibkopf nicht an der Position j steht, dann hat $\mathcal{M}$ auch zum Zeitpunkt $i+1$ an Position j das Symbol s_k.

Konstruktion:

$$\left(S[i,j,k] \wedge \overline{H[i,j]}\right) \Longrightarrow S[i+1,j,k] \qquad \text{für} \begin{cases} 0 \leq i < p(n) \\ -p(n) \leq j \leq p(n)+1 \\ 0 \leq k \leq \ell \end{cases}$$

Dies ergibt die Klausel

$$\overline{S[i,j,k]} \vee H[i,j] \vee S[i+1,j,k] \qquad \text{für} \begin{cases} 0 \leq i < p(n) \\ -p(n) \leq j \leq p(n)+1 \\ 0 \leq k \leq \ell \end{cases}$$

Klauselgruppe 6,2:

Der Wechsel von einer Konfiguration zur nächsten entspricht tatsächlich δ.

- Sei $\delta(q_a, s_u) = (q_b, s_v, d)$, $d \in \{-1, 0, 1\}$ (steht für L,N,R)

Konstruktion:

$$\begin{aligned} &(H[i,j] \wedge Q[i,a] \wedge S[i,j,u]) \Rightarrow H[i+1, j+d] \\ \text{und } &(H[i,j] \wedge Q[i,a] \wedge S[i,j,u]) \Rightarrow Q[i+1, b] \\ \text{und } &(H[i,j] \wedge Q[i,a] \wedge S[i,j,u]) \Rightarrow S[i+1, j, v] \end{aligned}$$

Dies ergibt folgende Klauseln

$$\begin{aligned} &\overline{H[i,j]} \vee \overline{Q[i,a]} \vee \overline{S[i,j,u]} \vee H[i+1, j+d] \\ &\overline{H[i,j]} \vee \overline{Q[i,a]} \vee \overline{S[i,j,u]} \vee Q[i+1, b] \\ &\overline{H[i,j]} \vee \overline{Q[i,a]} \vee \overline{S[i,j,u]} \vee S[i+1, j, v] \end{aligned}$$

für $0 \leq i < p(n),\ -p(n) \leq j \leq p(n) + 1,\ 0 \leq a, b \leq r,\ 0 \leq u, v \leq \ell$

Zwischenbilanz

- Wir beweisen, dass SAT $\mathcal{NP}$-vollständig ist.
- Für jede beliebige aber feste Sprache $L \in \mathcal{NP}$ konstruieren wir eine polynomiale Reduktion $L \propto L_{\text{SAT}}$, d.h.
  - wir konstruieren eine Abbildung $f_L : \Sigma^* \longrightarrow D_{\text{SAT}}$
  - so dass für alle $x \in \Sigma^*$ gilt: $x \in L \Leftrightarrow f_L(x) \in J_{\text{SAT}}$
  - und f_L polynomial berechenbar ist.

Zwischenbilanz

Karlsruher Institut für Technologie

- Wir beweisen, dass SAT $\mathcal{NP}$-vollständig ist.
- Für jede beliebige aber feste Sprache $L \in \mathcal{NP}$ konstruieren wir eine polynomiale Reduktion $L \propto L_{\text{SAT}}$, d.h.
  - wir konstruieren eine Abbildung $f_L : \Sigma^* \longrightarrow D_{\text{SAT}}$
  - so dass für alle $x \in \Sigma^*$ gilt: $\quad x \in L \Leftrightarrow f_L(x) \in J_{\text{SAT}}$
  - und f_L polynomial berechenbar ist.
- Wir betrachten eine NTM $\mathcal{M}$ die L entscheidet.
- Wir betrachten Polynom $p(n)$ das die Laufzeit von $\mathcal{M}$ beschränkt.
- Für beliebiges $x \in \Sigma^*$ konstruieren wir $f_L(x)$ mit Variablen $Q[i,j]$, $H[i,j]$, $S[i,j,k]$ und Klauselmenge $C := G_1 \cup G_2 \cup \cdots \cup G_6$.

Zwischenbilanz

- Wir beweisen, dass SAT $\mathcal{NP}$-vollständig ist.
- Für jede beliebige aber feste Sprache $L \in \mathcal{NP}$ konstruieren wir eine polynomiale Reduktion $L \propto L_{\text{SAT}}$, d.h.
  - wir konstruieren eine Abbildung $f_L : \Sigma^* \longrightarrow D_{\text{SAT}}$
  - so dass für alle $x \in \Sigma^*$ gilt: $\quad x \in L \Leftrightarrow f_L(x) \in J_{\text{SAT}}$
  - und f_L polynomial berechenbar ist.
- Wir betrachten eine NTM $\mathcal{M}$ die L entscheidet.
- Wir betrachten Polynom $p(n)$ das die Laufzeit von $\mathcal{M}$ beschränkt.
- Für beliebiges $x \in \Sigma^*$ konstruieren wir $f_L(x)$ mit Variablen $Q[i,j]$, $H[i,j]$, $S[i,j,k]$ und Klauselmenge $C := G_1 \cup G_2 \cup \cdots \cup G_6$.

$x \in L \quad \Longrightarrow \quad$ $\exists$ akzeptierende Berechnung von $\mathcal{M}$ für Eingabe x $\quad \Longrightarrow \quad$ $\exists$ Wahrheitsbelegung die alle Klauseln in C erfüllt.

Zwischenbilanz

- Wir beweisen, dass SAT $\mathcal{NP}$-vollständig ist.
- Für jede beliebige aber feste Sprache $L \in \mathcal{NP}$ konstruieren wir eine polynomiale Reduktion $L \propto L_{\text{SAT}}$, d.h.
  - wir konstruieren eine Abbildung $f_L : \Sigma^* \longrightarrow D_{\text{SAT}}$
  - so dass für alle $x \in \Sigma^*$ gilt: $\quad x \in L \Leftrightarrow f_L(x) \in J_{\text{SAT}}$
  - und f_L polynomial berechenbar ist.
- Wir betrachten eine NTM $\mathcal{M}$ die L entscheidet.
- Wir betrachten Polynom $p(n)$ das die Laufzeit von $\mathcal{M}$ beschränkt.
- Für beliebiges $x \in \Sigma^*$ konstruieren wir $f_L(x)$ mit Variablen $Q[i,j]$, $H[i,j]$, $S[i,j,k]$ und Klauselmenge $C := G_1 \cup G_2 \cup \cdots \cup G_6$.

$x \in L \iff$ ∃ akzeptierende Berechnung von $\mathcal{M}$ für Eingabe x $\iff$ ∃ Wahrheitsbelegung die alle Klauseln in C erfüllt.

Polynomialität der Transformation

Wir schätzen die Anzahl der Literale in den Klauselgruppen ab.

Polynomialität — Klauselgruppe 1:

Zum Zeitpunkt i ist $\mathcal{M}$ in genau einem Zustand.

Konstruktion:

- Zu jedem Zeitpunkt i ist $\mathcal{M}$ in mindestens einem Zustand
$$Q[i,0] \vee \ldots \vee Q[i,r] \quad \text{für } 0 \leq i \leq p(n)$$
- Zu jedem Zeitpunkt i ist $\mathcal{M}$ in nicht mehr als einem Zustand
$$\overline{Q[i,j]} \vee \overline{Q[i,j']} \quad \text{für } 0 \leq i \leq p(n),\ 0 \leq j < j' \leq r$$

Abschätzung:

$$(r+1) \cdot (p(n)+1) + 2 \cdot (p(n)+1)\frac{1}{2}r(r+1)$$

Polynomialität — Klauselgruppe 2:

Zum Zeitpunkt *i* hat der Lese-/Schreibkopf genau eine Position

Konstruktion:

- Zum Zeitpunkt i hat der Lese-/Schreibkopf mindestens eine Position
$$H[i,-p(n)] \vee \ldots \vee H[i,p(n)+1] \quad \text{für } 0 \leq i \leq p(n)$$
- Zum Zeitpunkt i hat der Lese-/Schreibkopf höchstens eine Position
$$\overline{H[i,j]} \vee \overline{H[i,j']} \quad \text{für } 0 \leq i \leq p(n),\ -p(n) \leq j < j' \leq p(n)+1$$

Abschätzung:

$$(2p(n)+2)\cdot(p(n)+1)+2\cdot(p(n)+1)\frac{1}{2}(2p(n)+1)(2p(n)+2)$$

Polynomialität — Klauselgruppe 3:

Zum Zeitpunkt *i* enthält jede Bandstelle genau ein Symbol

Konstruktion:

- Zum Zeitpunkt *i* enthält jede Bandstelle mindestens ein Symbol

$$S[i,j,0] \vee S[i,j,1] \vee \ldots \vee S[i,j,\ell] \quad \text{für } 0 \leq i \leq p(n),\ -p(n) \leq j \leq p(n)+1$$

- Zum Zeitpunkt *i* enthält jede Bandstelle höchstens ein Symbol

$$\overline{S[i,j,k]} \vee \overline{S[i,j,k']} \quad \text{für } 0 \leq i \leq p(n),\ -p(n) \leq j \leq p(n)+1,\ 0 \leq k < k' \leq \ell$$

Abschätzung:

$$(\ell+1) \cdot (p(n)+1)(2p(n)+2) + 2 \cdot (p(n)+1)(2p(n)+2)\frac{1}{2}\ell(\ell+1)$$

Polynomialität — Klauselgruppe 4:

Festlegung der Anfangskonfiguration zum Zeitpunkt 0

- $\mathcal{M}$ ist im Zustand q_0
$$Q[0,0]$$
- der Lese-/Schreibkopf steht an Position 1 des Bandes
$$H[0,1]$$
- in den Zellen 1 bis n steht das Wort $x = s_{k_1} \dots s_{k_n}$
$$\begin{cases} S[0,0,0], S[0,1,k_1], \dots, S[0,n,k_n] & \text{für Eingabe } x = s_{k_1} \cdots s_{k_n} \\ S[0,n+1,0], \dots, S[0,p(n)+1,0] & \text{für rechts der Eingabe} \end{cases}$$

Abschätzung:

$$2 + (n+1) + (p(n) + 1 - n)$$

Polynomialität — Klauselgruppe 5:

Bis zum Zeitpunkt $p(n)$ hat $\mathcal{M}$ den Zustand q_J erreicht.

Konstruktion:

$$Q[p(n), 1]$$

Abschätzung:

$$1$$

Polynomialität — Klauselgruppe 6,1:

Falls $\mathcal{M}$ zum Zeitpunkt i an der Position j das Symbol s_k hat und der Lese-/Schreibkopf nicht an der Position j steht, dann hat $\mathcal{M}$ auch zum Zeitpunkt $i+1$ an Position j das Symbol s_k.

Konstruktion:

$$\left(S[i,j,k] \wedge \overline{H[i,j]}\right) \Longrightarrow S[i+1,j,k] \qquad \text{für} \begin{cases} 0 \leq i < p(n) \\ -p(n) \leq j \leq p(n)+1 \\ 0 \leq k \leq \ell \end{cases}$$

Dies ergibt die Klausel

$$\overline{S[i,j,k]} \vee H[i,j] \vee S[i+1,j,k] \qquad \text{für} \begin{cases} 0 \leq i < p(n) \\ -p(n) \leq j \leq p(n)+1 \\ 0 \leq k \leq \ell \end{cases}$$

Abschätzung:

$$3 \cdot p(n)(2p(n)+2)(\ell+1)$$

Polynomialität — Klauselgruppe 6,2:

Der Wechsel von einer Konfiguration zur nächsten entspricht tatsächlich δ.

- Sei $\delta(q_a, s_u) = (q_b, s_v, d)$, $d \in \{-1, 0, 1\}$ (steht für L,N,R)

$$\overline{H[i,j]} \vee \overline{Q[i,a]} \vee \overline{S[i,j,u]} \vee H[i+1, j+d]$$
$$\overline{H[i,j]} \vee \overline{Q[i,a]} \vee \overline{S[i,j,u]} \vee Q[i+1, b]$$
$$\overline{H[i,j]} \vee \overline{Q[i,a]} \vee \overline{S[i,j,u]} \vee S[i+1, j, v]$$

für $0 \leq i < p(n),\ -p(n) \leq j \leq p(n)+1,\ 0 \leq a, b \leq r,\ 0 \leq u, v \leq \ell$

Abschätzung:

$$4 \cdot 3 \cdot p(n)(2p(n)+2)(r+1)(\ell+1)$$

Polynomialität der Transformation

Wir schätzen die Anzahl der Literale in den Klauselgruppen ab.

- $G_1:\ (p(n)+1)(r+1)^2$
- $G_2:\ 4(p(n)+1)^3$
- $G_3:\ 2(p(n)+1)^2(\ell+1)^2$
- $G_4:\ p(n)+4$
- $G_5:\ 1$
- $G_6:\ \underbrace{p(n)(\ell+1)(2p(n)+2)\cdot 3}_{G_{6,1}} + \underbrace{p(n)(p(n)+2)(r+1)(\ell+1)\cdot 3\cdot 4}_{G_{6,2}}$

Polynomialität der Transformation

Wir schätzen die Anzahl der Literale in den Klauselgruppen ab.

- $G_1: \; (p(n)+1)(r+1)^2$
- $G_2: \; 4(p(n)+1)^3$
- $G_3: \; 2(p(n)+1)^2(\ell+1)^2$
- $G_4: \; p(n)+4$
- $G_5: \; 1$
- $G_6: \; \underbrace{p(n)(\ell+1)(2p(n)+2)\cdot 3}_{G_{6,1}} + \underbrace{p(n)(p(n)+2)(r+1)(\ell+1)\cdot 3\cdot 4}_{G_{6,2}}$

- r und ℓ sind Konstanten, die durch $\mathcal{M}$ (und damit durch L) induziert werden
- $p(n)$ ist ein Polynom in n

Polynomialität der Transformation

Wir schätzen die Anzahl der Literale in den Klauselgruppen ab.

- G_1: $(p(n)+1)(r+1)^2$
- G_2: $4(p(n)+1)^3$
- G_3: $2(p(n)+1)^2(\ell+1)^2$
- G_4: $p(n)+4$
- G_5: 1
- G_6: $\underbrace{p(n)(\ell+1)(2p(n)+2)\cdot 3}_{G_{6,1}} + \underbrace{p(n)(p(n)+2)(r+1)(\ell+1)\cdot 3\cdot 4}_{G_{6,2}}$

- Also sind alle Größen polynomial in n.
- Die angegebene Funktion f_L ist damit eine polynomiale Transformation von L nach L_{SAT}.

Der Satz von Cook im Rückblick:

Satz von Cook.

SAT ist $\mathcal{NP}$-vollständig.

Beweisidee:

- Zu gegebener Sprache $L \in \mathcal{NP}$ und Eingabe $x \in \Sigma^*$ konstruiere eine SAT-Instanz $f_L(x) \in D_{\text{SAT}}$.
- Variablen der SAT-Instanz kodieren mögliche Zustände der NTM zu verschiedenen Zeitpunkten.
- Klauseln der SAT-Instanz garantieren
  - sinnvolle Zustandsübergänge, so wie von der NTM definiert
  - Erfüllbarkeit genau dann, wenn die NTM akzeptiert
- Dazu brauchen wir nur polynomial viele Variablen und Klauseln.

Der Satz von Cook im Rückblick:

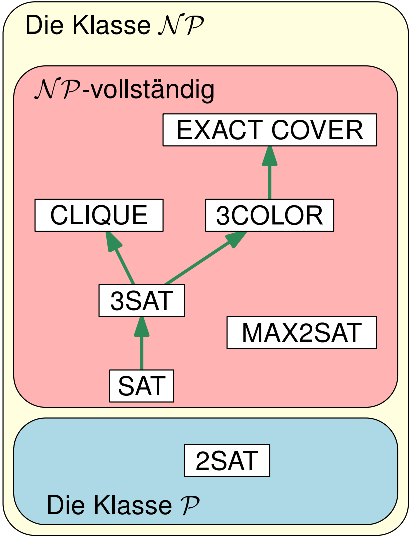
Satz von Cook.
SAT ist $\mathcal{NP}$-vollständig.

Damit haben wir gezeigt:

- SAT gehört zu den schwersten Problemen in $\mathcal{NP}$.
- Könnte man SAT in polynomialer Zeit lösen, so könnte man alle Probleme in $\mathcal{NP}$ in polynomialer Zeit lösen.
- Lässt sich SAT in polynomialer Zeit auf ein Problem Π transformieren, so muss Π $\mathcal{NP}$-vollständig sein.
  - Das haben wir schon in der letzten Vorlesung gemacht: SAT $\propto$ 3SAT

Der Plan

- 3SAT ist $\mathcal{NP}$-vollständig
  $\rightsquigarrow$ 3SAT $\in \mathcal{NP}$
  $\rightsquigarrow$ SAT $\propto$ 3SAT
- 2SAT ist in $\mathcal{P}$
- MAX2SAT ist $\mathcal{NP}$-vollständig
  $\rightsquigarrow$ Übung
- CLIQUE ist $\mathcal{NP}$-vollständig
  $\rightsquigarrow$ CLIQUE $\in \mathcal{NP}$
  $\rightsquigarrow$ 3SAT $\propto$ CLIQUE



- 3COLOR ist $\mathcal{NP}$-vollständig
  $\rightsquigarrow$ 3COLOR $\in \mathcal{NP}$
  $\rightsquigarrow$ 3SAT $\propto$ 3COLOR
- EXACT COVER ist $\mathcal{NP}$-vollständig
  $\rightsquigarrow$ EXACT COVER $\in \mathcal{NP}$
  $\rightsquigarrow$ 3COLOR $\propto$ EXACT COVER

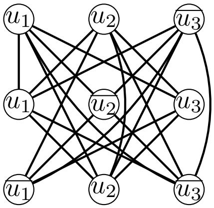
Das Problem CLIQUE

Eine **Clique** in einem Graphen $G = (V, E)$ ist eine Menge $V' \subseteq V$ so dass für alle $i, j \in V', i \neq j$, gilt: $\{i, j\} \in E$.

Problem CLIQUE

Gegeben: Graph $G = (V, E)$ und ein Parameter $K \leq |V|$

Frage: Gibt es in G eine Clique der Größe mindestens K?



Beweis: $\mathcal{NP}$-Vollständigkeit von CLIQUE

Satz.

Das Problem CLIQUE ist $\mathcal{NP}$-vollständig.

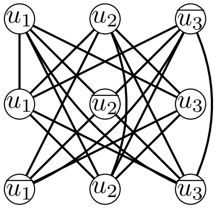
CLIQUE $\in \mathcal{NP}$:

- Für eine gegebene Menge $V' \subseteq V$ kann in polynomieller Zeit überprüft werden, ob
  - für alle $i, j \in V'$, $i \neq j$ gilt: $\{i, j\} \in E$
  - $|V'| \geq K$

Beweis: $\mathcal{NP}$-Vollständigkeit von CLIQUE

3SAT $\propto$ CLIQUE

- Sei $C = \{c_1, \ldots, c_n\}$ eine 3SAT-Instanz mit $c_i = x_{i1} \vee x_{i2} \vee x_{i3}$ und $x_{ij} \in \{u_1, \ldots, u_m, \overline{u_1}, \ldots, \overline{u_m}\}$.

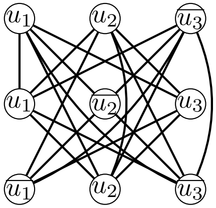


Wir transformieren C in eine CLIQUE-Instanz $(G = (V, E), K)$.

- V enthält $3n$ Knoten v_{ij} für $1 \leq i \leq n$, $1 \leq j \leq 3$.
- v_{ij} und $v_{k\ell}$ sind durch Kanten aus E verbunden genau dann, wenn:
  - $i \neq k$ (Literale sind in verschiedenen Klauseln) und
  - $x_{ij} \neq \overline{x_{k\ell}}$ (Literale sind gleichzeitig erfüllbar)
- Wir setzen $K := n$

Beweis: $\mathcal{NP}$-Vollständigkeit von CLIQUE

- V enthält $3n$ Knoten v_{ij} für $1 \leq i \leq n$, $1 \leq j \leq 3$.
- v_{ij} und $v_{k\ell}$ sind durch Kanten aus E verbunden genau dann, wenn:
  - $i \neq k$ (Literale sind in verschiedenen Klauseln) und
  - $x_{ij} \neq \overline{x_{k\ell}}$ (Literale sind gleichzeitig erfüllbar)



Beispiel: Sei $C = \{u_1 \vee u_2 \vee \overline{u_3},\ u_1 \vee \overline{u_2} \vee u_3,\ \overline{u_1} \vee u_2 \vee \overline{u_3}\}$.

Knotennummer	v_{11}	v_{12}	v_{13}	v_{21}	v_{22}	v_{23}	v_{31}	v_{32}	v_{33}
Literal	u_1	u_2	$\overline{u_3}$	u_1	$\overline{u_2}$	u_3	$\overline{u_1}$	u_2	$\overline{u_3}$.

Beweis: $\mathcal{NP}$-Vollständigkeit von CLIQUE

- Die Transformation kann in polynomieller Zeit berechnet werden.

Noch zu zeigen:

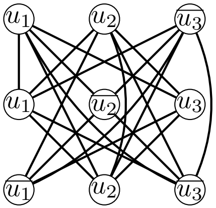
- 3SAT-Instanz C ist Ja-Instanz $\Leftrightarrow$ CLIQUE-Instanz (G, K) ist Ja-Instanz
  - C ist Ja-Instanz:
    Es existiert Wahrheitsbelegung $t\colon U \longrightarrow \{\texttt{wahr}, \texttt{falsch}\}$, so dass alle Klauseln in C unter t erfüllt sind.
  - (G, K) ist Ja-Instanz:
    Es existiert Knotenmenge $V' \subseteq V$, so dass $|V'| \geq K$ und alle Knoten in V' paarweise durch Kanten in G verbunden sind.

Beweis: $\mathcal{NP}$-Vollständigkeit von CLIQUE

3SAT-Instanz C ist Ja-Instanz $\Rightarrow$ CLIQUE-Instanz (G, K) ist Ja-Instanz:

- Wähle eine beliebige erfüllende Wahrheitsbelegung t von C.
- Wähle in jeder Klausel ein wahres Literal.
- Diese Knoten in G bilden eine Clique V' der Größe $K = n$.

Beispiel: Sei $C = \{u_1 \vee u_2 \vee \overline{u_3},\ u_1 \vee \overline{u_2} \vee u_3,\ \overline{u_1} \vee u_2 \vee \overline{u_3}\}$.

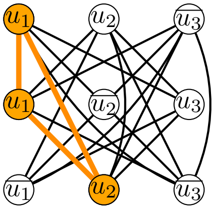


Beweis: $\mathcal{NP}$-Vollständigkeit von CLIQUE

3SAT-Instanz C ist Ja-Instanz $\Rightarrow$ CLIQUE-Instanz (G, K) ist Ja-Instanz:

- Wähle eine beliebige erfüllende Wahrheitsbelegung t von C.
- Wähle in jeder Klausel ein wahres Literal.
- Diese Knoten in G bilden eine Clique V' der Größe $K = n$.

Beispiel: Sei $C = \{u_1 \vee u_2 \vee \overline{u_3},\ u_1 \vee \overline{u_2} \vee u_3,\ \overline{u_1} \vee u_2 \vee \overline{u_3}\}$.



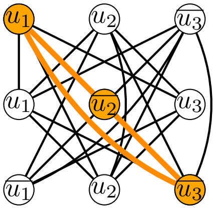
$$t(u_1) = \texttt{wahr}$$
$$t(u_2) = \texttt{wahr}$$
$$t(u_3) = \texttt{falsch}$$

Beweis: $\mathcal{NP}$-Vollständigkeit von CLIQUE

3SAT-Instanz C ist Ja-Instanz $\Rightarrow$ CLIQUE-Instanz (G, K) ist Ja-Instanz:

- Wähle eine beliebige erfüllende Wahrheitsbelegung t von C.
- Wähle in jeder Klausel ein wahres Literal.
- Diese Knoten in G bilden eine Clique V' der Größe $K = n$.

Beispiel: Sei $C = \{u_1 \vee u_2 \vee \overline{u_3},\ u_1 \vee \overline{u_2} \vee u_3,\ \overline{u_1} \vee u_2 \vee \overline{u_3}\}$.



$t(u_1) = \texttt{wahr}$

$t(u_2) = \texttt{falsch}$

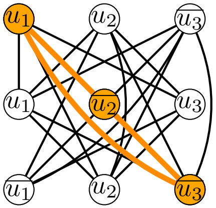
$t(u_3) = \texttt{falsch}$

Beweis: $\mathcal{NP}$-Vollständigkeit von CLIQUE

3SAT-Instanz C ist Ja-Instanz $\Leftarrow$ CLIQUE-Instanz (G, K) ist Ja-Instanz:

- Wähle eine Clique V' der Größe $K = n$ in G.
- Dies ist ein Knoten pro Klausel. Wir setzen dieses Literal in t auf `wahr`.
- Dann ist $t(u) \neq t(\overline{u})$ und alle Klauseln in C sind erfüllt.

Beispiel: Sei $C = \{u_1 \vee u_2 \vee \overline{u_3},\ u_1 \vee \overline{u_2} \vee u_3,\ \overline{u_1} \vee u_2 \vee \overline{u_3}\}$.



$t(u_1) = \texttt{wahr}$

$t(u_2) = \texttt{falsch}$

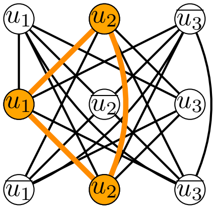
$t(u_3) = \texttt{falsch}$

Beweis: $\mathcal{NP}$-Vollständigkeit von CLIQUE

3SAT-Instanz C ist Ja-Instanz $\Leftarrow$ CLIQUE-Instanz (G, K) ist Ja-Instanz:

- Wähle eine Clique V' der Größe $K = n$ in G.
- Dies ist ein Knoten pro Klausel. Wir setzen dieses Literal in t auf `wahr`.
- Dann ist $t(u) \neq t(\overline{u})$ und alle Klauseln in C sind erfüllt.

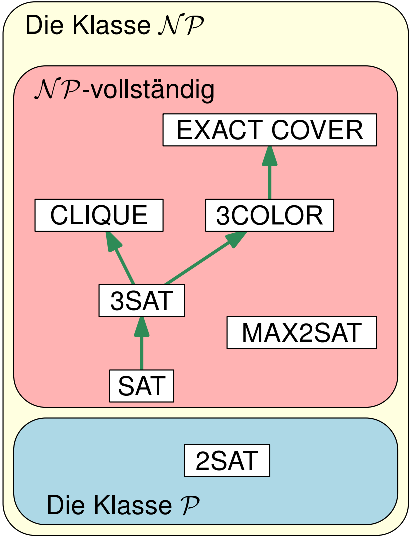
Beispiel: Sei $C = \{u_1 \vee u_2 \vee \overline{u_3},\ u_1 \vee \overline{u_2} \vee u_3,\ \overline{u_1} \vee u_2 \vee \overline{u_3}\}$.



$t(u_1) =$ `wahr`
$t(u_2) =$ `wahr`
$t(u_3) =$ beliebig

Der Plan

- 3SAT ist $\mathcal{NP}$-vollständig
  $\rightsquigarrow$ 3SAT $\in \mathcal{NP}$
  $\rightsquigarrow$ SAT $\propto$ 3SAT
- 2SAT ist in $\mathcal{P}$
- MAX2SAT ist $\mathcal{NP}$-vollständig
  $\rightsquigarrow$ Übung
- CLIQUE ist $\mathcal{NP}$-vollständig
  $\rightsquigarrow$ CLIQUE $\in \mathcal{NP}$
  $\rightsquigarrow$ 3SAT $\propto$ CLIQUE



- 3COLOR ist $\mathcal{NP}$-vollständig
  $\rightsquigarrow$ 3COLOR $\in \mathcal{NP}$
  $\rightsquigarrow$ 3SAT $\propto$ 3COLOR
- EXACT COVER ist $\mathcal{NP}$-vollständig
  $\rightsquigarrow$ EXACT COVER $\in \mathcal{NP}$
  $\rightsquigarrow$ 3COLOR $\propto$ EXACT COVER

Das Problem COLOR

Karlsruher Institut für Technologie

Problem COLOR

Gegeben: Graph $G = (V, E)$ und ein Parameter $K \in \mathbb{N}$.

Frage: Gibt es eine Knotenfärbung von G mit höchstens K Farben, so dass je zwei adjazente Knoten verschiedene Farben besitzen?

3COLOR bezeichnet das Problem COLOR mit festem Parameter $K = 3$.

Satz.

Das Problem 3COLOR ist $\mathcal{NP}$-vollständig.

Beweis: $\mathcal{NP}$-Vollständigkeit von 3COLOR

3COLOR $\in \mathcal{NP}$

- Es kann in Zeit $\mathcal{O}(|E|)$ überprüft werden, ob eine Färbung von Graph $G = (V, E)$ mit drei Farben zulässig ist.

Beweis: $\mathcal{NP}$-Vollständigkeit von 3COLOR

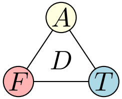
3SAT $\propto$ 3COLOR

- Sei I eine 3SAT-Instanz mit Variablen $U = \{u_1, \ldots, u_m\}$ und Klauseln $\{c_1, \ldots, c_n\}$.
- Wir konstruieren in Polynomialzeit eine 3COLOR-Instanz $G = f(I)$.
- Es soll gelten: I ist erfüllbar $\Leftrightarrow$ $G = f(I)$ ist 3-färbbar.

Konstruktion von 3COLOR-Instanz $G = f(I)$

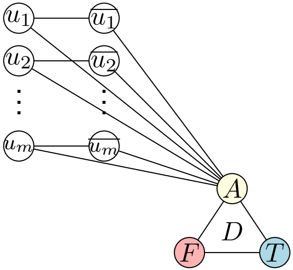
Karlsruher Institut für Technologie

- Ein Hauptdreieck D aus Knoten $\{T, F, A\}$ und Kanten $\{\{T, F\}, \{F, A\}, \{T, A\}\}$
- Interpretation: T, F, A sind die drei Farben mit denen G gefärbt wird.
- Interpretation: $T \longleftrightarrow$ `wahr`, $F \longleftrightarrow$ `falsch`



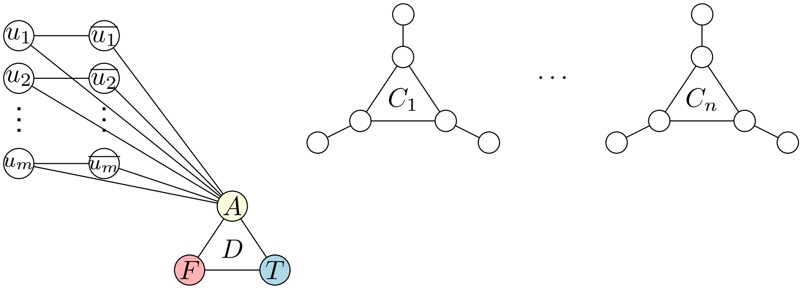
Konstruktion von 3COLOR-Instanz $G = f(I)$

- Ein Hauptdreieck D aus Knoten $\{T, F, A\}$ und Kanten $\{\{T, F\}, \{F, A\}, \{T, A\}\}$
- Interpretation: T, F, A sind die drei Farben mit denen G gefärbt wird.
- Interpretation: $T \longleftrightarrow$ `wahr`, $F \longleftrightarrow$ `falsch`
- Für jede Variable $u_i \in U$ zwei Knoten $u_i, \overline{u_i}$ und ein Dreieck $\{u_i, \overline{u_i}, A\}$.
- Interpretation: Falls u_i in Farbe T, muss $\overline{u_i}$ in Farbe F.



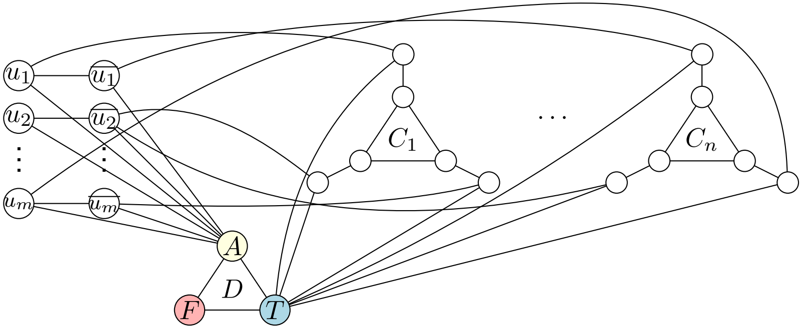
Konstruktion von 3COLOR-Instanz $G = f(I)$

- Für jede Klausel $c_j = x \vee y \vee z$ eine Komponente C_j wie folgt:
  - C_j besteht aus sechs Knoten, einem “inneren Dreieck” und drei “Satelliten”.



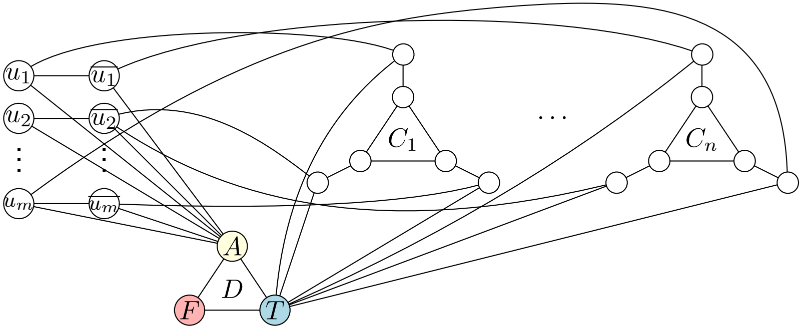
Konstruktion von 3COLOR-Instanz $G = f(I)$

- Für jede Klausel $c_j = x \vee y \vee z$ eine Komponente C_j wie folgt:
  - C_j besteht aus sechs Knoten, einem "inneren Dreieck" und drei "Satelliten".
- Jeder der drei Satelliten wird mit einem der Literale x, y, z verbunden.
- Alle drei Satelliten werden mit dem Knoten T in D verbunden.



Polynomialität der Reduktion

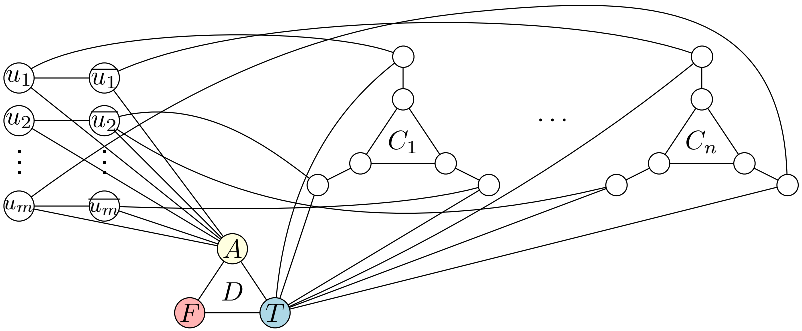
- Die Knotenanzahl von $G = f(I)$ liegt in $\mathcal{O}(n+m)$.
- Deswegen ist die Transformation polynomial.



Zu zeigen:
3SAT-Instanz I ist Ja-Instanz $\Leftrightarrow$ 3COLOR-Instanz $G = f(I)$ ist Ja-Instanz.

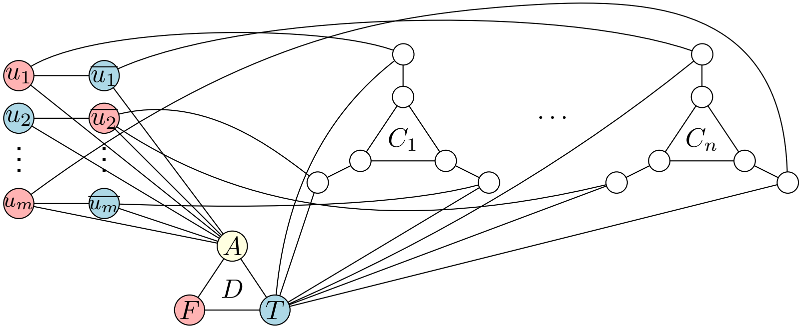
Instanz I erfüllbar $\Rightarrow$ Instanz $G = f(I)$ 3-färbbar

- Betrachte zulässige Wahrheitsbelegung t für I.
- Färbe wahre Literale wie Knoten T, falsche Literale wie Knoten F.



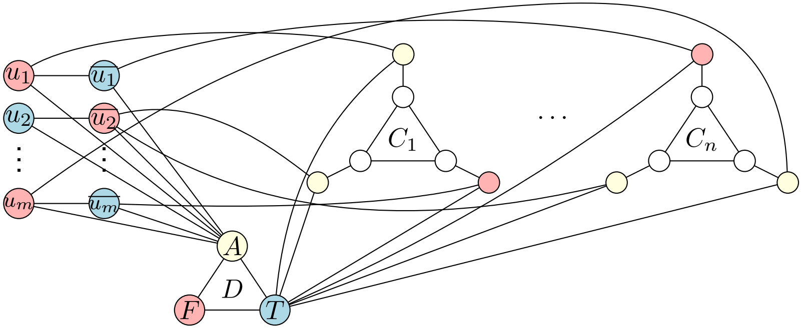
Instanz I erfüllbar $\Rightarrow$ Instanz $G = f(I)$ 3-färbbar

- Betrachte zulässige Wahrheitsbelegung t für I.
- Färbe wahre Literale wie Knoten T, falsche Literale wie Knoten F.
- Färbe Satelliten zu genau einem wahren Literal mit F,



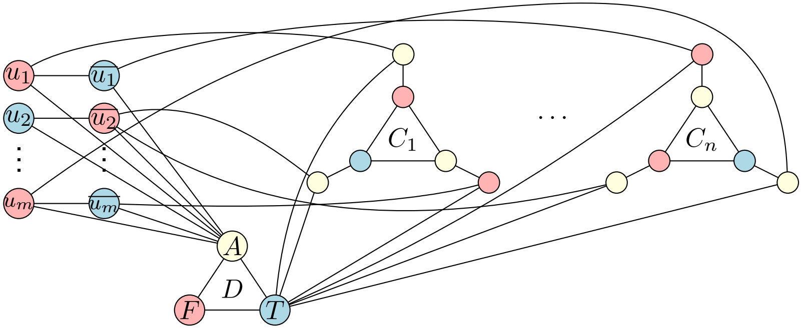
Instanz I erfüllbar $\Rightarrow$ Instanz $G = f(I)$ 3-färbbar

- Betrachte zulässige Wahrheitsbelegung t für I.
- Färbe wahre Literale wie Knoten T, falsche Literale wie Knoten F.
- Färbe Satelliten zu genau einem wahren Literal mit F, die beiden anderen Satelliten mit A.



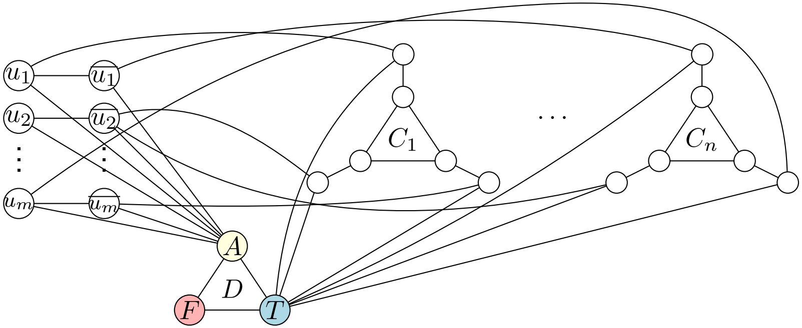
Instanz I erfüllbar $\Rightarrow$ Instanz $G = f(I)$ 3-färbbar

- Betrachte zulässige Wahrheitsbelegung t für I.
- Färbe wahre Literale wie Knoten T, falsche Literale wie Knoten F.
- Färbe Satelliten zu genau einem wahren Literal mit F, die beiden anderen Satelliten mit A.
- Inneres Dreieck kann dann zulässig gefärbt werden.



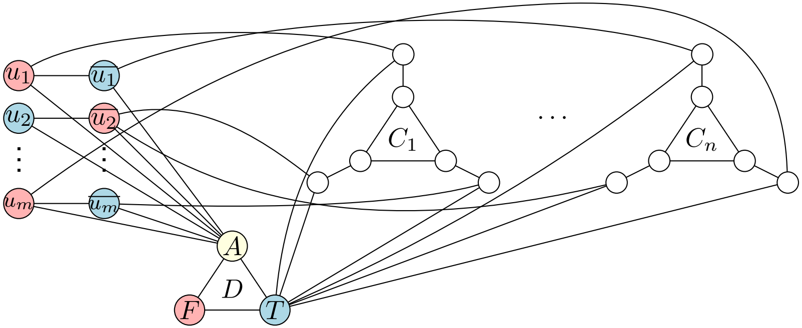
Instanz I erfüllbar $\Leftarrow$ Instanz $G = f(I)$ 3-färbbar

- Betrachte 3-Färbung von $G = f(I)$. Interpretiere $T \longleftrightarrow$ `wahr`, $F \longleftrightarrow$ `falsch`.



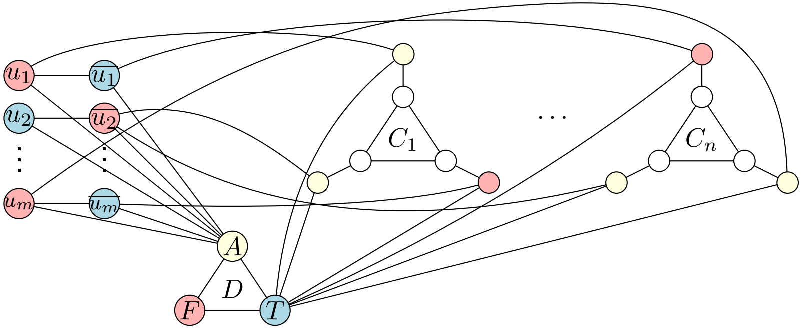
Instanz I erfüllbar $\Leftarrow$ Instanz $G = f(I)$ 3-färbbar

- Betrachte 3-Färbung von $G = f(I)$. Interpretiere $T \longleftrightarrow$ `wahr`, $F \longleftrightarrow$ `falsch`.
- Färbung von Literal-Knoten induziert Wahrheitsbelegung t von I.



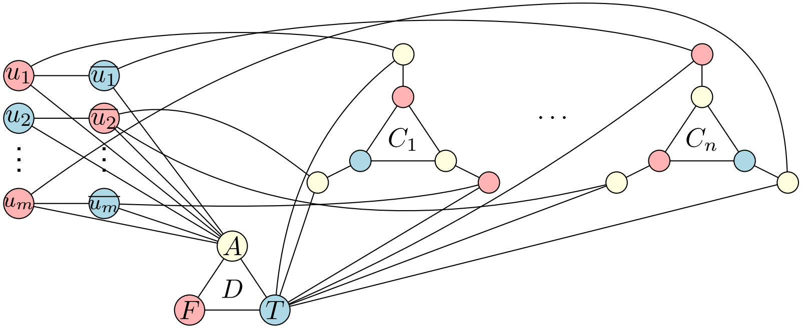
Instanz I erfüllbar $\Leftarrow$ Instanz $G = f(I)$ 3-färbbar

- Betrachte 3-Färbung von $G = f(I)$. Interpretiere $T \longleftrightarrow$ `wahr`, $F \longleftrightarrow$ `falsch`.
- Färbung von Literal-Knoten induziert Wahrheitsbelegung t von I. Kein Satellit hat Farbe von T.



Instanz I erfüllbar $\Leftarrow$ Instanz $G = f(I)$ 3-färbbar

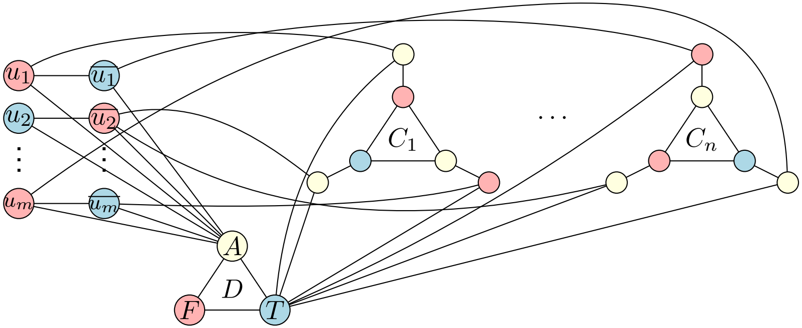
- Betrachte 3-Färbung von $G = f(I)$. Interpretiere $T \longleftrightarrow$ `wahr`, $F \longleftrightarrow$ `falsch`.
- Färbung von Literal-Knoten induziert Wahrheitsbelegung t von I. Kein Satellit hat Farbe von T.
- Nicht alle Satelliten sind in Farbe von A wegen des inneren Dreiecks.



Instanz I erfüllbar $\Leftarrow$ Instanz $G = f(I)$ 3-färbbar

- Betrachte 3-Färbung von $G = f(I)$. Interpretiere $T \longleftrightarrow$ `wahr`, $F \longleftrightarrow$ `falsch`.
- Färbung von Literal-Knoten induziert Wahrheitsbelegung t von I. Kein Satellit hat Farbe von T.
- Nicht alle Satelliten sind in Farbe von A wegen des inneren Dreiecks.

  $\Rightarrow$ Mindestens ein Literal pro Klausel in Farbe T, also I erfüllbar.



Zwischenstand Polynomiale Reduktion

Testen Sie sich: Können Sie zeigen, dass folgende Probleme $\mathcal{NP}$-vollständig sind?

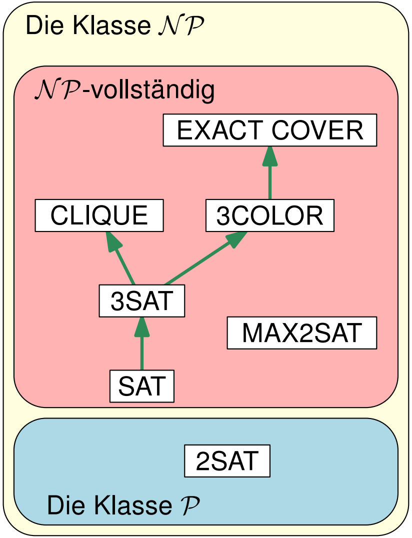
Geg.:	Graph G, Zahl K
Frage:	$\exists V' \subseteq V, \lvert V' \rvert = k$, keine zwei Knoten in V' verbunden?

Geg.:	Graph $G = (V, E)$
Frage:	$\exists$ 4-Färbung von V, keine zwei Knoten gleichfarbig verbunden?

Π'	$\propto$	Π
bekannt $\mathcal{NP}$-vollständig (wähle ähnlich zu Π)		$\mathcal{NP}$-vollständig zu zeigen
beliebige Instanzen	$\xrightarrow{f}$	spezielle Instanzen (größer, aber noch polynomial)
Ja-Instanzen	$\longleftrightarrow$	Ja-Instanzen

Der Plan

- 3SAT ist $\mathcal{NP}$-vollständig
  $\rightsquigarrow$ 3SAT $\in \mathcal{NP}$
  $\rightsquigarrow$ SAT $\propto$ 3SAT
- 2SAT ist in $\mathcal{P}$
- MAX2SAT ist $\mathcal{NP}$-vollständig
  $\rightsquigarrow$ Übung
- CLIQUE ist $\mathcal{NP}$-vollständig
  $\rightsquigarrow$ CLIQUE $\in \mathcal{NP}$
  $\rightsquigarrow$ 3SAT $\propto$ CLIQUE



- 3COLOR ist $\mathcal{NP}$-vollständig
  $\rightsquigarrow$ 3COLOR $\in \mathcal{NP}$
  $\rightsquigarrow$ 3SAT $\propto$ 3COLOR
- EXACT COVER ist $\mathcal{NP}$-vollständig
  $\rightsquigarrow$ EXACT COVER $\in \mathcal{NP}$
  $\rightsquigarrow$ 3COLOR $\propto$ EXACT COVER