# $\mathcal{NP}$-Vollständigkeit von 3-CNF-SAT

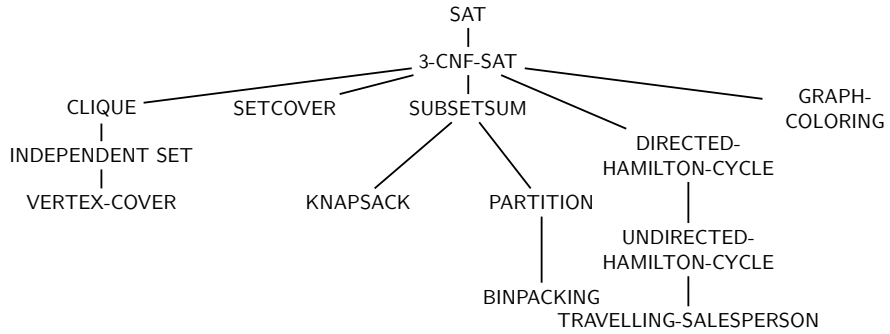
Prof. Dr. David Sabel

LFE Theoretische Informatik


Letzte Änderung der Folien: 19. Juli 2022

## Inhalt der kommenden Vorlesungen

$\mathcal{NP}$-Vollständigkeitsbeweise für eine Auswahl an Problemen.
$\mathcal{NP}$-Schwere durch Polynomialzeitreduktion bekannter $\mathcal{NP}$-vollständiger Probleme auf das neue Problem.




In dieser Vorlesung: $\mathcal{NP}$-Vollständigkeit von 3-CNF-SAT



## Wiederholung: NP-Vollständigkeit

**Definition ($\mathcal{NP}$-Vollständigkeit)**

Eine Sprache $L$ heißt $\mathcal{NP}$-vollständig, wenn gilt

1. $L \in \mathcal{NP}$ und
2. $L$ ist $\mathcal{NP}$-schwer (manchmal auch $\mathcal{NP}$-hart genannt):
   Für alle $L' \in \mathcal{NP}$ gilt $L' \leq_p L$

Ausreichend, um $\mathcal{NP}$-Vollständigkeit von $L$ zu zeigen:

1. Zeige $L \in \mathcal{NP}$
2. Zeige $L_0 \leq_p L$ für ein bekanntes $\mathcal{NP}$-vollständiges Problem (z.B. $L_0 = \text{SAT}$)



## 3-CNF-SAT

**Konjunktive Normalform**

Aussagenlogische Formel $F$ ist in konjunktiver Normalform (CNF), wenn:

$$F = \bigwedge_{i=1}^{m} (\bigvee_{j=1}^{n_i} L_{i,j})$$

- Literal $L_{i,j}$ ist aussagenlogische Variable oder deren Negation
- $(\bigvee_{j=1}^{n_i} L_{i,j})$ ist eine Klausel
- Erfüllende Belegung muss $\geq 1$ Literal pro Klausel wahr machen.

Klauselmengenschreibweise: $\{\{L_{1,1}, \ldots, L_{1,n_1}\}, \ldots, \{L_{m,1}, \ldots, L_{m,n_m}\}\}$

Annahme: Keine Klauseln, die $x$ und $\neg x$ enthalten
Diese Klauseln sind immer wahr und können gelöscht werden.

## Konjunktive Normalform

Jede aussagenlogische Formel kann in eine **äquivalente** konjunktive Normalform gebracht werden:

- $\iff$, $\implies$ auflösen
- Negationen nach innen schieben und anschließend Ausmultiplizieren (Distributivität, Kommutativität anwenden, um konjunktive Normalform herzustellen)

Aber:

- Algorithmus hat im worst case **exponentielle Laufzeit**
- Algorithmus kann daher **nicht** für eine **Polynomialzeitreduktion** verwendet werden.



## 3-CNF-SAT

**Definition (3-CNF-SAT)**

Das 3-CNF-SAT-Problem lässt sich in der gegeben/gefragt-Notation formulieren als:

gegeben: Eine aussagenlogische Formel $F$ in konjunktiver Normalform, sodass jede Klausel höchstens 3 Literale enthält.

gefragt: Ist $F$ erfüllbar? Genauer: Gibt es eine erfüllende Belegung der Variablen mit den Wahrheitswerten 0 und 1, sodass $F$ den Wert 1 erhält?



## $\mathcal{NP}$-Vollständigkeit von 3-CNF-SAT

**Satz**

3-CNF-SAT ist $\mathcal{NP}$-vollständig.

Beweis, Teil 1: 3-CNF-SAT ist in $\mathcal{NP}$

- Rate nicht-deterministisch eine Belegung der Variablen
- Prüfe in jedem nicht-deterministischen Zweig deterministisch, ob die Belegung die 3-CNF wahr macht. Akzeptiere in diesem Fall, sonst verwerfe.
  Dies geht in Polynomialzeit in der Größe der 3-CNF.
- Daher kann 3-CNF-SAT auf einer NTM in Polynomialzeit entschieden werden.



## $\mathcal{NP}$-Vollständigkeit von 3-CNF-SAT (2)

Beweis, Teil 2: 3-CNF-SAT ist $\mathcal{NP}$-schwer

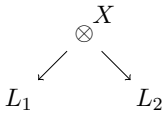
- Wir zeigen SAT $\leq_p$ 3-CNF-SAT.
- Gesucht: Polynomiell berechenbare, totale Funktion $f$, sodass $F$ erfüllbar g.d.w. 3-CNF $f(F)$ erfüllbar.
- $f$ muss **die Erfüllbarkeit erhalten**, aber nicht die **Äquivalenz**
- Verfahren, um $F$ in erfüllbarkeitsäquivalente 3-CNF $f(F)$ umzuformen, sodass $f(F)$ **polynomielle Größe** in $|F|$ hat:

**Tseitin-Transformation**

## $\mathcal{NP}$-Vollständigkeit von 3-CNF-SAT (3)

**Tseitin-Transformation:**

- Schiebe alle **Negationen nach innen** vor die Literale, dabei werden die Regeln $\neg\neg F \to F$, $\neg(F \wedge G) \to \neg F \vee \neg G$, $\neg(F \vee G) \to \neg F \wedge \neg G$, $\neg(F \iff G) \to (\neg F) \iff G$ und $\neg(F \implies G) \to F \wedge \neg G$ angewendet.
- Syntaxbaum der Formel (Blätter = Literale):
  Für jeden Nichtblatt-Knoten: neue aussagenlogische Variable
- pro Gabelung: erzeuge




$(X \iff (L_1 \otimes L_2))$

wobei $\otimes \in \{ \iff, \wedge, \vee, \implies \}$ und $L_1, L_2$ entweder die neue Variable oder das Literal am Blatt.

- Konjugiere diese Formeln zu $F'$ und schließlich erzeuge $W \wedge F'$, mit $W$ Variable für die Wurzel.



## $\mathcal{NP}$-Vollständigkeit von 3-CNF-SAT (3)

**Tseitin-Transformation** (Fortsetzung)

- Insgesamt: $W \wedge \bigwedge_{i,j,k}(X_i \iff (X_j \otimes_i X_k))$, wobei $X_r$ Literale sind.
- Berechne für jede Subformel $(X_i \iff (X_j \otimes_i X_k))$ die CNF mit dem üblichen Algorithmus.
- Lösche doppelte Vorkommen von Literalen.
- Ergibt 3-CNF, da pro Klausel nur 3 verschiedene Variablen vorkommen können.



## $\mathcal{NP}$-Vollständigkeit von 3-CNF-SAT (4)

Komplexität:

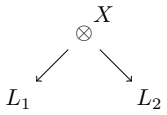
- Größe der kleinen CNFs ist konstant.
- Jede Klausel hat nur 3 Literale: Mehr Variablen gibt es in den kleinen Formeln nicht, doppelte Literale löschen
- Größe der 3-CNF ist polynomiell in der ursprünglichen Formel
- Berechnung in Polynomialzeit geht daher!



## $\mathcal{NP}$-Vollständigkeit von 3-CNF-SAT (5)

$F$ erfüllbar g.d.w. $f(F)$ erfüllbar:

- „$\Rightarrow$“: Sei $I$ Belegung mit $I(F) = 1$. Sei $I'$ Belegung mit
  - $I'(X) = I(X)$ für alle Variablen, die in $F$ vorkommen
  - $I'(W) = 1$ für die Variable $W$ an der Wurzel
  - $I'(X) = I'(L_1 \otimes L_2)$ für




  Dann gilt: $I'(f(F)) = 1$
- „$\Leftarrow$“: Sei $J$ Belegung von $f(F)$ mit $J(f(F)) = 1$
  Dann: $I = J_{|Var(F)}$ macht $F$ wahr.

Damit: SAT $\leq_p$ 3-CNF-SAT.
Da SAT $\mathcal{NP}$-schwer, ist somit auch 3-CNF-SAT $\mathcal{NP}$-schwer.

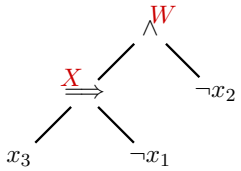
## Beispiel

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

Negationen nach innen schieben:

$$\begin{aligned} & \neg(\neg(x_3 \implies \neg x_1) \vee x_2) \\ \to\ & (\neg(\neg(x_3 \implies \neg x_1)) \wedge \neg x_2) \\ \to\ & (x_3 \implies \neg x_1) \wedge \neg x_2) \end{aligned}$$

Syntaxbaum dazu:




Formel: $W \wedge (W \iff (X \wedge \neg x_2)) \wedge (X \iff (x_3 \implies \neg x_1))$



## Beispiel (2)

Berechnung der CNFs der kleinen Subformeln:

$$\begin{aligned} & W \wedge (W \iff (X \wedge \neg x_2)) \wedge (X \iff (x_3 \implies \neg x_1)) \\ \to\ & W \wedge ((W \vee \neg(X \wedge \neg x_2)) \wedge (\neg W \vee (X \wedge \neg x_2))) \wedge (X \iff (x_3 \implies \neg x_1)) \\ \to\ & W \wedge ((W \vee (\neg X \vee x_2)) \wedge (\neg W \vee (X \wedge \neg x_2))) \wedge (X \iff (x_3 \implies \neg x_1)) \\ \to\ & W \wedge (W \vee \neg X) \wedge (W \vee x_2) \wedge (\neg W \vee X) \wedge (\neg W \vee \neg x_2) \wedge (X \iff (x_3 \implies \neg x_1)) \\ \to\ & W \wedge (W \vee \neg X) \wedge (W \vee x_2) \wedge (\neg W \vee X) \wedge (\neg W \vee \neg x_2) \\ & \wedge(\neg X \vee (x_3 \implies \neg x_1)) \wedge (X \vee \neg(x_3 \implies \neg x_1)) \\ \to\ & W \wedge (W \vee \neg X) \wedge (W \vee x_2) \wedge (\neg W \vee X) \wedge (\neg W \vee \neg x_2) \\ & \wedge(\neg X \vee \neg x_3 \vee \neg x_1) \wedge (X \vee \neg(\neg x_3 \vee \neg x_1)) \\ \to\ & W \wedge (W \vee \neg X) \wedge (W \vee x_2) \wedge (\neg W \vee X) \wedge (\neg W \vee \neg x_2) \\ & \wedge(\neg X \vee \neg x_3 \vee \neg x_1) \wedge (X \vee (x_3 \wedge x_1)) \\ \to\ & W \wedge (W \vee \neg X) \wedge (W \vee x_2) \wedge (\neg W \vee X) \wedge (\neg W \vee \neg x_2) \\ & \wedge(\neg X \vee \neg x_3 \vee \neg x_1) \wedge (X \vee x_3) \wedge (X \vee x_1) \end{aligned}$$

Erfüllende Belegung $J$:

$J(W) = 1,\ J(X) = 1,\ J(x_1) = 0,\ J(x_2) = 0,\ J(x_3) = 1$

Belegung $I$:

$I(x_1) = 0,\ I(x_2) = 0,\ I(x_3) = 1$ erfüllt $F = \neg(\neg(x_3 \implies \neg x_1) \vee x_2)$