Satz 3CNF-SAT ist NP-vollständig

$\in NP$: ✓

NP-Härte: Gegeben eine Formel F, erzeuge eine Formel F', so dass

F ist erfüllbar gdw. F' ist erfüllbar.

1. Überführung in Negationsnormalform (NNF), d.h. Negationszeichen nur direkt vor Variablen
   $(\neg x \wedge \neg y)$ $\cancel{\neg(x \vee y)}$.

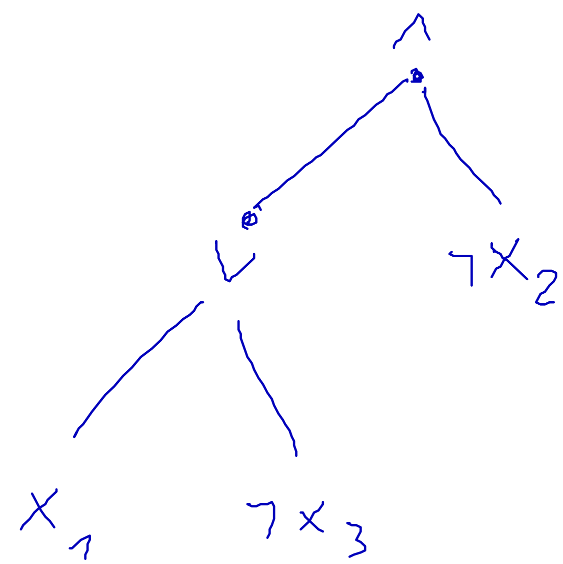
   Herstellen der NNF: de Morgan, dabei Negationszeichen nach innen schieben.

Bsp: $\neg(\neg(x_1 \vee \neg x_3) \vee x_2)$

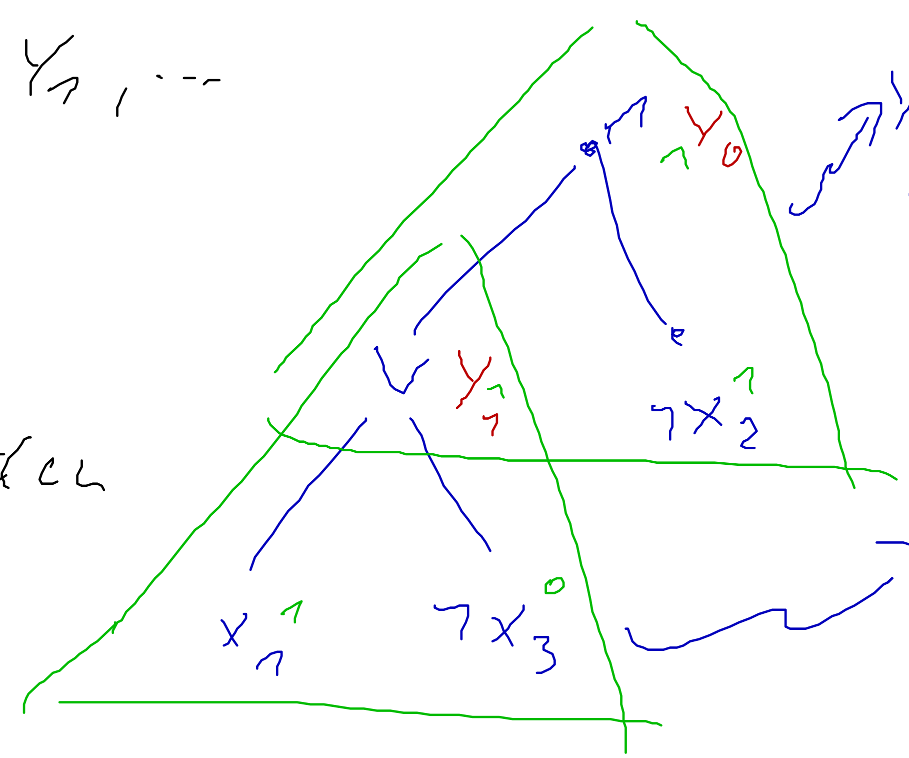
$\equiv (\neg\neg(x_1 \vee \neg x_3) \wedge \neg x_2) \equiv (x_1 \vee \neg x_3) \wedge \neg x_2$

Schritt 2: Betrachte Formel als Baum, bei denen Operatoren innere Knoten sind, Literale sind Blätter

Bsp $(x_1 \lor \neg x_3) \land \neg x_2$



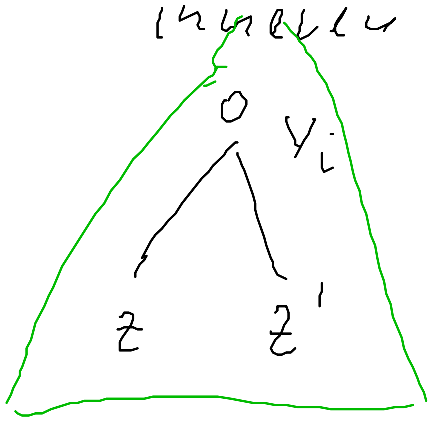
Schritt 3: Ordne inneren Knoten jeweils eine neue Variable zu $y_0, y_1, \dots$



$y_0 \leftrightarrow (y_1 \land \neg x_2)$

$y_1 \leftrightarrow (x_1 \lor \neg x_3)$

Schritt 4: Erzeuge für jeden inneren Knoten eine Formel



$\leadsto y_i \leftrightarrow (z \circ z')$

$\circ \in \{\lor, \land\}$

Schritt 5: Verknüpfe alle erzeugten Formeln und y_0 mit $\wedge$

Bsp: $\underline{y_0} \wedge \underline{[y_0 \Leftrightarrow (y_1 \wedge \neg x_2)]} \wedge [y_1 \Leftrightarrow (x_1 \vee \neg x_3)]$

Schritt 6: Wandle in 3CNF um:

$$[x \Leftrightarrow (y \vee z)] \equiv [x \rightarrow (y \vee z)] \wedge [x \leftarrow (y \vee z)]$$
$$\equiv [\neg x \vee (y \vee z)] \wedge [x \vee \neg (y \vee z)]$$
$$\equiv (\neg x \vee y \vee z) \wedge (x \vee \neg y) \wedge (x \vee \neg z)$$

$$[x \Leftrightarrow (y \wedge z)] \equiv [x \rightarrow (y \wedge z)] \wedge [x \leftarrow (y \wedge z)]$$
$$\equiv [\neg x \vee (y \wedge z)] \wedge [x \vee \neg (y \wedge z)]$$
$$\equiv (\neg x \vee y) \wedge (\neg x \vee z) \wedge (x \vee \neg y \vee \neg z)$$

Bsp: $y_6 \wedge [y_6 \leftrightarrow (y_1 \wedge \neg x_2)] \wedge [y_1 \leftrightarrow (x_1 \vee \neg x_3)]$

$$\boxed{\begin{aligned} & y_6 \wedge \\ & (\neg y_6 \vee y_1) \wedge \\ & (\neg y_6 \vee \neg x_2) \wedge \\ & (y_6 \vee \neg y_1 \vee x_2) \wedge \\ & (\neg y_1 \vee x_1 \vee \neg x_3) \wedge \\ & (y_1 \vee \neg x_1) \wedge \\ & (y_1 \vee x_3) \end{aligned}} = F'$$

F und F' sind erfüllbarkeitsäquivalent.

Sei F erfüllbar mittels einer Belegung α. Erweitere α zu α' über F', in dem die Variablen y_i gemäß des Wahrheitswertes der inneren belegt werden. Alle Klausel von F' werden durch α' wahr gemacht.

Sei F' mittels α' erfüllbar. Schränke α' auf α über F ein. Diese Belegung macht F wahr. D.h. F ist erfüllbar.

Da die Umformung offensichtlich nur polynomielle Zeit braucht, gilt: SAT $\leq_p$ 3CNF-SAT. $\square$

Wir haben gezeigt:

- SAT ist NP-vollständig, d.h. so schwer wie alle anderen Probleme in NP.
- DNF-SAT ist in P.
- 3CNF-SAT ist immer noch NP-vollst.
- 2CNF-SAT $\in$ P
- 1CNF-SAT $\in$ P $\quad (x_1) \wedge (\neg x_2) \wedge \underline{(x_3)} \wedge \ldots \wedge (\neg x_3) \wedge \underline{(\neg x_3)}$
- CNF-SAT NP-vollst trivial (Reduktionsfkt. = Identitätsfkt.)
- 4CNF-SAT? — " —

Viele Probleme (Diagnose, Soft-/Hardware-Verifikation, ...) können als SAT-Probleme formuliert werden (oft CNF-SAT).

Bei n Variablen brauchen wir 2^n Tests der Formel.

Ann: Ein Test dauert 10^{-8} Sekunden.

Größe $n = 10$: $2^{10} \times \underline{10^{-8}} \approx 10^{3} \times 10^{-8} \approx 10^{-5}$ Sek.

$n = 100$

1 Jahr hat 3×10^{7} Sekunde

$\underline{1 \text{ Jahr}: 3 \times 10^{15} \text{ Tests}}$

2^{100} Tests, d.h. 10^{30} Tests

$2^{10} \approx 10^{3}$

d.h. wir brauche rund 10^{14} Jahre

Fazit: Oft genug kann auch große Instanzen von NP-vollst. Problem lösen. Aber es gibt keine Laufzeitgarantien!

Weitere NP-vollständige Probleme

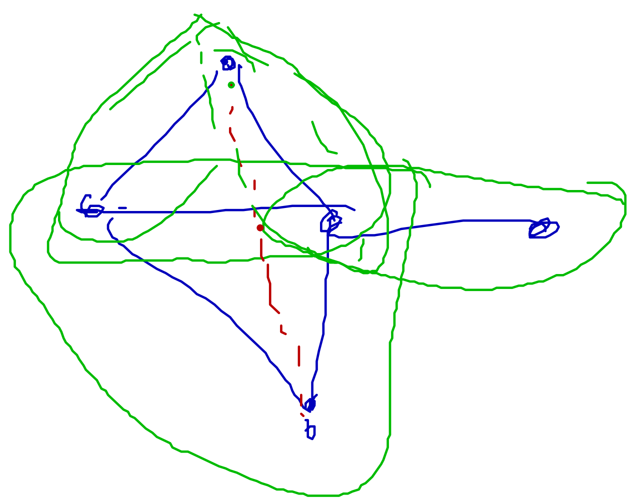
CLIQUE:

Gegeben: Ein ungerichteter Graph $G = (V, E)$ und natürliche Zahl k.

Gefragt: Besitzt G eine Clique der Größe k

(= vollständiger Teilgraph der Größe k)

Bsp:



Satz CLIQUE ist NP-vollständig.

Bew:

$\in NP$: Rate $V' \subseteq V$. Verifiziere $|V'| \geq k$ und dass für alle $(x, y) \in V' \times V'$ gelten dass $(x, y) \in E$.

NP-Härte: 3CNF-SAT $\leq_p$ CLIQUE zu zeigen!

Sei: $F = (l_{11} \vee l_{12} \vee l_{13}) \wedge \ldots \wedge (l_{m1} \vee l_{m2} \vee l_{m3})$

$$l_{ij} \in \{x_i \mid 1 \leq i \leq n\} \cup \{\neg x_i \mid 1 \leq i \leq n\}$$

Konstruiere $G_F = (V_F, E_F)$

$$V_F = \{(1,1), (1,2), (1,3), \ldots, (m,1), (m,2), (m,3)\}$$

→ keine kompl. Literale

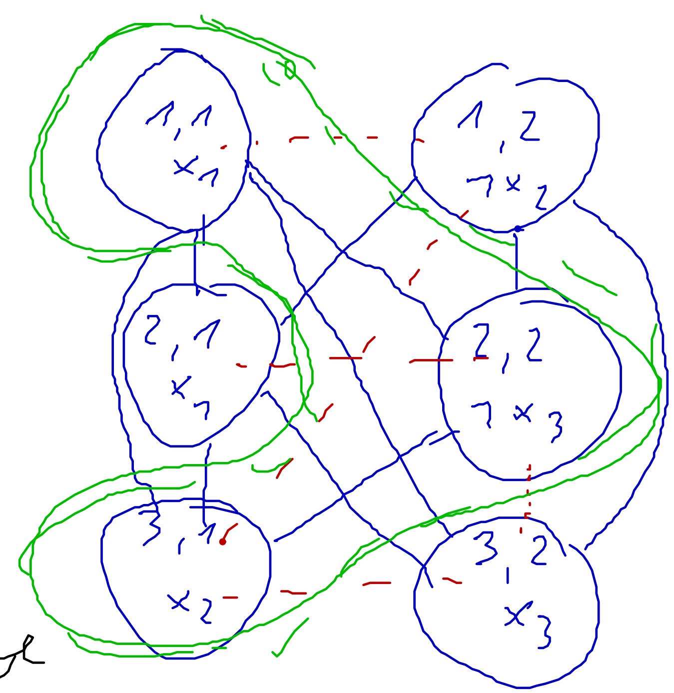
$$E_F = \{\{(i,j), (p,q)\} \mid i \neq p \text{ und } l_{ij} \not\equiv \neg l_{pq}\}$$

↳ jede Klausel muss wählen

$k_F = m$

Bsp $F = (\overset{1,1}{x_1} \vee \overset{1,2}{\neg x_2}) \wedge (\overset{2,1}{x_1} \vee \overset{2,2}{\neg x_3}) \wedge (\overset{3,1}{x_2} \vee \overset{3,2}{x_3})$

Es ex. eine Clique der Größe der k: Es können Kanten aus allen Klauseln vor, und alle vorkommende Literale sind nicht komplementär. D.h. man kann daraus eine erfüllende Belegung für F erzeugen



Es ex. eine erfüllende Belegung. Dann aus der Belegung eine Clique konstruiert werde. Da G_F in poly. Zeit konstr., haben wir eine poly. Red. ◻