Reduktion von 3SAT auf DHAM

Theorem

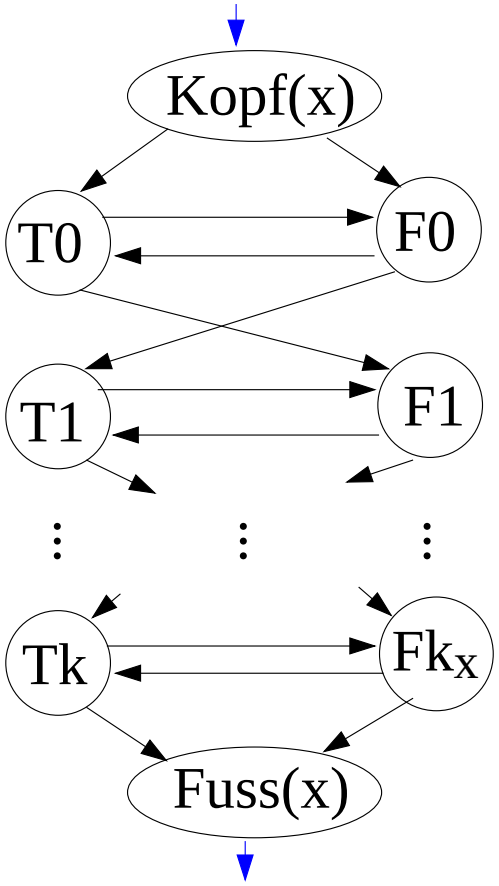
DHAM ist NP-vollständig.

Beweisidee: Reduziere 3SAT auf DHAM.

Reduktion von 3SAT auf DHAM

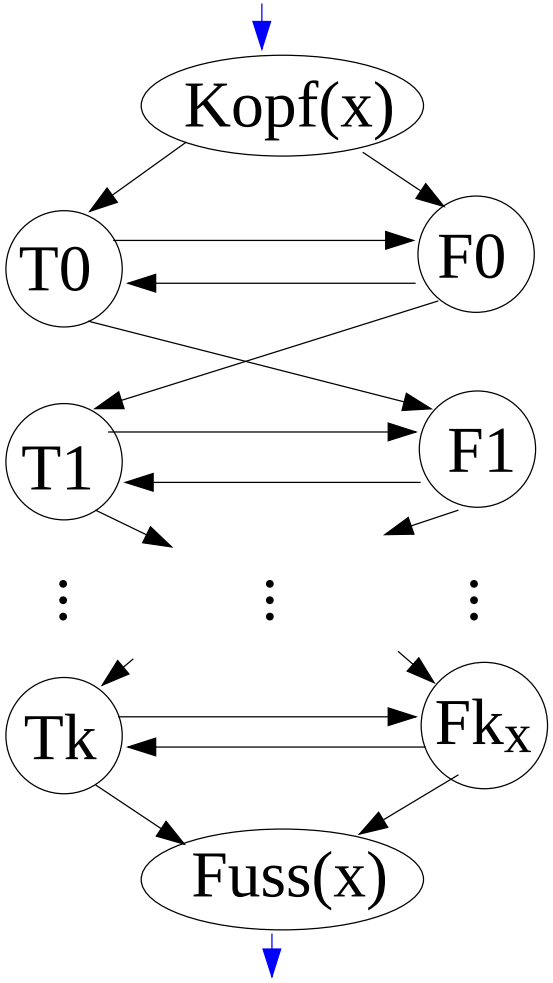
- f: Eingabeformel für 3SAT mit den Variablen $\{x_1, \ldots x_n\}$

1. Konstruiere für jede Variable x in f den Graphen $G(x)$



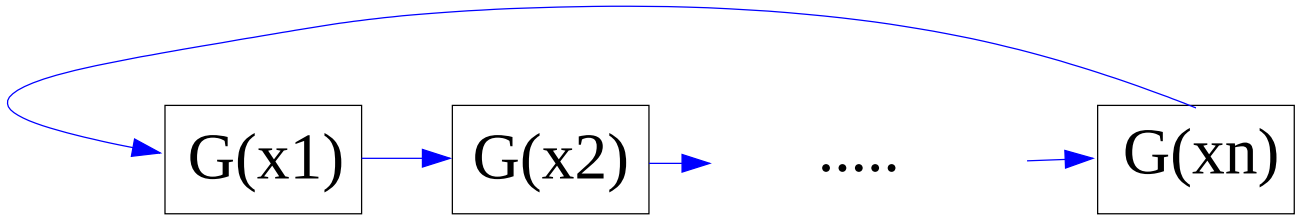
mit $k_x = max(count(x, f), count(\neg x, f))$

Idee



Durchlaufe den Graphen auf einem Rundweg von Kopf bis Fuß. Gehe dabei zuerst zu T_0, falls x mit $True$ belegt ist; sonst gehe zuerst zu F_0.

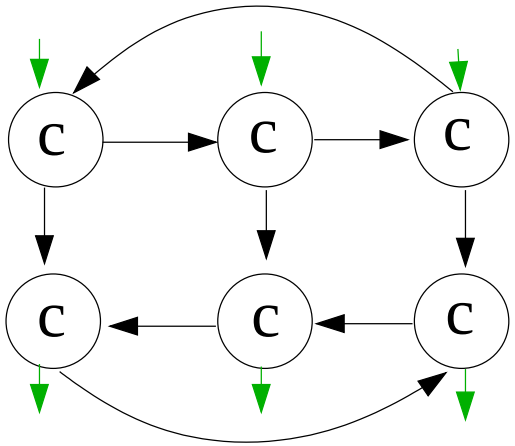
2. Verbinde diese Graphen miteinander zu G':



Beobachtung

Für jede Belegung der Variablen existiert ein Hamiltonscher Keis in G' (d.h. ein Rundweg, der jeden Knoten genau einmal besucht).

3. Konstruiere für jede Klausel c in f den Graphen $G(c)$:



Beobachtung

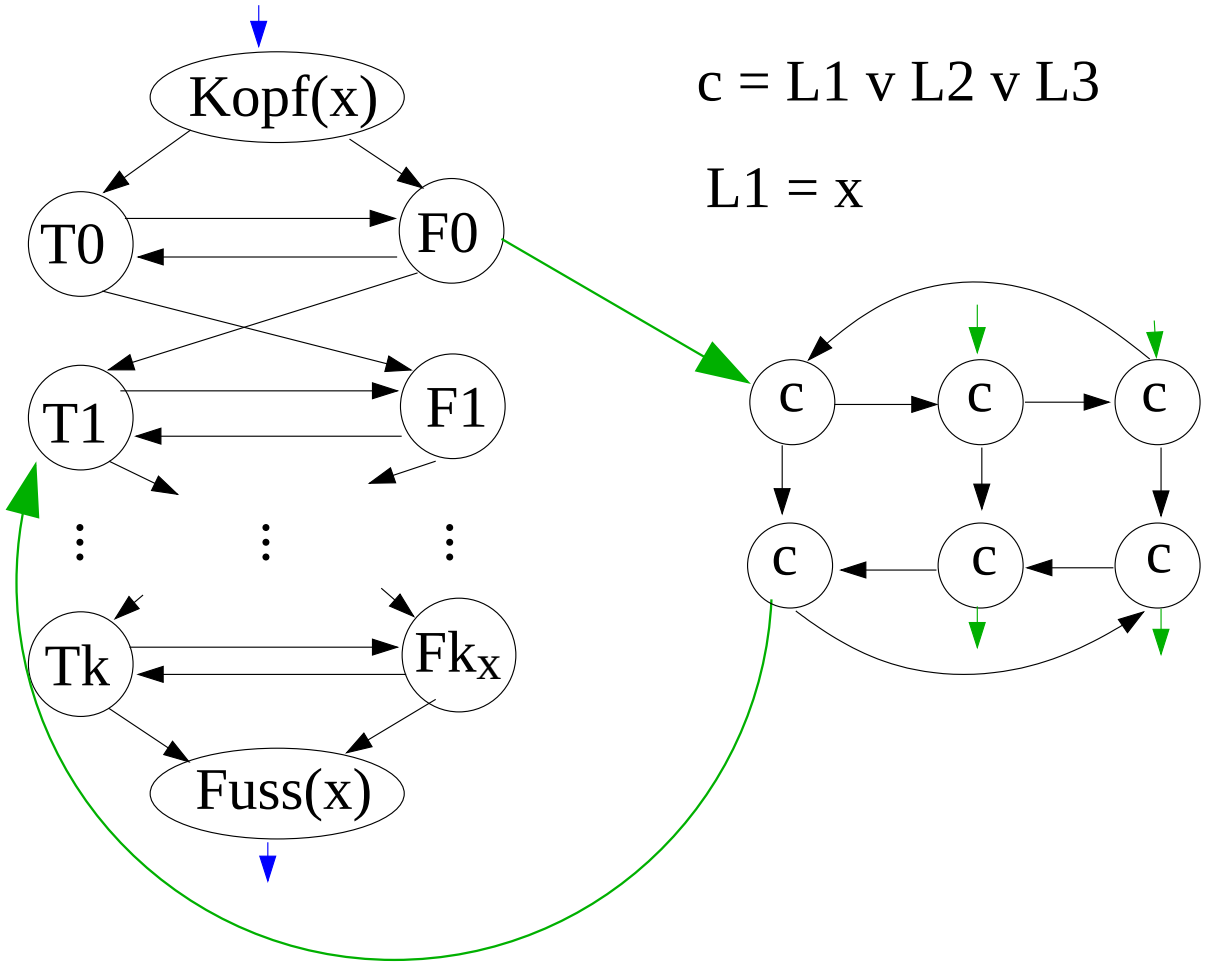
Läuft man in den i-ten oberen Knoten hinein, muss man aus dem i-ten unteren Knoten wieder herauslaufen, wenn man alle Knoten aus $G(c)$ besuchen will ($i = 1, 2, 3$).

4. Konstruiere $G(f)$, indem jeder Graph $G(c)$ mit dem Rest wie folgt verbunden wird:

   Sei $c = L_1 \vee L_2 \vee L_3$.

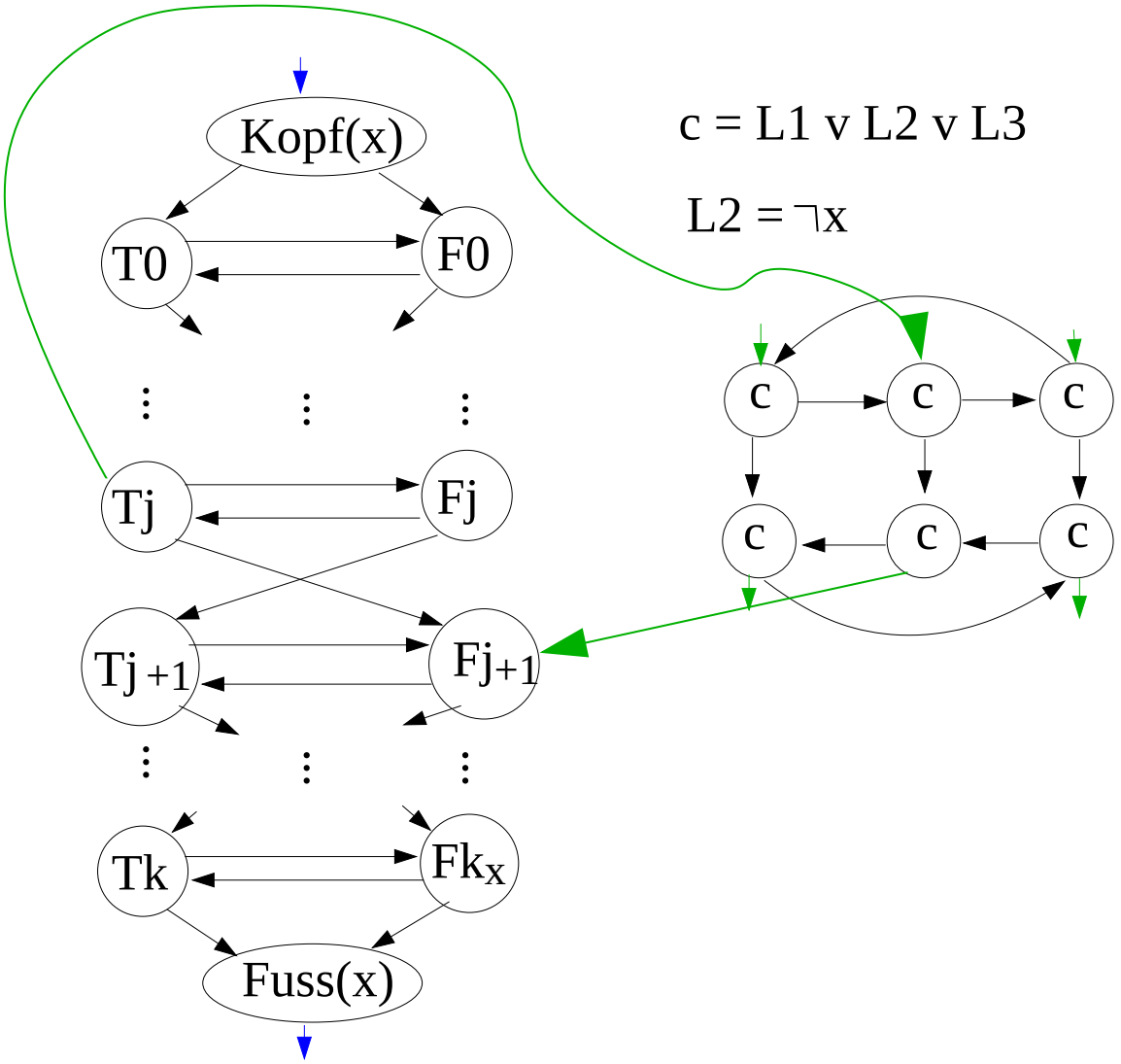
   Falls $L_i = x$, ziehe eine Kante von einem noch nicht benutzten F_j-Knoten von $G(x)$ ($j \in \{0, \ldots k_x - 1\}$) zu dem i-ten oberen Knoten in $G(c)$ und eine Kante von dem i-ten unteren Knoten in $G(c)$ zum Knoten T_{j+1} in $G(x)$.

Beispiel ($L_1 = x$)



Falls $L_i = \neg x$, ziehe eine Kante von einem noch nicht benutzten T_j-Knoten von $G(x)$ ($j \in \{0, \ldots k_x - 1\}$) zu dem i-ten oberen Knoten in $G(c)$ und eine Kante von dem i-ten unteren Knoten in $G(c)$ zum Knoten F_{j+1} in $G(x)$.

Beispiel ($L_2 = \neg x$)



Beobachtung

Der Hamiltonsche Kreis in G' für eine Belegung der Variablen kann einen Umweg über $G(c)$ machen gdw c $True$ ist.

Beobachtung

- DHAM ist in NP.
- f ist erfüllbar gdw es einen Hamiltonschen Weg durch $G(f)$ gibt.
- Die Konstruktion von $G(f)$ hat einen polynomiellen Zeitaufwand.

Polynomieller Platzbedarf

NPSPACE:

Probleme mit nichtdeterministischen Lösungsalgorithmen, die polynomiellen Speicherplatz brauchen (im Verhältnis zur Größe der Eingabe)

PSPACE:

Analog für deterministische Lösungen

Zusammenhänge zwischen Komplexitätsklassen

$$PSPACE =_{(1)} NPSPACE \supseteq_{(2)} NP \supseteq P$$

(1) Beweis durch Backtracking

(2) Konstanter Speicherplatz pro Rechenschritt

Offenes Problem: $P \supseteq PSPACE$

Wie weit reicht P und was kommt dahinter?

$$\boxed{\boxed{\boxed{P} =^? NP} =^? \begin{array}{l} PSPACE = \\ NPSPACE \end{array}} \neq EXP \quad \ldots \text{ Berechenbares}$$

- NP: Erfüllbarkeitsproblem, TSP u.v.a.m.
- $PSPACE$: Wortproblem monotoner Sprachen
- EXP: Hoffnungslos für große Eingaben
- Berechenbares: Interpreter für CE-S u. ä.; Aufwand oft nicht definiert (Berechnung unendlich)
- Unberechenbares: Halteproblem u.v.a.m.

Chomsky-Grammatiken

- Ursprünglich von Chomsky in den 1950er Jahren eingeführt zur Beschreibung natürlicher Sprachen.
- Enge Verwandschaft zu Automaten
- Grundlage wichtiger Softwarekomponenten
- Enthalten außer den rechtslinearen und den kontextfreien weitere Grammatiktypen

Chomsky-Grammatik (Typ 0)

$G = (N, T, P, S)$ mit

- N: endl. Menge nichtterminaler Zeichen,
- T: endl. Menge terminaler Zeichen mit $N \cap T = \emptyset$,
- $P \subseteq (N \cup T)^* N (N \cup T)^* \times (N \cup T)^*$: endliche Menge von Produktionen
- $S \in N$: Startsymbol

- Schreibweise für Produktionen $(u, v) \in P$: $u ::= v$

Direkte Ableitung

$$w = xuy \underset{p}{\longrightarrow} xvy = w'$$

mit $w, w', x, y, u, v \in (N \cup T)^*$, $p = (u ::= v)$.

1. Suche u als Teilwort eines Wortes
2. Ersetze u durch v

- Schreibweise: $w \underset{P}{\longrightarrow} w'$,
  falls P eine Menge von Produktionen ist mit $p \in P$.

Ableitung (Iteration direkter Ableitungen)

$$w_0 \underset{p_1}{\longrightarrow} w_1 \underset{p_2}{\longrightarrow} \cdots \underset{p_n}{\longrightarrow} w_n$$

für $w_0, \ldots, w_n \in (N \cup T)^*$ und Produktionen $p_1, \ldots, p_n$ $(n \geq 1)$

Schreibweisen:

- $w_0 \underset{P}{\longrightarrow} \cdots \underset{P}{\longrightarrow} w_n$ oder $w_0 \overset{n}{\underset{P}{\longrightarrow}} w_n$ oder $w_0 \overset{*}{\underset{P}{\longrightarrow}} w_n$, falls $p_1, \ldots, p_n \in P$.

- $w \overset{*}{\longrightarrow} w'$, falls P aus dem Kontext klar ist.

Nullableitung

$$w \xrightarrow[P]{0} w$$

für alle $w \in (N \cup T)^*$.

Erzeugte Sprache

Sei $G = (N, T, P, S)$ eine Chomsky-Grammatik.

Erzeugte Sprache

$$L(G) = \{w \in T^* \mid S \xrightarrow[P]{*} w\}$$