# Theoretische Informatik 2

Sabine Kuske
Linzer Str. 9a, OAS 3005
Tel.: 2335, 8794
kuske@informatik.uni-bremen.de
www.informatik.uni-bremen.de/theorie

7. April 2008

# Die Klasse $P$

Die Klasse $P$ enthält alle (Entscheidungs-)Probleme mit einer deterministischen polynomiellen Lösung.

Beispiele

$insort$, $quicksort$, $mergesort$, Matrizenmultiplikation, Wortproblem für kontextfreie Sprachen, . . .

# Die Klasse $NP$

Die Klasse $NP$ enthält alle (Entscheidungs-)Probleme mit einer nichtdeterministischen polynomiellen Lösung.

Beispiele

Traveling Salesperson, Hamiltonsche Kreise, Rucksack, Stundenplanung, Maschinenbelegung, Sudoku, . . .

# Beweis der $NP$-Vollständigkeit eines Problems $dp$ mittels Reduktion

**Theorem**

$dp$ ist $NP$-vollständig.

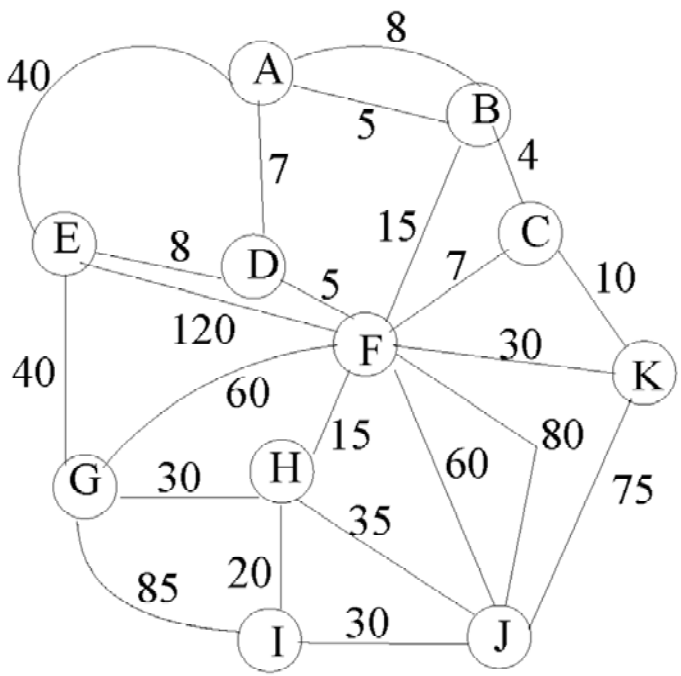
**Beweis:**

1. Zeige, dass $dp$ in $NP$ liegt.
2. Wähle ein $NP$-vollständiges Problem und reduziere es auf $dp$.

Anmerkung: Ist nur möglich, wenn man bereits ein NP-vollständiges Problem hat.

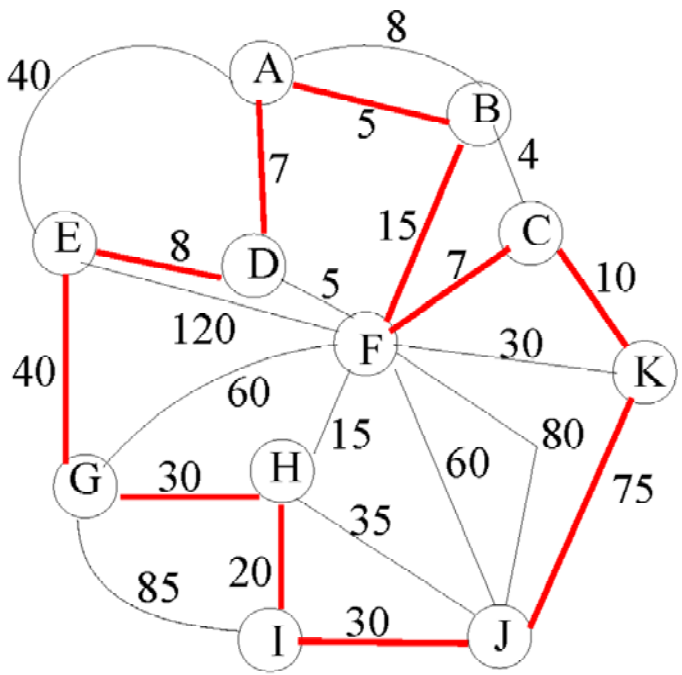
- TSP (Traveling Salesperson Problemm, Entscheidungsproblemvariante)
  - Eingabe: Ungerichteter Graph $G$ mit natürlichen Zahlen als Kantenmarkierungen und eine Zahl $k$.

    Beispiel für $G$:



- Ausgabe: T gdw $G$ einen Rundweg besitzt, der jeden Knoten genau einmal besucht und höchstens $k$ lang ist.

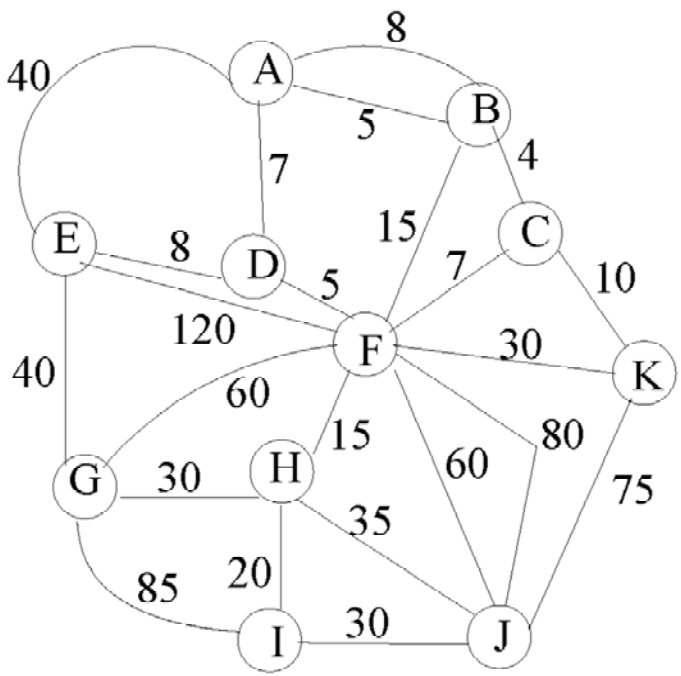
  Beispiel: T für k = 250




Rundweg: 247

- TSP (Traveling Salesperson Problemm, Entscheidungsproblemvariante)
  - Eingabe: Ungerichteter Graph $G$ mit natürlichen Zahlen als Kantenmarkierungen und eine Zahl $k$.

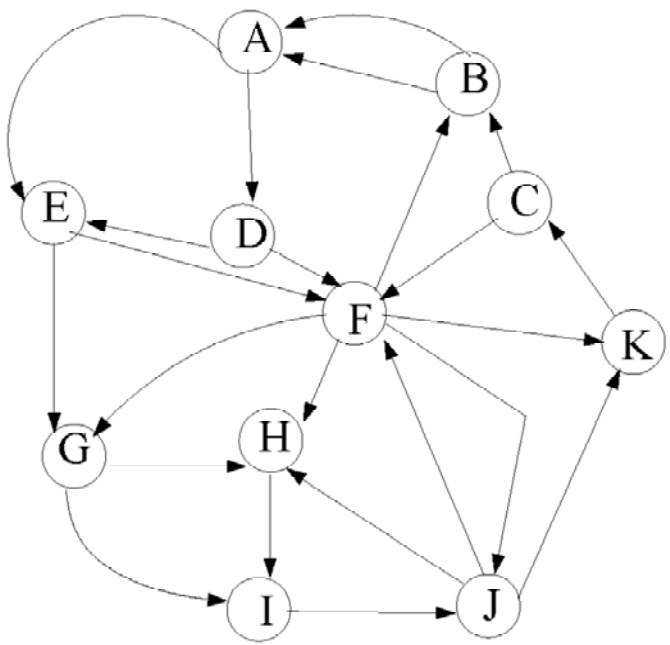
    Beispiel für $G$:



# DHC

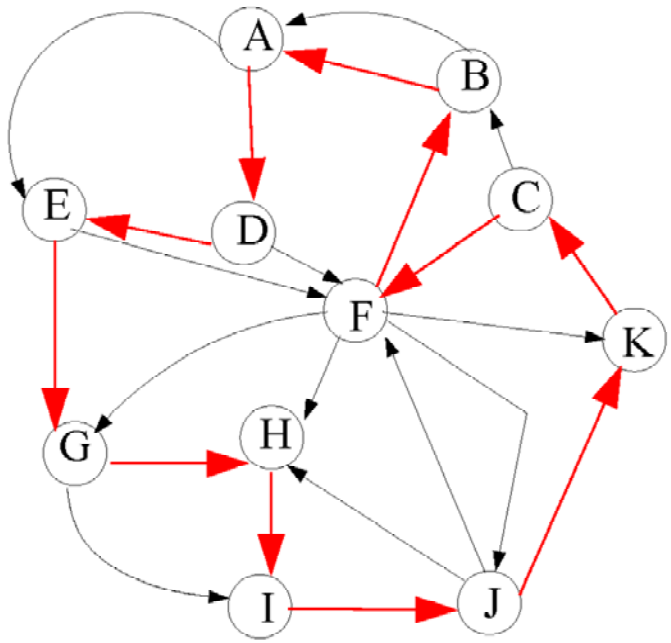
- DHC (Directed Hamiltonian Circuit Problem)
  - Eingabe: Ein gerichteter Graph $G$.

    Beispiel:



- Ausgabe: T gdw $G$ einen Rundweg (in Pfeilrichtung) besitzt, der jeden Knoten genau einmal besucht.

  Beispiel:




Beobachtung: DHC ist in $NP$.

# Probleme in $NP$

- 3SAT (Spezialfall von SAT)
  - Eingabe: Aussagenlogische Formel $f$ der Form

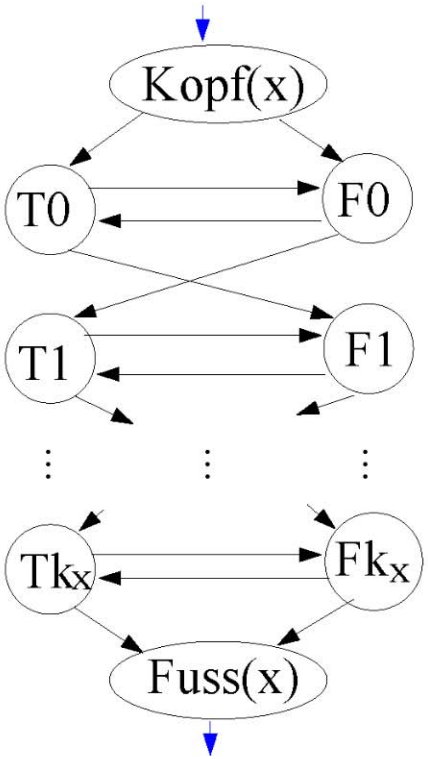
$$c_1 \wedge c_2 \wedge \cdots \wedge c_n,$$

    so dass
    - $c_i$: Klausel der Form $L_1 \vee L_2 \vee L_3$,
    - $L_j$: Literal der Form $x$ oder $\neg x$
    - $x$: Variable.
  - Ausgabe: T gdw eine Belegung der Variablen in $f$ mit 1 oder 0 existiert, so dass $f$ gilt.

# Reduktion von 3SAT auf DHC

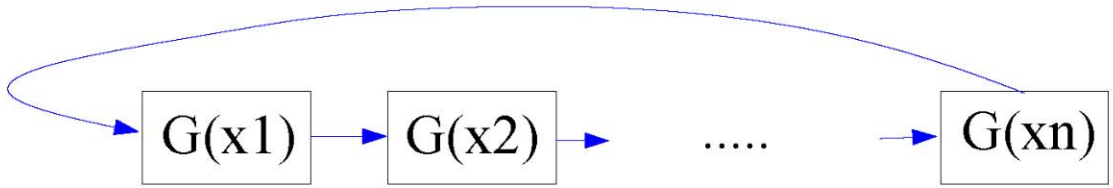
- $f$: Eingabeformel für 3SAT mit den Variablen $\{x_1, \ldots x_n\}$

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




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

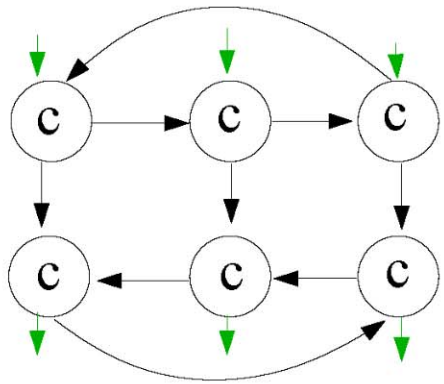
2. Verbinde diese Graphen miteinander zu $G'$:




**Beobachtung**

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

3. Konstruiere für jede Klausel $c$ in $f$ den Klausel-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 jeden Knoten aus $G(c)$ genau einmal besuchen will ($i = 1, 2, 3$).

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

$$\text{Sei } c = L_1 \vee L_2 \vee L_3.$$

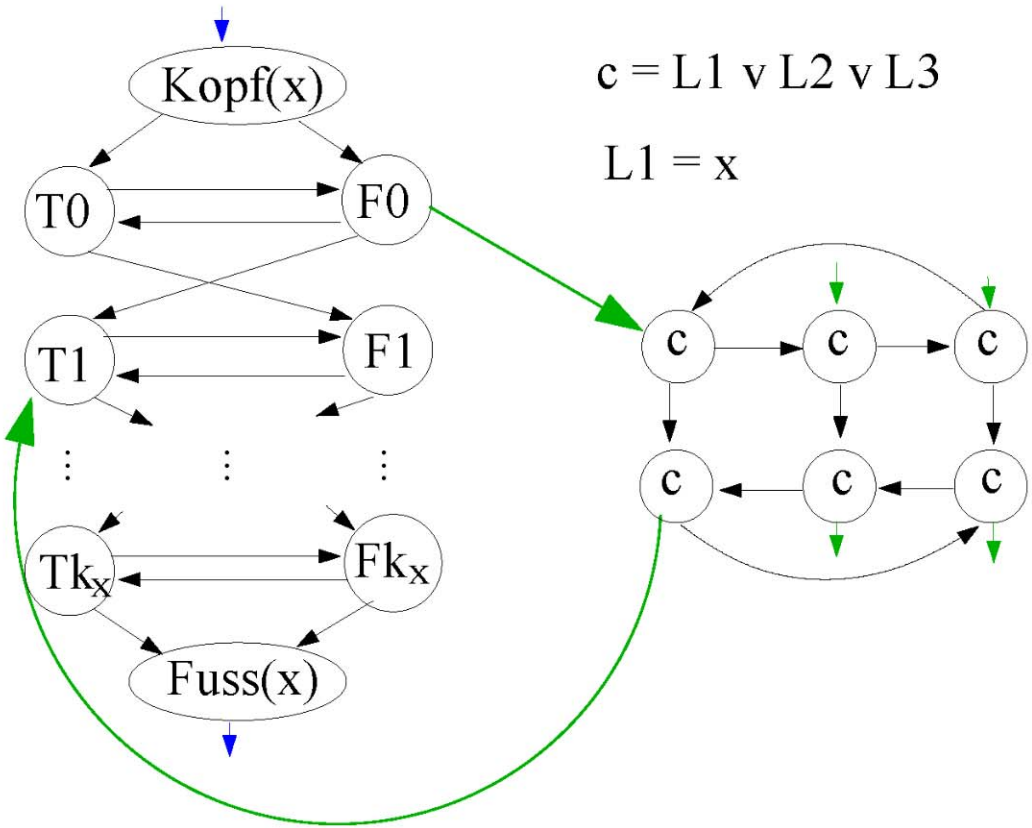
Falls $L_i = x$:

(a) Wähle ein $F_j$ aus $G(x)$, für das gilt:

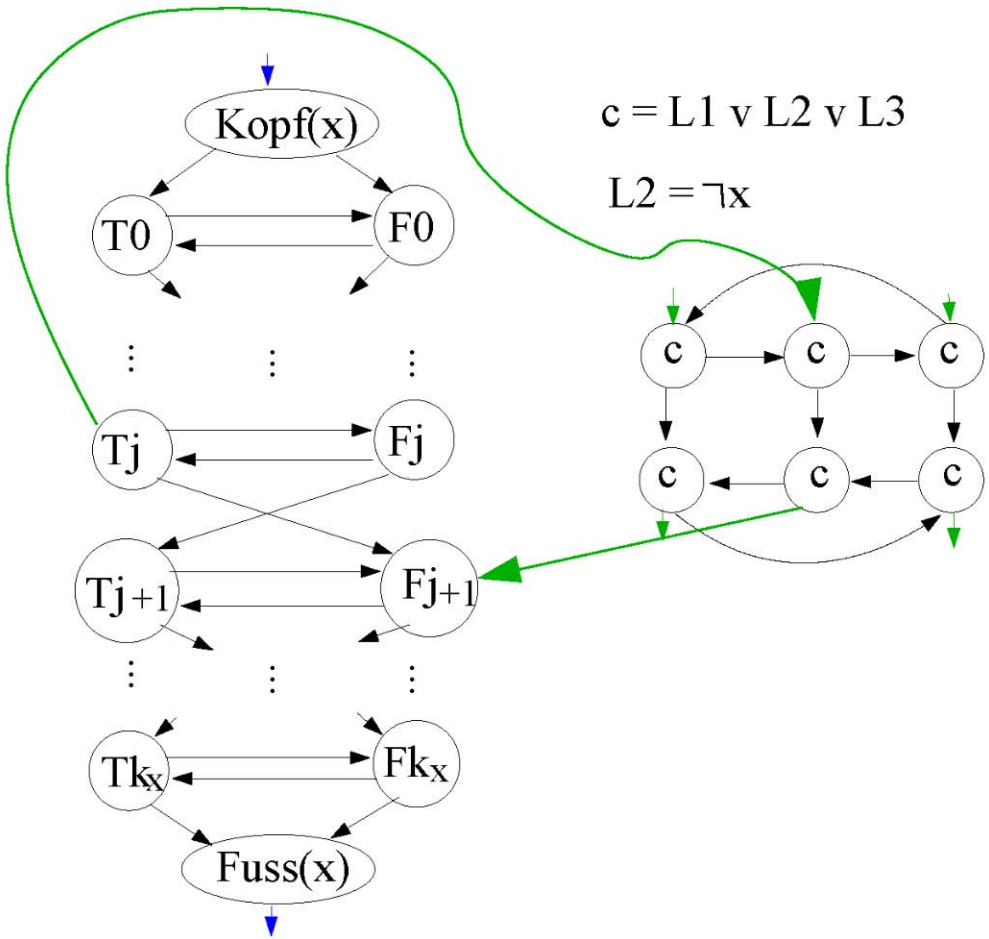
- ▷ Es gibt keine Kante zwischen $F_j$ und einem Klausel-Graphen.
- ▷ $F_j$ ist nicht der unterste $F$-Knoten in $G(x)$.

(b) Ziehe eine Kante von $F_j$ zu dem $i$-ten oberen Knoten in $G(c)$ und eine Kante von dem $i$-ten unteren Knoten in $G(c)$ zu $T_{j+1}$ in $G(x)$.

# Beispiel ($L_1 = x$)



# Beispiel ($L_2 = \neg x$)



**Beobachtung**

Der Hamiltonsche Kreis in $G'$ für eine Belegung der Variablen kann einen Umweg über $G(c)$ machen gdw $c$ $T$ ist.

**Beobachtung**

- DHC 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

# Wie weit reicht P und was kommt dahinter?

$$P \stackrel{?}{=} NP \stackrel{?}{=} PSPACE = NPSPACE \neq EXP \quad \ldots \text{ Berechenbares}$$

- $NP$: Erfüllbarkeitsproblem, TSP u.v.a.m.
- $PSPACE$: Wortproblem monotoner Sprachen
- $EXP$: Drachenkurve, 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 $p \in P$.