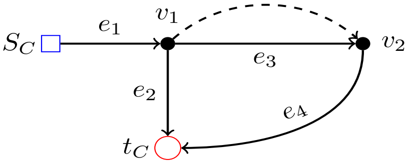
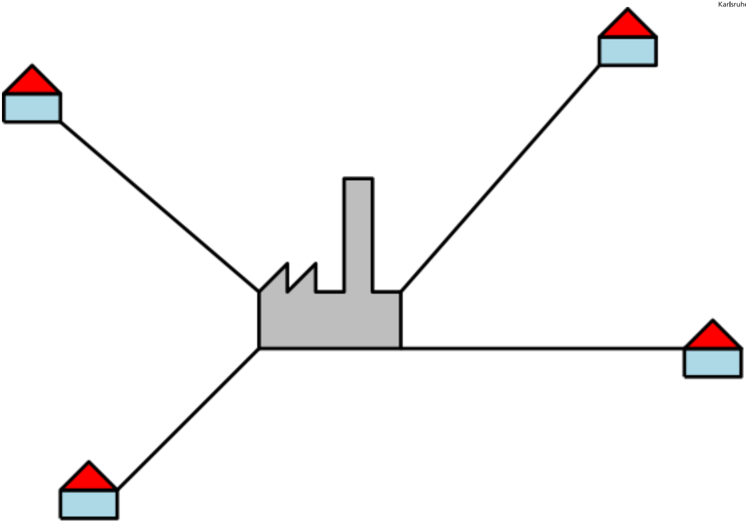
Karlsruher Institut für Technologie

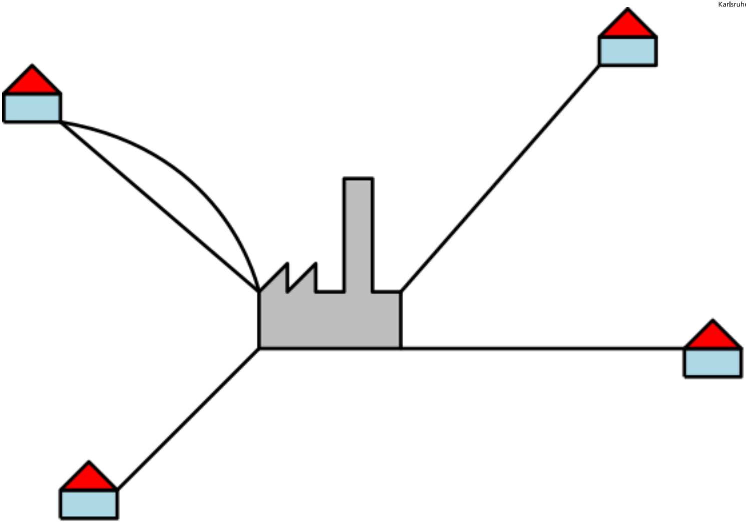
Complexity of Transmission Network Expansion Planning

Matthias Schimek
Betreuer: Matthias Wolf

Institut für Theoretische Informatik







Gliederung

TNEP–Definition

Was wird bewiesen?

Reduktion

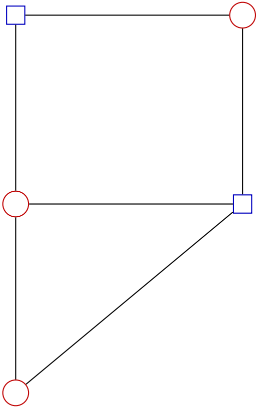
Beweis

Berechnung realer Instanzen

Modellierung

Elektr. Netzwerk/Leistungsfluss – Einführung

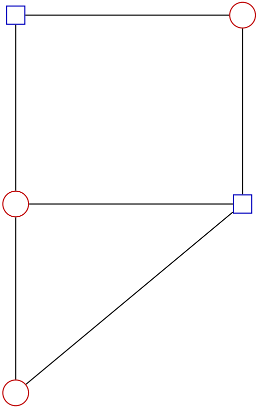
- Graph $G = (V, E)$:
  - Knotentyp □: Erzeuger
  - Knotentyp ○: Verbraucher



Modellierung

Elektr. Netzwerk/Leistungsfluss – Einführung

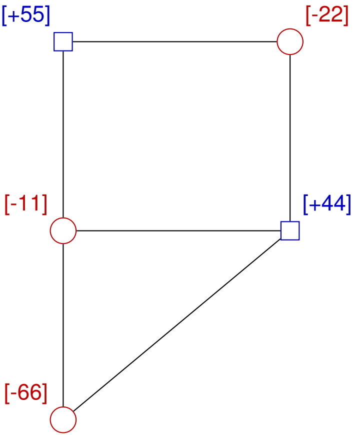
- Graph $G = (V, E)$:
  - Knotentyp $\square$: Erzeuger
  - Knotentyp $\bigcirc$: Verbraucher
  - Knotengewichtsfunktion
    $b : V \to \mathbb{R}$ Bereitstellung
    $b(\square) > 0$ Erzeuger
    $b(\bigcirc) \leq 0$ Verbraucher
  - $\sum_v b(v) = 0$



Modellierung

Elektr. Netzwerk/Leistungsfluss – Einführung

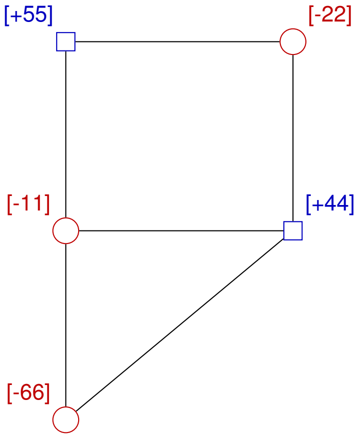
- Graph $G = (V, E)$:
  - Knotentyp $\square$: Erzeuger
  -
    Knotentyp $\bigcirc$: Verbraucher
  - Knotengewichtsfunktion
    $b : V \to \mathbb{R}$ Bereitstellung
    $b(\square) > 0$ Erzeuger
    $b(\bigcirc) \leq 0$ Verbraucher
  - $\sum_{v} b(v) = 0$



Modellierung

Elektr. Netzwerk/Leistungsfluss – Einführung

- Graph $G = (V, E)$:
  - Kantenkapazität
    $c : E \to \mathbb{R}_{\geq 0}$ max. Fluss

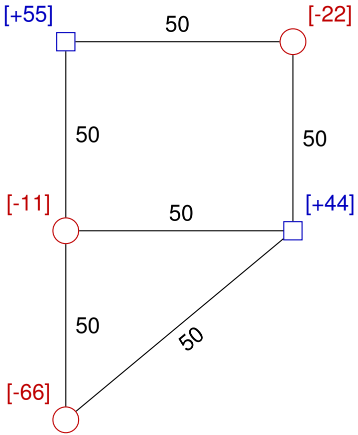


Modellierung

Elektr. Netzwerk/Leistungsfluss – Einführung

- Graph $G = (V, E)$:
  - Kantenkapazität
    $c : E \to \mathbb{R}_{\geq 0}$

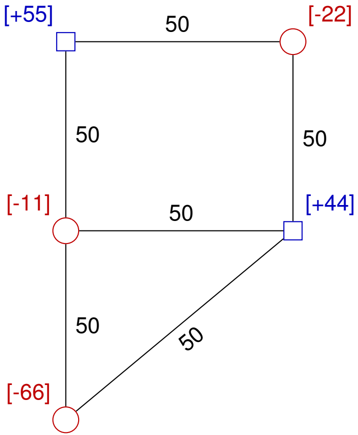
max. Fluss



Modellierung

Elektr. Netzwerk/Leistungsfluss – Einführung

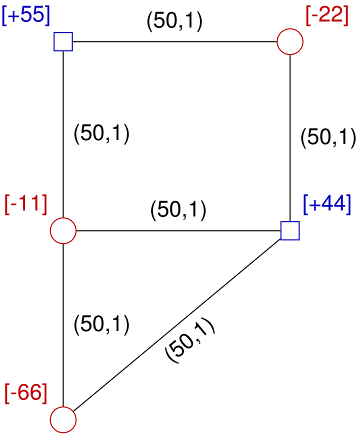
- Graph $G = (V, E)$:
  - Kantenkapazität $c : E \to \mathbb{R}_{\geq 0}$ max. Fluss
  - Kantenleitwert $s : E \to \mathbb{R}_{\geq 0}$ Suszeptanz



Modellierung

Elektr. Netzwerk/Leistungsfluss – Einführung

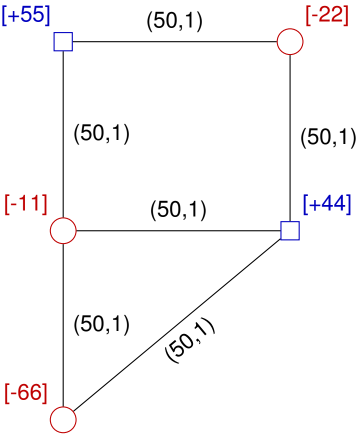
- Graph $G = (V, E)$:
  - Kantenkapazität $c : E \to \mathbb{R}_{\geq 0}$ max. Fluss
  - Kantenleitwert $s : E \to \mathbb{R}_{\geq 0}$ Suszeptanz



Modellierung

Elektr. Netzwerk/Leistungsfluss – Einführung

- Graph $G = (V, E)$:
  - Kantenkapazität
    $c : E \rightarrow \mathbb{R}_{\geq 0}$ max. Fluss
  - Kantenleitwert
    $s : E \rightarrow \mathbb{R}_{\geq 0}$ Suszeptanz
  - Phasenwinkel pro Knoten
    $\delta : V \rightarrow \mathbb{R}$ Phasenwinkel

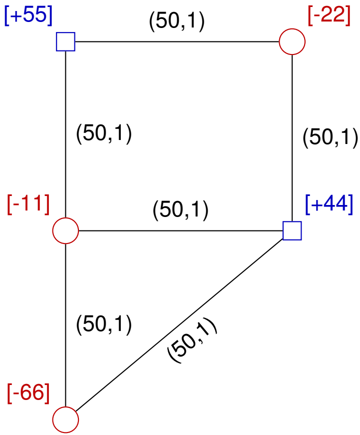


Modellierung

Elektr. Netzwerk/Leistungsfluss – Einführung

- Graph $G = (V, E, c, s, b)$:
- **Problem:**
  Finde Leistungsfluss
  $p : \boldsymbol{E} \to \mathbb{R}$

Fluss

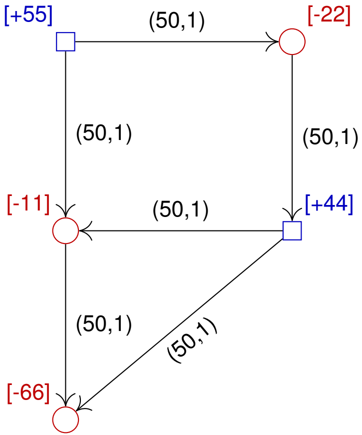


Modellierung

Elektr. Netzwerk/Leistungsfluss – Einführung

- Graph $G = (V, E, c, s, b)$:
- **Problem:**
  Finde Leistungsfluss
  $p : \boldsymbol{E} \to \mathbb{R}$

Fluss



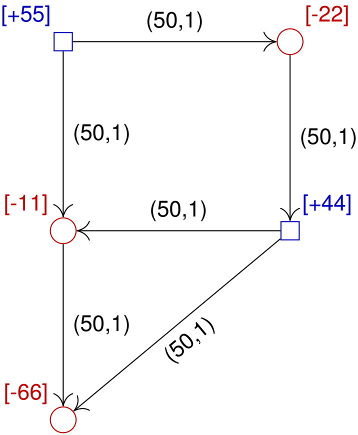
Modellierung

Elektr. Netzwerk/Leistungsfluss – Einführung

- Graph $G = (V, E, c, s, b)$:
- **Problem:**
  Finde Leistungsfluss
  $p : \boldsymbol{E} \to \mathbb{R}$ Fluss

Bedingungen:

1. $|p_{ij}| \leq c_{ij}, \quad i, j \in V$
2. $$b(v) + \sum_{iv} p_{iv} - \sum_{vi} p_{vi} = 0$$ (KCL)

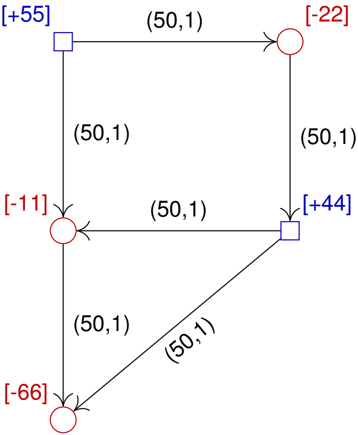


Modellierung

Elektr. Netzwerk/Leistungsfluss – Einführung

- Graph $G = (V, E, c, s, b)$:
- Unterschied zu klassischem Flussproblem:

$$p_{ij} = (\delta_i - \delta_j) \cdot s_{ij}$$



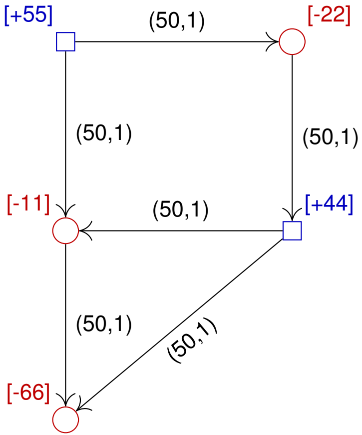
Modellierung

Elektr. Netzwerk/Leistungsfluss – Einführung

- Graph $G = (V, E, c, s, b)$:
- Unterschied zu klassischem Flussproblem:

$$p_{ij} = (\delta_i - \delta_j) \cdot s_{ij}$$

- **Problem**:
  Finde Phasenwinkel δ so, dass
  p Bedinungen (1) - (2) erfüllt



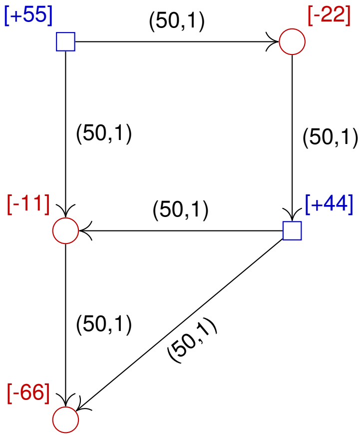
Modellierung

Elektr. Netzwerk/Leistungsfluss – Einführung

- Graph $G = (V, E, c, s, b)$:
- Unterschied zu klassischem Flussproblem:

$$p_{ij} = (\delta_i - \delta_j) \cdot s_{ij}$$

- **Problem**:
  Finde Phasenwinkel δ so, dass
  p Bedinungen (1) - (2) erfüllt



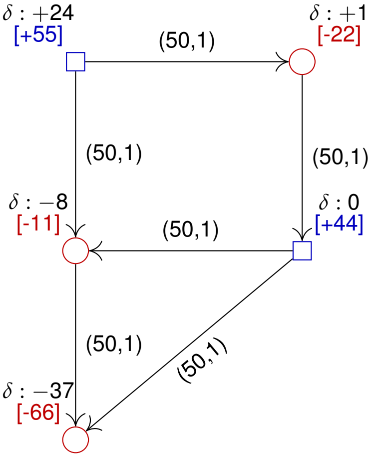
Modellierung

Elektr. Netzwerk/Leistungsfluss – Einführung

- Graph $G = (V, E, c, s, b)$:
- Unterschied zu klassischem Flussproblem:

$$p_{ij} = (\delta_i - \delta_j) \cdot s_{ij}$$

- **Problem**: Finde Phasenwinkel δ so, dass p Bedinungen (1) - (2) erfüllt



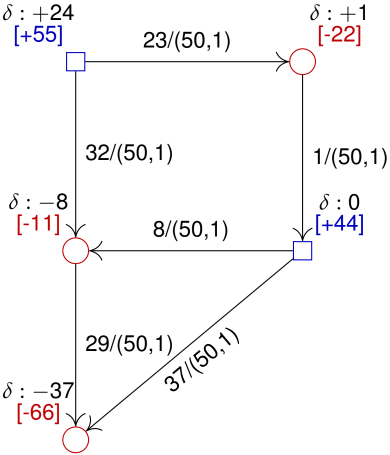
Modellierung

Elektr. Netzwerk/Leistungsfluss – Einführung

- Graph $G = (V, E, c, s, b)$:
- Unterschied zu klassischem Flussproblem:

$$p_{ij} = (\delta_i - \delta_j) \cdot s_{ij}$$

- **Problem**: Finde Phasenwinkel δ so, dass p Bedinungen (1) - (2) erfüllt



Fakultät für Informatik
Institut für Theoretische Informatik

Modellierung

Elektrisches Netzwerk - Übersicht

- Elektrisches Netzwerk:
  - Graph $G = (V, E)$
  - Bereitstellung b
  - Kapazitäten c
  - Suszeptanz s

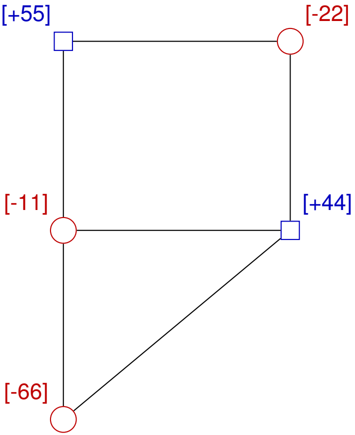
- Leistungsfluss im Netzwerk:
  - Leistungsfluss pro Kante: $p_{ij} = (\delta_i - \delta_j) \cdot s_{ij}$
  - gültige Phasenwinkel: induzierter Leistungsfluss erfüllt KCL
  - gültige Phasenwinkel δ Lösung von:

$$L \cdot \delta = b$$

L - Laplace Matrix $\in \mathbb{R}^{n \times n}$

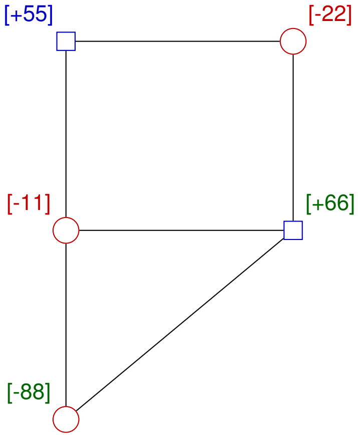
TNEP – Problem
Einführung (1)

- TNEP: Verbrauch ändert sich



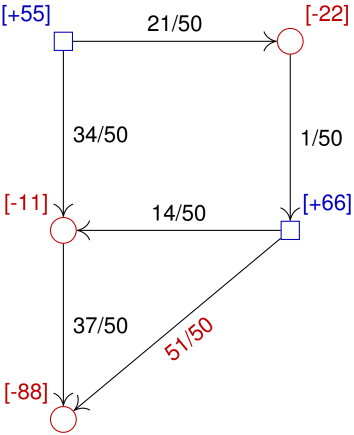
TNEP – Problem
Einführung (1)

- TNEP: Verbrauch ändert sich



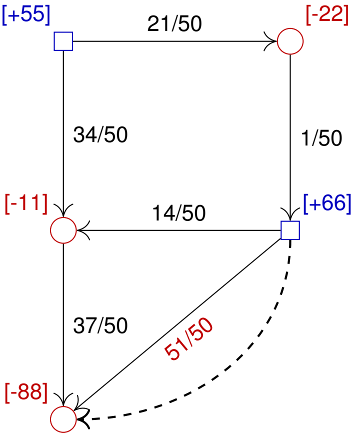
TNEP – Problem
Einführung (1)

- TNEP: Verbrauch ändert sich
- neue Leistungsflüsse
- Kapazitätsschranken verletzt



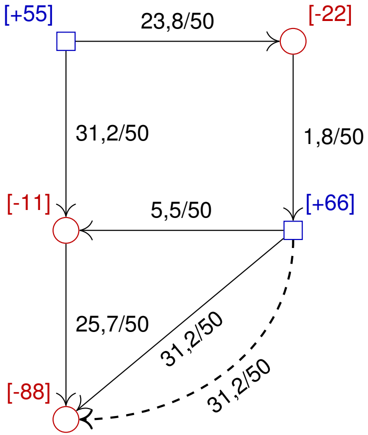
TNEP – Problem
Einführung (1)

- TNEP: Verbrauch ändert sich
- neue Leistungsflüsse
- Kapazitätsschranken verletzt
- Lösung: neue Kante



TNEP – Problem
Einführung (1)

- TNEP: Verbrauch ändert sich
- neue Leistungsflüsse
- Kapazitätsschranken verletzt
- Lösung: neue Kante
- neuer Fluss p (gerundet)
  $\Rightarrow$ Schranken eingehalten



TNEP–Problem

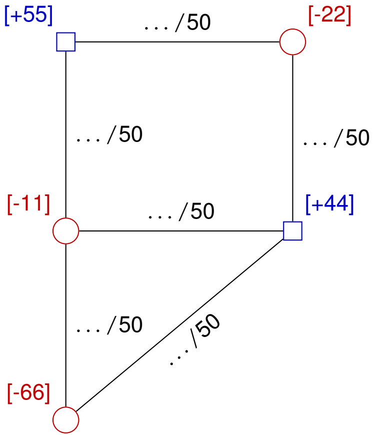
Einführung (2)

- Elektrisches Netzwerk $G = (V, E, s, c, b)$:
- **Frage:** Existieren gültige Phasenwinkel, sodass ind. Leistung p Kapazitätsschranken c einhält?
- Falls nein: Füge Kanten hinzu $\Rightarrow$ Schranken eingehalten

TNEP–Problem

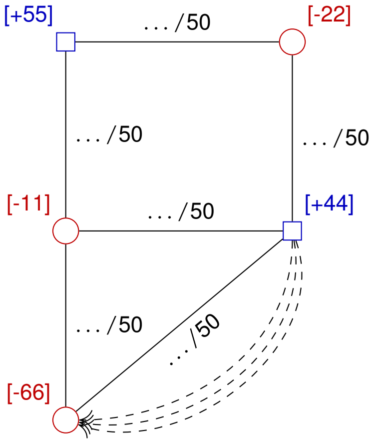
Einführung (3)

- Elektrisches Netzwerk:
  - Graph $G = (V, E)$
  - Bedarfsvektor b
  - Kapazitäten c
  - Suszeptanz s (überall 1)



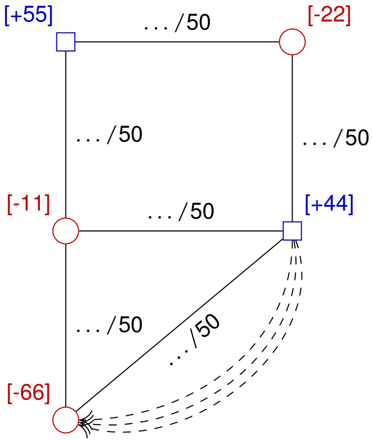
TNEP–Problem
Einführung (3)

- Elektrisches Netzwerk:
  - Graph $G = (V, E)$
  - Bedarfsvektor b
  - Kapazitäten c
  - Suszeptanz s (überall 1)
- Zusätzlich:
  - Für jedes Knotenpaar: #hinzufügbarer Kanten



TNEP–Problem
Einführung (3)

- Elektrisches Netzwerk:
  - Graph $G = (V, E)$
  - Bedarfsvektor b
  - Kapazitäten c
  - Suszeptanz s (überall 1)
- Zusätzlich:
  - Für jede Kante:
    #hinzufügbarer Kanten
  - Kosten w_{ij} pro hinzufügbarer Kante



TNEP (formal)

TNEP - Instanz

TNEP Instanz ist Tupel $I = (V, s, c, b, N)$

$$N = (n^{(0)}, \hat{n}, w)$$

- $n_{ij}^{(0)}$ - initiale #Kanten zwischen i, j
- $\hat{n}_{ij}$ - maximale # hinzufügbarer Kanten zwischen i, j
- w_{ij} - Kosten pro hinzugefügter Kante zwischen i, j

TNEP (formal)

TNEP - Kantenausbau

Gegeben eine **TNEP**-Instanz $I = (V, s, c, b, N)$.
Funktion $n : V^2 \to \mathbb{N}_0$ ist **TNEP-Kantenausbau** für I gdw.:

$$0 \leq n_{ij} \leq \hat{n}_{ij}$$

TNEP (formal)

Karlsruher Institut für Technologie

TNEP - Kantenausbau

Gegeben eine **TNEP**-Instanz $I = (V, s, c, b, N)$.
Funktion $n : V^2 \to \mathbb{N}_0$ ist **TNEP-Kantenausbau** für I gdw.:

$$0 \leq n_{ij} \leq \hat{n}_{ij}$$

Die Kosten eines **TNEP**-Kantenausbaus n:

$$cost_S(n) = \sum_{\{i,j\} \in V^2} (n_{ij}) \cdot w_{ij}$$

TNEP (formal)

TNEP - Lösung

- **TNEP**-Kantenausbau n ist **TNEP-Lösung** für Instanz I gdw.:
  - für gültige Phasenwinkel δ

$$|(\delta_i - \delta_j)| \cdot (n_{ij}^{(0)} + n_{ij}) \cdot s_{ij} \leq (n_{ij}^{(0)} + n_{ij}) \cdot c_{ij}$$

$\Rightarrow$ Schranke für Leistungsfluss pro Kante wird nicht überschritten

Gliederung

TNEP–Definition

Was wird bewiesen?

Reduktion

Beweis

Berechnung realer Instanzen

NP-Vollständigkeit von TNEP

- Es gibt zwei "natürliche" Probleme für **TNEP**

  1. **Entscheidungsproblem:** Gibt es **TNEP**-Lösung n für I mit $cost_I(n) \leq k$?
  2. **Optimierungsproblem:** Welche **TNEP**-Lösung hat minimale Kosten?

- Wir betrachten Entscheidungsproblem
- Aussage: **TNEP** ist **NP**-vollständig

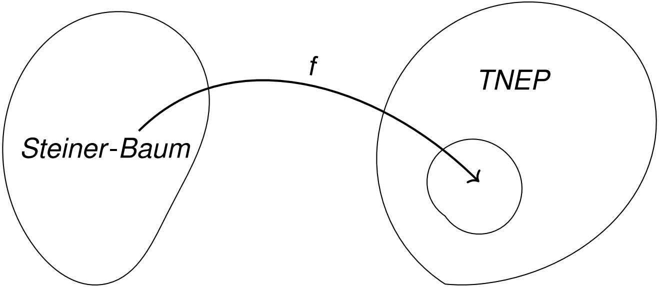
Schritte für NP-Vollständigkeit

1. Zeige **TNEP** $\in$ **NP**
2. Jedes Problem in **NP** kann auf **TNEP** reduziert werden

   Gezeigt durch: Reduktion von **NP**-vollständigem Problem auf **TNEP**

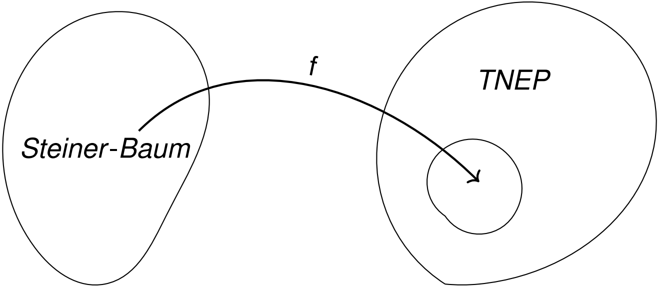
Stand vor Veröffentlichung

- **TNEP** ist **NP**-vollständig
- Reduktion *f* von Steiner-Baum-Problem in [1]
- **Problem**: *f* bildet nur auf nicht-verbundene **TNEP**-Szenarien ab



Stand vor Veröffentlichung

- **TNEP** ist **NP**-vollständig
- Reduktion *f* von Steiner-Baum-Problem in [1]
- **Problem**: In Realität, Netzwerk meist verbunden



Beitrag des Papers

Karlsruher Institut für Technologie

- Polynomial-Zeit Reduktion von 3**SAT** auf **TNEP**
- Reduktion mappt auf (schon) **verbundene TNEP**-Instanzen

Beitrag des Papers

- Polynomial-Zeit Reduktion von 3**SAT** auf **TNEP**

- Reduktion mappt auf (schon) **verbundene TNEP**-Instanzen

Wiederholung 3SAT

- Aussagenlogische Formel F in KNF
- (Höchstens) 3 Literale pro Klausel
- Frage: Ist F erfüllbar?

Beitrag des Papers

- Polynomial-Zeit Reduktion von 3**SAT** auf **TNEP**
- Reduktion mappt auf (schon) **verbundene TNEP**-Instanzen

Wiederholung 3SAT

- Beispiel:
- $(\neg x_1 \lor x_2 \lor x_3) \land (x_1 \lor \neg x_2 \lor \neg x_3) \land (x_1 \lor x_2 \lor \neg x_3)$

Gliederung

TNEP–Definition

Was wird bewiesen?

Reduktion

Beweis

Berechnung realer Instanzen

Reduktion

Reduktion

Funktion $f : 3\textbf{SAT} \to \textbf{TNEP}$

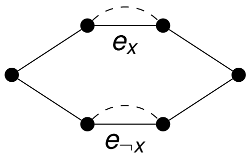
- f in Polynomialzeit berechenbar
- I ist erfüllbare 3**SAT** Formel $\Leftrightarrow$ $f(I)$ **TNEP**-Instanz mit Lösung

Vorgehen:

- Erfüllung 3**SAT**-Formel
  $\Rightarrow$ Einhalten max. Leistungsfluss
- Setzen von Variable (wahr/falsch)
  $\Rightarrow$ Eine von zwei Kanten hinzufügen

Variablen–Zelle

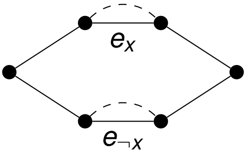
- Für jede Variable x: Variablen–Zelle
- Hinzufügen von Kante $\leftrightsquigarrow$ setzen von Literal
- **Sicherstellen**: Genau eine zusätzliche Kante



[2]

Variablen–Zelle

- Für jede Variable x:
  Variablen–Zelle
- Hinzufügen von Kante
  $\leftrightsquigarrow$ setzen von Literal
- Kantenausbau
  SAT-konsistent $\Leftrightarrow$ genau eine Kante pro Zelle hinzugefügt



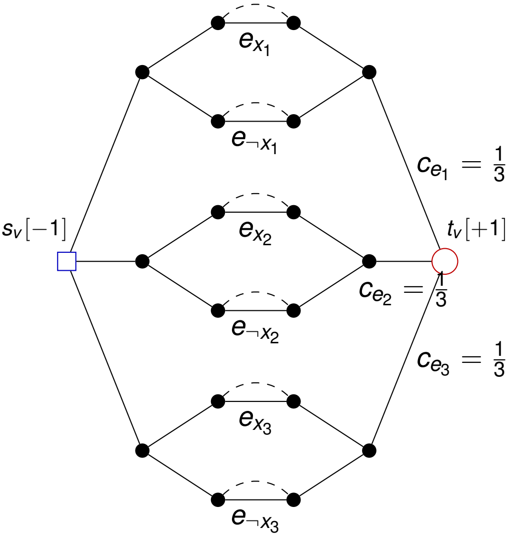
[2]

Variablen–Zelle: Symmetrie

- Formel F: m Klauseln, k Literale
- Für Kanten $e_1, \ldots, e_k$:

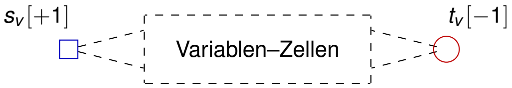
$$c_{e_1} = \cdots = c_{e_k} = \frac{1}{k}$$

- **Symmetriefilter:** Alle Variablen–Zellen gleiche #Kanten



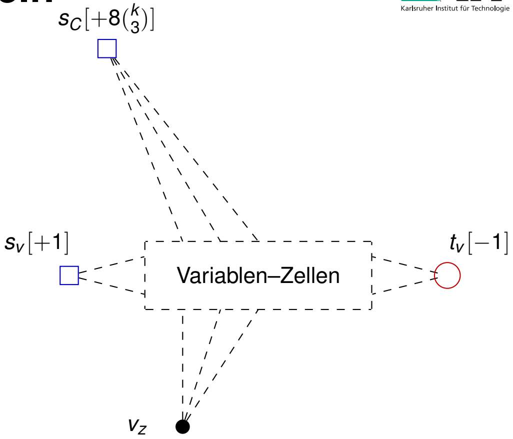
[2]

Modellierung Klauseln



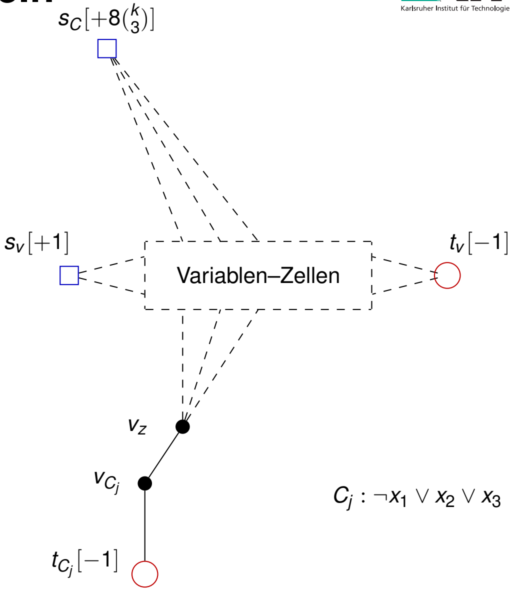
Modellierung Klauseln

- Füge Knoten hinzu:
  - global:
    S_C mit $b(S_C) = 8\binom{k}{3}$
    v_z



Modellierung Klauseln

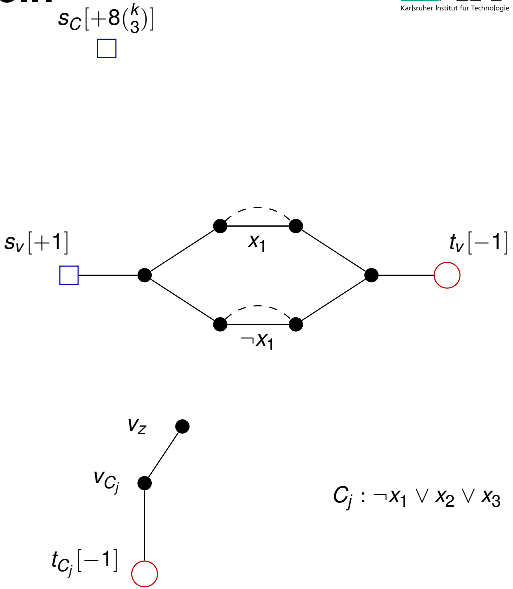
- Füge Knoten hinzu:
  - global:
    S_C mit $b(S_C) = 8\binom{k}{3}$
    v_z
  - Für jede Klausel C_j:
    v_{C_j}
    t_{C_j} mit $b(t_{C_j}) = -1$
- Jetzt: Anschluss an Variablen-Zellen



$C_j : \neg x_1 \vee x_2 \vee x_3$

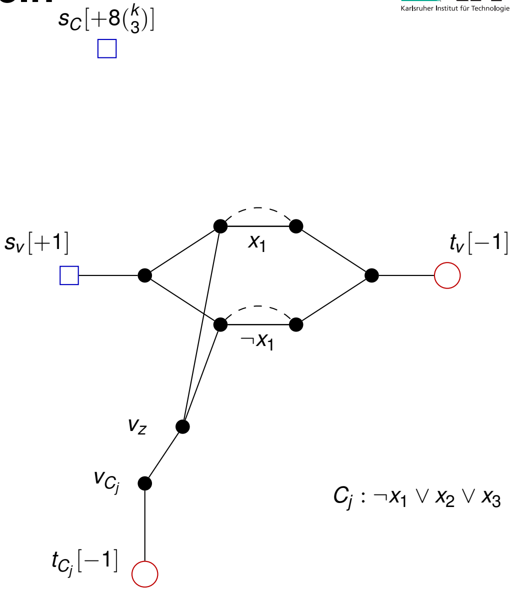
Modellierung Klauseln

- Füge Knoten hinzu:
  - global:
    S_C mit $b(S_C) = 8\binom{k}{3}$
    v_z
  - Für jede Klausel C_j:
    v_{C_j}
    t_{C_j} mit $b(t_{C_j}) = -1$
- Jetzt: Anschluss an Variablen-Zellen



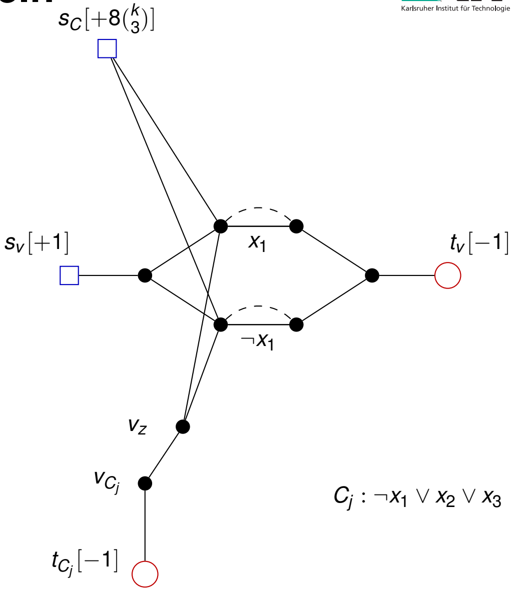
Modellierung Klauseln

- Füge Knoten hinzu:
  - global:
    S_C mit $b(S_C) = 8\binom{k}{3}$
    v_z
  - Für jede Klausel C_j:
    v_{C_j}
    t_{C_j} mit $b(t_{C_j}) = -1$
- Jetzt: Anschluss an Variablen-Zellen



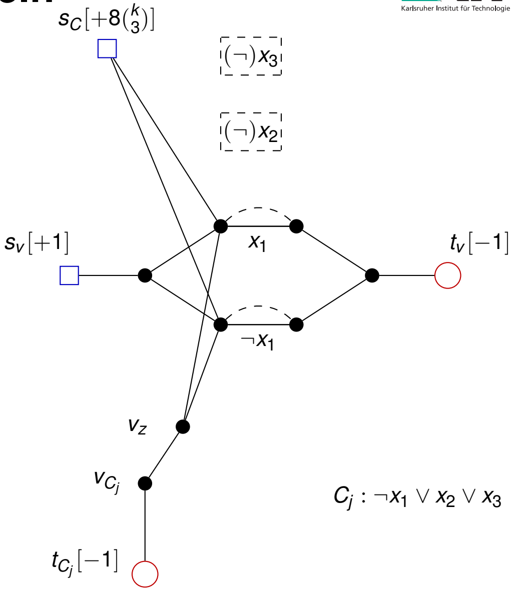
Modellierung Klauseln

- Füge Knoten hinzu:
  - global:
    S_C mit $b(S_C) = 8\binom{k}{3}$
    v_z
  - Für jede Klausel C_j:
    v_{C_j}
    t_{C_j} mit $b(t_{C_j}) = -1$
- Jetzt: Anschluss an Variablen-Zellen



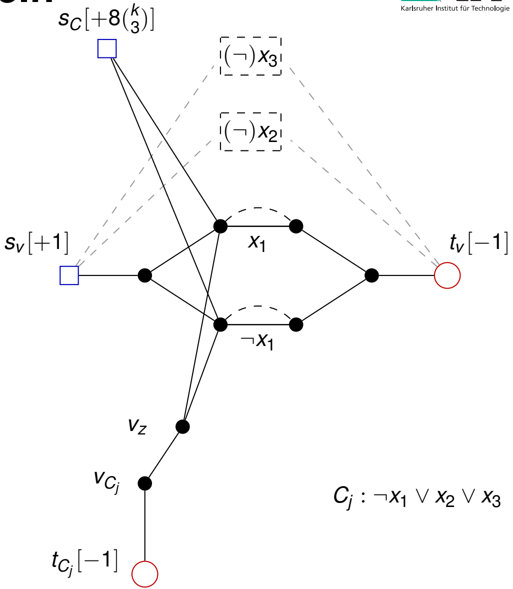
Modellierung Klauseln

- Füge verbleibende Variablen-Zellen hinzu
- Verbinde analog



Modellierung Klauseln

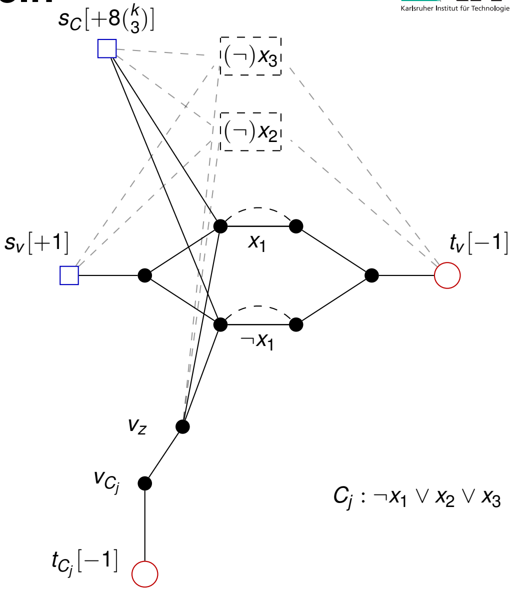
- Füge verbleibende Variablen-Zellen hinzu
- Verbinde analog



$C_j : \neg x_1 \vee x_2 \vee x_3$

Modellierung Klauseln

- Füge verbleibende Variablen-Zellen hinzu
- Verbinde analog

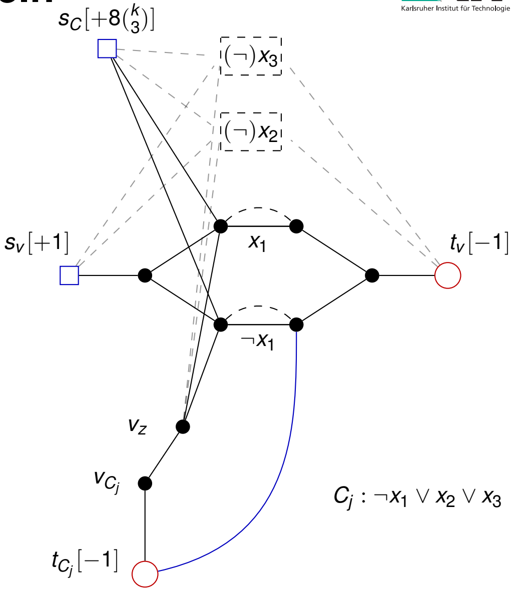


$C_j : \neg x_1 \vee x_2 \vee x_3$

Fakultät für Informatik
Institut für Theoretische Informatik

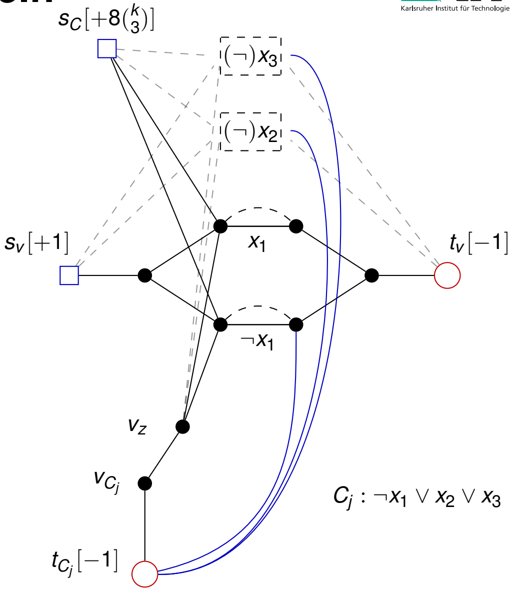
Modellierung Klauseln

- Füge verbleibende Variablen-Zellen hinzu
- Verbinde analog
- Kanten zu Klausel-Block hinzufügen



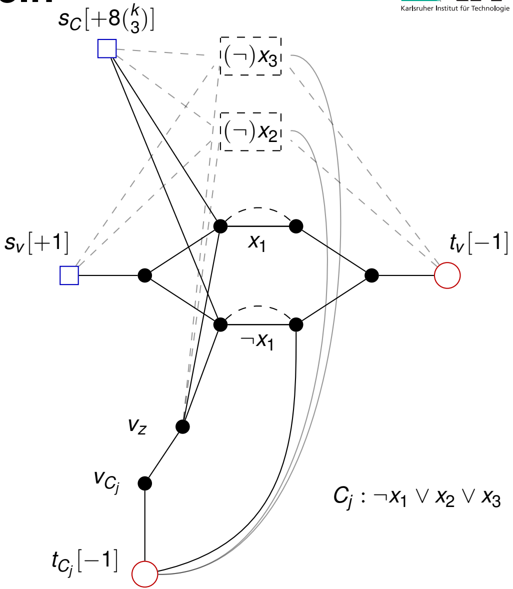
Modellierung Klauseln

- Füge verbleibende Variablen-Zellen hinzu
- Verbinde analog
- Kanten zu Klausel-Block hinzufügen



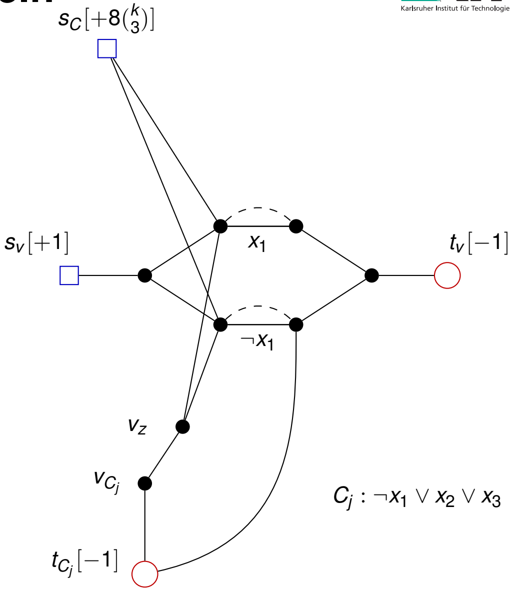
Modellierung Klauseln

- Bisher Topologie der Instanz festgelegt.
- Es fehlen noch:
  - Suszeptanz der Kanten
  - Kapazitäten der Kanten



Modellierung Klauseln

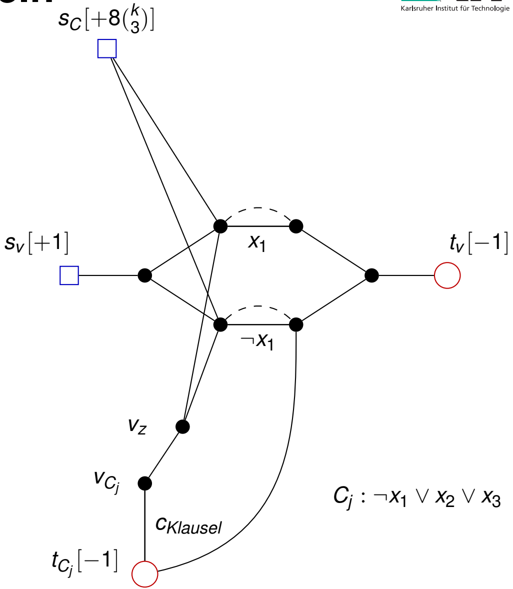
- Bisher Topologie der Instanz festgelegt.
- Es fehlen noch:
  - Suszeptanz der Kanten
  - Kapazitäten der Kanten



Modellierung Klauseln

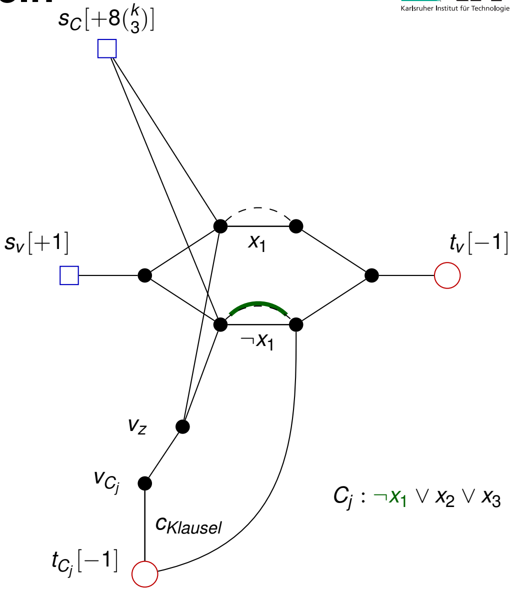
- Bisher Topologie der Instanz festgelegt.
- Es fehlen noch:
  - Suszeptanz der Kanten
  - Kapazitäten der Kanten:

  - Kapazitätschranken nur in Klausel-Blöcken



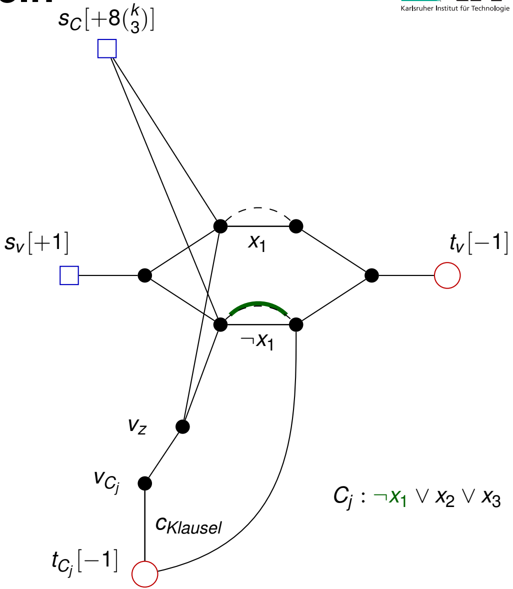
Modellierung Klauseln

- Jeder Klausel-Block mit 3 Variablenzellen verbunden
- Klausel-Block 0/1/2/3-erhöht
- Klausel-Block *i*-erhöht :⇔ Block mit *i* hinzugefügten Kanten verbunden



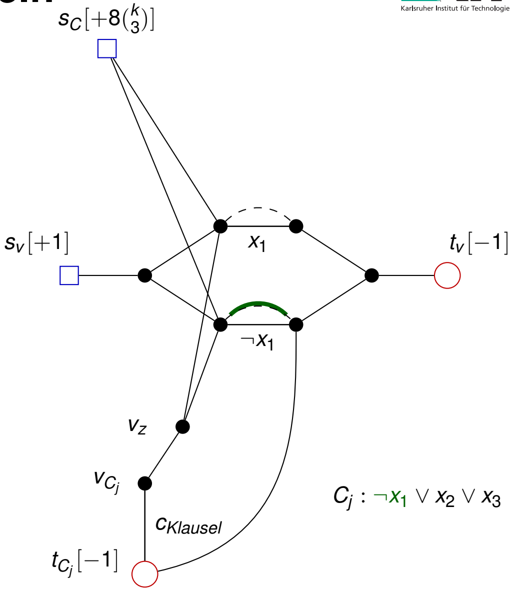
Modellierung Klauseln

- Jeder Klausel-Block mit 3 Variablenzellen verbunden
- Klausel-Block 0/1/2/3-erhöht
- Klausel-Block *i*-erhöht :⇔ Block mit *i* hinzugefügten Kanten verbunden
- $p_0 > p_1 > p_2 > p_3$



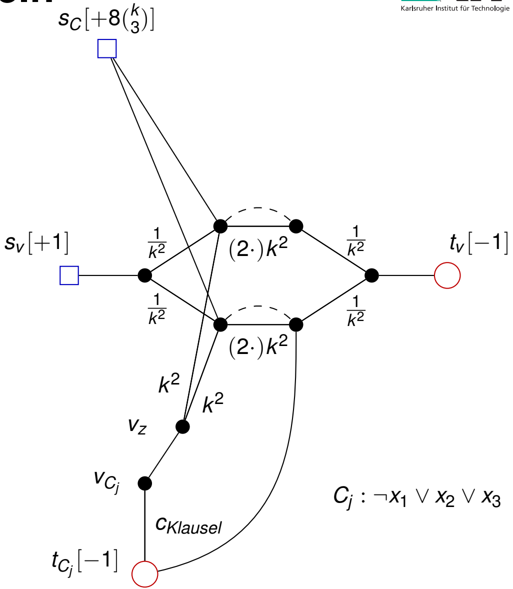
Modellierung Klauseln

- Jeder Klausel-Block mit 3 Variablenzellen verbunden
- Klausel-Block 0/1/2/3-erhöht
- Klausel-Block *i*-erhöht :⇔ Block mit *i* hinzugefügten Kanten verbunden
- Schranke eingehalten ⇔ t_{C_j} mindestens 1-erhöht



Modellierung Klauseln

- Bisher Topologie der Instanz festgelegt.
- Es fehlen noch:
  - Suszeptanz der Kanten
  - keine Beschriftung ⇝ Suszeptanz = 1



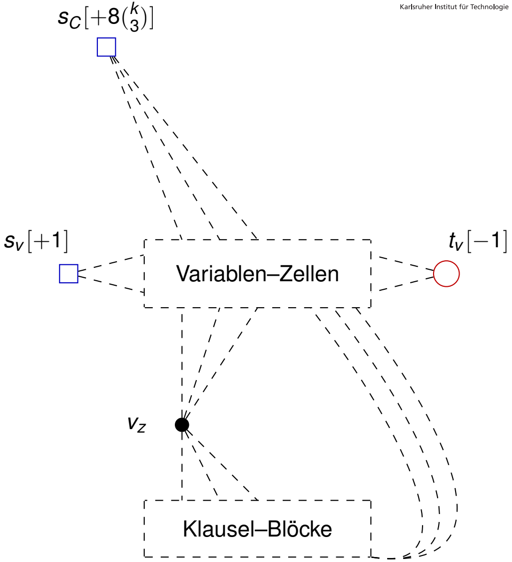
Gesamtübersicht

- Schranken an t_v:
  Nur SAT-konsistenter
  Ausbau erfüllt diese

  $\Rightarrow$ wohldef.
  Variablenbelegung

- In dieser Var.-Belegung:
  nur mind. 1-erhöhte
  Klauseln sind wahr

- Schranke $c_{Klausel}$:
  Nur durch mind.
  1-erhöhte Klauseln erfüllt



Reduktion - Zusammenfassung

- 3**SAT** Formel F mit m Klauseln und k Variablen
- Für jede Variable x_i: konstruiere Variablen-Zelle i
- Anzahl an möglichen Klauseln:

$$|C_{total}(k)| = 2^3 \cdot \binom{k}{3}$$

$\Rightarrow$ Konstruiere für jede mögliche Klausel einen Klausel-Block

- Setze Suszeptanz für Kanten
- Setze Leistungsflussschranken für Kanten
- Einzige Asymmetrie: $c_{Klausel}$ nur für m Klauseln aus F gesetzt

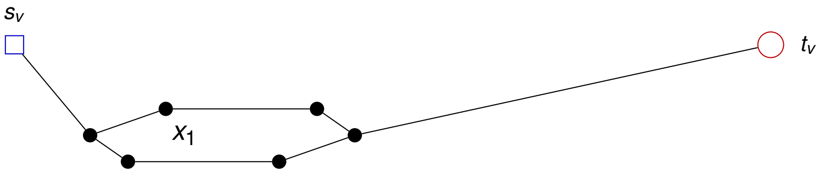
Beispiel

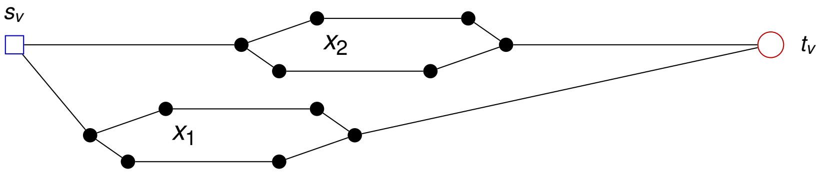
- 3**SAT**-Formel

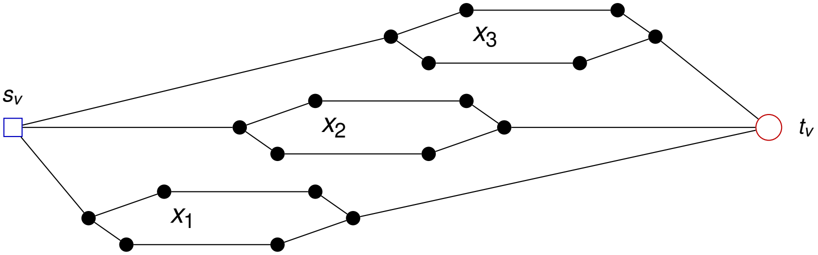
$$F = (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (x_1 \vee \neg x_2 \vee x_3)$$

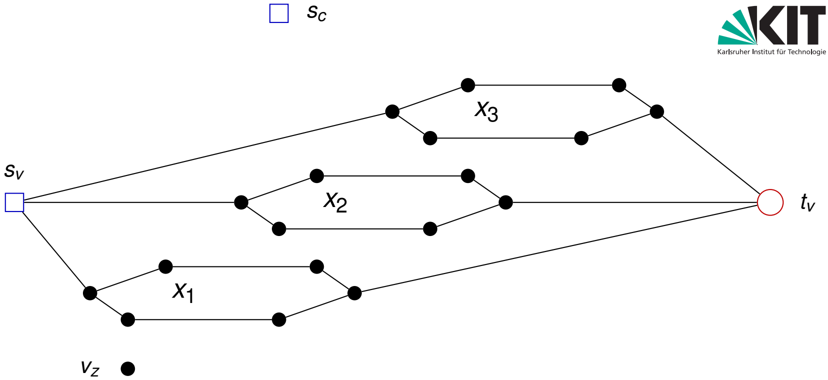
- Auf folgender Folie wird $f(F)$ konstruiert:
- Entnommen aus [2]

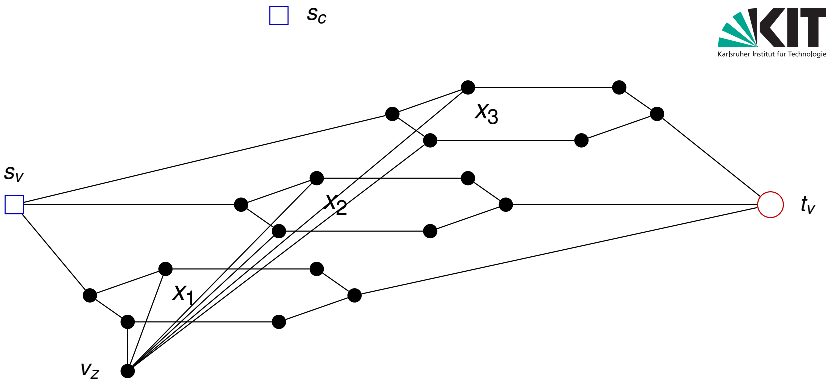

s_V
t_V

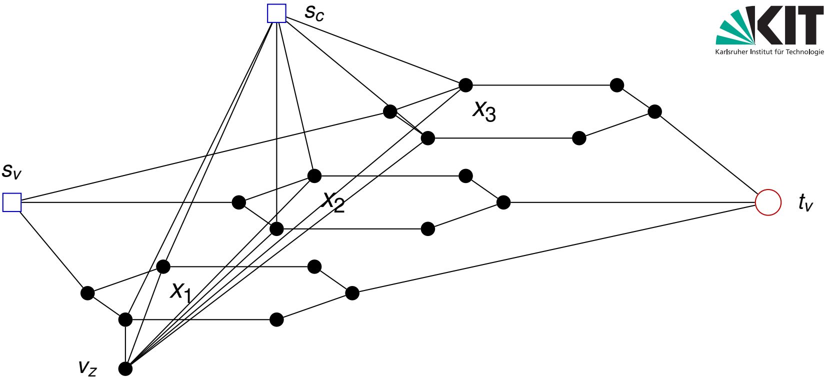

s_v
t_v
x_1

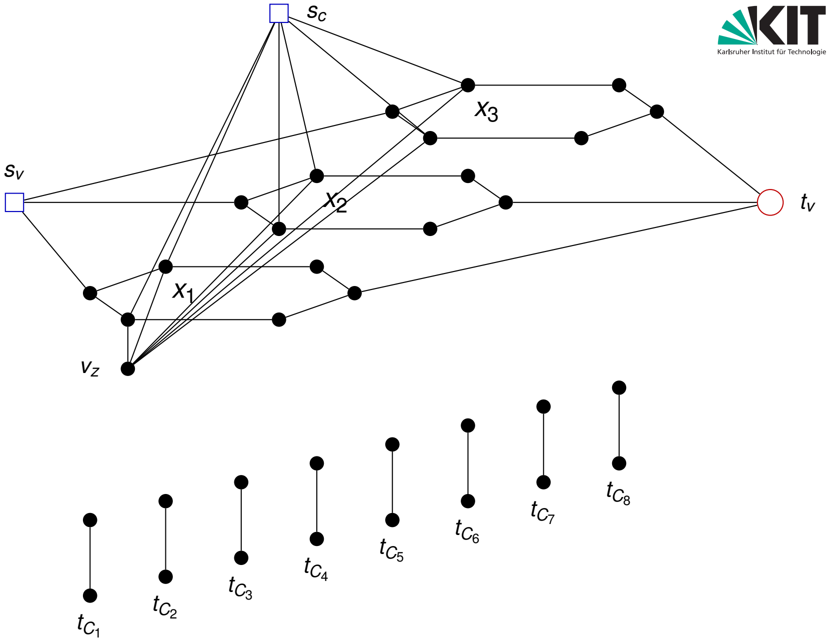

s_v
x_2
t_v
x_1

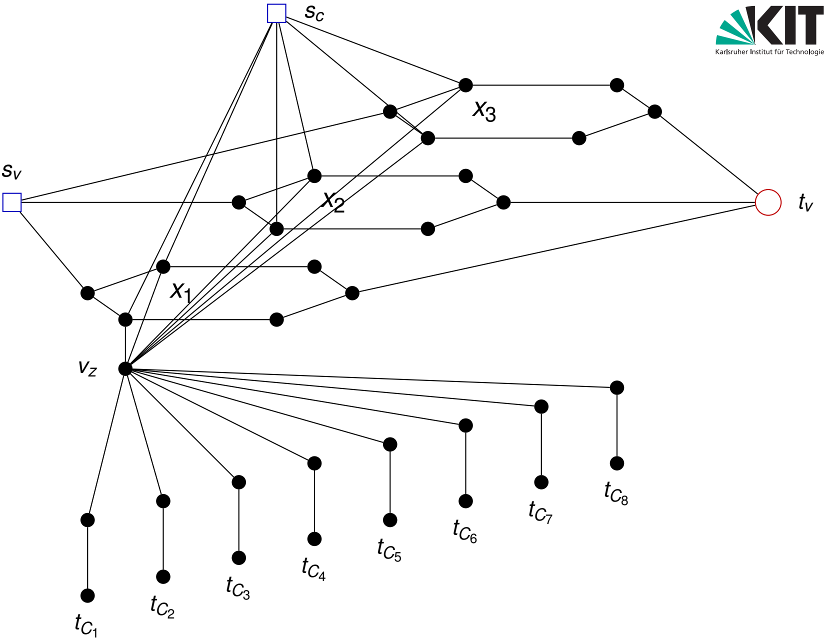

x_3
s_v
x_2
t_v
x_1


s_c
s_v
x_3
x_2
t_v
x_1
v_z


s_c
s_v
x_3
x_2
x_1
t_v
v_z

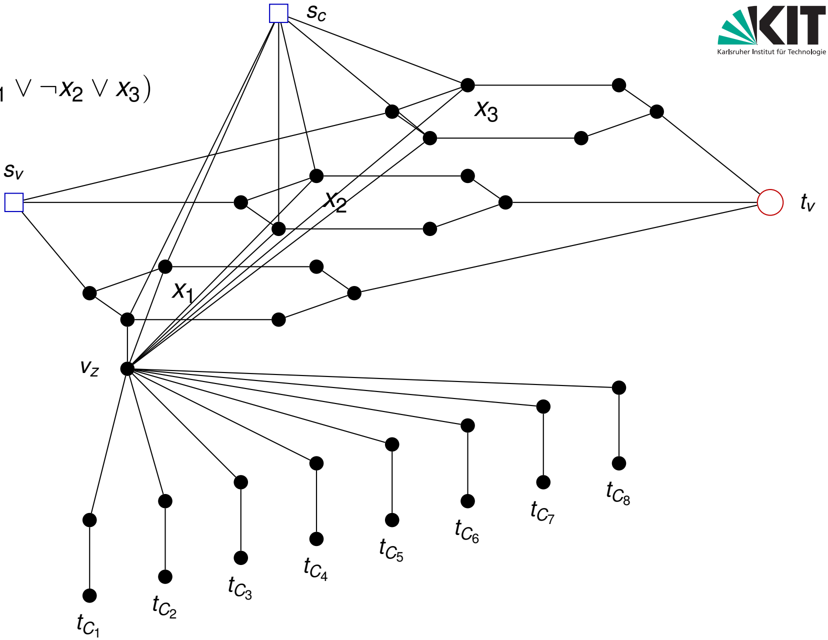

s_c
x_3
s_v
x_2
t_v
x_1
v_z

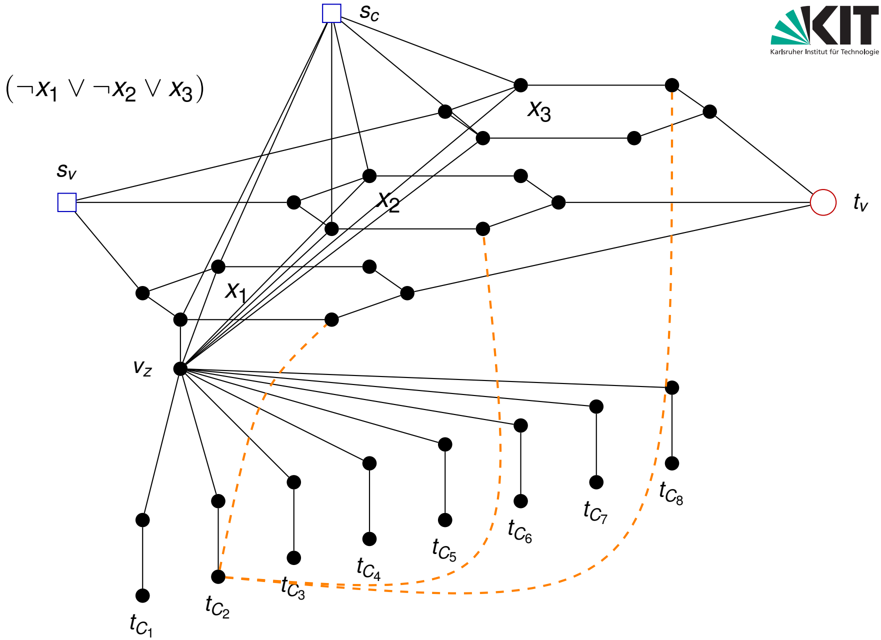

s_c
x_3
s_v
x_2
t_v
x_1
v_z
t_{C_1}
t_{C_2}
t_{C_3}
t_{C_4}
t_{C_5}
t_{C_6}
t_{C_7}
t_{C_8}



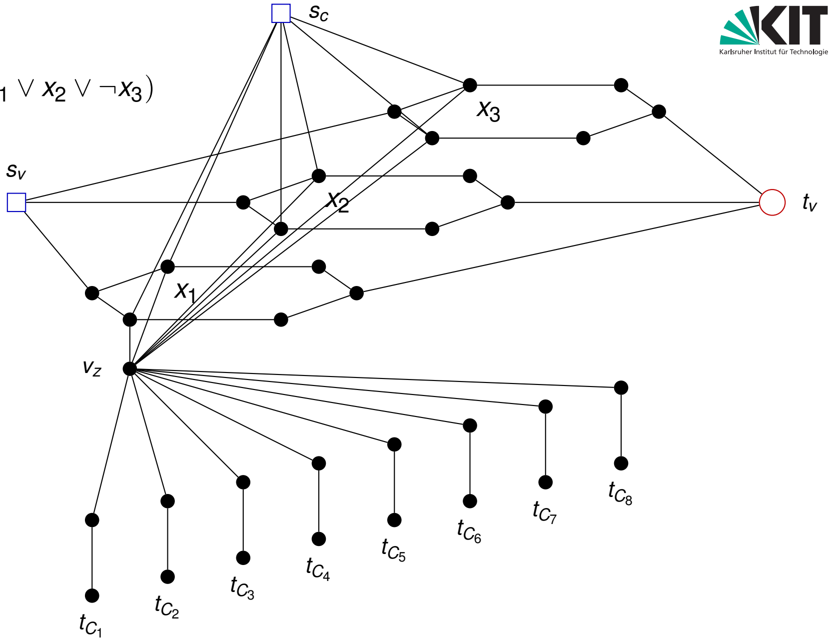
s_c
s_v
t_v
x_1
x_2
x_3
v_z
t_{C_1}
t_{C_2}
t_{C_3}
t_{C_4}
t_{C_5}
t_{C_6}
t_{C_7}
t_{C_8}

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

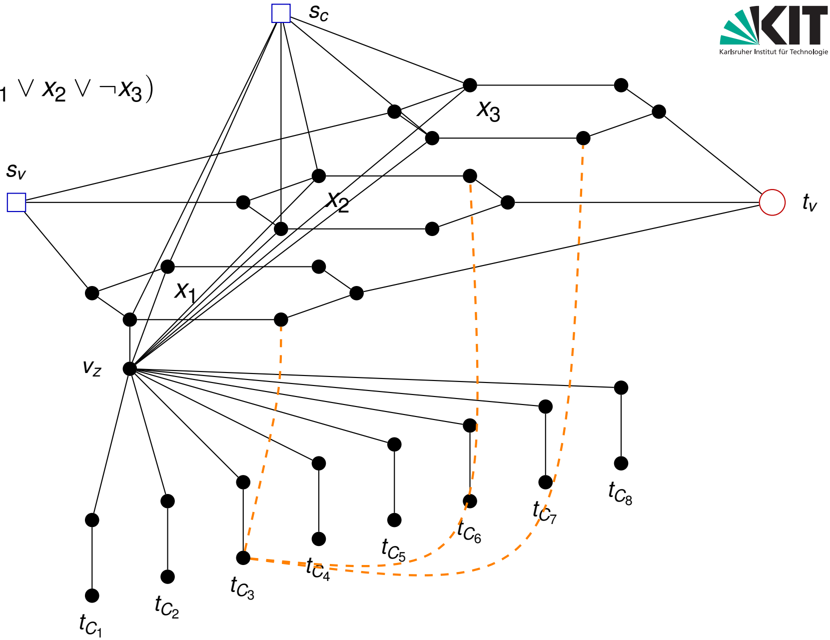


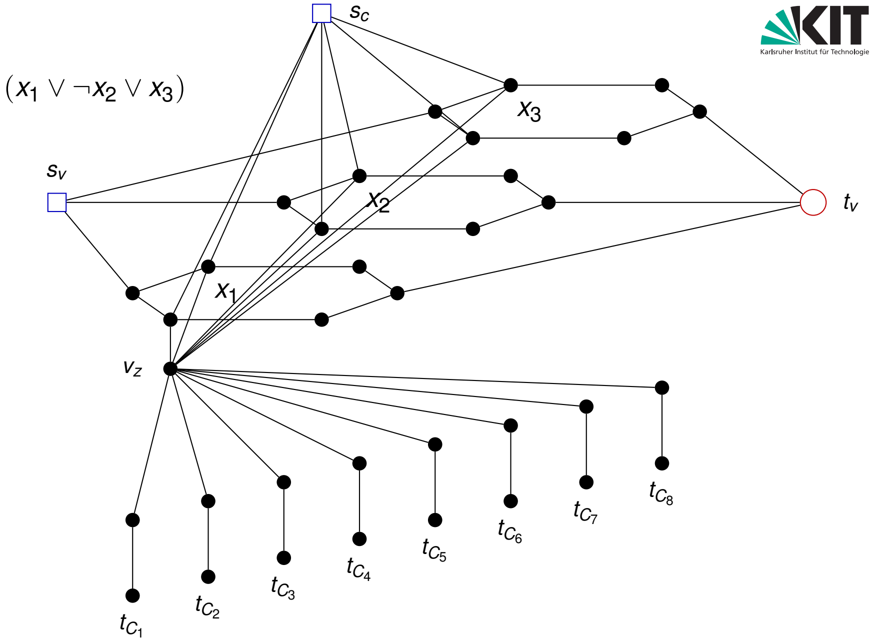

$(\neg x_1 \vee \neg x_2 \vee x_3)$
s_c
s_v
t_v
x_1
x_2
x_3
v_z
t_{C_1}
t_{C_2}
t_{C_3}
t_{C_4}
t_{C_5}
t_{C_6}
t_{C_7}
t_{C_8}

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

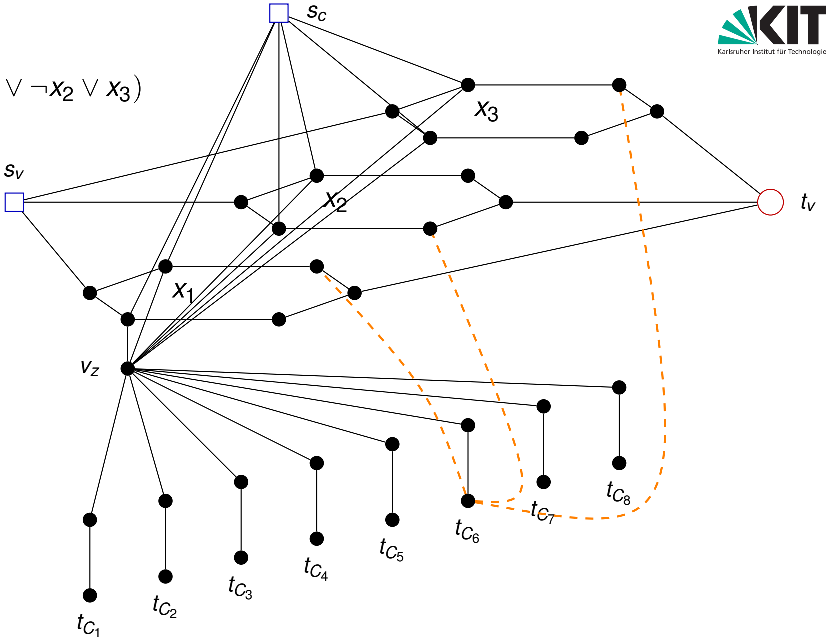


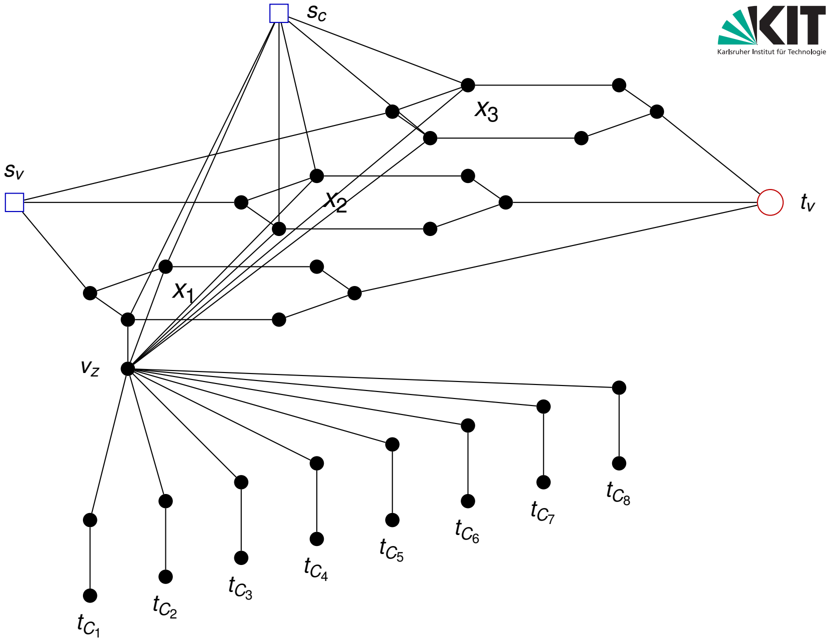
$(\neg x_1 \vee x_2 \vee \neg x_3)$

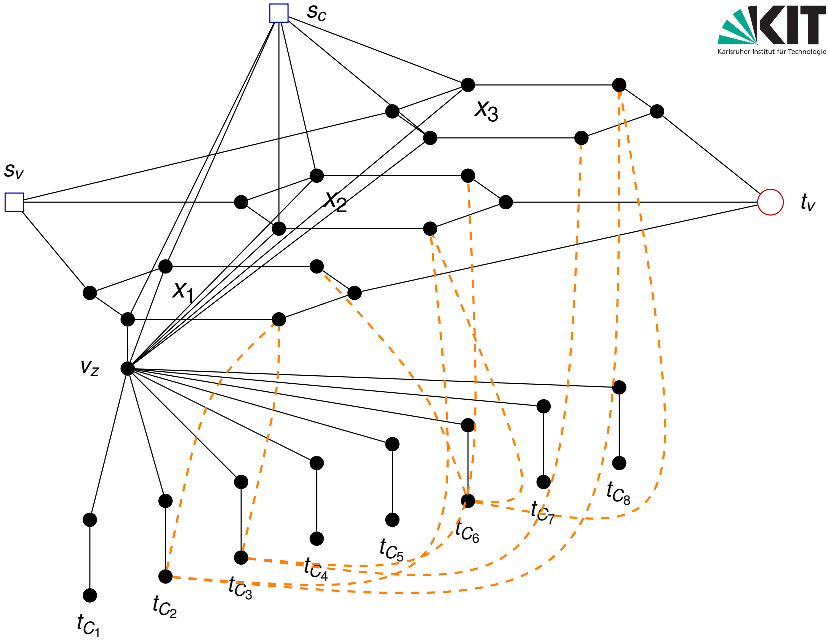



$(x_1 \vee \neg x_2 \vee x_3)$
s_c
s_v
t_v
x_3
x_2
x_1
v_z
t_{C_1}
t_{C_2}
t_{C_3}
t_{C_4}
t_{C_5}
t_{C_6}
t_{C_7}
t_{C_8}

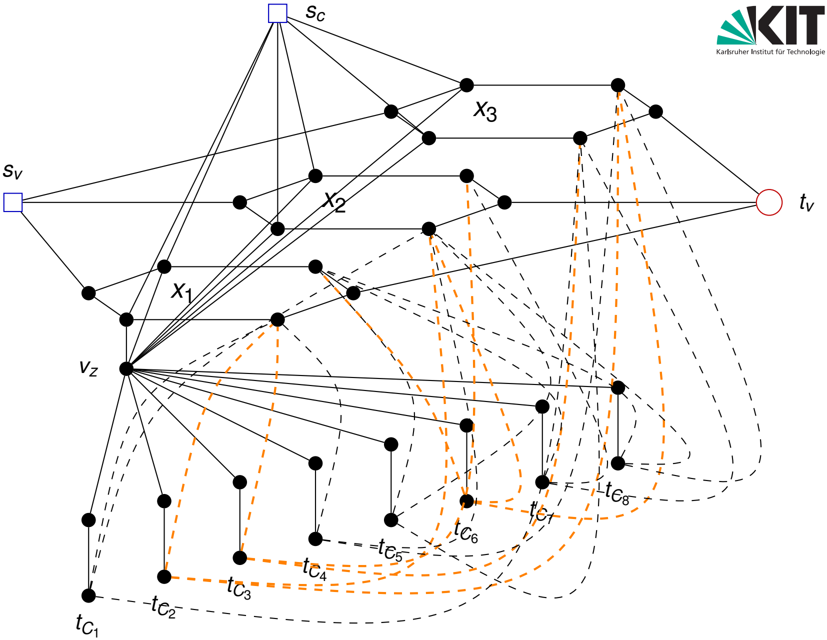
$(x_1 \vee \neg x_2 \vee x_3)$




s_C
s_V
t_V
x_1
x_2
x_3
v_z
t_{C_1}
t_{C_2}
t_{C_3}
t_{C_4}
t_{C_5}
t_{C_6}
t_{C_7}
t_{C_8}



s_c
x_3
s_v
x_2
t_v
x_1
v_z
t_{C_8}
t_{C_7}
t_{C_6}
t_{C_5}
t_{C_4}
t_{C_3}
t_{C_2}
t_{C_1}



s_c
s_v
t_v
x_3
x_2
x_1
v_z
t_{C_1}
t_{C_2}
t_{C_3}
t_{C_4}
t_{C_5}
t_{C_6}
t_{C_7}
t_{C_8}

Gliederung

TNEP–Definition

Was wird bewiesen?

Reduktion

Beweis

Berechnung realer Instanzen

1. Reduktion hat poly. Laufzeit

Nötige Schritte für Formel F (m Klauseln, k Variablen)

- Konstruktion k Variablen-Zellen $\mathcal{O}(6k)$
- Konstruktion $2^3 \cdot \binom{k}{3}$ KLausel-Blöcken $\mathcal{O}(k^3)$
- Hinzufügen von $\binom{k}{3} \cdot 3$ Kanten $\mathcal{O}(k^3)$
- Berechnung der Leistungschranke $c_{Klausel}$ $\mathcal{O}(k^9)$

$\Rightarrow$ Gesamtlaufzeit in $\mathcal{O}(k^9)$

2. Reduktion ist korrekt

Beweis besteht aus 4 Schritten:

1. Reduktionslemma

Netzwerk kann auf höchstens 38 Knoten reduziert werden.

2. Reduktion ist korrekt

Beweis besteht aus 4 Schritten:

- **TNEP**-Kantenausbau ist **SAT**-Konsistent $\Leftrightarrow$ genau eine Kante pro Variablen-Zelle hinzugefügt
- **TNEP**-Kantenausbau ist trivial $\Leftrightarrow$ keine Kante wird hinzugefügt

2. Konsistenzlemma

TNEP-Kantenausbau n ist **SAT**-konsistent (oder trivial)

$$\Leftrightarrow$$

alle Leistungsschranken an t_v sind erfüllt.

2. Reduktion ist korrekt

Beweis besteht aus 4 Schritten:

3. Flusslemma

- n: **SAT**-konsistente **TNEP**-Kantenausbau

  $\Rightarrow$ Genau 4 Klausel-Leistungsflüsse p_0, p_1, p_2 und p_3

  $\Rightarrow 0 < p_3 < p_2 < p_1 < p_0$

2. Reduktion ist korrekt

Beweis besteht aus 4 Schritten:

3. Flusslemma

- n: **SAT**-konsistente **TNEP**-Kantenausbau

  $\Rightarrow$ Genau 4 Klausel-Leistungsflüsse p_0, p_1, p_2 und p_3
  $\Rightarrow 0 < p_3 < p_2 < p_1 < p_0$

4. Trivialitätslemma

n: trivialer **TNEP**-Kantenausbau
$\Rightarrow$
Leistungsfluss-Schranke $c_{Klausel}$ wird nicht erfüllt

2. Reduktion ist korrekt
Übersicht

" $\Rightarrow$ " : Formel F ist erfüllbar:

2. Reduktion ist korrekt
Übersicht

" $\Rightarrow$ " : Formel F ist erfüllbar:

- Zeige, es gibt Lösung n_F der **TNEP**-Instanz I:

2. Reduktion ist korrekt

Übersicht

" $\Rightarrow$ " : Formel F ist erfüllbar:

- Zeige, es gibt Lösung n_F der **TNEP**-Instanz I:
- Betrachte **TNEP**-Kantenausbau n_F mit:

  n_F setzt Kanten, die erfüllender Belegung von F entsprechen
  $\Rightarrow$ n_F ist **SAT**-konsistent

2. Reduktion ist korrekt
Übersicht

" $\Rightarrow$ " : Formel F ist erfüllbar:

- Zeige, es gibt Lösung n_F der **TNEP**-Instanz I:
- Betrachte **TNEP**-Kantenausbau n_F mit:

  n_F setzt Kanten, die erfüllender Belegung von F entsprechen
  $\Rightarrow$ n_F ist **SAT**-konsistent

- mit Konsistenzlemma $\Rightarrow$ n_F erfüllt Schranke an t_v
- mit Flusslemma $\Rightarrow$ n_F erfüllt Schranke $c_{Klausel}$ für alle $C_j \in F$

2. Reduktion ist korrekt
Übersicht

" $\Rightarrow$ " : Formel F ist erfüllbar:

- Zeige, es gibt Lösung n_F der **TNEP**-Instanz I:
- Betrachte **TNEP**-Kantenausbau n_F mit:

  n_F setzt Kanten, die erfüllender Belegung von F entsprechen
  $\Rightarrow n_F$ ist **SAT**-konsistent

- mit Konsistenzlemma $\Rightarrow n_F$ erfüllt Schranke an t_v
- mit Flusslemma $\Rightarrow n_F$ erfüllt Schranke $c_{Klausel}$ für alle $C_j \in F$

$\Rightarrow n_F$ ist Lösung für I

2. Reduktion ist korrekt

Übersicht

" $\Leftarrow$ ": **TNEP**-Instanz *I* hat Lösung *n*

2. Reduktion ist korrekt
Übersicht

" $\Leftarrow$ ": **TNEP**-Instanz *I* hat Lösung *n*

- Mit Konsistenzlemma $\Rightarrow$ *n* ist **SAT**-konsistent oder trivial

2. Reduktion ist korrekt
Übersicht

"$\Leftarrow$": **TNEP**-Instanz I hat Lösung n

- Mit Konsistenzlemma $\Rightarrow$ n ist **SAT**-konsistent oder trivial
- Mit Trivialitätslemma $\Rightarrow$ n ist nicht trivial

2. Reduktion ist korrekt
Übersicht

"$\Leftarrow$": **TNEP**-Instanz I hat Lösung n

- Mit Konsistenzlemma $\Rightarrow$ n ist **SAT**-konsistent oder trivial
- Mit Trivialitätslemma $\Rightarrow$ n ist nicht trivial
- Mit Flusslemma und Definition von $c_{Klausel}$ $\Rightarrow$ jede Klausel aus F ist mindestens 1-erhöht

2. Reduktion ist korrekt

Übersicht

" $\Leftarrow$ ": **TNEP**-Instanz I hat Lösung n

- Mit Konsistenzlemma $\Rightarrow$ n ist **SAT**-konsistent oder trivial
- Mit Trivialitätslemma $\Rightarrow$ n ist nicht trivial
- Mit Flusslemma und Definition von $c_{Klausel}$ $\Rightarrow$ jede Klausel aus F ist mindestens 1-erhöht
- n induziert erfüllende Variablenbelegung von F

2. Reduktion ist korrekt
Übersicht

" $\Leftarrow$ ": **TNEP**-Instanz I hat Lösung n

- Mit Konsistenzlemma $\Rightarrow$ n ist **SAT**-konsistent oder trivial
- Mit Trivialitätslemma $\Rightarrow$ n ist nicht trivial
- Mit Flusslemma und Definition von $c_{Klausel}$ $\Rightarrow$ jede Klausel aus F ist mindestens 1-erhöht
- n induziert erfüllende Variablenbelegung von F

$\Rightarrow$ F ist erfüllbar

Beweis
Gesamt

- Reduktion kann in poly. Zeit berechnet werden
- F erfüllbar $\Leftrightarrow f(F)$ Ja-Instanz von **TNEP**
- **TNEP** $\in$ **NP** (Berechnung mittels Laplace-Matrix)

$\Rightarrow$ **TNEP** ist **NP**-vollständig

Gliederung

TNEP–Definition

Was wird bewiesen?

Reduktion

Beweis

Berechnung realer Instanzen

TNEP in der Praxis

- **TNEP** kann als **MINLP** Problem formuliert werden
- Anwendung von **MINLP** Solvern
- Verwendete Solver in [2]: NEOS-Server mit BARON

TNEP in der Praxis

Test-Instanzen

Name	$\lvert V \rvert$	$\lvert \mathrm{conn}(E) \rvert$
Garver	6	15
IEEE 24 bus	24	34
Brazil South	46	79
Brazil South East	79	143

Test-Instanzen aus [2]

TNEP in der Praxis

Test-Instanzen: Laufzeit

Name	$\epsilon_r = 0.1$	$\epsilon_r = 0.01$	$\epsilon_r = 0.001$	$\epsilon_a = 0.999$
Garver	0:00:01	0:00:01	0:00:01	0:00:01
IEEE 24 bus	0:00:02	0:00:04	0:00:04	0:00:05
Brazil South	0:16:12	1:29:07	1:46:58	1:38:56
Brazil South East	8:00:00	8:00:00	8:00:00	8:00:00

Laufzeit für Test-Instanzen aus [2], Format h:mm:ss

Zusammenfassung

- **TNEP** ist **NP**-vollständig
- Gilt auch, wenn nur verbundene **TNEP**-Instanzen betrachtet werden
- Laufzeitmessung an Testfällen deckt sich mit Theorie

Literatur

Karlsruher Institut für Technologie

[1] Luciano S. Moulin, Michael Poss, and Claudia Sagastizábal.
Transmission expansion planning with re-design.
Energy systems, 1(2):113–139, 2010.

[2] David Oertel and R. Ravi.
Complexity of transmission network expansion planning.
Energy Systems, 5(1):179–207, 2014.