1 Einleitung

1.1 NP-Vollständigkeit

Eine Sprache B ist **NP-Vollständig** wenn gilt:

1. $B \in NP$
2. $\forall A \in NP : A \prec_p B$

Notiz:
$A \prec_p B$: A ist polynomialzeitreduzierbar auf B

1.2 Polynomialzeitreduktion

Eine Sprache A ist polynomialzeitreduzierbar auf Sprache B, $A \prec_p B$, wenn eine in polynomialer Zeit berechenbare Funktion $f : \Sigma^* \to \Sigma^*$ existiert, für die gilt:
$\forall w : w \in A \iff f(w) \in B$

Die Funktion f heißt dann Polynomialzeitreduktion von A nach B.

1.3 Definition 3SAT

Spezialform des Erfüllbarkeitsproblems

- Literal:
  x_i oder $\overline{x_i}$
- Variable:
  x_i (l: Anzahl der Variablen)
- Klausel:
  $(x_1 \vee \overline{x_2} \vee \overline{x_3} \vee x_4)$ (k: Anzahl der Klauseln)
- CNF-Formel(cnf-formula - conjunctive normal form):
  $(x_1 \vee \overline{x_2} \vee \overline{x_3} \vee x_4) \wedge (x_3 \vee \overline{x_5} \vee x_6) \wedge (x_3 \vee \overline{x_6})$
- ϕ: $(x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (\overline{x_2} \vee x_4 \vee ...) \wedge ... \wedge (\overline{x_3} \vee ... \vee ...)$

$3SAT = \{ \langle\phi\rangle | \phi$ ist eine erfüllbare 3CNF-Formel$\}$

$\phi = 1 \iff \forall c_j$: mindestens ein Literal ist $true$

2 Hamilton-Pfad-Problem

2.1 Definition

- Geht durch jeden Knoten genau einmal
- Startet in s
- Endet in t

Notiz: Gesucht: Pfad von s nach t, der durch jeden knoten genau einmal geht.

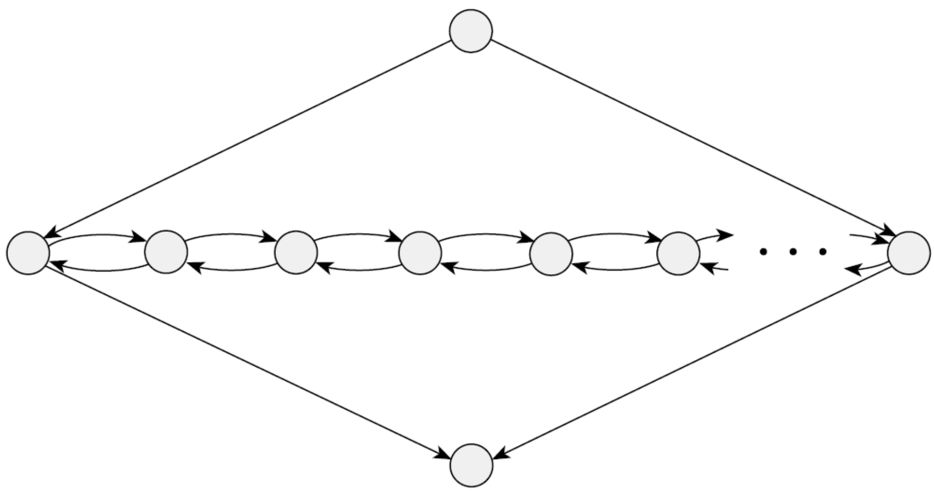
2.2 Beweis Gerichtet

2.2.1 Konstruktion von G

$\phi \to G$

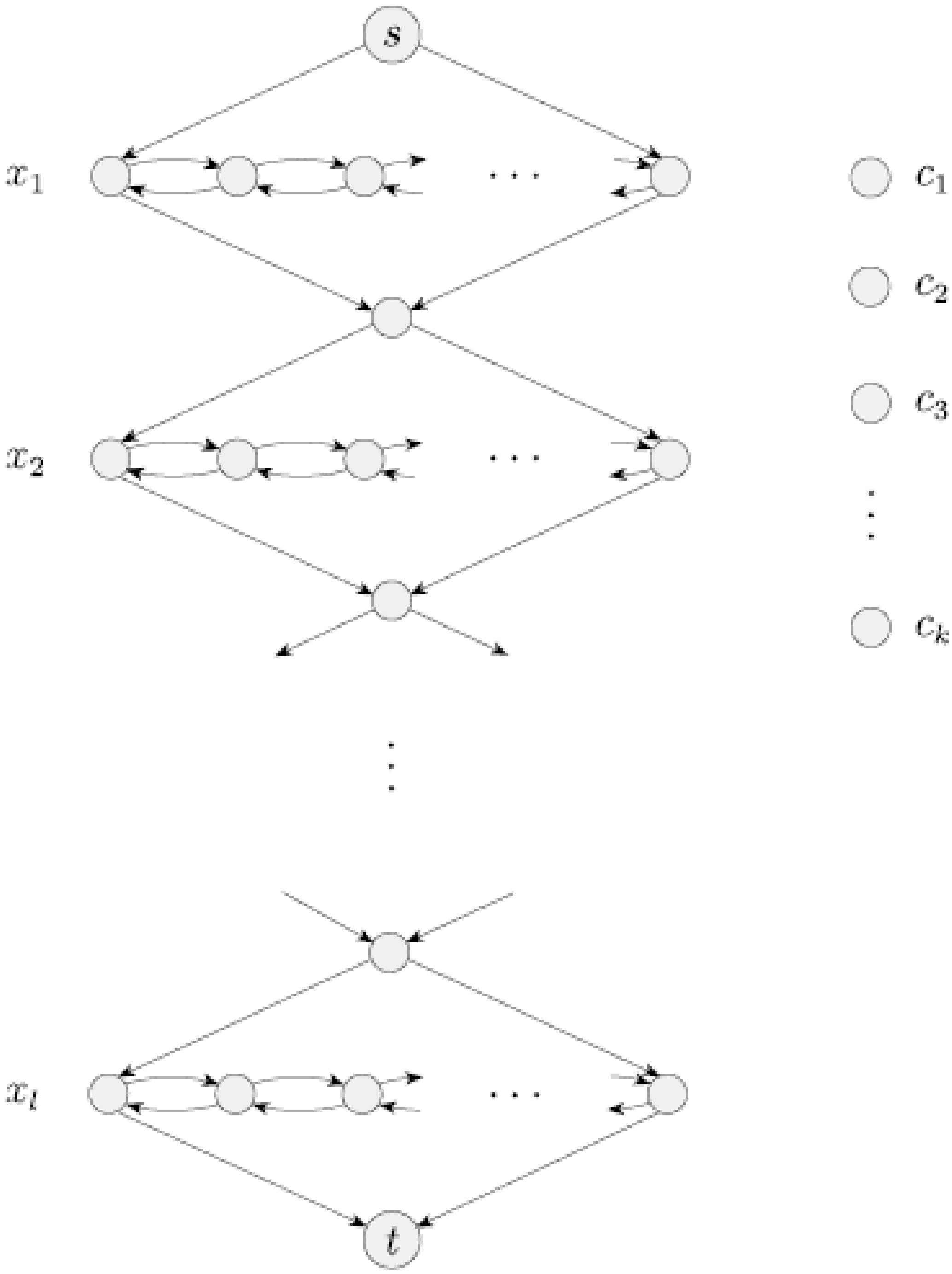
Darstellung Variable x_i

x_i

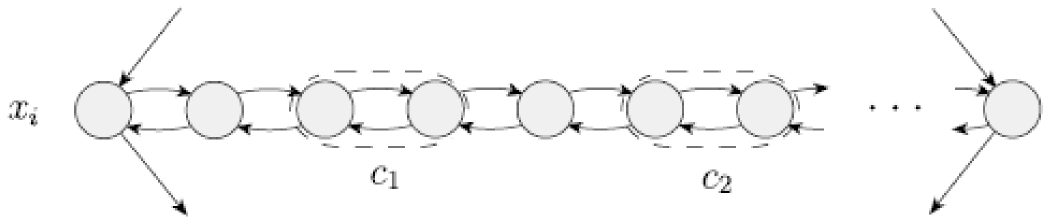


Darstellung Klausel c_j

High-level structure of G

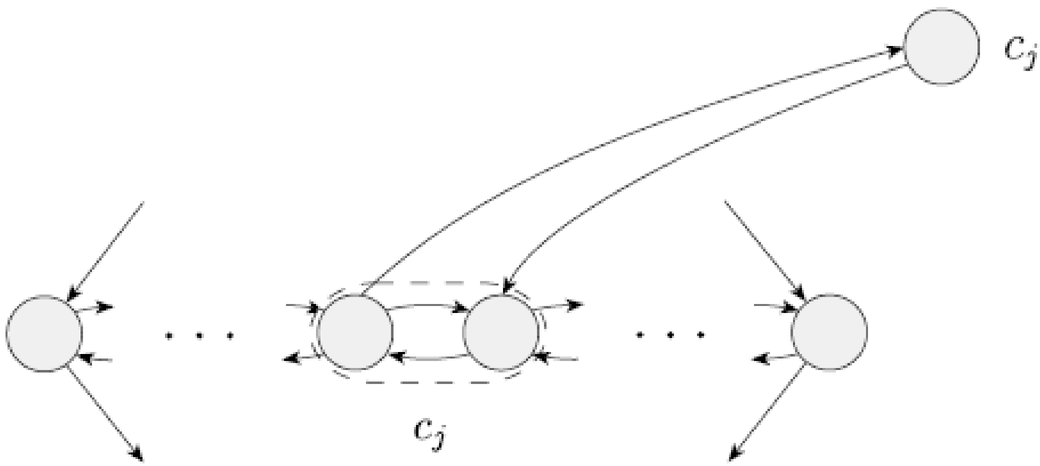


Horizontale Struktur im Diamant



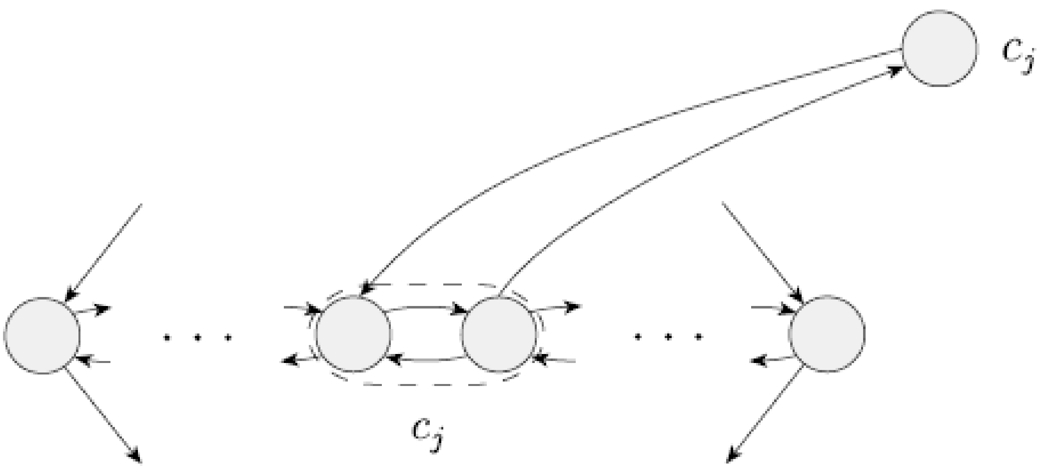
$\langle 3k+1 \rangle$ Knoten

Zusätzliche Kanten wenn x_i in c_j ist



Notiz: Am Beispiel von ϕ zeigen!

Zusätzliche Kanten wenn $\overline{x_i}$ in c_j ist



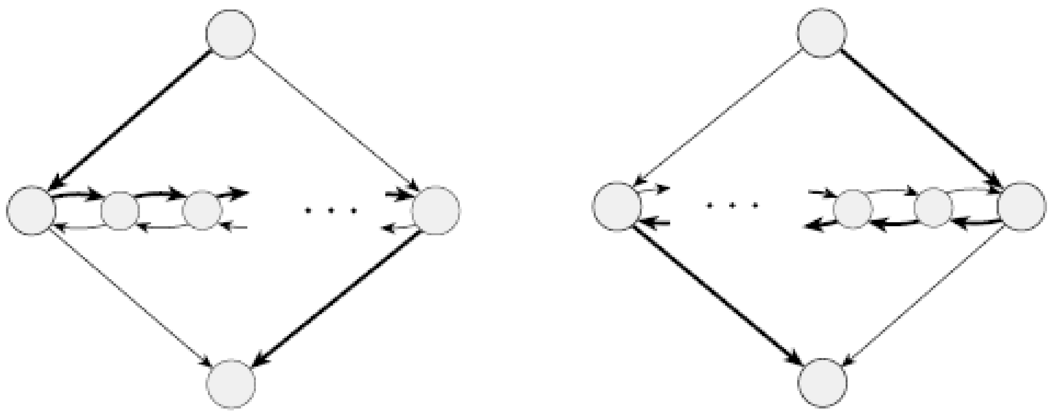
2.2.2 ϕ ist erfüllbar

Notiz: Zunächst werden die Knoten c_j ignoriert.

Notiz: Der Pfad geht von s nach t durch jeden Diamanten nach einander.

Notiz: Um alle Knoten zu treffen muss der Pfad entweder zig-zaggen oder zag-ziggen.

Zig-zagging and Zag-zigging



zig-zag zag-zig

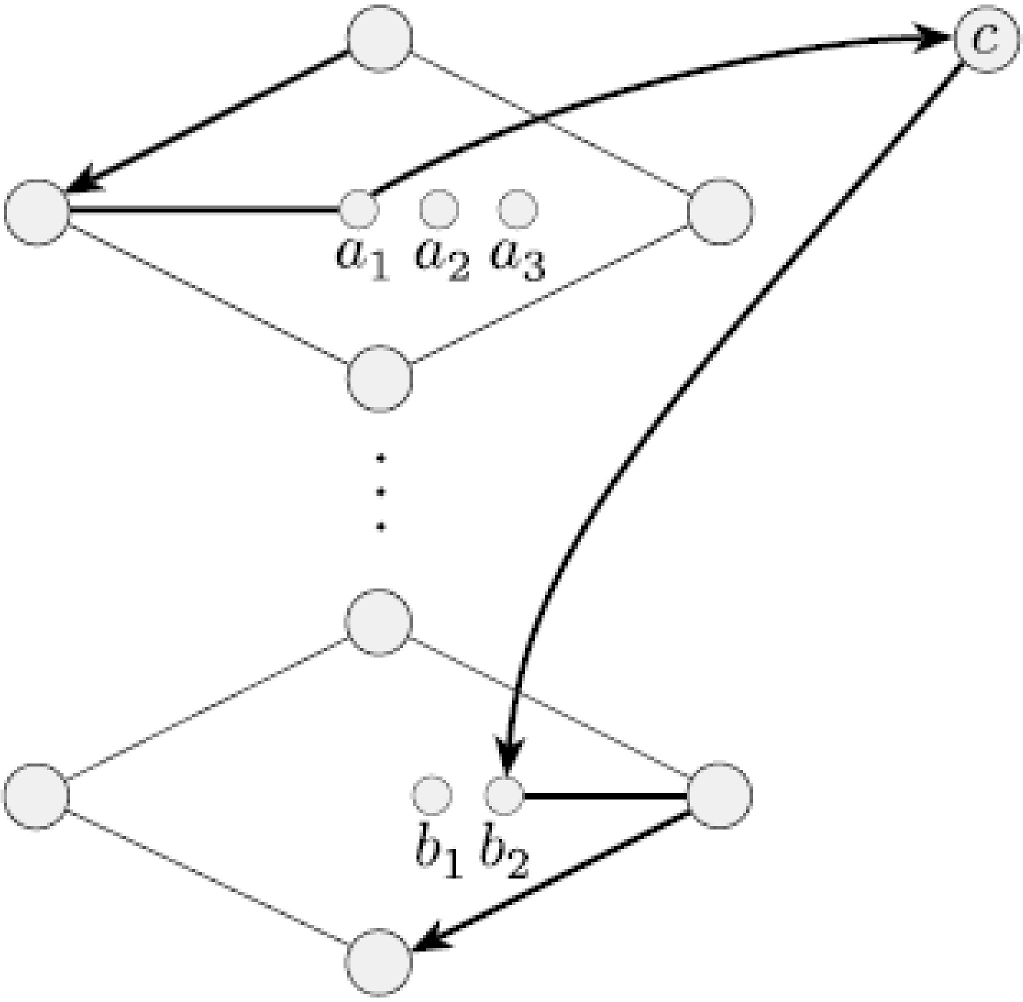
Notiz: Zig-zagging wenn Variable $x_i = true$, Zag-zigging wenn Variable $x_i = false$

Notiz: Jetzt fehlen nurnoch die Knoten c_j.

Notiz: In jeder Klausel c_j wählen wir einen Literal aus, dem wir $true$ zuweisen.

Notiz: Jeder wahre Literal in einer Klausel ist nur eine Option für einen Umweg über einen Klauselnoten. $\rightarrow$ Es wird immer nur ein Umweg zu jedem Klauselknoten genommen. $\rightarrow$ Knostruktion von G ist beendet.

This situation cannot occur



Laufzeit

Notiz: Offentsichlich nicht polynomial

2.2.3 Ungerichtet

TODO: muss der auch rein?

3 SUBSET-SUM-Problem

3.1 Definition

- Integer-Arithmetik
- Menge von Nummern: $x_1, ..., x_k$
- Ziel t
- Kann t durch eine Teilmenge erreicht werden?

3.2 Beweis

3SAT $\prec_p$ SUBSET-SUM

ϕ : Boolesche Formel
$x_1, x_2, ..., x_l$ $c_1, c_2, ..., c_k$ Teilausdrücke

3.2.1 Gesucht

$\prec_p$: $\phi \mapsto \langle S, t \rangle$

$\phi : (x_1 \vee \overline{x_2} \vee x_3) \wedge (x_2 \vee x_3 \vee ...) \wedge ... \wedge (\overline{x_3} \vee ... \vee ...)$

Notiz: c_1, c_2, c_k dran schreiben

3.2.2 Annahme

Es existiert eine Konfiguration für die ϕ erfüllt ist.

$\Rightarrow$ Subset bauen
y_i iff $x_i = true$ else z_i

$\Rightarrow t = 11...1133...33$
Notiz: l mal 1 und k mal 3

$\Rightarrow$ Füge solange g_i unf h_i hinzu bis das Target t erreicht ist.

	1	2	3	4	...	1	c_1	c_2	...	c_k
y_1	1	0	0	0	...	0	1	0	...	0
z_1	1	0	0	0	...	0	0	0	...	0
y_2		1	0	0	...	0	0	1	...	0
z_2		1	0	0	...	0	1	0	...	0
y_3			1	0	...	0	1	1	...	0
z_3			1	0	...	0	0	0	...	1
...					...				...	
y_l						1	0	0	...	0
z_l						1	0	0	...	0
g_1						1	0	0	...	0
h_1						1	0	0	...	0
g_2							1	0	...	0
h_2							1	0	...	0
...					...				...	
g_k										1
h_k										1
t	1	1	1	1	...	1	3	3	...	3

Notiz: Zeigen, dass ϕ erfüllbar ist mit einem subset von S dass sich auf t summiert.

Notiz: $\forall c_j$: Mindestens eine Variable von c_j muss 1 sein, da maximal 2 von g_j und h_j kommen können.

Notiz: $\phi = true$ iff $\forall c_j : c_j = true \Rightarrow$ reduzierbar

3.2.3 Laufzeit

Tabelle hat etwa eine Größe von $(l + k)^2$
Jeder Eintrag ist leicht zu berechnen $\Rightarrow O(n^2)$