Formalising Polynomial-Time Reductions Using the Call-by-Value λ-Calculus in Coq

Lennard Gäher

Advisor: Fabian Kunze

Saarland University

8 November 2019

Timeline

1965: time/space complexity, hierarchy theorems, . . .
On the Computational Complexity of Algorithms
Hartmanis and Stearns

Timeline

1965: time/space complexity, hierarchy theorems, . . .
On the Computational Complexity of Algorithms
Hartmanis and Stearns

1971: Cook's Theorem
The Complexity of Theorem-Proving Procedures
Stephen A. Cook

Timeline

1965: time/space complexity, hierarchy theorems, . . .
On the Computational Complexity of Algorithms
Hartmanis and Stearns

1971: Cook's Theorem
The Complexity of Theorem-Proving Procedures
Stephen A. Cook

1972: Karp's 21 NP-complete problems
Reducibility Among Combinatorial Problems
Richard M. Karp

The Call-by-Value λ-calculus L[1] ...

$$s, t : \text{Term} := n \mid s\ t \mid \lambda s$$

$$\frac{}{(\lambda s)(\lambda t) \succ s^0_{\lambda t}} \qquad \frac{s \succ s'}{s\ t \succ s'\ t} \qquad \frac{t \succ t'}{s\ t \succ s\ t'}$$

$$k^k_u := u \qquad n^k_u := n \quad \text{if } n \neq k$$

$$(s\ t)^k_u := (s^k_u)\ (t^k_u) \qquad (\lambda s)^k_u := \lambda(s^{1+k}_u)$$

[1] [Forster and Smolka, 2017]

. . . as a Model of Computation

- certifying extraction from Coq to L[2]
  - inductive datatypes
  - recursive functions

[2] [Forster and Kunze, 2019]
[3] [Forster et al., 2019]

. . . as a Model of Computation

- certifying extraction from Coq to L[2]
  - inductive datatypes
  - recursive functions
- reasonable for time and space[3]

$$\text{size } n := 1$$
$$\text{size } (s\ t) := \text{size } s + \text{size } t + 1$$
$$\text{size } (\lambda s) := \text{size } s + 1$$

[2] [Forster and Kunze, 2019]
[3] [Forster et al., 2019]

. . . as a Model of Computation

- certifying extraction from Coq to L^2, including time bounds
  - inductive datatypes
  - recursive functions
- reasonable for time and space[3]

$$\text{size } n := 1$$
$$\text{size } (s\ t) := \text{size } s + \text{size } t + 1$$
$$\text{size } (\lambda s) := \text{size } s + 1$$

[2] [Forster and Kunze, 2019]
[3] [Forster et al., 2019]

Basic Definitions: NP

Language $A : X \to \mathbb{P}$

inNP A, if:

- there is a verifier $V : X \to \text{Term} \to \mathbb{P}$
- $\forall x, A\ x \leftrightarrow \exists\ cert, V\ x\ cert$
- V is polynomial-time computable (in $\text{size}(\overline{x}), \text{size}(\overline{cert})$)
- V only accepts certificates of polynomial size

Basic Definitions: Polynomial-time Reductions

$$A : X \to \mathbb{P} \qquad B : Y \to \mathbb{P}$$

$$A \preceq_p B := \exists f : X \to Y, \quad (\forall x, A\ x \leftrightarrow B(f\ x))$$
$$\wedge \text{ polynomial-time computable } f$$

Basic Definitions: Polynomial-time Reductions

$$A : X \to \mathbb{P} \qquad B : Y \to \mathbb{P}$$

$$A \preceq_p B := \exists f : X \to Y, \quad (\forall x, A\ x \leftrightarrow B(f\ x))$$
$$\wedge \text{ polynomial-time computable } f$$
$$\wedge \exists\ p, \forall x, \text{size}(\overline{f\ x}) \leq p(\text{size}(\overline{x}))$$

Basic Definitions: Polynomial-time Reductions

$$A : X \to \mathbb{P} \qquad B : Y \to \mathbb{P}$$

$$A \preceq_p B := \exists f : X \to Y, \quad (\forall x, A\ x \leftrightarrow B(f\ x))$$
$$\wedge \text{ polynomial-time computable } f$$
$$\wedge \exists\ p, \forall x, \text{size}(\overline{f\ x}) \leq p(\text{size}(\overline{x}))$$

$$\text{NP-hard } A := \forall B, \text{inNP } B \to B \preceq_p A$$

SAT

$$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$$

SAT

$$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$$

$$\{x_0 \mapsto \mathsf{T}, x_1 \mapsto \mathsf{F}, x_2 \mapsto \mathsf{F}, x_3 \mapsto \mathsf{F}\}$$

SAT

$$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$$

$$(\mathsf{T} \vee \overline{\mathsf{F}} \vee \mathsf{F}) \wedge (\mathsf{T} \vee \mathsf{F} \vee \mathsf{F})$$

$$\{x_0 \mapsto \mathsf{T}, x_1 \mapsto \mathsf{F}, x_2 \mapsto \mathsf{F}, x_3 \mapsto \mathsf{F}\}$$

SAT

$$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$$

$$\mathsf{T} \quad \wedge \quad \mathsf{T}$$

$$\{x_0 \mapsto \mathsf{T}, x_1 \mapsto \mathsf{F}, x_2 \mapsto \mathsf{F}, x_3 \mapsto \mathsf{F}\}$$

SAT

$$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$$

$$\mathsf{T}$$

$$\{x_0 \mapsto \mathsf{T}, x_1 \mapsto \mathsf{F}, x_2 \mapsto \mathsf{F}, x_3 \mapsto \mathsf{F}\}$$

$$\text{Evaluation } \mathcal{E} : \text{assgn} \to \text{cnf} \to \mathcal{O}(\mathbb{B})$$

Definition (SAT)

$$\textbf{SAT}\ N := \exists\ a, \mathcal{E}\ a\ N \quad = \quad {}^{\circ}\mathsf{T}$$

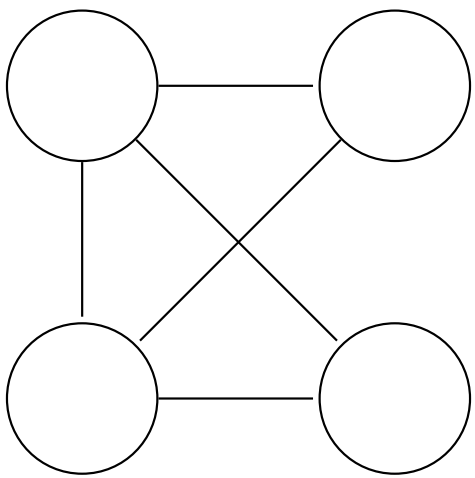
SAT

Definition (k-SAT)

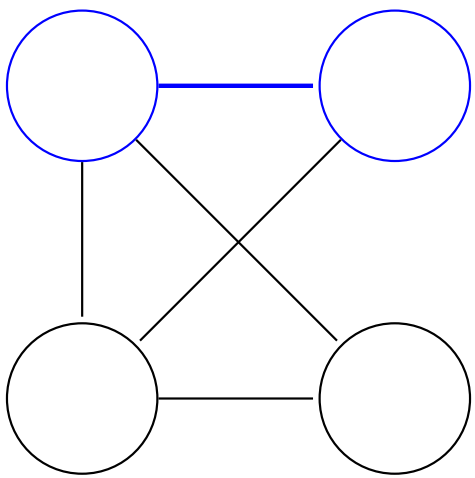
$$k\text{-}\mathbf{SAT}\ N := k\text{-CNF}\ N \wedge \exists\ a, \mathcal{E}\ a\ N \quad = \quad {}^{\circ}\mathsf{T}$$

$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$ is a 3-CNF

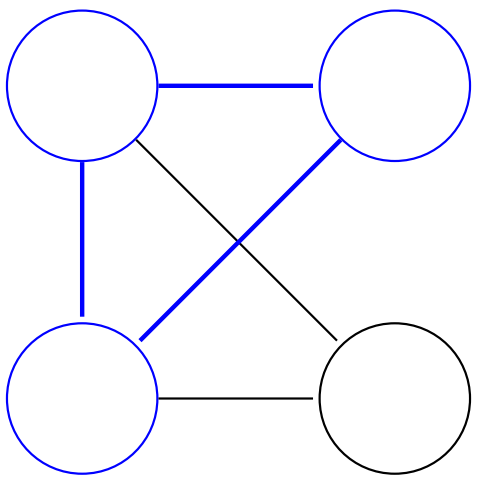
Clique



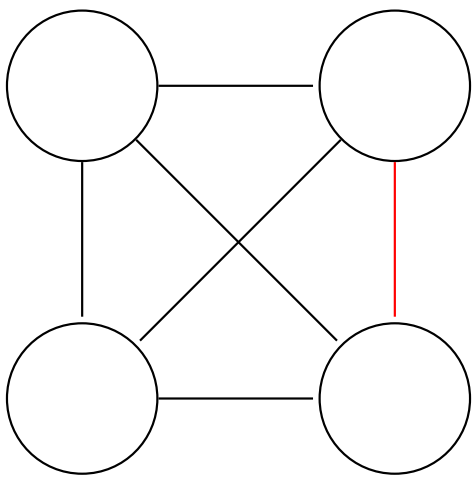
Clique



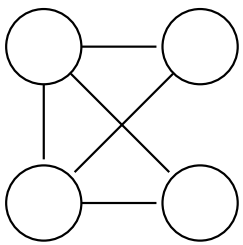
Clique



Clique



Clique



u, v : node

e : edge

g : graph

cl : $\mathcal{L}$ node

Definition (Clique)

Clique $(g, k) := \exists\ cl, \text{isClique}\ g\ cl\ k$

3-SAT $\prec_p$ Clique

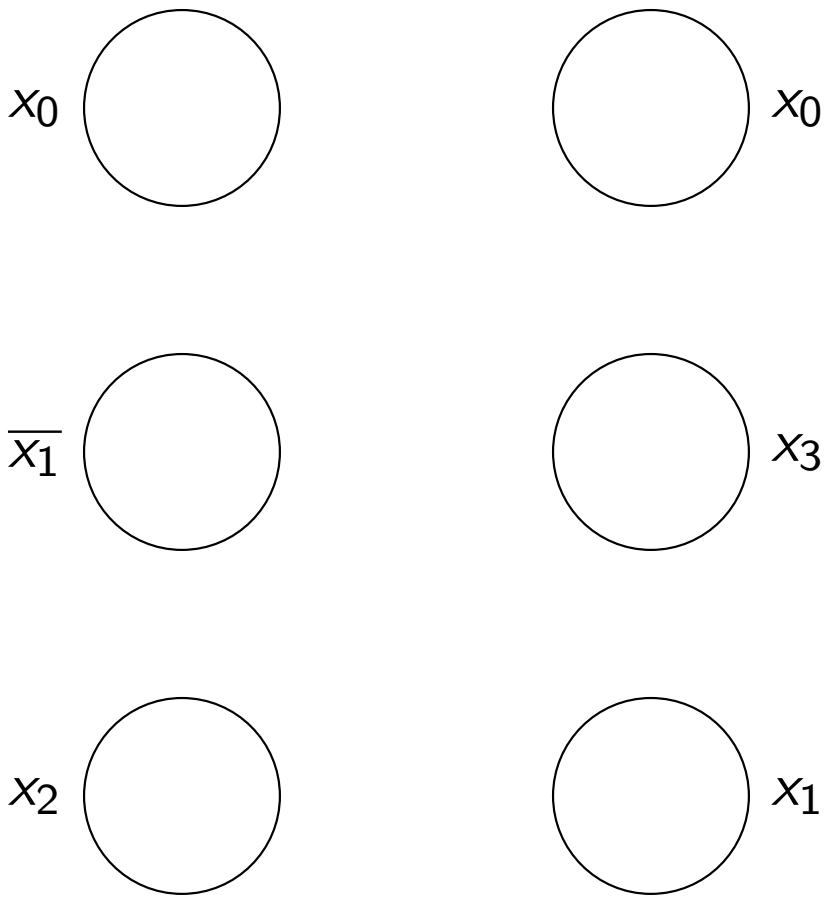
$$N \mapsto (g, k)$$

3-**SAT** $N \leftrightarrow$ **Clique** (g, k)

Reduction: Example

$N \mapsto (g, k)$

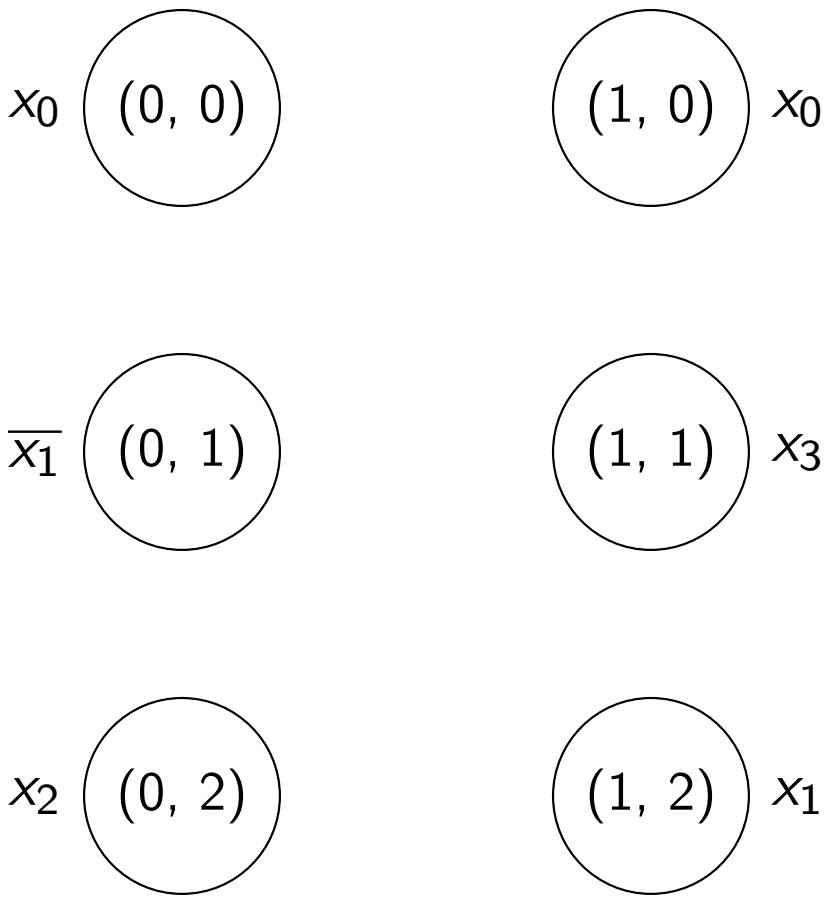
$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$



Reduction: Example

$N \mapsto (g, k)$

$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$

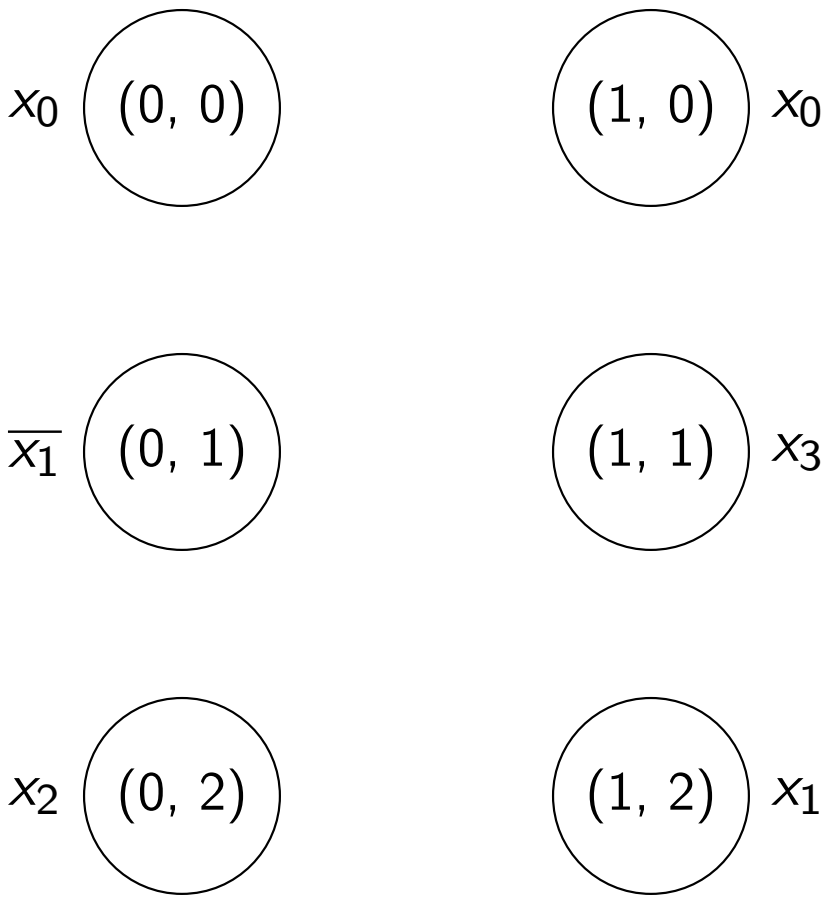


Reduction: Example

$N \mapsto (g, k)$

$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$

edge $\{L_1, L_2\}$ iff:
different clause L_1 L_2 and
non-conflicting L_1 L_2

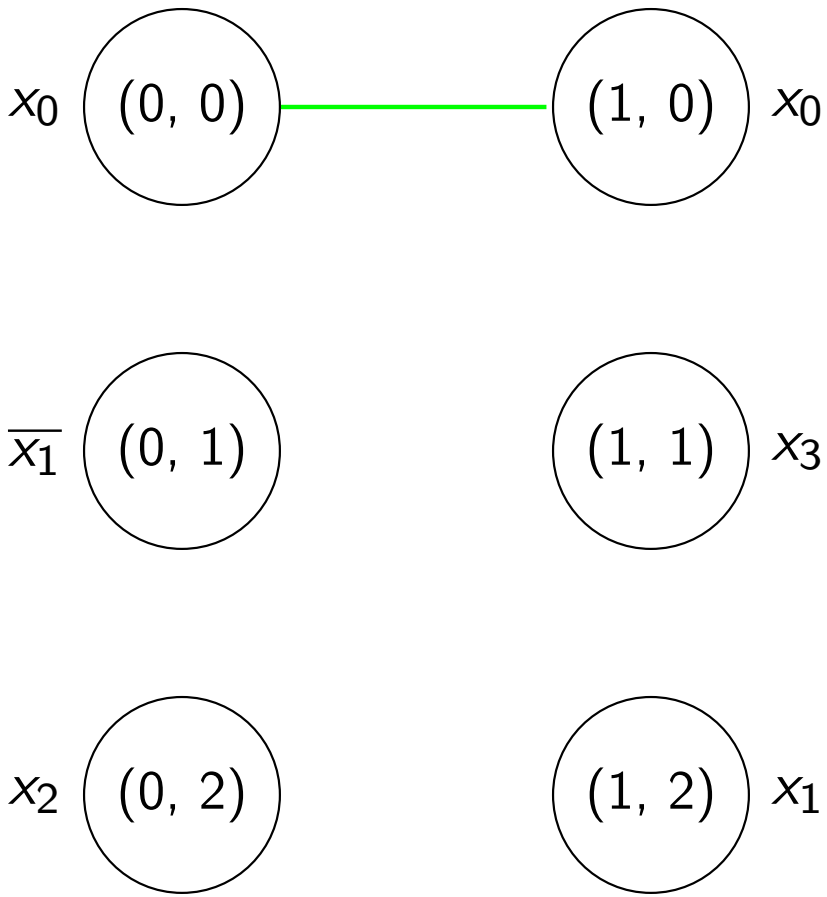


Reduction: Example

$N \mapsto (g, k)$

$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$

edge $\{L_1, L_2\}$ iff:
different clause L_1 L_2 and
non-conflicting L_1 L_2

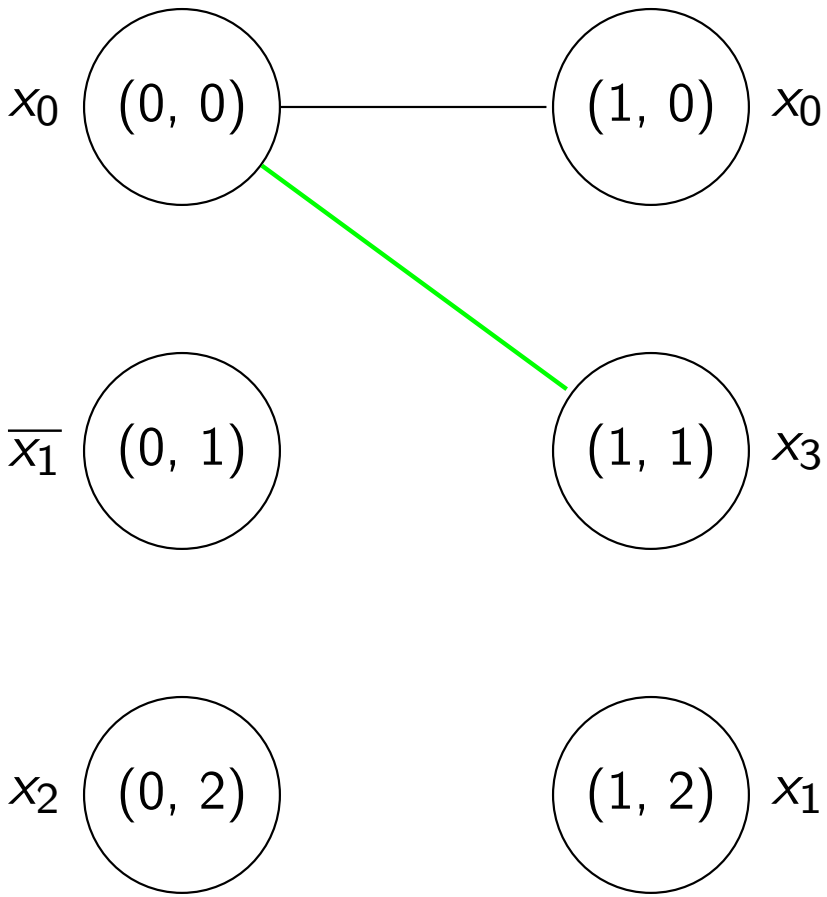


Reduction: Example

$N \mapsto (g, k)$

$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$

edge $\{L_1, L_2\}$ iff:
different clause L_1 L_2 and
non-conflicting L_1 L_2

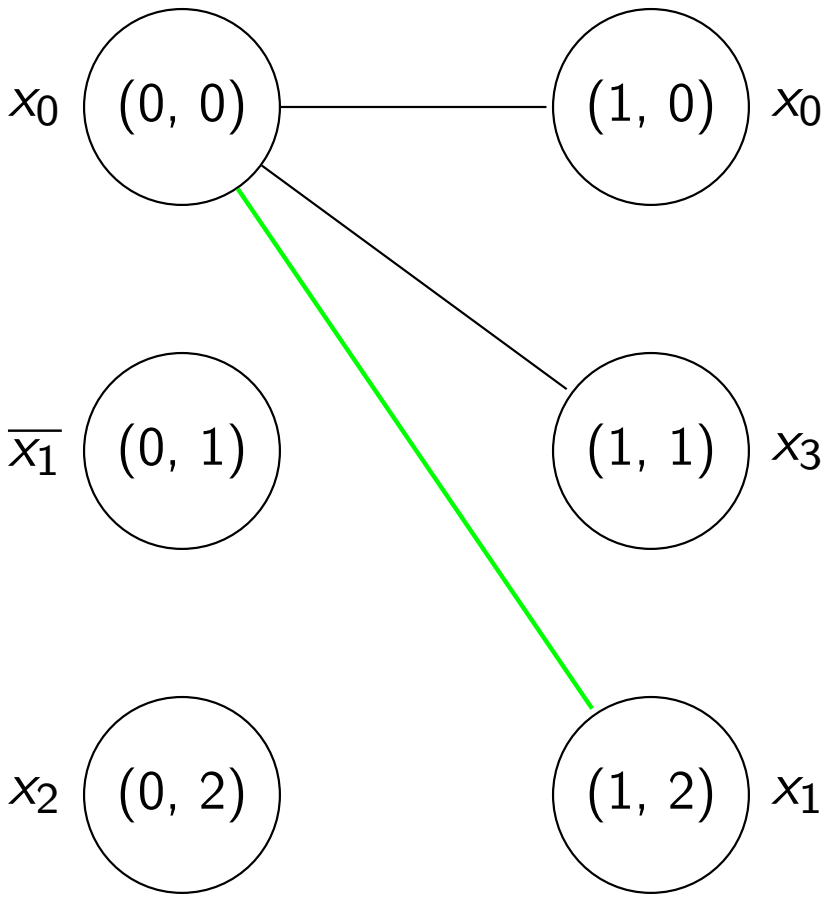


Reduction: Example

$N \mapsto (g, k)$

$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$

edge $\{L_1, L_2\}$ iff:
different clause L_1 L_2 and
non-conflicting L_1 L_2

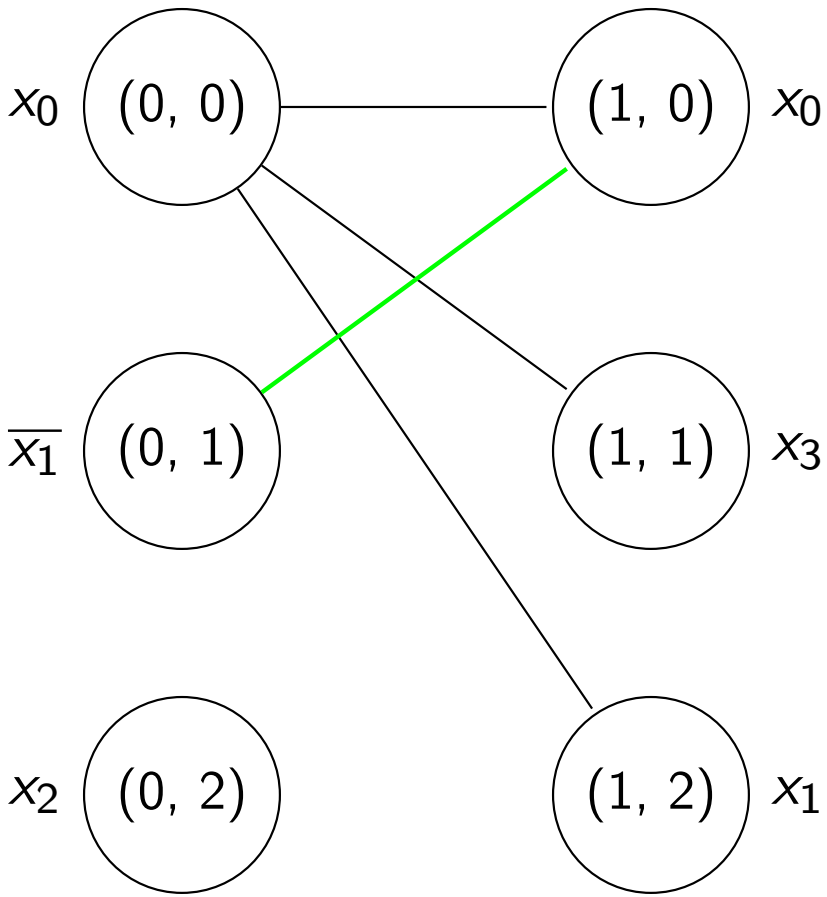


Reduction: Example

$N \mapsto (g, k)$

$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$

edge $\{L_1, L_2\}$ iff:
different clause L_1 L_2 and
non-conflicting L_1 L_2

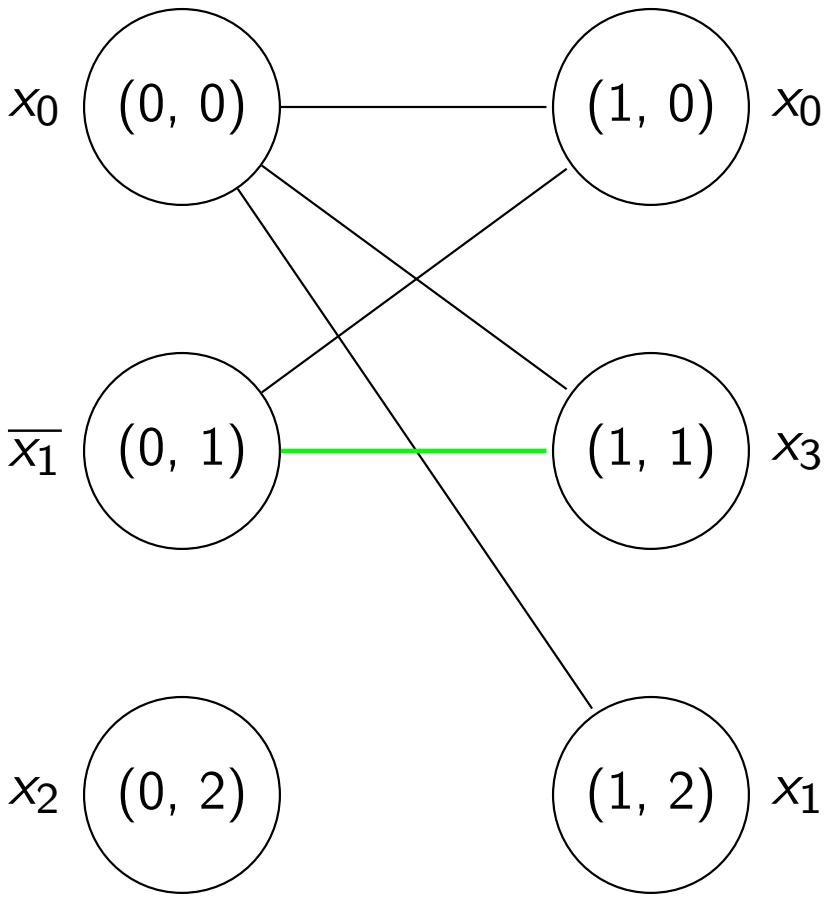


Reduction: Example

$N \mapsto (g, k)$

$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$

edge $\{L_1, L_2\}$ iff:
different clause L_1 L_2 and
non-conflicting L_1 L_2



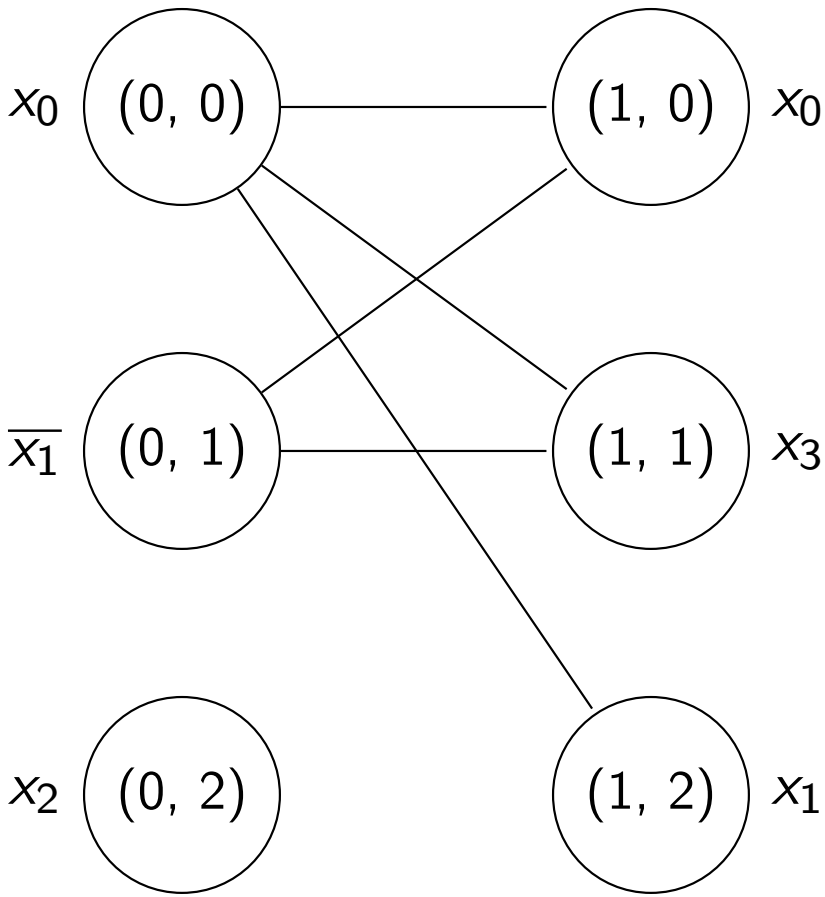
Reduction: Example

$N \mapsto (g, k)$

$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$

edge $\{L_1, L_2\}$ iff:
different clause L_1 L_2 and
non-conflicting L_1 L_2

$\overline{x_1} \not\downarrow x_1$

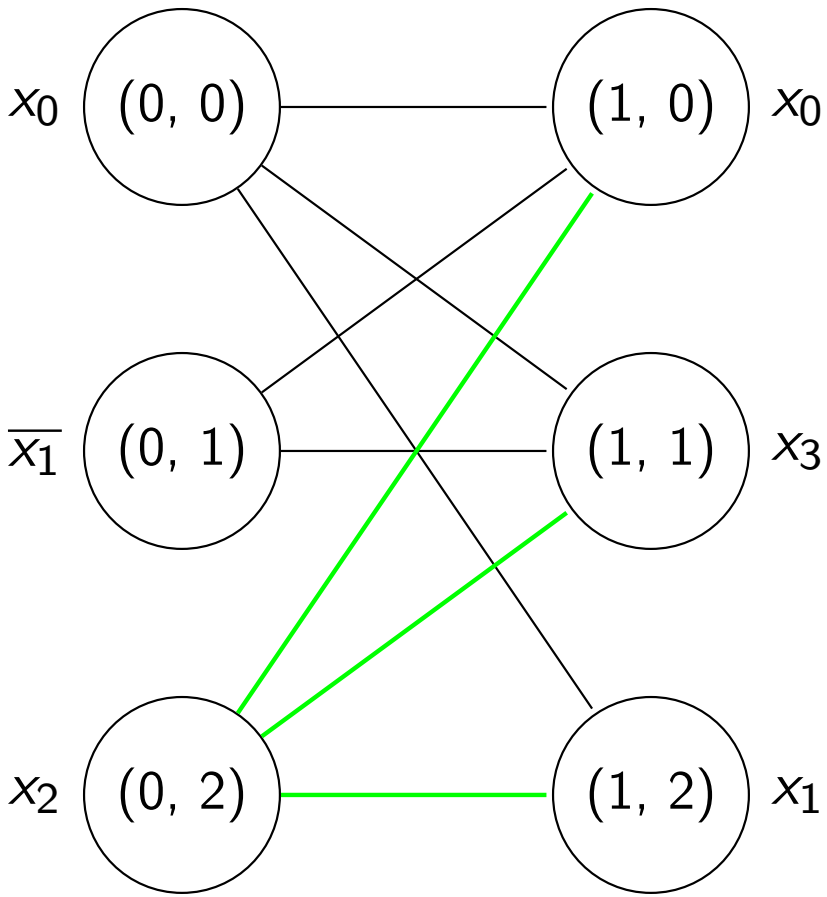


Reduction: Example

$N \mapsto (g, k)$

$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$

edge $\{L_1, L_2\}$ iff:
different clause L_1 L_2 and
non-conflicting L_1 L_2

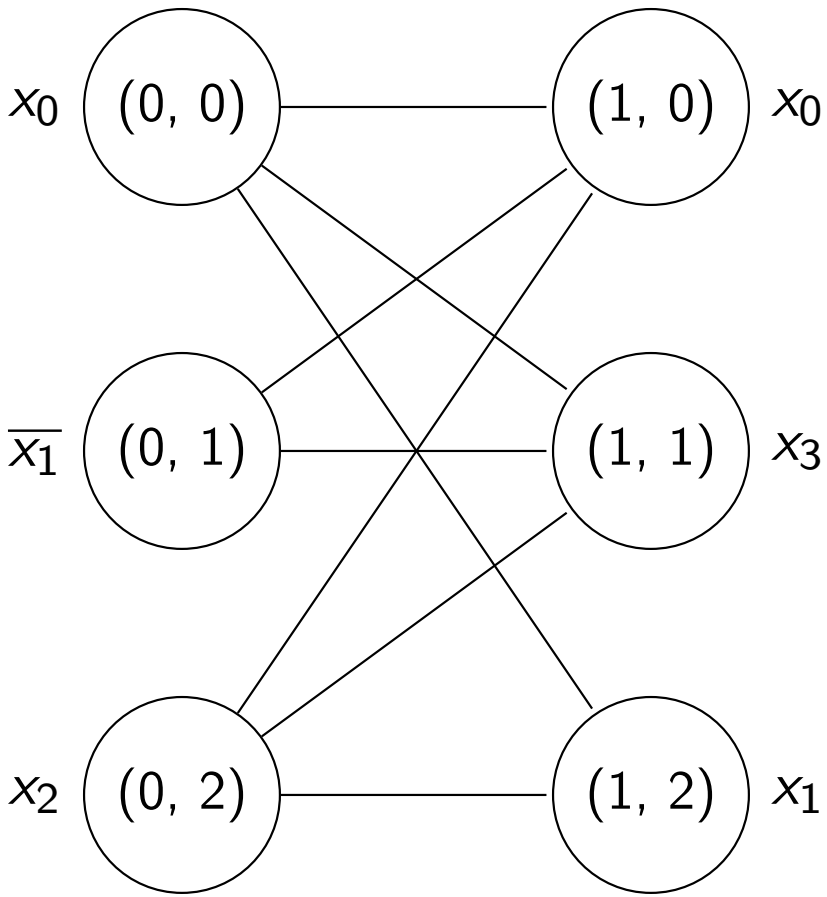


Reduction: Example

$N \mapsto (g, |N|)$

$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$

edge $\{L_1, L_2\}$ iff:
different clause L_1 L_2 and
non-conflicting L_1 L_2

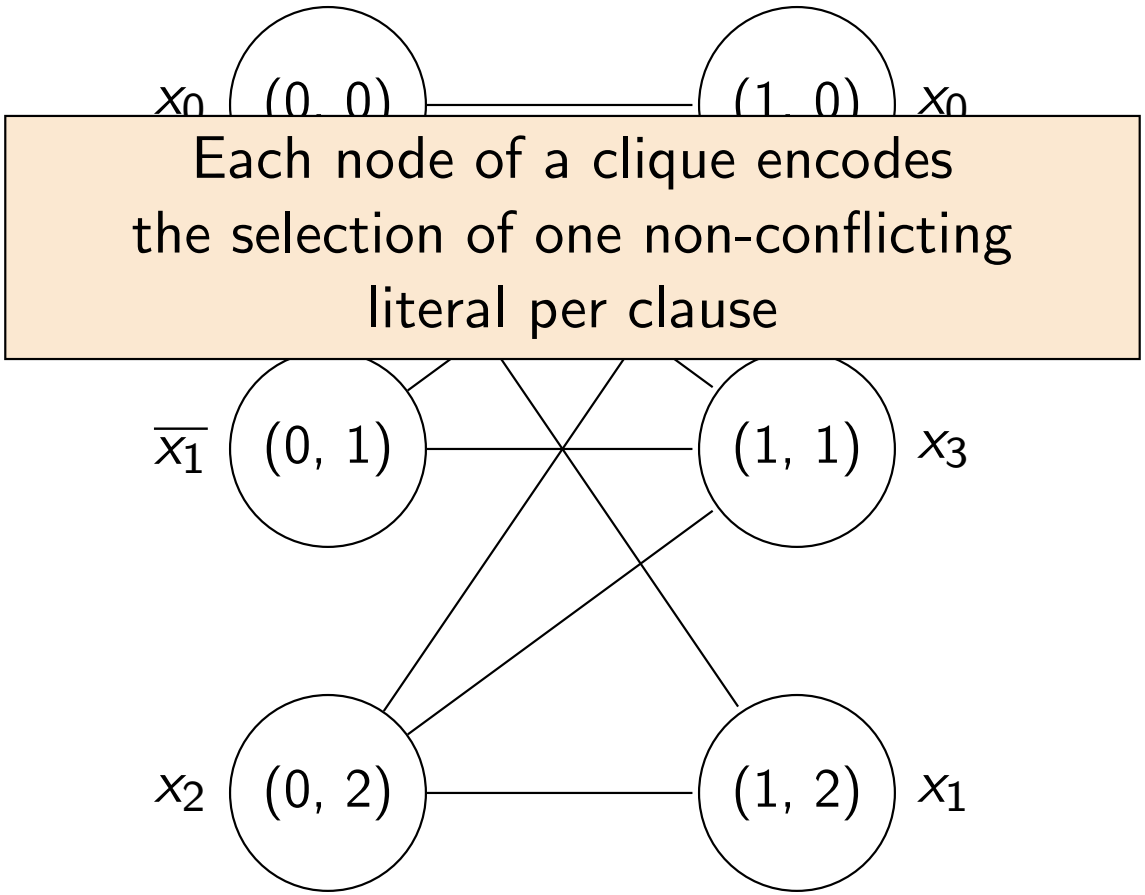


Reduction: Example

$N \mapsto (g, |N|)$

$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$

edge $\{L_1, L_2\}$ iff:
different clause L_1 L_2 and
non-conflicting L_1 L_2



Reduction Relation

$$N \overset{f}{\mapsto} (g, |N|)$$

$$\text{3-}\mathbf{SAT}\ N \leftrightarrow \mathbf{Clique}\ (g, |N|)$$

Reduction Relation

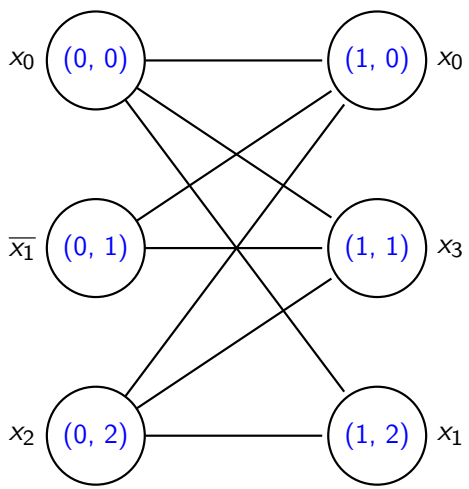
$$N \sim^{\textbf{3-SAT}}_{\textbf{Clique}} (f\ N)$$

$$N \sim^{\textbf{3-SAT}}_{\textbf{Clique}} (g, |N|) \rightarrow \textbf{3-SAT}\ N \leftrightarrow \textbf{Clique}\ (g, |N|)$$

Reduction Relation

$$N \sim^{\textbf{3-SAT}}_{\textbf{Clique}} (f\ N)$$

$$N \sim^{\textbf{3-SAT}}_{\textbf{Clique}} (g, |N|) \rightarrow \textbf{3-SAT}\ N \leftrightarrow \textbf{Clique}\ (g, |N|)$$



Reduction Function

$$f : \text{cnf} \to \text{graph} \times \mathbb{N} \text{ with } N \sim^{\textbf{3-SAT}}_{\textbf{Clique}} (f\ N)$$

$$A \preceq_p B := \exists f : X \to Y, \quad (\forall x, A\ x \leftrightarrow B(f\ x))$$
$$\wedge \text{ polynomial-time computable } f$$
$$\wedge \exists\ p, \forall x, \text{size}(\overline{f\ x}) \leq p(\text{size}(\overline{x}))$$

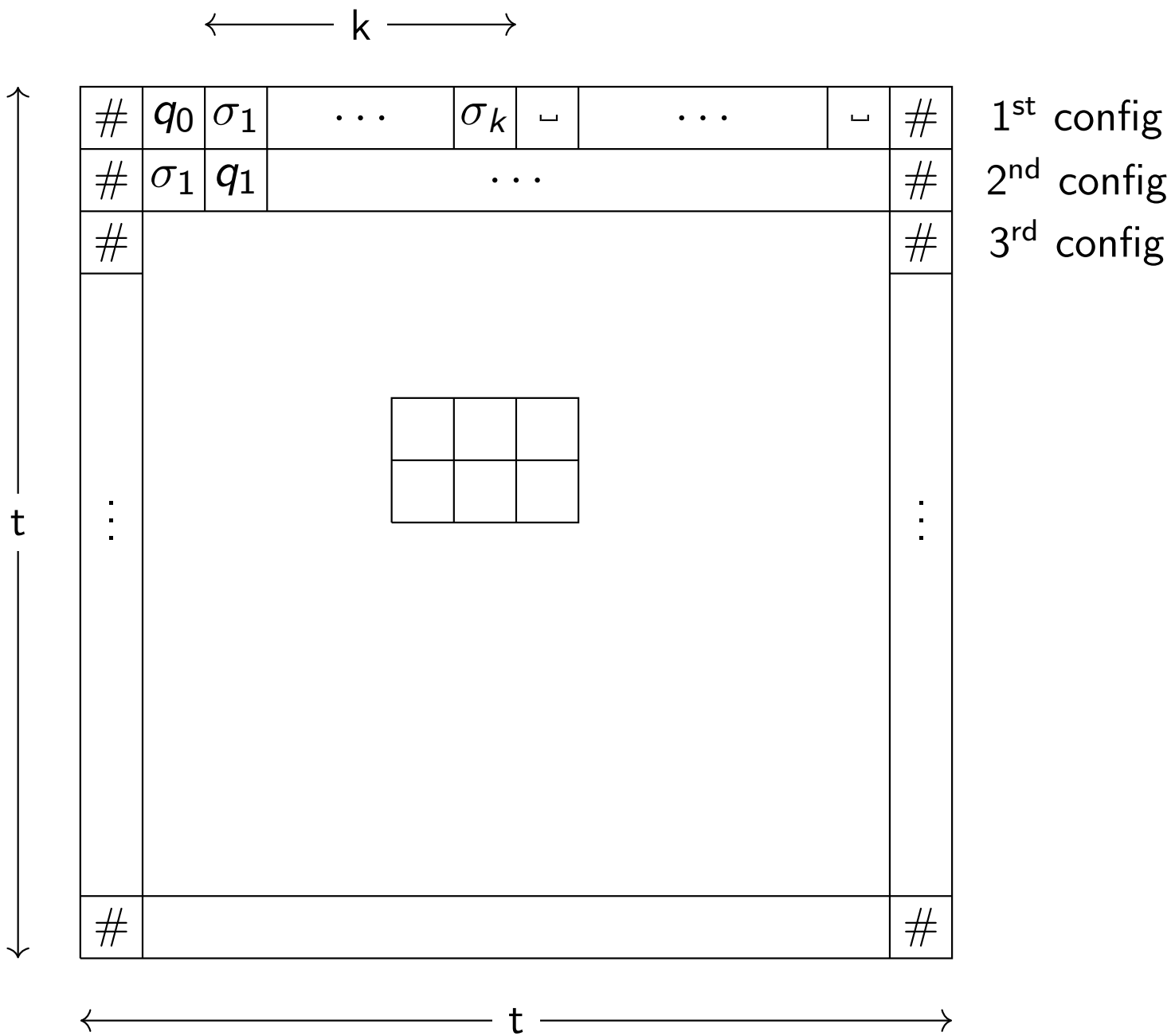
Cook's Theorem

SAT is NP-hard.

Cook's Theorem

SAT is NP-hard.

Encode the computation of a 1-tape Turing machine on an input of size k for t steps


k
t
q_0 σ_1 ⋯ σ_k ␣ ⋯ ␣
σ_1 q_1 ⋯
#
#
#
#
1st config
2nd config
3rd config
t

Conclusion

- a first formalised polynomial-time reduction from **3SAT** to **Clique**
- proofs that **SAT**, *k*-**SAT** and **Clique** are in NP
- nice proofs hard to obtain

Roadmap:

- work out the details of Cook's Theorem
- tools for SAT-programming/ intermediate problems
- a formalisation of Cook's Theorem
- ...

LOC

Component		Spec	Proof
Basic definitions[4]		271	619
Preliminaries		118	297
Higher-order RT		64	182
SAT	Main	69	225
	RT	75	227
k-SAT		24	70
Clique	Main	59	152
	RT	30	112
3-SAT to Clique	Main	264	667
	RT	41	199
Total		1015	2750

[4] due to F. Kunze

Cook's Theorem

SAT is NP-hard.

Cook's Theorem

SAT is NP-hard.

Generic Problem

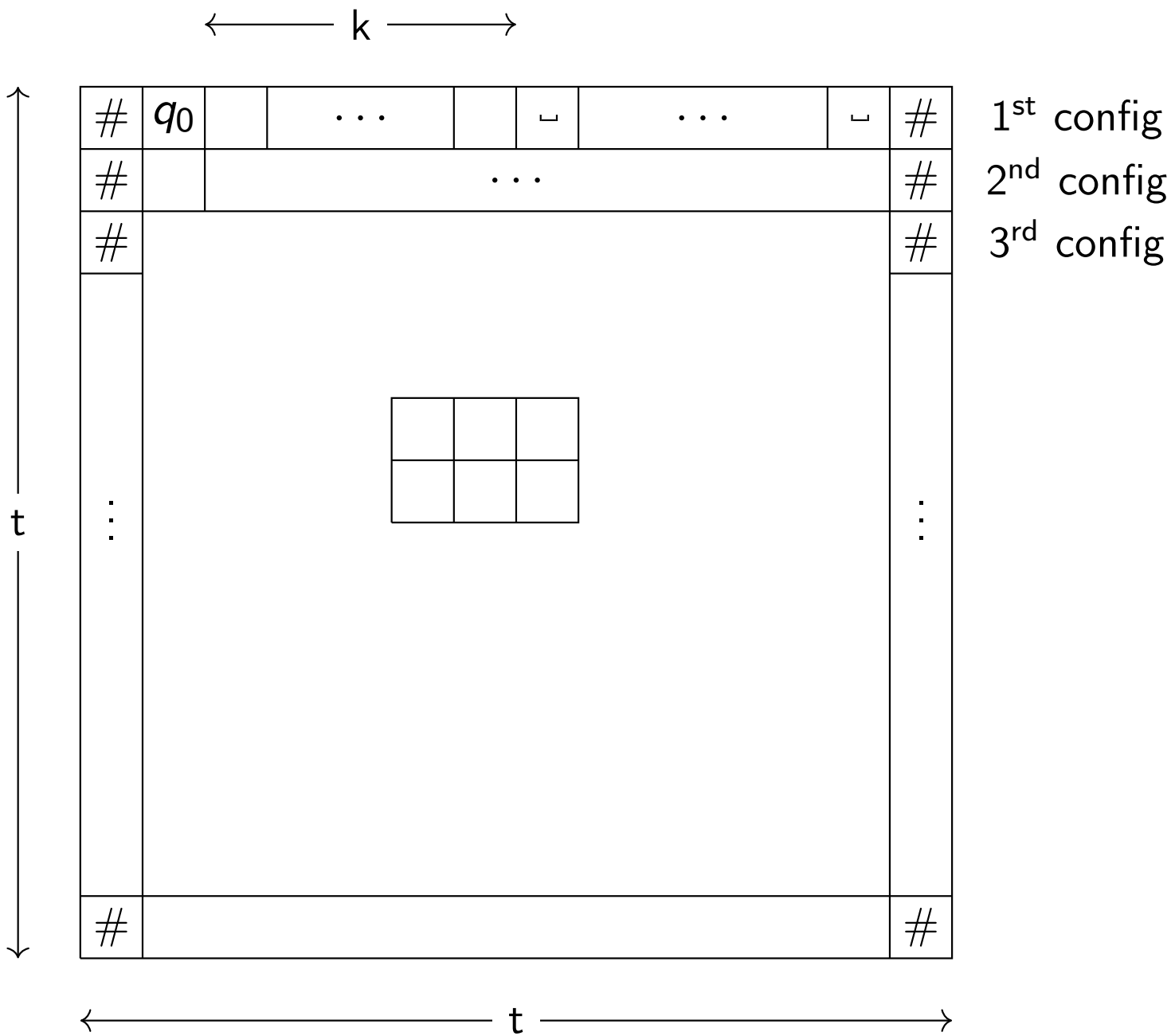
GenNP $(M, input, t) := M$ is a nondet. TM
$\wedge M$ accepts on $input$ in $\leq t$ steps

Cook's Theorem

SAT is NP-hard.

Generic Problem

$$\begin{aligned}\textbf{GenNP}\ (M,k,t) :=& M \text{ is a det. TM} \\ &\wedge \exists\ input, |input| \leq k \\ &\wedge M \text{ accepts on } input \text{ in } \leq t \text{ steps}\end{aligned}$$


k
t
t
q_0 ⊔ ⊔
1st config
2nd config
3rd config

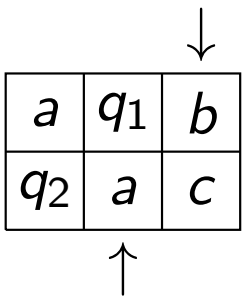
Cook's Theorem: Configuration Windows

$$\delta(q_1, a) = (q_1, b, R), \delta(q_1, b) = (q_2, c, L)$$

a	q_1	b
q_2	a	c

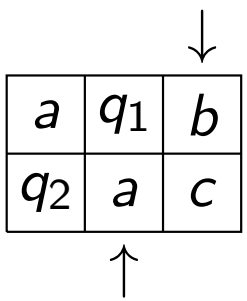
Cook's Theorem: Configuration Windows

$$\delta(q_1, a) = (q_1, b, R), \delta(q_1, b) = (q_2, c, L)$$



Cook's Theorem: Configuration Windows

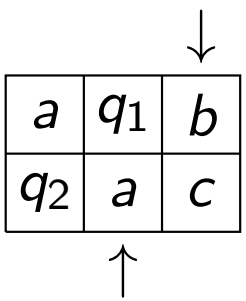
$$\delta(q_1, a) = (q_1, b, R), \delta(q_1, b) = (q_2, c, L)$$



✓

Cook's Theorem: Configuration Windows

$$\delta(q_1, a) = (q_1, b, R), \delta(q_1, b) = (q_2, c, L)$$

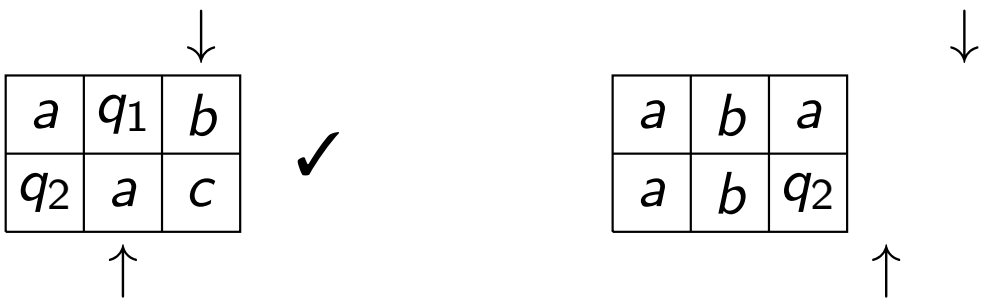


✓

a	b	a
a	b	q_2

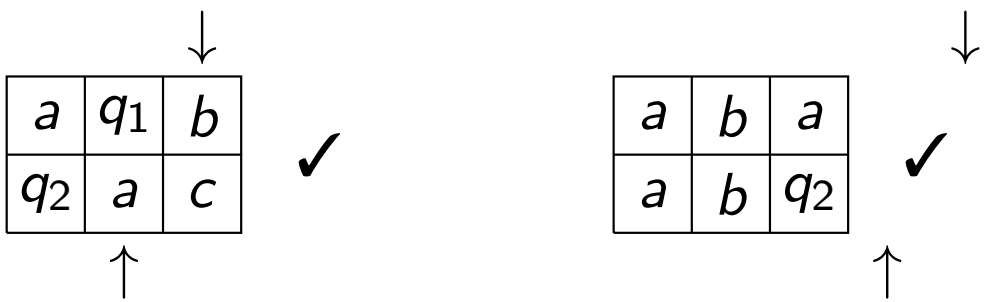
Cook's Theorem: Configuration Windows

$$\delta(q_1, a) = (q_1, b, R), \delta(q_1, b) = (q_2, c, L)$$



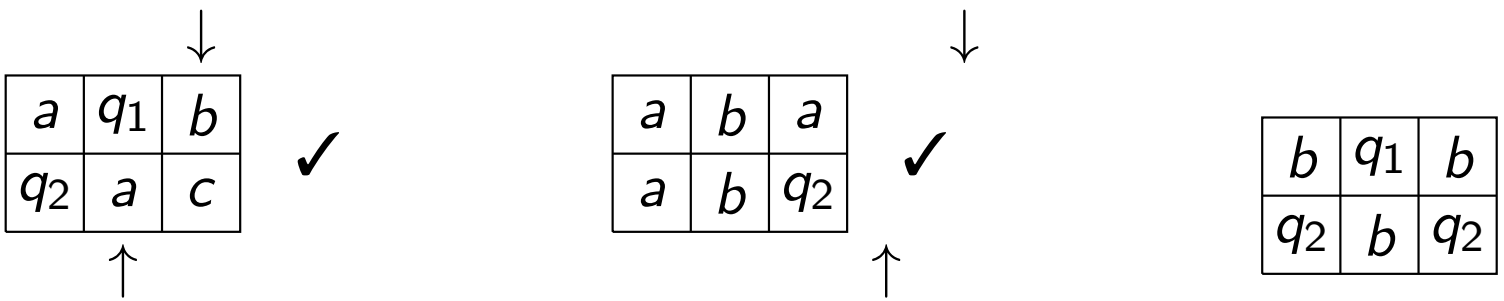
Cook's Theorem: Configuration Windows

$$\delta(q_1, a) = (q_1, b, R), \delta(q_1, b) = (q_2, c, L)$$



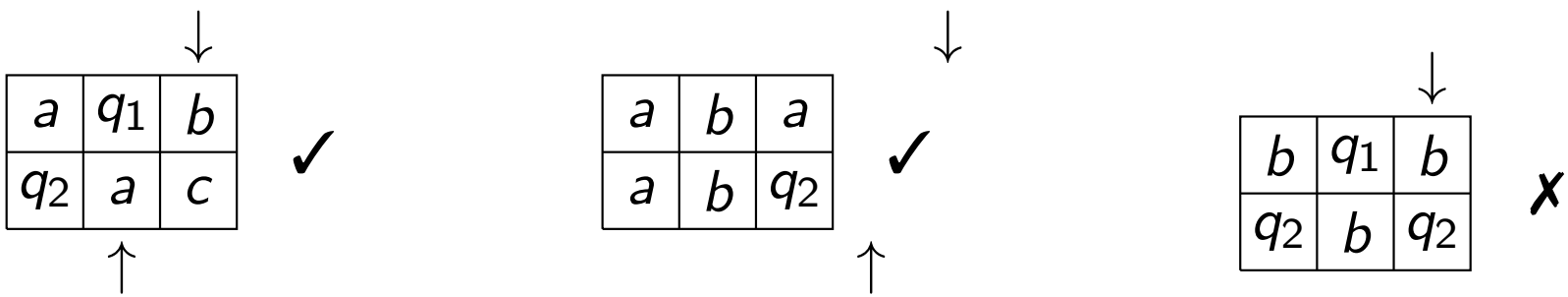
Cook's Theorem: Configuration Windows

$$\delta(q_1, a) = (q_1, b, R), \delta(q_1, b) = (q_2, c, L)$$



Cook's Theorem: Configuration Windows

$$\delta(q_1, a) = (q_1, b, R), \delta(q_1, b) = (q_2, c, L)$$



Problem: Abstractions

- explicit construction and analysis of reduction function

Problem: Abstractions

- explicit construction and analysis of reduction function
- simple representations due to extraction and time analysis

$$u, v : \text{node} := \mathbb{N}$$
$$e : \text{edge} := \text{node} \times \text{node}$$
$$g : \text{graph} := \mathbb{N} \times \mathcal{L}\ \ \text{edge}$$

Problem: Abstractions

- explicit construction and analysis of reduction function
- simple representations due to extraction and time analysis

$$u, v : \text{node} := \mathbb{N}$$
$$e : \text{edge} := \text{node} \times \text{node}$$
$$g : \text{graph} := \mathbb{N} \times \mathcal{L}\ \ \text{edge}$$

- "global" structure as in **Clique**

Time Analysis: Example

$$\begin{aligned}
&\text{forallb} \ : (A \to \mathbb{B}) \to \mathcal{L}\ A \to \mathbb{B} \\
&\text{forallb}\ f\ [] := \mathsf{T} \\
&\text{forallb}\ f\ (l :: ls) := f\ a\ \&\&\ \text{forallb}\ f\ ls
\end{aligned}$$

Time Analysis: Example

$$\begin{aligned}
&\text{forallb} \ : (A \to \mathbb{B}) \to \mathcal{L}\ A \to \mathbb{B} \\
&\text{forallb}\ f\ [] := \mathsf{T} \\
&\text{forallb}\ f\ (l :: ls) := f\ a\ \&\&\ \text{forallb}\ f\ ls
\end{aligned}$$

$$\text{forallb\_time}\ (fT : A \to \mathbb{N})\ l := 8 + \sum_{el \in l} (fT\ el + 15)$$

Time Analysis: Example

$$\begin{aligned}&\text{forallb} \ : (A \to \mathbb{B}) \to \mathcal{L}\ A \to \mathbb{B}\\&\text{forallb}\ f\ [] := \mathsf{T}\\&\text{forallb}\ f\ (l :: ls) := f\ a\ \&\&\ \text{forallb}\ f\ ls\end{aligned}$$

$$\text{forallb\_time}\ (fT : A \to \mathbb{N})\ l := 8 + \sum_{el \in l}(fT\ el + 15)$$

$$\begin{aligned}&\forall\ (fT : A \to \mathbb{N}),\\&\quad (\exists\ (p : \mathbb{N} \to \mathbb{N}), \forall\ el, fT\ el \leq p(\text{size}(\overline{el})))\\&\to \exists\ (p : \mathbb{N} \to \mathbb{N}), \forall\ l, \text{forallb\_time}\ fT\ l \leq p(\text{size}(\bar{l}))\end{aligned}$$

Time Analysis: Example

$$\begin{aligned}&\text{forallb} \ : (A \to \mathbb{B}) \to \mathcal{L}\ A \to \mathbb{B}\\ &\text{forallb}\ f\ [] := \mathsf{T}\\ &\text{forallb}\ f\ (l :: ls) := f\ a\ \&\&\ \text{forallb}\ f\ ls\end{aligned}$$

$$\text{forallb\_time}\ (fT : A \to \mathbb{N})\ l := 8 + \sum_{el \in l}(fT\ el + 15)$$

$$\begin{aligned}&\forall\ (fT : E \to A \to \mathbb{N}),\\ &\quad (\exists\ (p : \mathbb{N} \to \mathbb{N}), \forall\ el\ env, fT\ env\ el \leq p(\text{size}(\overline{el})+\text{size}(\overline{env})))\\ &\to \exists\ (p : \mathbb{N} \to \mathbb{N}), \forall\ l\ env, \text{forallb\_time}\ (fT\ env)\ l\\ &\qquad\qquad \leq p(\text{size}(\overline{l})+\text{size}(\overline{env}))\end{aligned}$$

Representation of SAT

$$x : \text{var} := \mathbb{N} \qquad L : \text{literal} := \mathbb{B} \times \text{var}$$

$$C : \text{clause} := \mathcal{L}\ \text{literal} \qquad N : \text{cnf} := \mathcal{L}\ \text{clause}$$

$$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$$

$$[[(\mathsf{T}, 0), (\mathsf{F}, 1), (\mathsf{T}, 2)], [(\mathsf{T}, 0), (\mathsf{T}, 3), (\mathsf{T}, 1)]]$$

$$a : \text{assgn} := \mathcal{L}\ \mathbb{B}$$

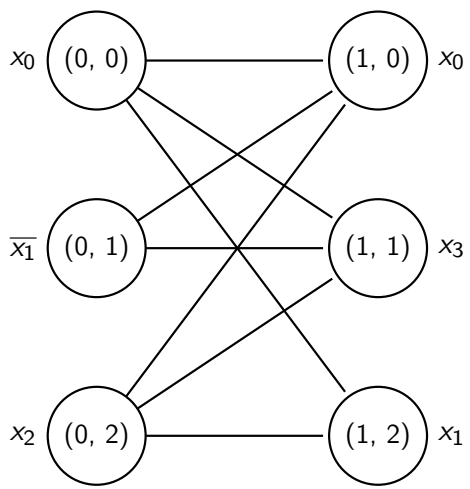
$$\{x_0 \mapsto \mathsf{T}, x_1 \mapsto \mathsf{F}, x_2 \mapsto \mathsf{F}, x_3 \mapsto \mathsf{F}\} \qquad [\mathsf{T}, \mathsf{F}, \mathsf{F}, \mathsf{F}]$$

Correctness

$N \sim^{\textbf{3-SAT}}_{\textbf{Clique}} (g, |N|)$

$$\textbf{3-SAT}\ N \leftrightarrow \textbf{Clique}\ (g, |N|)$$

$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$

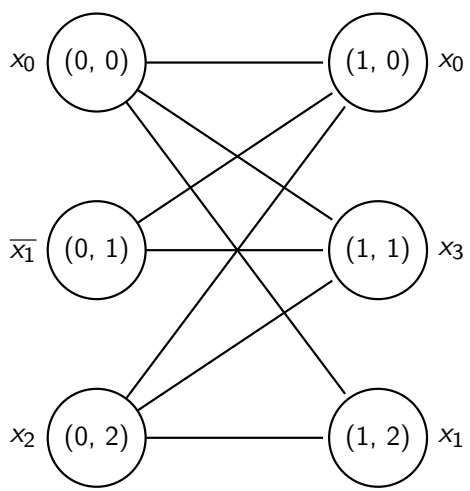


Correctness

$N \sim^{\textbf{3-SAT}}_{\textbf{Clique}} (g, |N|)$

$$(\exists\ a, \mathcal{E}\ a\ N\ =^{\circ} \mathsf{T}) \leftrightarrow (\exists\ cl, \mathsf{isClique}\ g\ cl\ |N|)$$

$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$



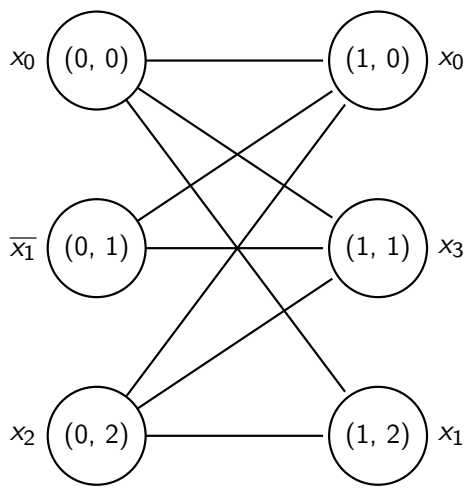
Correctness

$N \sim^{\textbf{3-SAT}}_{\textbf{Clique}} (g, |N|)$

$$(\exists\ a, \mathcal{E}\ a\ N\ =^{\circ} \mathsf{T}) \rightarrow (\exists\ cl, \mathsf{isClique}\ g\ cl\ |N|)$$

- Pick a satisfied literal for each clause
- A clique is given by the corresponding nodes

$$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$$

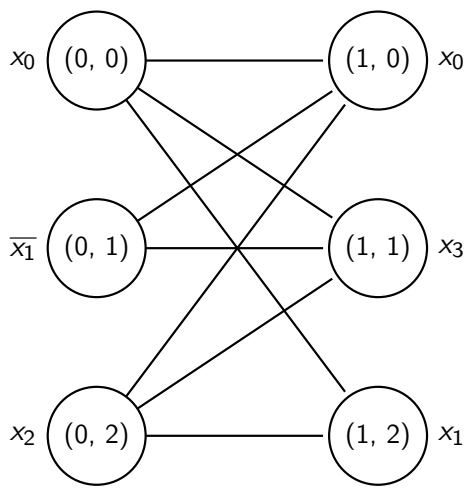


Correctness

$N \sim^{\textbf{3-SAT}}_{\textbf{Clique}} (g, |N|)$

$$(\exists\ a, \mathcal{E}\ a\ N\ =^{\circ} \mathsf{T}) \leftarrow (\exists\ cl, \mathsf{isClique}\ g\ cl\ |N|)$$

$$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$$

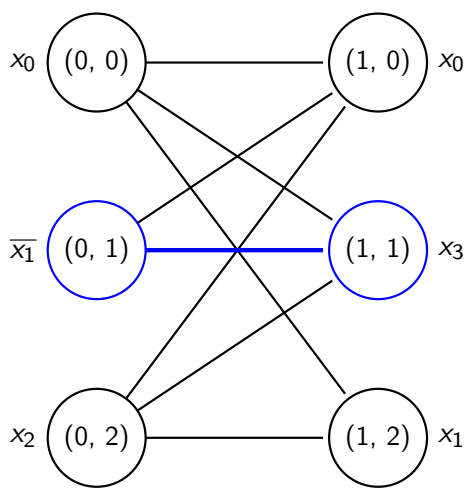


Correctness

$N \sim^{\textbf{3-SAT}}_{\textbf{Clique}} (g, |N|)$

$$(\exists\ a, \mathcal{E}\ a\ N\ =^{\circ} \mathsf{T}) \leftarrow (\exists\ cl, \mathsf{isClique}\ g\ cl\ |N|)$$

$$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$$



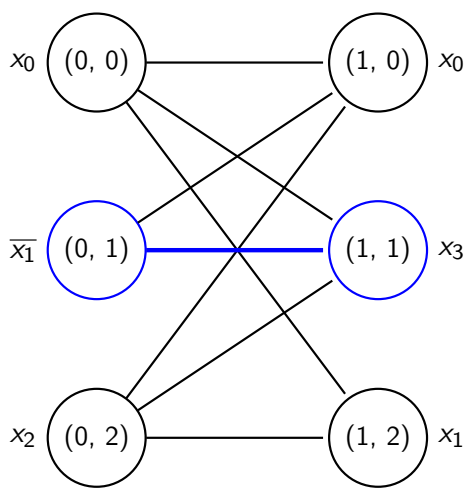
Correctness

$N \sim^{\textbf{3-SAT}}_{\textbf{Clique}} (g, |N|)$

$$(\exists\ a, \mathcal{E}\ a\ N\ =^{\circ} \mathsf{T}) \leftarrow (\exists\ cl, \mathsf{isClique}\ g\ cl\ |N|)$$

1. Map clique nodes to literal positions

$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$



$\rightarrow \quad [(0,1),(1,1)]$

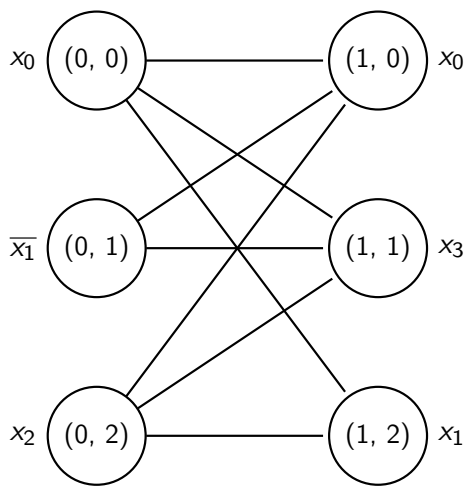
Correctness

$N \sim^{\textbf{3-SAT}}_{\textbf{Clique}} (g, |N|)$

$$(\exists\ a, \mathcal{E}\ a\ N\ =^{\circ} \mathsf{T}) \leftarrow (\exists\ cl, \mathsf{isClique}\ g\ cl\ |N|)$$

1. Map clique nodes to literal positions
2. Map literal positions to syntax of literals

$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$



$\rightarrow \quad [(0,1),(1,1)]$

$\rightarrow \quad [\overline{x_1}, x_3]$

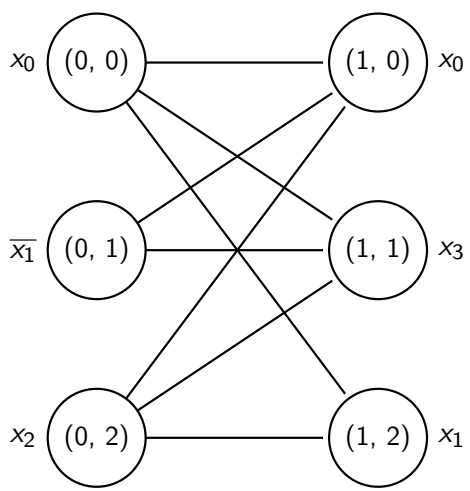
Correctness

$N \sim^{\textbf{3-SAT}}_{\textbf{Clique}} (g, |N|)$

$$(\exists\ a, \mathcal{E}\ a\ N\ =^{\circ} \mathsf{T}) \leftarrow (\exists\ cl, \mathsf{isClique}\ g\ cl\ |N|)$$

1. Map clique nodes to literal positions
2. Map literal positions to syntax of literals
3. Expand to full assignment

$(x_0 \vee \overline{x_1} \vee x_2) \wedge (x_0 \vee x_3 \vee x_1)$



$\rightarrow \quad [(0,1),(1,1)]$

$\rightarrow \quad [\overline{x_1}, x_3]$

$\rightarrow \quad \{x_0 \mapsto \mathsf{F}, x_1 \mapsto \mathsf{F}, x_2 \mapsto \mathsf{F}, x_3 \mapsto \mathsf{T}\}$

The Reduction Relation

Representation of graphs

$$\text{node} := \mathbb{N} \qquad \text{edge} := \text{node} \times \text{node} \qquad \text{graph} := \mathbb{N} \times \mathcal{L}\ \ \text{edge}$$

Labellings

$$\text{labG} := \mathbb{N} \to \mathbb{N} \times \mathbb{N} \qquad \text{labG}^{-1} := \mathbb{N} \times \mathbb{N} \to \mathbb{N}$$

$$\text{isLabelling}\ N\ (f : \text{labG})\ (f^{-1} : \text{labG}^{-1}) := \text{inverseOn}\ f\ f^{-1}$$
$$\{v \in \mathbb{N} | v < 3 \cdot |N|\}$$
$$\{(c, l) \in \mathbb{N}^2 | c < |N| \wedge l < 3\}$$

The Reduction Relation

$N \sim^{\textbf{3-SAT}}_{\textbf{Clique}} ((n, e), k)$, iff :

- N is a 3-CNF
- $n = 3 \cdot |N| \wedge k = |N|$
- there is a labelling (f, f^{-1}) s.t. $\forall u\ v < n$, edgeIn $(n, e)\ u\ v$ is equivalent to
  - $f\ u$ and $f\ v$ being in different clauses, and
  - $\forall L_1\ L_2$, if L_1 is the $(f\ u)$-th literal of N and L_2 is the $(f\ v)$-th literal of N, then L_1 and L_2 do not conflict

The Reduction Function

$$\text{red } N := (\textbf{if } \text{kCNF\_dec } 3 \ N \textbf{ then } (3 \cdot |N|, \text{makeEdges } N) \textbf{ else } (0, []), |N|)$$

$$labF \ n := (n/3, n \mod 3) \qquad labF^{-1} \ (cl, l) := 3 \cdot cl + l$$

L: Size Explosion

Church encoding of numbers: $\overline{n} := \lambda\ f\ x.(f(f(f\cdots(f\ x)\cdots)))$

$$c := \lambda\ x.x\ \overline{2}\ (\lambda\ x.x)$$

$$\begin{aligned} c\ \overline{n} &\succ^3 (\overline{2}(\overline{2}(\overline{2}\cdots(\overline{2}(\lambda\ x.x))\cdots))) \\ &\succ (\overline{2}(\overline{2}(\overline{2}\cdots(\lambda\ x.(\lambda\ x.x)((\lambda\ x.x)\ x))\cdots))) \\ &\succ \cdots \end{aligned}$$

References


Bläser, M.
Theoretical computer science: An introduction.


Forster, Y. and Kunze, F. (2019).
A certifying extraction with time bounds from coq to call-by-value lambda-calculus.
In *Interactive Theorem Proving - 10th International Conference, ITP 2019, Portland, USA*.
Also available as arXiv:1904.11818.


Forster, Y., Kunze, F., and Roth, M. (2019).
The weak call-by-value lambda-calculus is reasonable for both time and space.
Technical report.
Full version appeared as arXiv:1902.07515 To appear.

References


Forster, Y. and Smolka, G. (2017).
Weak call-by-value lambda calculus as a model of computation in coq.
In *Interactive Theorem Proving - 8th International Conference, ITP 2017, Brasilia, Brazil, September 26-29, 2017*.


Sipser, M.
Introduction to Theory of Computation.
1 edition.