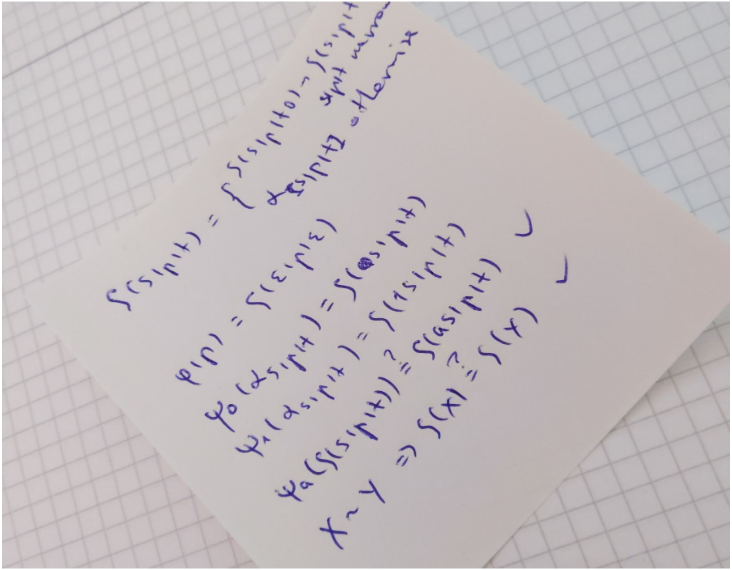
# Undecidability of Semi-unification on a Napkin

Andrej Dudenhefner

Saarland University, Saarbrücken, Germany

FSCD 2020
2020-07-02, Paris, France

# Undecidability of Semi-unification on a Napkin



# Semi-unification

## Definition (Terms $\mathbb{T}$)

$$\mathbb{T} \ni \sigma, \tau ::= \alpha \mid \sigma \to \tau \quad \text{where } \alpha \text{ ranges over variables } \mathbb{V}$$

- Semi-unification $\sim$ first-order unification combined with matching

## Problem (Semi-unification)

*Given inequalities* $\mathcal{I} = \{\sigma_1 \leq \tau_1, \ldots, \sigma_n \leq \tau_n\}$,
*is there a substitution* $\varphi : \mathbb{V} \to \mathbb{T}$ *such that*
*for each inequality* $(\sigma \leq \tau) \in \mathcal{I}$
*there is a substitution* $\psi : \mathbb{V} \to \mathbb{T}$ *such that*
$\psi(\varphi(\sigma)) = \varphi(\tau)$*?*

## Theorem ([Kfoury, Tiuryn, and Urzyczyn 1993a])

*Semi-unification is undecidable.*

# Semi-unification Occurrences

- Type inference in polymorphic functional programming
  [Leiß 1989; Kfoury, Tiuryn, and Urzyczyn 1993b; Henglein 1993]
- Type inference in polymorphic logic programming
  [Mycroft and O'Keefe 1984]
- System F type checking
  [Wells 1999]
- Loop detection in term rewriting
  [Purdom 1987]
- Program flow analysis
  [Fähndrich, Rehof, and Das 2000]
- Natural language processing
  [Dörre and Rounds 1990]
- ...

# Semi-unification Example

## Example (Composed Iteration)

- `iter2 :: Nat -> (a -> b) -> (b -> a) -> a -> a`
- `iter2 0 f g x = x`
- `iter2 1 f g x = g (f x)`
- `iter2 2 f g x = g (f (g (f x)))`

  ...

In Haskell

```
iter2 0 f g x = x
iter2 n f g x = g (iter2 (n-1) g f (f x))
```

has type

```
iter2 :: Nat -> (a -> a) -> (a -> a) -> a -> a
```

where types of `f` and `g` are unified.

# Semi-unification Example

## Example (Composed Iteration)

```
iter2 0 f g x = x
iter2 n f g x = g (iter2 (n-1) g f (f x))
```

- **Parametric polymorphism:** *monomorphic* recursive calls
  $\rightsquigarrow$ find substitution $\varphi$ such that
  $$\begin{aligned}&\varphi(\mathsf{Nat} \to (\alpha \to \beta) \to (\beta \to \alpha) \to \alpha \to \alpha)\\ =\,&\varphi(\mathsf{Nat} \to (\beta \to \alpha) \to (\alpha \to \beta) \to \beta \to \beta)\end{aligned}$$
  $\rightsquigarrow$ $\varphi = \{\alpha \Rrightarrow \mathsf{a}, \beta \Rrightarrow \mathsf{a}\}$
- **Recursive polymorphism:** *instantiated* recursive calls
  $\rightsquigarrow$ find substitutions $\varphi, \psi$ such that
  $$\begin{aligned}&\psi(\varphi(\mathsf{Nat} \to (\alpha \to \beta) \to (\beta \to \alpha) \to \alpha \to \alpha))\\ =\,&\varphi(\mathsf{Nat} \to (\beta \to \alpha) \to (\alpha \to \beta) \to \beta \to \beta)\end{aligned}$$
  $\rightsquigarrow$ $\varphi = \{\alpha \Rrightarrow \alpha, \beta \Rrightarrow \beta\}$, $\psi = \{\alpha \Rrightarrow \beta, \beta \Rrightarrow \alpha\}$
  Different $\psi$ for individual recursive calls $\rightsquigarrow$ **semi-unification**

# Semi-unification Undecidability

## Original Proof Synopsis.

**Turing machine immortality** [Hooper 1966]
(is there an non-terminating configuration?)

$\leq$ **Turing machine uniform boundedness**
(is the number of reachable configurations uniformly bounded?)

$\leq$ **Symmetric intercell Turing machine uniform boundedness**
(as above; returning to potential past configurations)

$\leq$ **Path equation derivability**
(reachability in a tailored rewriting system)

$\leq$ **Unification constraint normalization**
(halting in a tailored redex contraction system)

$\leq$ **Semi-unification**

- Uses excluded middle and König's lemma $\square$

# Semi-unification Undecidability

**New Proof Synopsis.**

**Turing machine immortality** [Hooper 1966]
(is there an non-terminating configuration?)

$\leq$ **Stack machine uniform boundedness**
(is the number of reachable configurations uniformly bounded?)

$\leq$ **Semi-unification**

- First step uses fan theorem (Brouwer's intuitionism)
- Second step is fully constructive (axiom-free Coq, 1500 loc) $\square$

# Simple Stack Machine

## Definition (Simple Stack Machine)

**Instruction:** $ap \longrightarrow qb$ or $pb \longrightarrow aq$
where $p, q$ are *states* and $a, b \in \{0, 1\}$ are *symbols*

**Simple stack machine:** list of instructions $\mathcal{M}$

**Configuration:** $s|p|t$
where $p$ is a state and $s, t \in \{0, 1\}^*$ are words

**Step relation:**
$sa|p|t \longrightarrow_{\mathcal{M}} s|q|bt$ if $(ap \longrightarrow qb) \in \mathcal{M}$
$s|p|bt \longrightarrow_{\mathcal{M}} sa|q|t$ if $(pb \longrightarrow aq) \in \mathcal{M}$

- Simple stack machine $\sim$ space-bounded intercell Turing machine

## Problem (Uniform Boundedness)

*Given a simple stack machine* $\mathcal{M}$*,*
*is there an* $n \in \mathbb{N}$ *such that for any configuration* $X$ *we have*
$|\{Y \mid X \longrightarrow^*_{\mathcal{M}} Y\}| \leq n$*?*

# Simple Stack Machine Properties

- Mechanization-friendly
  (specification 30 loc)
- No infinite tape
  (linear automaton)
- Decidable reachability and termination
  (every run operates in bounded space)
- Undecidable uniform boundedness
  (similar to **Turing machine immortality** $\leq$ **Turing machine uniform boundedness**)

# Simple Semi-unification

## Definition (Simple Constraint)

**Simple constraint:** $a_{|}\alpha_{|}\epsilon \doteq \epsilon_{|}\beta_{|}b$
where $\alpha \in \mathbb{V}$ and $a, b \in \{0, 1\}$

**Model:** $(\varphi, \psi_0, \psi_1) \models a_{|}\alpha_{|}\epsilon \doteq \epsilon_{|}\beta_{|}b$ if either

- $b = 0$ and $\psi_a(\varphi(\alpha)) \to \tau = \varphi(\beta)$ for some $\tau$
- $b = 1$ and $\sigma \to \psi_a(\varphi(\alpha)) = \varphi(\beta)$ for some $\sigma$

## Definition (Simple Semi-unification)

Given a finite set $\mathcal{C}$ of simple constraints,
are there substitutions $\varphi, \psi_0, \psi_1 : \mathbb{V} \to \mathbb{T}$ such that
for all constraints $C \in \mathcal{C}$ we have $(\varphi, \psi_0, \psi_1) \models C$?

- Undecidable fragment of semi-unification

# Not Uniformly Bounded Example

## Example (Not Uniformly Bounded Stack Machine)

$\mathcal{M} = \{\mathbf{0}p \longrightarrow p\mathbf{1}\}$

# Not Uniformly Bounded Example

## Example (Not Uniformly Bounded Stack Machine)

$\mathcal{M} = \{0p \longrightarrow p1\}$

$$000_{|}p_{|}\epsilon$$

# Not Uniformly Bounded Example

## Example (Not Uniformly Bounded Stack Machine)

$\mathcal{M} = \{\mathbf{0}\boldsymbol{p} \longrightarrow \boldsymbol{p}\mathbf{1}\}$

$$\mathbf{000}_{\shortmid}\boldsymbol{p}_{\shortmid}\epsilon \longrightarrow_{\mathcal{M}} \mathbf{00}_{\shortmid}\boldsymbol{p}_{\shortmid}\mathbf{1}$$

# Not Uniformly Bounded Example

## Example (Not Uniformly Bounded Stack Machine)

$\mathcal{M} = \{\mathbf{0}p \longrightarrow p\mathbf{1}\}$

$$\mathbf{000}_{|}p_{|}\epsilon \longrightarrow_{\mathcal{M}} \mathbf{00}_{|}p_{|}\mathbf{1} \longrightarrow_{\mathcal{M}} \mathbf{0}_{|}p_{|}\mathbf{11}$$

# Not Uniformly Bounded Example

## Example (Not Uniformly Bounded Stack Machine)

$\mathcal{M} = \{0p \longrightarrow p1\}$

$$000_{|}p_{|}\epsilon \longrightarrow_{\mathcal{M}} 00_{|}p_{|}1 \longrightarrow_{\mathcal{M}} 0_{|}p_{|}11 \longrightarrow_{\mathcal{M}} \epsilon_{|}p_{|}111$$

# Not Uniformly Bounded Example

## Example (Not Uniformly Bounded Stack Machine)

$\mathcal{M} = \{0p \longrightarrow p1\}$

$$000 \mid p \mid \epsilon \longrightarrow_{\mathcal{M}} 00 \mid p \mid 1 \longrightarrow_{\mathcal{M}} 0 \mid p \mid 11 \longrightarrow_{\mathcal{M}} \epsilon \mid p \mid 111$$

$\rightsquigarrow$ $0^n \mid p \mid \epsilon$ reaches $n + 1$ distinct configurations

$\rightsquigarrow$ no *uniform* bound on number of reachable configurations

# Not Uniformly Bounded Example

## Example (Not Uniformly Bounded Stack Machine)

$\mathcal{M} = \{\mathbf{0}p \longrightarrow p\mathbf{1}\}$

$$\mathbf{000}|p|\epsilon \longrightarrow_{\mathcal{M}} \mathbf{00}|p|\mathbf{1} \longrightarrow_{\mathcal{M}} \mathbf{0}|p|\mathbf{11} \longrightarrow_{\mathcal{M}} \epsilon|p|\mathbf{111}$$

$\rightsquigarrow$ $\mathbf{0}^n|p|\epsilon$ reaches $n+1$ distinct configurations

$\rightsquigarrow$ no *uniform* bound on number of reachable configurations

- $\mathbf{0}p \longrightarrow p\mathbf{1} \rightsquigarrow \mathbf{0}|p|\epsilon \doteq \epsilon|p|\mathbf{1}$

## Example (Unsolvable Constraints)

$\mathcal{C} = \{\mathbf{0}|p|\epsilon \doteq \epsilon|p|\mathbf{1}\}$

- $\sigma \rightarrow \psi_0(\varphi(p)) = \varphi(p)$

$\rightsquigarrow$ no model

# Uniformly Bounded Example

### Example (Uniformly Bounded Stack Machine)

$\mathcal{M} = \{0p \longrightarrow q1, q1 \longrightarrow 1p, 1p \longrightarrow q0, q0 \longrightarrow 0p\}$

# Uniformly Bounded Example

## Example (Uniformly Bounded Stack Machine)

$$\mathcal{M} = \{\mathbf{0}p \longrightarrow q\mathbf{1}, q\mathbf{1} \longrightarrow \mathbf{1}p, \mathbf{1}p \longrightarrow q\mathbf{0}, q\mathbf{0} \longrightarrow \mathbf{0}p\}$$

$\mathbf{0}|p|\epsilon$

# Uniformly Bounded Example

## Example (Uniformly Bounded Stack Machine)

$$\mathcal{M} = \{\mathbf{0}p \longrightarrow q\mathbf{1}, q\mathbf{1} \longrightarrow \mathbf{1}p, \mathbf{1}p \longrightarrow q\mathbf{0}, q\mathbf{0} \longrightarrow \mathbf{0}p\}$$

$$\mathbf{0}|p|\epsilon \longrightarrow_{\mathcal{M}} \epsilon|q|\mathbf{1}$$

# Uniformly Bounded Example

## Example (Uniformly Bounded Stack Machine)

$$\mathcal{M} = \{\mathbf{0}p \longrightarrow q\mathbf{1}, q\mathbf{1} \longrightarrow \mathbf{1}p, \mathbf{1}p \longrightarrow q\mathbf{0}, q\mathbf{0} \longrightarrow \mathbf{0}p\}$$

$$\mathbf{0}_{|}p_{|}\epsilon \longrightarrow_{\mathcal{M}} \epsilon_{|}q_{|}\mathbf{1} \longrightarrow_{\mathcal{M}} \mathbf{1}_{|}p_{|}\epsilon$$

# Uniformly Bounded Example

## Example (Uniformly Bounded Stack Machine)

$$\mathcal{M} = \{\mathbf{0}p \longrightarrow q\mathbf{1}, q\mathbf{1} \longrightarrow \mathbf{1}p, \mathbf{1}p \longrightarrow q\mathbf{0}, q\mathbf{0} \longrightarrow \mathbf{0}p\}$$

$$\mathbf{0}|p|\epsilon \longrightarrow_{\mathcal{M}} \epsilon|q|\mathbf{1} \longrightarrow_{\mathcal{M}} \mathbf{1}|p|\epsilon \longrightarrow_{\mathcal{M}} \epsilon|q|\mathbf{0}$$

# Uniformly Bounded Example

## Example (Uniformly Bounded Stack Machine)

$$\mathcal{M} = \{0p \longrightarrow q1, q1 \longrightarrow 1p, 1p \longrightarrow q0, q0 \longrightarrow 0p\}$$

$$0|p|\epsilon \longrightarrow_{\mathcal{M}} \epsilon|q|1 \longrightarrow_{\mathcal{M}} 1|p|\epsilon \longrightarrow_{\mathcal{M}} \epsilon|q|0 \longrightarrow_{\mathcal{M}} 0|p|\epsilon$$

# Uniformly Bounded Example

## Example (Uniformly Bounded Stack Machine)

$$\mathcal{M} = \{0p \longrightarrow q1, q1 \longrightarrow 1p, 1p \longrightarrow q0, q0 \longrightarrow 0p\}$$

$$0|p|\epsilon \longrightarrow_{\mathcal{M}} \epsilon|q|1 \longrightarrow_{\mathcal{M}} 1|p|\epsilon \longrightarrow_{\mathcal{M}} \epsilon|q|0 \longrightarrow_{\mathcal{M}} 0|p|\epsilon$$

$\rightsquigarrow$ any configuration reaches at most **4** distinct configurations

# Uniformly Bounded Example

## Example (Uniformly Bounded Stack Machine)

$$\mathcal{M} = \{0p \longrightarrow q1, q1 \longrightarrow 1p, 1p \longrightarrow q0, q0 \longrightarrow 0p\}$$

$$0_{|}p_{|}\epsilon \longrightarrow_{\mathcal{M}} \epsilon_{|}q_{|}1 \longrightarrow_{\mathcal{M}} 1_{|}p_{|}\epsilon \longrightarrow_{\mathcal{M}} \epsilon_{|}q_{|}0 \longrightarrow_{\mathcal{M}} 0_{|}p_{|}\epsilon$$

$\rightsquigarrow$ any configuration reaches at most **4** distinct configurations

## Example (Solvable Constraints)

$$\mathcal{C} = \{0_{|}p_{|}\epsilon \doteq \epsilon_{|}q_{|}1, 1_{|}p_{|}\epsilon \doteq \epsilon_{|}q_{|}1, 1_{|}p_{|}\epsilon \doteq \epsilon_{|}q_{|}0, 0_{|}p_{|}\epsilon \doteq \epsilon_{|}q_{|}0\}$$

- $\sigma \rightarrow \psi_0(\varphi(p)) = \varphi(q)$
- $\sigma \rightarrow \psi_1(\varphi(p)) = \varphi(q)$
- $\psi_1(\varphi(p)) \rightarrow \tau = \varphi(q)$
- $\psi_0(\varphi(p)) \rightarrow \tau = \varphi(q)$

$\rightsquigarrow$ model

$$\varphi(p) = \alpha$$
$$\varphi(q) = \beta \rightarrow \beta$$
$$\psi_0(\alpha) = \psi_1(\alpha) = \beta$$

# Reduction Soundness

## Definition (Machine Encoding)

Given simple stack machine $\mathcal{M}$, define

$$\mathcal{C} = \{ a_{|}p_{|}\epsilon \doteq \epsilon_{|}q_{|}b \mid (ap \longrightarrow qb) \in \mathcal{M} \text{ or } (qb \longrightarrow ap) \in \mathcal{M} \}$$

## Definition ($\zeta$)

$$\zeta(s_{|}p_{|}t) = \begin{cases} \zeta(s_{|}p_{|}t0) \to \zeta(s_{|}p_{|}t1) & \text{if } s_{|}p_{|}t \text{ is } \textit{narrow} \\ \alpha_{[s_{|}p_{|}t]} & \text{otherwise} \end{cases}$$

where narrowness is decidable and $[\cdot]$ is a total and computable

## Lemma (Reduction Soundness)

*If $\mathcal{M}$ is uniformly bounded, then* $(\varphi, \psi_0, \psi_1) \models \mathcal{C}$ *where*

$$\varphi(p) = \zeta(\epsilon_{|}p_{|}\epsilon) \qquad \psi_0(\alpha_{s_{|}p_{|}t}) = \zeta(0s_{|}p_{|}t) \qquad \psi_1(\alpha_{s_{|}p_{|}t}) = \zeta(1s_{|}p_{|}t)$$

# Reduction Completeness

## Definition (Machine Encoding)

Given simple stack machine $\mathcal{M}$, define

$$\mathcal{C} = \{ a_{|}p_{|}\epsilon \doteq \epsilon_{|}q_{|}b \mid (ap \longrightarrow qb) \in \mathcal{M} \text{ or } (qb \longrightarrow ap) \in \mathcal{M} \}$$

## Lemma

$X \longrightarrow^*_{\mathcal{M}} Y$ *and* $(\varphi, \psi_0, \psi_1) \models \mathcal{C}$ *implies* $(\varphi, \psi_0, \psi_1) \models X \doteq Y$

## Remark

Size of the syntax tree of $\varphi(p)$ uniformly bounds reachable configuration space from state $p$.

## Lemma (Reduction Completeness)

*If* $(\varphi, \psi_0, \psi_1) \models \mathcal{C}$*, then* $\mathcal{M}$ *is uniformly bounded.*

# Contribution

- Intuitionistic (in sense of Brouwer) Turing reduction from **Turing machine immortality** to **simple stack machine uniform boundedness**
- Fully constructive many-one reduction from **simple stack machine uniform boundedness** to **semi-unification**
  - simple and direct via $\zeta$
  - mechanized (axiom-free Coq) (specification 100 loc, argument 1400 loc) `https://github.com/uds-psl/2020-fscd-semi-unification`

# Ongoing Work

Mechanized reduction from
the **Turing machine halting problem**
to **semi-unification**

- comprehensive
  (current mechanization starts with boundedness)
- many-one
  (current proof requires Turing reductions)
- axiom-free
  (current proof requires fan theorem)
- part of the Coq library of Undecidability Proofs
  `https://github.com/uds-psl/coq-library-undecidability`

# Ongoing Work

Mechanized reduction from
the **Turing machine halting problem**
to **semi-unification**

- comprehensive
  (current mechanization starts with boundedness)
- many-one
  (current proof requires Turing reductions)
- axiom-free
  (current proof requires fan theorem)
- part of the Coq library of Undecidability Proofs
  `https://github.com/uds-psl/coq-library-undecidability`

# Thank You

# Bibliography I


Dörre, Jochen and William C. Rounds (1990). "On Subsumption and Semiunification in Feature Algebras." In: *Proceedings of the Fifth Annual Symposium on Logic in Computer Science (LICS '90), Philadelphia, Pennsylvania, USA, June 4-7, 1990*. IEEE Computer Society, pp. 300–310. DOI: 10.1109/LICS.1990.113756. URL: https://doi.org/10.1109/LICS.1990.113756.

Fähndrich, Manuel, Jakob Rehof, and Manuvir Das (2000). "Scalable context-sensitive flow analysis using instantiation constraints." In: *Proceedings of the 2000 ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI), Vancouver, Britith Columbia, Canada, June 18-21, 2000*. Ed. by Monica S. Lam. ACM, pp. 253–263. DOI: 10.1145/349299.349332. URL: https://doi.org/10.1145/349299.349332.

# Bibliography II


Henglein, Fritz (1993). "Type Inference with Polymorphic Recursion." In: *ACM Trans. Program. Lang. Syst.* 15.2, pp. 253–289. DOI: 10.1145/169701.169692. URL: https://doi.org/10.1145/169701.169692.

Hooper, Philip K. (1966). "The Undecidability of the Turing Machine Immortality Problem." In: *J. Symb. Log.* 31.2, pp. 219–234. DOI: 10.2307/2269811. URL: https://doi.org/10.2307/2269811.

Kfoury, Assaf J., Jerzy Tiuryn, and Pawel Urzyczyn (1993a). "The Undecidability of the Semi-unification Problem." In: *Inf. Comput.* 102.1, pp. 83–101. DOI: 10.1006/inco.1993.1003. URL: https://doi.org/10.1006/inco.1993.1003.

– (1993b). "Type Reconstruction in the Presence of Polymorphic Recursion." In: *ACM Trans. Program. Lang. Syst.* 15.2, pp. 290–311. DOI: 10.1145/169701.169687. URL: https://doi.org/10.1145/169701.169687.

# Bibliography III


Leiß, Hans (1989). "Polymorphic recursion and semi-unification." In: *International Workshop on Computer Science Logic*. Springer, pp. 211–224.

Mycroft, Alan and Richard A. O'Keefe (1984). "A Polymorphic Type System for Prolog." In: *Artif. Intell.* 23.3, pp. 295–307. DOI: 10.1016/0004-3702(84)90017-1. URL: https://doi.org/10.1016/0004-3702(84)90017-1.

Purdom, Paul Walton (1987). "Detecting Looping Simplifications." In: *Rewriting Techniques and Applications, 2nd International Conference, RTA-87, Bordeaux, France, May 25-27, 1987, Proceedings*. Ed. by Pierre Lescanne. Vol. 256. Lecture Notes in Computer Science. Springer, pp. 54–61. DOI: 10.1007/3-540-17220-3_5. URL: https://doi.org/10.1007/3-540-17220-3_5.

# Bibliography IV

Wells, J. B. (1999). "Typability and Type Checking in System F are Equivalent and Undecidable." In: *Ann. Pure Appl. Log.* 98.1-3, pp. 111–156. DOI: 10.1016/S0168-0072(98)00047-5. URL: https://doi.org/10.1016/S0168-0072(98)00047-5.

# Backup Slides

# Constraint-based Semi-unification

## Definition (Substitution Composition)

For substitutions $\psi_0, \psi_1 : \mathbb{V} \to \mathbb{T}$ and word $v \in \{0,1\}^*$ define

$$\psi_\epsilon(\sigma) = \sigma \qquad \psi_{v0}(\sigma) = \psi_v(\psi_0(\sigma)) \qquad \psi_{v1}(\sigma) = \psi_v(\psi_1(\sigma))$$

## Definition (Path Function)

For $w \in \{0,1\}^*$ define

$$\pi_\epsilon(\sigma) = \sigma \qquad \pi_{0w}(\sigma \to \tau) = \pi_w(\sigma) \qquad \pi_{1w}(\sigma \to \tau) = \pi_w(\tau)$$

## Definition (Constraint)

**Constraint:** $s|\alpha|t \doteq v|\beta|w$
where $\alpha, \beta \in \mathbb{V}$ and $s, t, v, w \in \{0,1\}^*$

**Model:** $(\varphi, \psi_0, \psi_1) \models (s|\alpha|t \doteq v|\beta|w)$ if
$\pi_t(\psi_s(\varphi(\alpha))) = \pi_w(\psi_v(\varphi(\beta)))$

# Narrow Configuration, Representative

## Definition (Joinable Configurations)

Configurations $\boldsymbol{X}, \boldsymbol{Y}$ are *joinable* in $\mathcal{M}$,
if $\boldsymbol{X} \longrightarrow^*_{\mathcal{M}} \boldsymbol{Z} \longleftarrow^*_{\mathcal{M}} \boldsymbol{Y}$ for some configuration $\boldsymbol{Z}$.

## Definition (Narrow Configuration)

A configuration $\boldsymbol{X}$ is *narrow* in $\mathcal{M}$,
if $\boldsymbol{X}$ and $\boldsymbol{s}|\boldsymbol{p}|\epsilon$ are joinable in $\mathcal{M}$ for some state $\boldsymbol{p}$ and a word $\boldsymbol{s} \in \mathbb{B}^*$.

## Definition (Representative $[\boldsymbol{X}]_{\mathcal{M}}$)

The *representative* of $\boldsymbol{X}$ in $\mathcal{M}$ is the lexicographically smallest configuration $\boldsymbol{Y}$ such that $\boldsymbol{X}$ and $\boldsymbol{Y}$ are joinable in $\mathcal{M}$.

- Joinability is decidable
- Narrowness is decidable
- Representative is computable