

Contents

1	Introduction	2
2	Basic Definitions	2
2.1	λ -calculus $\lambda 2$	2
2.2	first-order logic	3
2.3	two-counter automaton	6
3	System P	7
3.1	Definitions	7
3.2	CONS is undecidable	9
4	INHAB is undecidable	16

1 Introduction

2 Basic Definitions

2.1 λ -calculus $\lambda 2$

In the following let $\mathcal{V}_T = \{\alpha, a, \beta, b, \dots\}$ be a countably infinite set (of type-variables) and $\mathcal{V}_V = \{x, x_1, x_2, \dots\}$ be a countably infinite set (of value-variables).

Definition 1. The set of all $\lambda 2$ types over \mathcal{V}_T , denoted by $T_{\lambda 2}$, is the smallest set T satisfying the following conditions:

- $\mathcal{V}_T \subseteq T$,
- if $t_1, t_2 \in T$ then $(t_1 \rightarrow t_2) \in T$, and
- if $t \in T$ and $\alpha \in \mathcal{V}_T$ then $\forall \alpha. t \in T$.

The set of all $\lambda 2$ terms over \mathcal{V}_T and \mathcal{V}_V , denoted by $\Lambda_{T_{\lambda 2}}$, is the smallest set Λ_T satisfying the following conditions:

- $\mathcal{V}_V \subseteq \Lambda_T$,
- if $e_1, e_2 \in \Lambda_T$ then $e_1 e_2 \in \Lambda_T$,
- if $x \in \mathcal{V}_V$, $t \in T_{\lambda 2}$, and $e \in \Lambda_T$ then $\lambda x : t. e \in \Lambda_T$,
- if $\alpha \in \mathcal{V}_T$ and $e \in \Lambda_T$ then $\Lambda \alpha. e \in \Lambda_T$, and
- if $e \in \Lambda_T$ and $t \in T_{\lambda 2}$ then $e t \in \Lambda_T$.

If we have a type of the form $(t_1 \rightarrow (t_2 \rightarrow (\dots \rightarrow (t_{n-1} \rightarrow t_n) \dots)))$ we will often omit the brackets and just write $(t_1 \rightarrow t_2 \rightarrow \dots \rightarrow t_{n-1} \rightarrow t_n)$ or $t_1 \rightarrow t_2 \rightarrow \dots \rightarrow t_{n-1} \rightarrow t_n$ instead.

Definition 2. Let $e \in \Lambda_{T_{\lambda 2}}$. The free variables of e , denoted by $FV(e)$, are defined inductively as follows:

$$FV(e) = \begin{cases} \{x\} & \text{if } e = x \\ FV(e_1) \cup FV(e_2) & \text{if } e = e_1 e_2 \\ FV(e') \setminus \{x\} & \text{if } e = \lambda x : t. e' \\ FV(e') & \text{if } e = \Lambda \alpha. e' \\ FV(e') & \text{if } e = e' t \end{cases}$$

Definition 3. Let $\mathcal{V} = \{x_1, \dots, x_n\}$ be a finite subset of \mathcal{V}_V and $t_1, \dots, t_n \in \Lambda_{T_{\lambda 2}}$. A $\lambda 2$ -basis $\Gamma = \{(x_1 : t_1), \dots, (x_n : t_n)\}$ is a mapping from \mathcal{V} to $T_{\lambda 2}$. If the kind of basis is clear from the context we abbreviate $\lambda 2$ -basis to basis.

The free variables of a basis Γ , denoted by $FV(\Gamma)$, are $\bigcup \{FV(t) \mid (x : t) \in \Gamma\}$.

For a basis Γ and another basis Σ , $x \in \mathcal{V}_V \setminus \text{dom}(\Gamma)$, and $t \in T_{\lambda 2}$ we will abbreviate $\Gamma \cup \{(x : t)\}$ to $\Gamma, x : t$ and $\Gamma \cup \Sigma$ to Γ, Σ .

Definition 4. Let e be in $\Lambda_{T_{\lambda 2}}$, t in $T_{\lambda 2}$, and Γ be a basis. A statement $e : t$ is derivable from Γ , denoted by $\Gamma \vdash e : t$, if $e : t$ can be produced using the following rules.

(Axiom)	$\Gamma, x : t \vdash x : t$	
(λ -Introduction)	$\frac{\Gamma, x : t_1 \vdash e : t_2}{\Gamma \vdash \lambda x : t_1. e : t_1 \rightarrow t_2}$	
(λ -Elimination)	$\frac{\Gamma \vdash e_1 : t_1 \rightarrow t_2 \quad \Gamma \vdash e_2 : t_1}{\Gamma \vdash e_1 e_2 : t_2}$	
(\forall -Introduction)	$\frac{\Gamma \vdash e : t}{\Gamma \vdash \Lambda \alpha. e : \forall \alpha. t}$	$\alpha \notin \text{FV}(\Gamma)$
(\forall -Elimination)	$\frac{\Gamma \vdash e : \forall \alpha. t}{\Gamma \vdash e t' : t[\alpha := t']}$	

Definition 5. The inhabitation problem for $\lambda 2$, denoted by **INHAB**, is defined as follows. Given a $\lambda 2$ type t .

Is there a $\lambda 2$ term M such that $\emptyset \vdash M : t$?

But we can rephrase this problem so that it becomes more general: Given a basis Γ and a $\lambda 2$ type t .

Is there a $\lambda 2$ term M such that $\Gamma \vdash M : t$?

Obviously the second version is a special case of the first one. For the other direction consider a basis $\Gamma = \{(x_1 : t_1), \dots, (x_n : t_n)\}$ and a $\lambda 2$ type t . Clearly, for every term M , $\Gamma \vdash M : t$ holds iff $\emptyset \vdash \lambda x_1 : t_1. \dots \lambda x_n : t_n. M : t_1 \rightarrow \dots \rightarrow t_n \rightarrow t$.

2.2 first-order logic

Definition 6. A ranked set is a tuple (Σ, rk) , where Σ is a countable set and $rk : \Sigma \rightarrow \mathbb{N}$ is a function that maps every symbol from Σ to a natural number (its rank).

If the function rk is understood we will just write Σ instead of (Σ, rk) . The set of all elements in Σ with a certain rank k , denoted by $\Sigma^{(k)}$, is defined as $\Sigma^{(k)} := rk^{-1}(k)$.

For the remainder of this subsection let $\mathcal{V} = \{y, y_1, y_2, \dots\}$ be a countable set (of variables), \mathcal{F} a ranked set (of function symbols), and \mathcal{P} a ranked set (of predicate symbols).

Definition 7. The set of terms over \mathcal{V} and \mathcal{F} , denoted by $\mathcal{T}_{(\mathcal{V}, \mathcal{F})}$, is the smallest set \mathcal{T} satisfying the following conditions:

- $\mathcal{V} \subseteq \mathcal{T}$, and
- for every $k \in \mathbb{N}$, if $f \in \mathcal{F}^{(k)}$ and $t_1, t_2, \dots, t_k \in \mathcal{T}$ then $f(t_1, t_2, \dots, t_k) \in \mathcal{T}$.

The set of first-order formulas over \mathcal{V} , \mathcal{F} , and \mathcal{P} , denoted by $\mathcal{L}_{(\mathcal{V}, \mathcal{F}, \mathcal{P})}$, is the smallest set \mathcal{L} satisfying the following conditions:

- for every $k \in \mathbb{N}$, if $P \in \mathcal{P}^{(k)}$ and $t_1, t_2, \dots, t_k \in \mathcal{T}_{(\mathcal{V}, \mathcal{F})}$ then $P(t_1, t_2, \dots, t_k) \in \mathcal{L}$.
- If $\varphi, \psi \in \mathcal{L}$ then $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $\neg \varphi \in \mathcal{L}$, and
- if $y \in \mathcal{V}$ and $\varphi \in \mathcal{L}$ then $\exists y.\varphi$, $\forall y.\varphi \in \mathcal{L}$.

We introduce an additional binary operation \rightarrow on formulas, where for some $\varphi, \psi \in \mathcal{L}_{(\mathcal{V}, \mathcal{F}, \mathcal{P})}$ the formula $(\varphi \rightarrow \psi)$ is defined as $(\neg \varphi \vee \psi)$, if we have a formula of the form $(\varphi_1 \rightarrow (\varphi_2 \rightarrow (\dots \rightarrow (\varphi_{n-1} \rightarrow \varphi_n) \dots)))$ we will often omit the brackets and just write $(\varphi_1 \rightarrow \varphi_2 \rightarrow \dots \rightarrow \varphi_{n-1} \rightarrow \varphi_n)$ or $\varphi_1 \rightarrow \varphi_2 \rightarrow \dots \rightarrow \varphi_{n-1} \rightarrow \varphi_n$ instead.

For nullary relation symbols P we will abbreviate $P()$ to P . If a formula φ is of the form $Qy.(\psi)$ (where $Q \in \{\exists, \forall\}$, $y \in \mathcal{V}$, and $(\psi) \in \mathcal{L}_{(\mathcal{V}, \mathcal{F}, \mathcal{P})}$) we often drop the dot and write $Qy(\psi)$ instead. If a formula φ has multiple variables bound by the same quantifier (i.e. $\varphi = Qy_1.Qy_2.\dots.Qy_n.\psi$ for $Q \in \{\exists, \forall\}$, some $n \in \mathbb{N}$, $y_1, y_2, \dots, y_n \in \mathcal{V}$, and $\psi \in \mathcal{L}_{(\mathcal{V}, \mathcal{F}, \mathcal{P})}$) we abbreviate φ to $Qy_1y_2.\dots.y_n.\psi$ or to $Q\vec{y}.\psi$ where $\vec{y} = (y_1, y_2, \dots, y_n)^\top$.

Definition 8. The variables of a term $t \in \mathcal{T}_{(\mathcal{V}, \mathcal{F})}$, denoted by $V(t)$, are defined by:

$$V(t) = \begin{cases} \{y\} & \text{if } t = y \\ V(t_1) \cup V(t_2) \cup \dots \cup V(t_k) & \text{if } t = f(t_1, t_2, \dots, t_k) \end{cases}$$

The free variables of a formula $\varphi \in \mathcal{L}_{(\mathcal{V}, \mathcal{F}, \mathcal{P})}$, denoted by $FV(\varphi)$, are defined as follows:

$$FV(\varphi) = \begin{cases} V(t_1) \cup V(t_2) \cup \dots \cup V(t_k) & \text{if } \varphi = P(t_1, t_2, \dots, t_k) \\ FV(\psi) & \text{if } \varphi = \neg \psi \\ FV(\varphi_1) \cup FV(\varphi_2) & \text{if } \varphi = (\varphi_1 \wedge \varphi_2) \text{ or } \varphi = (\varphi_1 \vee \varphi_2) \\ FV(\psi) \setminus \{y\} & \text{if } \varphi = \forall y.\psi \text{ or } \varphi = \exists y.\psi \end{cases}$$

Definition 9. Let y be in \mathcal{V} and $t, t' \in \mathcal{T}_{(\mathcal{V}, \mathcal{F})}$. The substitution of y by t' in t , denoted by $t[y := t']$, is defined as follows:

$$t[y := t'] = \begin{cases} t' & \text{if } t = y \\ z & \text{if } t = z \text{ and } z \neq y \\ f(t_1[y := t'], \dots, t_k[y := t']) & \text{if } t = f(t_1, \dots, t_k) \end{cases}$$

Now we can lift this definition to formulas, let φ be in $\mathcal{L}_{(\mathcal{V}, \mathcal{F}, \mathcal{P})}$. The substitution of y by t' in φ , denoted by $\varphi[y := t']$, is defined as follows:

$$\varphi[y := t'] = \begin{cases} P(t_1[y := t'], \dots, t_k[y := t']) & \text{if } \varphi = P(t_1, \dots, t_k) \\ \neg(\psi[y := t']) & \text{if } \varphi = \neg\psi \\ \varphi_1[y := t'] \circ \varphi_2[y := t'] & \text{if } \varphi = (\varphi_1 \circ \varphi_2), \circ \in \{\wedge, \vee\} \\ \varphi & \text{if } \varphi = \forall y.\psi \text{ or } \varphi = \exists y.\psi \\ Qz.(\psi[y := t']) & \text{if } \varphi = Qz.\psi, Q \in \{\forall, \exists\} \text{ and } z \neq y \end{cases}$$

Now we come to the semantics of first-order formulas.

Definition 10. An interpretation I over \mathcal{V} , \mathcal{F} , and \mathcal{P} is a triple $(\Delta, \cdot^I, \omega)$, where Δ is a nonempty set (which we call domain), \cdot^I is a function such that $f^I: \Delta^k \rightarrow \Delta$ is a function for every $k \in \mathbb{N}$, $f \in \mathcal{F}^{(k)}$ and $P^I \subseteq \Delta^k$ is a relation for every $k \in \mathbb{N}$, $P \in \mathcal{P}^{(k)}$ and ω is a function from \mathcal{V} to Δ .

Let $I = (\Delta, \cdot^I, \omega)$ be an interpretation, $y \in \mathcal{V}$, and $d \in \Delta$ the interpretation $I[y \mapsto d]$ is defined as $(\Delta, \cdot^I, \omega[y \mapsto d])$ where

$$(\omega[y \mapsto d])(z) = \begin{cases} d & \text{if } z = y \\ \omega(y) & \text{otherwise.} \end{cases}$$

Definition 11. Let $I = (\Delta, \cdot^I, \omega)$ be an interpretation and t a term. The interpretation of t under I , denoted by t^I , is defined as follows:

$$t^I = \begin{cases} \omega(y) & \text{if } t = y \\ f^I(t_1^I, \dots, t_k^I) & \text{if } t = f(t_1, \dots, t_k) \end{cases}$$

Let φ be a formula. The interpretation of φ under I , denoted by φ^I , is defined recursively as follows:

$$\varphi^I = \begin{cases} \top & \text{if } \varphi = P(t_1, \dots, t_k) \text{ and } (t_1^I, \dots, t_k^I) \in P^I \\ \perp & \text{if } \varphi = P(t_1, \dots, t_k) \text{ and } (t_1^I, \dots, t_k^I) \notin P^I \\ \text{not } \psi^I & \text{if } \varphi = \neg\psi \\ \varphi_1^I \text{ and } \varphi_2^I & \text{if } \varphi = (\varphi_1 \wedge \varphi_2) \\ \varphi_1^I \text{ or } \varphi_2^I & \text{if } \varphi = (\varphi_1 \vee \varphi_2) \\ \text{exists } d \in \Delta \psi^I[y \mapsto d] & \text{if } \varphi = \exists y.\psi \\ \text{forall } d \in \Delta \psi^I[y \mapsto d] & \text{if } \varphi = \forall y.\psi \end{cases}$$

The interpretation I is a model of φ , denoted by $I \models \varphi$, if $\varphi^I = \top$.

When we define an interpretation I and we have a nullary predicate symbol P we write $P^I = \top$ instead of $P^I = \{()\}$ and $P^I = \perp$ for $P^I = \emptyset$ (this works because $P()^I = \top$ iff $() \in P^I$).

Definition 12. Let Γ be a finite set of first-order formulas.

We say that an interpretation I is a model of Γ , denoted by $I \models \Gamma$, if $I \models \psi$ for every ψ in Γ .

The formula φ is a semantic consequence of Γ , denoted by $\Gamma \vdash \varphi$, if every model of Γ is also a model of φ .

The free variables of Γ , denoted by $\text{FV}(\Gamma)$, are $\bigcup \{\text{FV}(\varphi) \mid \varphi \in \Gamma\}$.

2.3 two-counter automaton

We will use a version of two-counter automaton which only has two types of transitions. First it can increment a register and second it can decrement a register or jump if the register is already zero. Formally:

Definition 13. A deterministic two-counter automaton is a 4-tuple $M = (\mathcal{Q}, Q_0, Q_f, R)$,

- where \mathcal{Q} is a finite set (of states),
- Q_0 is in \mathcal{Q} (the initial state),
- Q_f is in \mathcal{Q} (the final state), and
- R is a function from $\mathcal{Q} \setminus \{Q_f\}$ to $\mathcal{R}_{\mathcal{Q}}$,
where $\mathcal{R}_{\mathcal{Q}} = \{+(i, Q') \mid i \in \{1, 2\}, Q' \in \mathcal{Q}\} \cup \{-(i, Q_1, Q_2) \mid i \in \{1, 2\}, Q_1, Q_2 \in \mathcal{Q}\}$

A configuration C of our automaton is a triple $\langle Q, m, n \rangle$, where $Q \in \mathcal{Q}$ and $m, n \in \mathbb{N}$. Let r be in $R(\mathcal{Q} \setminus \{Q_f\})$, then \Rightarrow_M^r is a binary relation on the configurations of M such that two configurations $\langle Q, m, n \rangle, \langle \hat{Q}, \hat{m}, \hat{n} \rangle$ of M are in the relation if all of the following conditions hold:

- $Q \neq Q_f, r = R(Q)$,
- if $r = +(1, Q')$ for some $Q' \in \mathcal{Q}$ then $\hat{Q} = Q', \hat{m} = m + 1$, and $\hat{n} = n$,
- if $r = +(2, Q')$ for some $Q' \in \mathcal{Q}$ then $\hat{Q} = Q', \hat{m} = m$, and $\hat{n} = n + 1$,
- if $r = -(1, Q_1, Q_2)$ for some $Q_1, Q_2 \in \mathcal{Q}$ then
 - if $m = 0$ then $\hat{Q} = Q_2, \hat{m} = 0$, and $\hat{n} = n$,
 - if $m \geq 1$ then $\hat{Q} = Q_1, \hat{m} = m - 1$, and $\hat{n} = n$,
- if $r = -(2, Q_1, Q_2)$ for some $Q_1, Q_2 \in \mathcal{Q}$ then
 - if $n = 0$ then $\hat{Q} = Q_2, \hat{m} = m$, and $\hat{n} = 0$,
 - if $n \geq 1$ then $\hat{Q} = Q_1, \hat{m} = m$, and $\hat{n} = n - 1$.

The transition relation of M , denoted by \Rightarrow_M , is defined as $\bigcup_{r \in R(Q \setminus \{Q_f\})} \Rightarrow_M^r$. We denote the transitive reflexive closure of \Rightarrow_M by \Rightarrow_M^* .

Let m, n be in \mathbb{N} , we say that M terminates on input (m, n) if there exist $\hat{m}, \hat{n} \in \mathbb{N}$ such that $\langle Q_0, m, n \rangle \Rightarrow_M^* \langle Q_f, \hat{m}, \hat{n} \rangle$ (It follows that there exists an $i \in \mathbb{N}$ and configurations D_1, \dots, D_i of M such that $\langle Q_0, m, n \rangle = D_1 \Rightarrow_M \dots \Rightarrow_M D_i = \langle Q_f, \hat{m}, \hat{n} \rangle$, we call this chain a computation with length i).

Definition 14. The halting problem for two-counter automaton, denoted by **HALT**, is defined as follows. Given a two-counter automaton M .

Does M terminate on input $(0, 0)$?

It is well known that **HALT** is undecidable.

3 System P

3.1 Definitions

In the following let $\mathcal{V}_P = \{\alpha, a, \beta, b, \dots\}$ be a countably infinite subset of \mathcal{V}_T (of variables). Let $\mathcal{P}_P = \{P, Q, \dots\}$ be a set (of predicate symbols) and \mathcal{P} a ranked set such that $\mathcal{P}^{(0)} = \{\mathbf{false}\}$, $\mathcal{P}^{(2)} = \mathcal{P}_P$, and $\mathcal{P}^{(k)} = \emptyset$ for all $k \in \mathbb{N} \setminus \{0, 2\}$. A first-order logic formula φ over \mathcal{V}_P , \emptyset , and \mathcal{P} is an

atomic formula if $\varphi = \mathbf{false}$ or $\varphi = P(a, b)$ for some $P \in \mathcal{P}_P$ and $a, b \in \mathcal{V}_P$.

universal formula if $\varphi = \forall \vec{\alpha}(A_1 \rightarrow A_2 \rightarrow \dots \rightarrow A_n)$ for some $n \in \mathbb{N}$ and where A_i is an atomic formula for $i \in \{1, \dots, n\}$, $A_i \neq \mathbf{false}$ for $i \in \{1, \dots, n-1\}$ and for each $\alpha \in \text{FV}(A_n)$ there exists an $i \in \{1, \dots, n-1\}$ such that $\alpha \in \text{FV}(A_i)$.

existential formula if there is an $n \in \mathbb{N}^+$, atomic formulas $A_i \neq \mathbf{false}$ for $i \in \{1, \dots, n\}$ such that $\varphi = \forall \vec{\alpha}(A_1 \rightarrow A_2 \rightarrow \dots \rightarrow A_{n-1} \rightarrow \forall \beta(A_n \rightarrow \mathbf{false}) \rightarrow \mathbf{false})$.

The set of formulas of System **P** (= set of **P**-formulas) over \mathcal{V}_P and \mathcal{P}_P is the set of all first-order formulas in $\mathcal{L}_{(\mathcal{V}_P, \emptyset, \mathcal{P})}$ that are either an atomic, universal or existential formula.

Definition 15. A finite set of **P**-formulas Γ is called **P**-basis, or basis if it is clear from the context whether a **P**-basis or a **$\lambda 2$** -basis is meant.

For a **P**-basis Γ , another **P**-basis Σ , and a **P**-formula A we will abbreviate $\Gamma \cup \{A\}$ to Γ, A and $\Gamma \cup \Sigma$ to Γ, Σ (c.f. **$\lambda 2$** -basis).

Definition 16. Let A be a **P**-formula, and Γ be a basis. The formula A is a semantic consequence of Γ , denoted by $\Gamma \vdash A$, if A can be produced using the following deduction rules.

(Axiom)	$\Gamma, A \vdash A$	
(\rightarrow -Introduction)	$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B}$	
(\rightarrow -Elimination)	$\frac{\Gamma \vdash A \rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}$	
(\forall -Introduction)	$\frac{\Gamma \vdash B}{\Gamma \vdash \forall \alpha B}$	$\alpha \notin FV(\Gamma)$
(\forall -Elimination)	$\frac{\Gamma \vdash \forall \alpha B}{\Gamma \vdash B[\alpha := b]}$	$b \in \mathcal{V}_P$

We define a more general consequence relation in which we demand that **false** is interpreted with \perp . In this relation existential formulas will behave like the name suggests. Formally:

Definition 17. Let Γ be a basis. The **P**-formula A is a semantic consequence with falsity of Γ , denoted by $\Gamma \vdash_f A$, if for every interpretation I

$$I \models \Gamma \text{ and } \mathbf{false}^I = \perp \text{ implies } I \models A.$$

This allows us to add the following deduction rule.

(\exists -Introduction)	$\frac{\Gamma, A[\alpha := a] \vdash_f B}{\Gamma, A' := \forall \alpha (A \rightarrow \mathbf{false}) \rightarrow \mathbf{false} \vdash_f B}$	$a \notin FV(\Gamma, A', B)$
----------------------------	---	------------------------------

Proof. Let $I = (\Delta, \cdot^I, \omega)$ be a model of $\Gamma, \forall \alpha (A \rightarrow \mathbf{false}) \rightarrow \mathbf{false}$ with $\mathbf{false}^I = \perp$ and $a \in \mathcal{V}_P$ a variable such that $a \notin FV(\Gamma, A', B)$.

$$\begin{aligned}
I \models \Gamma, \forall \alpha (A \rightarrow \mathbf{false}) \rightarrow \mathbf{false} &\Rightarrow I \models \forall \alpha (A \rightarrow \mathbf{false}) \rightarrow \mathbf{false} \\
&\Rightarrow (\forall \alpha (A \rightarrow \mathbf{false}))^I \rightarrow \mathbf{false}^I \\
&\Rightarrow (\forall \alpha (A \rightarrow \mathbf{false}))^I \rightarrow \perp \\
&\Rightarrow \neg(\forall \alpha (A \rightarrow \mathbf{false}))^I \\
&\Rightarrow \neg(\forall d \in \Delta: (A \rightarrow \mathbf{false})^{I[\alpha \mapsto d]}) \\
&\Rightarrow \exists d \in \Delta: \neg(A^{I[\alpha \mapsto d]} \rightarrow \mathbf{false}^{I[\alpha \mapsto d]}) \\
&\Rightarrow \exists d \in \Delta: \neg(A^{I[\alpha \mapsto d]} \rightarrow \perp) \\
&\Rightarrow \exists d \in \Delta: \neg(\neg A^{I[\alpha \mapsto d]}) \\
&\Rightarrow \exists d \in \Delta: A^{I[\alpha \mapsto d]}
\end{aligned}$$

Together with $a \notin FV(\Gamma, A')$, it follows that $I[a \mapsto d]$ is a model of $\Gamma, A[\alpha := a]$. Which implies $I[a \mapsto d] \models B$. Since a is not free in B we conclude that I is also a model of B . \square

Definition 18. The problem to decide whether a given set of **P**-formulas is consistent, denoted by **CONS**, is defined as follows. Given a set of **P**-formulas Γ .

Does $\Gamma \vdash \mathbf{false}$ not hold?

3.2 CONS is undecidable

We will show that $\mathbf{HALT} \leq \mathbf{CONS}$ then the undecidability of **CONS** directly follows from the undecidability of **HALT**. For a given two-counter automaton M we will effectively construct a **P**-basis Γ_M such that

M terminates on input $(0, 0)$ iff $\Gamma_M \vdash \mathbf{false}$ holds in System **P**.

Let $M = (\mathcal{Q}, Q_0, Q_f, R)$ be a two-counter automaton, w.l.o.g. $S, P, R_1, R_2, E, D \notin \mathcal{Q}$. In the following we will consider **P**-formulas over \mathcal{V}_P and \mathcal{P}_P , where $\mathcal{P}_P = \mathcal{Q} \uplus \{S, P, R_1, R_2, E, D\}$. We will abbreviate $P(a, a)$ to $P(a)$, note that this way we can use binary predicate symbols as unary ones.

The intended informal meaning for these new relation symbols is the following:

- The meaning of $Q(a)$ is “ a represents a configuration and Q is the state of this configuration”.
- For $i \in \{1, 2\}$, $R_i(a, m)$ denotes that “the value of register i in the configuration represented by a is represented by m ” (we call m anchor of a for register i).
- With $S(a, b)$ we state that “ b is a successor of a ”.
- The meaning of $P(a, b)$ is “ b is a predecessor of a ”.
- And $E(a)$ marks “ a as the end of chain”.
- Finally $D(a)$ states that “ a is not the end of a chain”.

For a configuration $C = \langle Q, m, n \rangle$ of M we define a set of **P**-formulas Γ_C . It contains the following formulas:

- $Q(a)$
- $R_1(a, a_0), P(a_{i-1}, a_i)$ for $i \in \{1, \dots, m\}$
- $R_2(a, b_0), P(b_{i-1}, b_i)$ for $i \in \{1, \dots, n\}$
- $D(a_i), D(b_j)$ for $i \in \{0, \dots, m-1\}$ and $j \in \{0, \dots, n-1\}$
- $E(a_m), E(b_n)$

Next we need sets of **P**-formulas for all possible transitions. For every $Q \in \mathcal{Q} \setminus \{Q_f\}$ and $r \in \mathcal{R}_{\mathcal{Q}}$ we define $\Gamma_{Q,r}$. If $r = +(1, Q')$ for some $Q' \in \mathcal{Q}$ then $\Gamma_{Q,+(1,Q')}$ contains the following formulas:

- $\forall \alpha \beta (Q(\alpha) \rightarrow S(\alpha, \beta) \rightarrow Q'(\beta))$
change of state
- $\forall \alpha \beta \gamma \delta (Q(\alpha) \rightarrow S(\alpha, \beta) \rightarrow R_1(\alpha, \gamma) \rightarrow R_1(\beta, \delta) \rightarrow P(\delta, \gamma))$
increment register 1
- $\forall \alpha \beta \delta (Q(\alpha) \rightarrow S(\alpha, \beta) \rightarrow R_1(\beta, \delta) \rightarrow D(\delta))$
prevent zero in register 1
- $\forall \alpha \beta \gamma (Q(\alpha) \rightarrow S(\alpha, \beta) \rightarrow R_2(\alpha, \gamma) \rightarrow R_2(\beta, \gamma))$
do not change the value register 2

If $r = -(1, Q_1, Q_2)$ for some $Q_1, Q_2 \in \mathcal{Q}$ then $\Gamma_{Q,-(1,Q_1,Q_2)}$ contains the following formulas:

- $\forall \alpha \beta \gamma (Q(\alpha) \rightarrow S(\alpha, \beta) \rightarrow R_1(\alpha, \gamma) \rightarrow E(\gamma) \rightarrow Q_2(\beta))$
jump to Q_2 if register 1 is zero
- $\forall \alpha \beta \gamma (Q(\alpha) \rightarrow S(\alpha, \beta) \rightarrow R_1(\alpha, \gamma) \rightarrow E(\gamma) \rightarrow R_1(\beta, \gamma))$
if register 1 is zero it stays zero
- $\forall \alpha \beta \gamma (Q(\alpha) \rightarrow S(\alpha, \beta) \rightarrow R_1(\alpha, \gamma) \rightarrow D(\gamma) \rightarrow Q_1(\beta))$
change state to Q_1 if register 1 is greater zero
- $\forall \alpha \beta \gamma \delta (Q(\alpha) \rightarrow S(\alpha, \beta) \rightarrow R_1(\alpha, \gamma) \rightarrow D(\gamma) \rightarrow P(\gamma, \delta) \rightarrow R_1(\beta, \delta))$
decrement register 1 if possible
- $\forall \alpha \beta \gamma (Q(\alpha) \rightarrow S(\alpha, \beta) \rightarrow R_2(\alpha, \gamma) \rightarrow R_2(\beta, \gamma))$
do not change register 2 in both cases

For $r = +(2, Q')$ for some $Q' \in \mathcal{Q}$ or $r = -(2, Q_1, Q_2)$ for some $Q_1, Q_2 \in \mathcal{Q}$ the sets $\Gamma_{Q,r}$ are defined analogously.

We also need a set Γ_1 to ensure that our representation works correctly. The following formula are in Γ_1 :

- $\forall \alpha (\forall \beta (R_1(\alpha, \beta) \rightarrow \mathbf{false}) \rightarrow \mathbf{false})$
every element represents a configuration so it has a value for register 1
- $\forall \alpha (\forall \beta (R_2(\alpha, \beta) \rightarrow \mathbf{false}) \rightarrow \mathbf{false})$
every element represents a configuration so it has a value for register 2
- $\forall \alpha (\forall \beta (S(\alpha, \beta) \rightarrow \mathbf{false}) \rightarrow \mathbf{false})$
every element has a successor

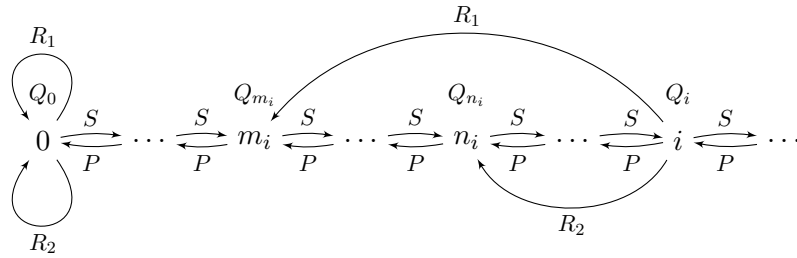
We define $\Gamma_{\overline{M}}$ as $\bigcup_{Q \in \mathcal{Q} \setminus \{Q_f\}} \Gamma_{Q, R(Q)} \cup \{\forall \alpha (Q_f(\alpha) \rightarrow \mathbf{false})\} \cup \Gamma_1$. We have added the formula $\forall \alpha (Q_f(\alpha) \rightarrow \mathbf{false})$ to be able to deduce **false** if our automaton terminates. Finally we can define Γ_M as $\Gamma_{C_0} \cup \Gamma_{\overline{M}}$, where $C_0 = \langle Q_0, 0, 0 \rangle$ is the initial configuration.

Claim 19.

$$\Gamma_M \vdash \mathbf{false} \text{ holds in system } P \quad \implies \quad M \text{ terminates on input } (0, 0)$$

Proof. Assume M does not terminate then there is an infinite chain $C_0 \Rightarrow_M C_1 \Rightarrow_M C_2 \Rightarrow_M \dots$ ($C_i = \langle Q_i, m_i, n_i \rangle$ for $i \in \mathbb{N}$). Now we construct a model of Γ_M which interprets **false** with \perp this contradicts $\Gamma_M \vdash \mathbf{false}$.

To illustrate the idea we will use a graphical notation for an interpretation I . By $d_1 \xrightarrow{R} d_2$ we say that $(d_1, d_2) \in R^I$. And we use $\frac{P}{d}$ to say that $(d, d) \in P^I$ for predicate symbols that are used as unary predicate symbols. As domain for our interpretation we will use the natural numbers. Every number will have two tasks: firstly it will represent itself as a possible value for register 1 or 2 and secondly every number i will also represent the i^{th} configuration of our infinite computation. Now the idea for our model of Γ_M looks like this:



We have $0 \in E^I$ and all other numbers are in D^I .

Here is the more formal definition of our model $I = (\mathbb{N}, \cdot^I, \omega)$.

$$\begin{aligned} P^I &= \{(i+1, i) \mid i \in \mathbb{N}\} & R_1^I &= \{(i, m_i) \mid i \in \mathbb{N}\} & R_2^I &= \{(i, n_i) \mid i \in \mathbb{N}\} \\ S^I &= \{(i, i+1) \mid i \in \mathbb{N}\} & D^I &= \{(i, i) \mid i \in \mathbb{N}^+\} & E^I &= \{(0, 0)\} \\ Q^I &= \{(i, i) \mid i \in \mathbb{N}, Q = Q_i\} \text{ for every } Q \in \mathcal{Q} & \mathbf{false}^I &= \perp \end{aligned}$$

$$a^I = 0 \qquad a_0^I = 0 \qquad b_0^I = 0$$

Since there are no free variables in Γ_M we can just set $\omega(x) = 0$ for every $x \in \mathcal{V}_P$. It is easy to see that I is indeed a model of Γ_M . \square

We proof the other direction by induction on the length of the computation. But to be able to use the induction hypothesis we need a slightly more general statement (this is why we defined $\Gamma_{\overline{M}}$ and not just Γ_M right away).

Claim 20. Let $C = \langle Q, m, n \rangle$ be a configuration of M . If a final configuration (i.e. a configuration $\langle Q_f, \hat{m}, \hat{n} \rangle$ for some $\hat{m}, \hat{n} \in \mathbb{N}$) is reachable from C then $\Gamma_C \cup \Gamma_{\overline{M}} \vdash \mathbf{false}$.

Proof. By induction on the length i of the computation.

Induction Base: $i = 0$

Since a final configuration is reachable in 0 steps C must be this final configuration. So $C = \langle Q_f, m, n \rangle$ for some $m, n \in \mathbb{N}$. Hence, $Q_f(a)$ is in Γ_C for some $a \in \mathcal{V}_P$ and $\forall \alpha (Q_f(\alpha) \rightarrow \mathbf{false})$ is in $\Gamma_{\overline{M}}$, we can easily deduce \mathbf{false} .

$$\frac{\frac{\Gamma_C \cup \Gamma_{\overline{M}} \vdash \forall \alpha (Q_f(\alpha) \rightarrow \mathbf{false})}{\Gamma_C \cup \Gamma_{\overline{M}} \vdash Q_f(a) \rightarrow \mathbf{false}} \quad \Gamma_C \cup \Gamma_{\overline{M}} \vdash Q_f(a)}{\Gamma_C \cup \Gamma_{\overline{M}} \vdash \mathbf{false}}$$

Induction Step: $i = i' + 1$

Since $I \models \mathbf{false}$ holds trivially if I interprets \mathbf{false} with \top we only need to consider models of $\Gamma_C \cup \Gamma_{\overline{M}}$ that interpret \mathbf{false} with \perp (note that there are no such models if M terminates which is exactly what we want to proof). As result of this observation we can use the \exists -Introduction rule.

From the fact that a final configuration is reachable from C in i steps we can deduce that there exists a configuration $D = \langle \hat{Q}, \hat{m}, \hat{n} \rangle$ such that $C \Rightarrow_M^{r_M} D$ for some $r \in \mathcal{R}_Q$ and a final configuration is reachable from D in i' steps. We also know that $C = \langle Q, m, n \rangle$ for some $Q \in \mathcal{Q} \setminus \{Q_f\}$ and some $m, n \in \mathbb{N}$. The set Γ_C contains the formulas:

$$R_1(a, a_0), P(a_{i-1}, a_i) \text{ and } D(a_{i-1}) \text{ for } i \in \{1, \dots, m\},$$

$$R_2(a, b_0), P(b_{i-1}, b_i) \text{ and } D(b_{i-1}) \text{ for } i \in \{1, \dots, n\},$$

$$Q(a), E(a_m) \text{ and } E(b_n).$$

And Γ_D contains the formulas:

$$R_1(\hat{a}, \hat{a}_0), P(\hat{a}_{i-1}, \hat{a}_i) \text{ and } D(\hat{a}_{i-1}) \text{ for } i \in \{1, \dots, \hat{m}\},$$

$$R_2(\hat{a}, \hat{b}_0), P(\hat{b}_{i-1}, \hat{b}_i) \text{ and } D(\hat{b}_{i-1}) \text{ for } i \in \{1, \dots, \hat{n}\},$$

$$\hat{Q}(\hat{a}), E(\hat{a}_{\hat{m}}) \text{ and } E(\hat{b}_{\hat{n}}).$$

The basic idea is to deduce Γ_D from $\Gamma_C \cup \Gamma_{\overline{M}}$ and then apply the induction hypothesis to $\Gamma_D \cup \Gamma_{\overline{M}}$.

$$\frac{\frac{\text{Induction Hypothesis}}{\Gamma_C \cup \Gamma_{\overline{M}} \cup \Gamma_D \vdash_f \mathbf{false}} \quad \Gamma_C \cup \Gamma_{\overline{M}} \vdash_f \Gamma_D}{\Gamma_C \cup \Gamma_{\overline{M}} \vdash_f \mathbf{false}}$$

We achieve this by looking at the four possible cases for the type of the rule r . We will only consider the cases $r = +(1, Q')$ and $r = -(1, Q_1, Q_2)$, because the two remaining

cases $r = +(2, Q')$ and $r = -(2, Q_1, Q_2)$ follow by exchanging the roles of register 1 and register 2 in the first two cases.

First we need a new free variable representing the configuration D . Also the value in register 2 does not change, because in both cases we are only concerned with register 1. For the succeeding tableau proofs we will abbreviate **false** by **f** and we will drop $\Gamma_C \cup \Gamma_{\overline{M}}$ and only write new formulas on the left side of \vdash_f .

We first introduce a new variable representing the new configuration D (let $b \in \mathcal{V}_P \setminus \text{FV}(\Gamma_C)$, note that $\text{FV}(\Gamma_{\overline{M}}) = \emptyset$).

$$\frac{\begin{array}{c} \vdots \\ \hline S(a, b) \vdash_f \mathbf{f} \\ \hline \end{array} \quad \frac{\frac{\forall \beta(S(a, \beta) \rightarrow \mathbf{f}) \rightarrow \mathbf{f} \vdash_f \mathbf{f}}{\vdash_f (\forall \beta(S(a, \beta) \rightarrow \mathbf{f}) \rightarrow \mathbf{f}) \rightarrow \mathbf{f}} \quad \frac{\vdash_f \forall \alpha(\forall \beta(S(\alpha, \beta) \rightarrow \mathbf{f}) \rightarrow \mathbf{f})}{\vdash_f \forall \beta(S(a, \beta) \rightarrow \mathbf{f}) \rightarrow \mathbf{f}}}{\vdash_f \mathbf{f}}$$

Since register 2 should not change we need $R_2(b, b_0)$. Again we will just drop $S(a, b)$ on the left side for comprehensibility.

$$\frac{\begin{array}{c} \vdots \\ \hline R_2(b, b_0) \vdash_f \mathbf{f} \\ \hline \end{array} \quad \frac{\frac{\frac{\vdash_f \forall \alpha \beta \gamma (Q(\alpha) \rightarrow S(\alpha, \beta) \rightarrow R_2(\alpha, \gamma) \rightarrow R_2(\beta, \gamma))}{\vdash_f Q(a) \rightarrow S(a, b) \rightarrow R_2(a, b_0) \rightarrow R_2(b, b_0)} \quad \vdash_f Q(a)}{\vdash_f S(a, b) \rightarrow R_2(a, b_0) \rightarrow R_2(b, b_0)} \quad \frac{\vdash_f S(a, b)}{\vdash_f R_2(a, b_0) \rightarrow R_2(b, b_0)} \quad \vdash_f R_2(a, b_0)}{\vdash_f R_2(b, b_0) \rightarrow \mathbf{f}} \quad \frac{\vdash_f R_2(b, b_0) \rightarrow \mathbf{f}}{\vdash_f \mathbf{f}}$$

For the case that $\mathbf{r} = +(\mathbf{1}, Q')$, we have that $\widehat{Q} = Q'$, $\widehat{m} = m + 1$, and $\widehat{n} = n$. So we need to increment register 1 and ensure that the state of b is Q' .

$$\frac{\begin{array}{c} \vdots \\ \hline Q'(b) \vdash_f \mathbf{f} \\ \hline \end{array} \quad \frac{\frac{\vdash_f \forall \alpha \beta (Q(\alpha) \rightarrow S(\alpha, \beta) \rightarrow Q'(\beta))}{\vdash_f Q(a) \rightarrow S(a, b) \rightarrow Q'(b)} \quad \vdash_f Q(a)}{\vdash_f S(a, b) \rightarrow Q'(b)} \quad \vdash_f S(a, b)}{\vdash_f Q'(b) \rightarrow \mathbf{f}} \quad \frac{\vdash_f Q'(b) \rightarrow \mathbf{f}}{\vdash_f \mathbf{f}}$$

To increment register 1 we need a new free variable as anchor for register 1 (let $d \in \mathcal{V}_P \setminus \text{FV}(\Gamma_C)$ and $d \neq b$).

$$\begin{array}{c}
\vdots \\
\hline
R_1(b, d) \vdash_f \mathbf{f} \\
\hline
\frac{\forall \beta (R_1(b, \beta) \rightarrow \mathbf{f}) \rightarrow \mathbf{f} \vdash_f \mathbf{f}}{\vdash_f (\forall \beta (R_1(b, \beta) \rightarrow \mathbf{f}) \rightarrow \mathbf{f}) \rightarrow \mathbf{f}} \quad \frac{\vdash_f \forall \alpha (\forall \beta (R_1(\alpha, \beta) \rightarrow \mathbf{f}) \rightarrow \mathbf{f})}{\vdash_f \forall \beta (R_1(b, \beta) \rightarrow \mathbf{f}) \rightarrow \mathbf{f}} \\
\hline
\vdash_f \mathbf{f}
\end{array}$$

Now we need to connect d with a_0 (the anchor of a for register 1).

$$\begin{array}{c}
\vdots \\
\hline
\frac{\vdash_f \forall \alpha \beta \gamma \delta (Q(\alpha) \rightarrow S(\alpha, \beta) \rightarrow R_1(\alpha, \gamma) \rightarrow R_1(\beta, \delta) \rightarrow P(\delta, \gamma))}{\vdash_f Q(a) \rightarrow S(a, b) \rightarrow R_1(a, a_0) \rightarrow R_1(b, d) \rightarrow P(d, a_0)} \quad \vdash_f Q(a) \\
\hline
\frac{\vdash_f S(a, b) \rightarrow R_1(a, a_0) \rightarrow R_1(b, d) \rightarrow P(d, a_0)}{\vdash_f R_1(a, a_0) \rightarrow R_1(b, d) \rightarrow P(d, a_0)} \quad \vdash_f S(a, b) \\
\hline
\frac{P(d, a_0) \vdash_f \mathbf{f}}{\vdash_f P(d, a_0) \rightarrow \mathbf{f}} \quad \frac{\vdash_f R_1(b, d) \rightarrow P(d, a_0)}{\vdash_f R_1(b, d)} \\
\hline
\vdash_f \mathbf{f}
\end{array}$$

At last we have to make sure that we do not get an artificial zero. We achieve this by deducing $D(d)$.

$$\begin{array}{c}
\vdots \\
\hline
\frac{\vdash_f \forall \alpha \beta \delta (Q(\alpha) \rightarrow S(\alpha, \beta) \rightarrow R_1(\beta, \delta) \rightarrow D(\delta))}{\vdash_f Q(a) \rightarrow S(a, b) \rightarrow R_1(b, d) \rightarrow D(d)} \quad \vdash_f Q(a) \\
\hline
\frac{D(d) \vdash_f \mathbf{f}}{\vdash_f D(d) \rightarrow \mathbf{f}} \quad \frac{\vdash_f S(a, b) \rightarrow R_1(b, d) \rightarrow D(d)}{\vdash_f R_1(b, d) \rightarrow D(d)} \quad \vdash_f S(a, b) \\
\hline
\vdash_f \mathbf{f}
\end{array}$$

Now we already have deduced Γ_D , to see why define $\hat{a} := b$, $\hat{b}_i := b_i$ for $i \in \{0, \dots, n\}$, $\hat{a}_0 := d$, and $\hat{a}_{i+1} := a_i$ for $i \in \{0, \dots, m\}$. Hence we can deduce **false** by induction hypothesis.

The other case, that $\mathbf{r} = -(\mathbf{Q}, \mathbf{1}, \mathbf{Q}_1, \mathbf{Q}_2)$, has to be split into two cases again. If $\mathbf{m} = \mathbf{0}$ then $\hat{Q} = Q_2$, $\hat{m} = 0$, and $\hat{n} = n$. We only need to ensure that the successor state is Q_2 and that register 1 is still zero.

$$\begin{array}{c}
\vdots \\
\hline
\frac{\vdash_f \forall \alpha \beta \gamma (Q(\alpha) \rightarrow S(\alpha, \beta) \rightarrow R_1(\alpha, \gamma) \rightarrow E(\gamma) \rightarrow Q_2(\beta))}{\vdash_f Q(a) \rightarrow S(a, b) \rightarrow R_1(a, a_0) \rightarrow E(a_0) \rightarrow Q_2(b)} \quad \vdash_f Q(a) \\
\hline
\frac{\vdash_f S(a, b) \rightarrow R_1(a, a_0) \rightarrow E(a_0) \rightarrow Q_2(b)}{\vdash_f R_1(a, a_0) \rightarrow E(a_0) \rightarrow Q_2(b)} \quad \vdash_f S(a, b) \\
\hline
\frac{Q_2(b) \vdash_f \mathbf{f}}{\vdash_f Q_2(b) \rightarrow \mathbf{f}} \quad \frac{\vdash_f E(a_0) \rightarrow Q_2(b)}{\vdash_f E(a_0)} \\
\hline
\vdash_f \mathbf{f}
\end{array}$$

Register 1 stays zero.

$$\begin{array}{c}
\frac{\frac{\frac{\vdash_f \forall \alpha \beta \gamma (Q(\alpha) \rightarrow S(\alpha, \beta) \rightarrow R_1(\alpha, \gamma) \rightarrow E(\gamma) \rightarrow R_1(\beta, \gamma))}{\vdash_f Q(a) \rightarrow S(a, b) \rightarrow R_1(a, a_0) \rightarrow E(a_0) \rightarrow R_1(b, a_0)} \quad \vdash_f Q(a)}{\vdash_f S(a, b) \rightarrow R_1(a, a_0) \rightarrow E(a_0) \rightarrow R_1(b, a_0)} \quad \vdash_f S(a, b) \\
\vdots \\
\frac{\frac{R_1(b, a_0) \vdash_f \mathbf{f}}{\vdash_f R_1(b, a_0) \rightarrow \mathbf{f}}}{\vdash_f \mathbf{f}} \quad \frac{\frac{\vdash_f R_1(a, a_0) \rightarrow E(a_0) \rightarrow R_1(b, a_0)}{\vdash_f E(a_0) \rightarrow R_1(b, a_0)} \quad \vdash_f E(a_0)}{\vdash_f R_1(b, a_0)} \\
\hline
\vdash_f \mathbf{f}
\end{array}$$

If we define $\hat{a} := b$, $\hat{b}_i := b_i$ for $i \in \{0, \dots, n\}$, and $\hat{a}_0 := a_0$ then it is clear that we have deduced all formulas required for Γ_D . So we can use the induction hypothesis to deduce **false**.

In the last case $m > 0$, so $\hat{Q} = Q_1$, $\hat{m} = m - 1$, and $\hat{n} = n$. First we ensure that b is in state Q_1 .

$$\begin{array}{c}
\frac{\frac{\frac{\vdash_f \forall \alpha \beta \gamma (Q(\alpha) \rightarrow S(\alpha, \beta) \rightarrow R_1(\alpha, \gamma) \rightarrow D(\gamma) \rightarrow Q_1(\beta))}{\vdash_f Q(a) \rightarrow S(a, b) \rightarrow R_1(a, a_0) \rightarrow D(a_0) \rightarrow Q_1(b)} \quad \vdash_f Q(a)}{\vdash_f S(a, b) \rightarrow R_1(a, a_0) \rightarrow D(a_0) \rightarrow Q_1(b)} \quad \vdash_f S(a, b) \\
\vdots \\
\frac{\frac{Q_1(b) \vdash_f \mathbf{f}}{\vdash_f Q_1(b) \rightarrow \mathbf{f}}}{\vdash_f \mathbf{f}} \quad \frac{\frac{\vdash_f R_1(a, a_0) \rightarrow D(a_0) \rightarrow Q_1(b)}{\vdash_f D(a_0) \rightarrow Q_1(b)} \quad \vdash_f R_1(a, a_0)}{\vdash_f Q_1(b)} \\
\hline
\vdash_f \mathbf{f}
\end{array}$$

Now we decrement register 1 by taking a_1 (the predecessor of a_0) as anchor of b for register 1.

$$\begin{array}{c}
\frac{\frac{\frac{\vdash_f \forall \alpha \beta \gamma \delta (Q(\alpha) \rightarrow S(\alpha, \beta) \rightarrow R_1(\alpha, \gamma) \rightarrow D(\gamma) \rightarrow P(\gamma, \delta) \rightarrow R_1(\beta, \delta))}{\vdash_f Q(a) \rightarrow S(a, b) \rightarrow R_1(a, a_0) \rightarrow D(a_0) \rightarrow P(a_0, a_1) \rightarrow R_1(b, a_1)} \quad \vdash_f Q(a)}{\vdash_f S(a, b) \rightarrow R_1(a, a_0) \rightarrow D(a_0) \rightarrow P(a_0, a_1) \rightarrow R_1(b, a_1)} \quad \vdash_f S(a, b) \\
\vdots \\
\frac{\frac{R_1(b, a_1) \vdash_f \mathbf{f}}{\vdash_f R_1(b, a_1) \rightarrow \mathbf{f}}}{\vdash_f \mathbf{f}} \quad \frac{\frac{\vdash_f R_1(a, a_0) \rightarrow D(a_0) \rightarrow P(a_0, a_1) \rightarrow R_1(b, a_1)}{\vdash_f D(a_0) \rightarrow P(a_0, a_1) \rightarrow R_1(b, a_1)} \quad \vdash_f R_1(a, a_0)}{\vdash_f P(a_0, a_1) \rightarrow R_1(b, a_1)} \\
\hline
\vdash_f \mathbf{f}
\end{array}$$

Again it is obvious that we have deduced Γ_D ($\hat{a} := b$, $\hat{b}_i := b_i$ for $i \in \{0, \dots, n\}$, and $\hat{a}_{i-1} := a_i$ for $i \in \{1, \dots, m\}$). Hence, by induction hypothesis, we can deduce **false**. \square

Lemma 21.

M terminates on input $(0, 0)$ *iff* $\Gamma_M \vdash \mathbf{false}$ holds in system P .

Proof. The \Leftarrow direction is proven in Claim 19. And the \Rightarrow direction is a direct consequence of Claim 20 with $C = \langle Q_0, 0, 0 \rangle$. \square

Theorem 22. *CONS is undecidable.*

Proof. Since by Lemma 21 for a given two-counter automaton M we can effectively construct a set of **P**-formulas Γ_M such that M terminates on input $(0, 0)$ iff Γ_M is not consistent. It follows that **HALT** \leq **CONS**. Since **HALT** is undecidable we have shown that **CONS** is undecidable too. \square

4 INHAB is undecidable

Now we can show that the inhabitation problem in $\lambda 2$ is undecidable by reducing **CONS** to **INHAB**. Given a **P**-basis Γ we construct a $\lambda 2$ -basis $\bar{\Gamma}$ such that

$$\Gamma \vdash \mathbf{false} \quad \text{iff} \quad \bar{\Gamma} \vdash \mathbf{false}$$

where **false**

Definition 23. For a **P**-basis Γ and a $\lambda 2$ type t we define a set $\mathcal{U}(t)$, it contains the

$$(x_t : t \rightarrow \eta_2)$$

and for every P in *Predicate symbols of Γ* , $i \in \{1, 2\}$ the

$$(x_{t,p_i} : (t \rightarrow p_i) \rightarrow \eta_1).$$

Definition 24. We define a function ...

For a **P**-formula A , if A is an atomic formula then

$$\bar{A} = \begin{cases} \mathbf{false} & \text{if } A = \mathbf{false} \\ (\alpha \rightarrow p_1) \rightarrow (\beta \rightarrow p_2) \rightarrow p & \text{if } A = P(\alpha, \beta) \end{cases}$$

if A is an universal formula, it follows that there is an $n \in \mathbb{N}$ and atomic formulas A_1, A_2, \dots, A_n such that $A = \forall \vec{\alpha} (A_1 \rightarrow A_2 \rightarrow \dots \rightarrow A_n)$, then

$$\bar{A} = \forall \vec{\alpha} (\mathcal{U}(\vec{\alpha}) \rightarrow \bar{A}_1 \rightarrow \bar{A}_2 \rightarrow \dots \rightarrow \bar{A}_n)$$

if A is an existential formula, it follows that for some $n \in \mathbb{N}^+$ and some atomic formulas A_1, \dots, A_n it holds that $A = \exists \vec{\alpha} (A_1 \rightarrow \dots \rightarrow A_{n-1} \rightarrow \forall \beta ((A_n) \rightarrow \mathbf{false}) \rightarrow \mathbf{false})$, then

$$\bar{A} = \forall \vec{\alpha} (\mathcal{U}(\vec{\alpha}) \rightarrow \bar{A}_1 \rightarrow \dots \rightarrow \forall \beta (\mathcal{U}(\beta) \rightarrow \bar{A}_n \rightarrow \mathbf{false}) \rightarrow \mathbf{false})$$

For a **P**-basis Γ we define $\bar{\Gamma}$ as $\{\bar{A} \mid A \in \Gamma\} \cup \{\mathcal{U}(a) \mid a \in \text{FV}(\Gamma)\}$.

Claim 25. Let α, β be type-variables, x a value-variable, and Γ a basis such that β is not a target in Γ . Furthermore assume that $\Gamma, (x : \alpha \rightarrow \beta) \vdash e : \beta$ for some term e . It follows that there exists a Γ' such that $\Gamma \subseteq \Gamma'$, $\Gamma, (x : \alpha \rightarrow \beta) \vdash t$ for each $t \in \{t \mid (x : t) \in \Gamma'\}$, and $\Gamma', (x : \alpha \rightarrow \beta) \vdash e' : \alpha$ for some term e' .

Proof. By structural induction on the term e .

$\frac{e = y \text{ for some } y \in \mathcal{V}_V}{\text{asd}}$

$\frac{e = e_1 e_2 \text{ for some } e_1, e_2 \in \Lambda_{T_{\lambda_2}}}{\text{asd}}$

$$\frac{\Gamma, (x : \beta \rightarrow \alpha) \vdash e_1 e_2 : t}{\exists t' : \Gamma, (x : \beta \rightarrow \alpha) \vdash e_1 : t' \rightarrow t \quad \Gamma, (x : \beta \rightarrow \alpha) \vdash e_2 : t'}$$

$e = \lambda y : t_1. e'$ for some $e' \in \Lambda_{T_{\lambda_2}}$, some $y \in \mathcal{V}_V \setminus \text{dom}(\Gamma)$, $y \neq x$, and $t_1 \in T_{\lambda_2}$.
It follows that $t = t_1 \rightarrow t_2$ for some $t_2 \in T_{\lambda_2}$.

$$\frac{\Gamma, (x : \beta \rightarrow \alpha) \vdash \lambda y : t_1. e' : t_1 \rightarrow t_2}{\Gamma, (x : \beta \rightarrow \alpha), (y : t_1) \vdash e' : t_2}$$

$e = \Lambda \gamma. e'$ for some $e' \in \Lambda_{T_{\lambda_2}}$, $\gamma \in \mathcal{V}_T \setminus \text{FV}(\Gamma)$, $\gamma \neq \beta$, and $\gamma \neq \alpha$.
It follows that $t = \forall \gamma. t'$ for some $t' \in T_{\lambda_2}$.

$$\frac{\Gamma, (x : \beta \rightarrow \alpha) \vdash \Lambda \gamma. e' : \forall \gamma. t'}{\Gamma, (x : \beta \rightarrow \alpha) \vdash e' : t'}$$

$\frac{e = e' t' \text{ for some } e' \in \Lambda_{T_{\lambda_2}} \text{ and some } t' \in T_{\lambda_2}}{\text{asd}}$

$$\frac{\Gamma, (x : \beta \rightarrow \alpha) \vdash e' t' : t}{\exists \gamma \in \mathcal{V}_T, \hat{t} \in T_{\lambda_2} : \Gamma, (x : \beta \rightarrow \alpha) \vdash e' : \forall \gamma. \hat{t} \wedge \hat{t}[\gamma := t'] = t}$$

□

References

- [1] H.P. Barendregt, 1993. Lambda Calculi with Types, Handbook of Logic in Computer Science, Volume II, 34-68.