

# Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Basic Definitions</b>	<b>2</b>
2.1	$\lambda$ -calculus $\lambda_2$ . . . . .	2
2.2	first-order logic . . . . .	3
2.3	two-counter automaton . . . . .	6
<b>3</b>	<b>System P</b>	<b>7</b>
3.1	Definitions . . . . .	7
3.2	<b>CONS</b> is undecidable . . . . .	8

# 1 Introduction

## 2 Basic Definitions

We will denote the set  $\{1, \dots, n\}$  by  $[n]$ .

### 2.1 $\lambda$ -calculus $\lambda 2$

$$FV(\Gamma) = \bigcup \{FV(t) \mid (x : t) \in \Gamma\}$$

In the following let  $\mathcal{V}_T = \{\alpha, \beta, \dots\}$  be a countable set (of type-variables) and  $\mathcal{V}_V = \{x_1, x_2, \dots\}$  be a countable 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  $\forall \alpha. t \in T$ .

**Definition 2.** 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  $\Lambda_T$  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  $e t \in \Lambda_T$ .

**Definition 3.** Let  $e \in \Lambda_{T_{\lambda 2}}$ . The free variables of  $e$ , denoted by  $FV(e)$ , are defined inductively as follows:

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

Or is this definition better?

**Definition 4.** Let  $e \in \Lambda_{T_{\lambda 2}}$ . The free variables of  $e$ , denoted by  $FV(e)$ , are defined inductively as follows:

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

**Definition 5.** A basis is a finite subset of  $\mathcal{V}_V \times \Lambda_{T_{\lambda 2}}$

$\lambda 2$  deduction Rules

(Axiom)	$\Gamma, x : t \vdash x : t$	
( $\lambda$ -Introduction)	$\frac{\Gamma, x : t_1 \vdash e : t_2}{\Gamma \vdash \lambda x. e : t_1 \rightarrow t_2}$	
( $\lambda$ -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)	$\frac{\Gamma \vdash e : t}{\Gamma \vdash \Lambda \alpha. e : \forall \alpha. t}$	$\alpha \notin FV(\Gamma)$
( $\forall$ -Elimination)	$\frac{\Gamma \vdash e : \forall \alpha. t}{\Gamma \vdash e t' : t[\alpha := t']}$	$t' \in T_{\lambda 2}$

## 2.2 first-order logic

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

If the function  $rk$  is understood we will just write  $\Sigma$  instead of  $(\Sigma, rk)$ . The set of all elements with a certain rank  $k$  in  $\Sigma$ , denoted by  $\Sigma^{(k)}$ , is defined by  $\Sigma^{(k)} := rk^{-1}(k)$ . In the following we will write  $\Sigma = \{P^{(0)}, Q^{(3)}\}$  to say that  $\Sigma = \{P, Q\}$ ,  $rk(P) = 0$ , and  $rk(Q) = 3$ .

In the following let  $\mathcal{V} = \{x_0, x_1, \dots\}$  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  $(\mathcal{V}, \mathcal{F})$ , 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  $(\mathcal{V}, \mathcal{F}, \mathcal{P})$ , 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  $x \in \mathcal{V}$  and  $\varphi \in \mathcal{L}$  then  $\exists x \varphi, \forall x \varphi \in \mathcal{L}$ .

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  $(\neg \varphi \vee \psi)$ . For nullary relation symbols  $P$  we will abbreviate  $P()$  to  $P$ .

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

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

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

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

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

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

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

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

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

**Definition 10.** An interpretation  $I$  over  $(\mathcal{V}, \mathcal{F}, \mathcal{P})$  is a triple  $(\Delta, \cdot^I, \omega)$  where  $\Delta$  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  $k \in \mathbb{N}$ ,  $f \in \mathcal{P}^{(k)}$  and  $\omega$  is a function from  $\mathcal{V}$  to  $\Delta$ .

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

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

**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:

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

**Definition 12.** Let  $I = (\Delta, \cdot^I, \omega)$  be an interpretation and  $\varphi$  a formula the interpretation of  $\varphi$  under  $I$ , denoted by  $\varphi^I$ , is defined recursively as follows:

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

The interpretation  $I$  is a model of  $\varphi$ , 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 13.** Let  $\Gamma$  be a finite set of first-order formulas.

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

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

The free variables of  $\Gamma$ , 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 try to decrement a register and jump if the register is already zero. Formally:

**Definition 14.** 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  $\langle Q, m, n \rangle, \langle Q', m', n' \rangle$  of  $M$  are in the relation if all of the following conditions hold:

- $Q \neq Q_f, r = R(Q)$ ,
- if  $r = +(1, P)$  for some  $P \in \mathcal{Q}$  then  $Q' = P, m' = m + 1$ , and  $n' = n$ ,
- if  $r = +(2, P)$  for some  $P \in \mathcal{Q}$  then  $Q' = P, m' = m$ , and  $n' = n + 1$ ,
- if  $r = -(1, P_1, P_2)$  for some  $P_1, P_2 \in \mathcal{Q}$  then
  - if  $m = 0$  then  $Q' = P_2, m' = 0$ , and  $n' = n$ ,
  - if  $m \geq 1$  then  $Q' = P_1, m' = m - 1$ , and  $n' = n$ ,
- if  $r = -(2, P_1, P_2)$  for some  $P_1, P_2 \in \mathcal{Q}$  then
  - if  $n = 0$  then  $Q' = P_2, m' = m$ , and  $n' = 0$ ,
  - if  $n \geq 1$  then  $Q' = P_1, m' = m$ , and  $n' = n - 1$ .

The transition relation of  $M$ , denoted by  $\Rightarrow_M$ , is defined as  $\bigcup_{r \in R(\mathcal{Q} \setminus \{Q_f\})} \Rightarrow_M^r$ . 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  $m', n' \in \mathbb{N}$  such that  $\langle Q_0, m, n \rangle \Rightarrow_M^* \langle Q_f, m', n' \rangle$ .

**Definition 15.** 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 set (of variables). Let  $\mathcal{P}_P = \{P, Q, \dots\}$  be a set (of predicate symbols) and  $\mathcal{P}$  a ranked set such that  $\mathcal{P}^{(0)} = \{\text{false}\}$ ,  $\mathcal{P}^{(2)} = \mathcal{P}_P$ , and  $\mathcal{P}^{(k)} = \emptyset$  for all  $k \in \mathbb{N} \setminus \{0, 2\}$ . A first-order logic formula  $\varphi$  over  $(\mathcal{V}_P, \emptyset, \mathcal{P})$  is an

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

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

**existential formula** if there exists  $n \geq 0$ , atomic formulas  $A_i \neq \text{false}$  for  $i \in [n]$  such that  $\varphi = \forall \vec{\alpha} (A_1 \rightarrow A_2 \rightarrow \dots \rightarrow A_{n-1} \rightarrow \forall \beta (A_n \rightarrow \text{false}) \rightarrow \text{false})$ .

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

Deduction Rules

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

An Interpretation  $I$  of a P formula is a tuple  $I = (\Delta, \cdot^I)$  where  $\Delta$  is a set (called domain),  $P^I \subseteq \Delta^k$  and  $\alpha^I \in \Delta \dots$

If we interpret false with the logical constant false ( $\perp$ ) (denoted by  $\vdash_f$ ) we can add a new deduction rule.

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

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

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

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 16.** 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  $\Gamma$ .

Does  $\Gamma \vdash \text{false}$  not hold.

### 3.2 CONS is undecidable

We will show that **HALT**  $\leq$  **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 set of **P**-formulas  $\Gamma_M$  such that

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

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  $(\mathcal{V}_P, \mathcal{P}_P)$ , 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.

Intuitively  $Q(a)$  stands for “ $a$  is in state  $Q$ ”,  $R_i(a, m)$  stands for “in  $a$  the value of register  $i$  is  $m$ ” for  $i \in \{1, 2\}$ ,  $S(a, b)$  states that “ $b$  is a successor of  $a$ ”,  $P(a, b)$  states that “ $b$  is a predecessor of  $a$ ”,  $E(a)$  marks “ $a$  as the end of chain”, and  $D(a)$  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  $\Gamma_C$ . 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  $i \in \{1, \dots, n\}$
- $D(a), D(a_i), D(b_j)$  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  $r \in \mathcal{R}_Q$  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
- $\forall \alpha \beta \gamma (Q(\alpha) \rightarrow S(\alpha, \beta) \rightarrow R_2(\alpha, \gamma) \rightarrow R_2(\beta, \gamma))$   
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  $\Gamma_1$  to ensure that our representation works correctly. The following formula are in  $\Gamma_1$ :

- $\forall \alpha \beta (S(\alpha, \beta) \rightarrow D(\beta))$   
no successor is the end of a chain
- $\forall \alpha (D(\alpha) \rightarrow \forall \beta (R_1(\alpha, \beta) \rightarrow \text{false}) \rightarrow \text{false})$   
every element that represents a configuration has a value for register 1

- $\forall \alpha (D(\alpha) \rightarrow \forall \beta (R_2(\alpha, \beta) \rightarrow \text{false}) \rightarrow \text{false})$   
every element that represents a configuration has a value for register 2
- $\forall \alpha (\forall \beta (S(\alpha, \beta) \rightarrow \text{false}) \rightarrow \text{false})$   
every element has a successor

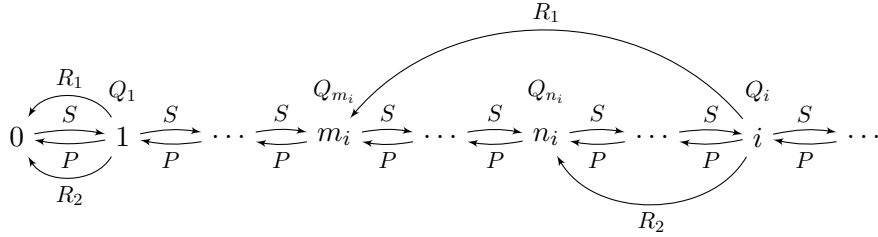
We define  $\Gamma_{\overline{M}}$  as  $\bigcup_{Q \in \mathcal{Q} \setminus \{Q_f\}} \Gamma_{Q, R(Q)} \cup \{\forall \alpha (Q_f(\alpha) \rightarrow \text{false})\} \cup \Gamma_1$ . Finally we can define  $\Gamma_M$  as  $\Gamma_{C_1} \cup \Gamma_{\overline{M}}$ , where  $C_1 = \langle Q_0, 0, 0 \rangle$  is the initial configuration.

**Claim 17.**

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

*Proof.* Assume  $M$  does not terminate then there is an infinite chain  $C_1 \Rightarrow_M C_2 \Rightarrow_M C_3 \Rightarrow_M \dots$  ( $C_i = \langle Q_i, m_i, n_i \rangle$  for  $i \in \mathbb{N}^+$ ). Now we construct a model of  $\Gamma_M$  which interprets false with  $\perp$  this contradicts  $\Gamma_M \vdash \text{false}$ .

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  $\overset{P}{d}$  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  $i$  greater than zero will also represent the  $i^{\text{th}}$  configuration of our infinite computation. Now the idea for our model of  $\Gamma_M$  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  $I = (\mathbb{N}, \cdot^I, \omega)$ .

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

$$a^I = 1 \quad a_0^I = 0 \quad b_0^I = 0$$

Since there are no free variables in  $\Gamma_M$  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  $\Gamma_M$ .  $\square$

**Claim 18.** *Let  $C = \langle Q, m, n \rangle$  be a configuration of  $M$ . If a final configuration (i.e. a configuration  $\langle Q_f, m', n' \rangle$  for some  $n', m' \in \mathbb{N}$ ) is reachable from  $C$  then  $\Gamma_C \cup \Gamma_{\overline{M}} \vdash \text{false}$ .*

*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  $n, m \in \mathbb{N}$ . Hence,  $Q_f(a)$  is in  $\Gamma_C$  for some  $a \in \mathcal{V}_P$  and  $\forall \alpha (Q_f(\alpha) \rightarrow \text{false})$  is in  $\Gamma_{\overline{M}}$ , we can easily deduce false.

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

Induction Step:  $i = i' + 1$

Since  $I \models \text{false}$  holds trivially if  $I$  interprets false with  $\top$  we only need to consider models of  $\Gamma_C \cup \Gamma_{\overline{M}}$  that interpret false with  $\perp$  (note that there are none 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  $D = \langle P, m', n' \rangle$  such that  $C \Rightarrow_M^r D$  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 basic idea is to deduce  $\Gamma_D$  from  $\Gamma_C \cup \Gamma_{\overline{M}}$  and then apply the induction hypothesis to  $\Gamma_D \cup \Gamma_{\overline{M}}$ .

$$\frac{\frac{IH}{\Gamma_C \cup \Gamma_{\overline{M}} \cup \Gamma_D \vdash_f \text{false}} \quad \Gamma_C \cup \Gamma_{\overline{M}} \vdash_f \Gamma_D}{\Gamma_C \cup \Gamma_{\overline{M}} \vdash_f \text{false}}$$

There are four possible cases for the type of the rule  $r$ . We will only consider the cases  $r = +(1, Q')$  and  $r = -(1, Q_1, Q_2)$ , the two 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. In both cases we need a new free variable  $b$  representing the configuration  $D$  and the value in register 2 does not change.

For the succeeding tableau proofs we will abbreviate false by  $\text{fand}$  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 variables representing  $D$  (let  $b \in \mathcal{V}_P \setminus \text{FV}(\Gamma_C \cup \Gamma_{\overline{M}})$ ).

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

The formula  $R_1(b, d)$  can be acquired in a similar way. Again we will just drop  $S(a, b)$  and  $D(b)$  on the left side for comprehensibility.

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

Now we have all the new free variables we need and we continue by ensuring that these variables fulfill all the formulas in  $\Gamma_D$ .

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

Starting from  $Q'(b) \vdash_f \text{false}$  we can connect  $d$  and  $a_0$ .

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

For register one we still need  $D(d)$ .

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

Since register two should not change we only need  $R_2(b, b_0)$ .

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

Now we have  $\Gamma_C$  (Since  $P(a_{i-1}, a_i)$  is already in  $\Gamma_D$ ) and can deduce false by induction hypothesis.

Case  $r = -(Q, 1, Q_1, Q_2)$   
 $r_1 = 0$

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

$r_1$  stays zero

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

$r_1 \geq 1$   
new state  $Q_1$

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

decrement  $r_1$

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

□

**Lemma 19.**

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

*Proof.* The  $\Leftarrow$  direction follows directly from Claim 17. And the  $\Rightarrow$  direction is a direct consequence of Claim 18 with  $C = \langle Q_0, 0, 0 \rangle$ . □

**Theorem 20.** *CONS is undecidable.*

*Proof.* Since by Lemma 19 for a given two-counter automaton  $M$  we can effectively construct a set of **P**-formulas  $\Gamma_M$  such that  $M$  terminates on input  $(0, 0)$  iff  $\Gamma_M$  is not consistent. It follows that **HALT**  $\leq$  **CONS**. Hence, since **HALT** is undecidable, we have shown that **CONS** is undecidable too. □