Undecidability of the validity problem

We prove the undecidability of the validity problem for formulas of predicate logic with equality.

Recall: there is an algorithm that given a formula of predicate logic with equality returns a sat-equivalent formula of predicate logic (without equality).

It follows the validity problem for formulas of predicate logic without equality is also undecidable.

Goto-**programs**

The proof is by reduction from the halting problem for **goto**-programs.

$$
\begin{array}{rcll}
\mathit{Prog} & ::= & \ell : \mathit{Assign} & \text{(assignment)} \\
 & & \ell : \ \mathbf{goto}\ \ell' & \text{(unconditional jump)} \\
 & & \ell : \mathbf{if}\ x_i = 0\ \mathbf{then}\ \mathbf{goto}\ \ell' & \text{(conditional jump)} \\
 & & \ell : \mathbf{halt} & \text{(termination)} \\
 & & \mathit{Prog}\ ;\ \mathit{Prog} & \text{(concatenation)}
\end{array}
$$

$$
\begin{array}{rcl}
\mathit{Assign} & ::= & x_i := 0 \ \mid \ x_i := x_j \\
 & & x_i := x_j + 1 \ \mid \ x_i := x_j - 1 \\
\ell & ::= & 1 \mid 2 \mid 3 \mid \ldots
\end{array}
$$

Example

1: **if** $x_1 = 0$ **then goto** 4;

2: $x_1 := x_1 - 1$;

3: **goto** 1;

4: **halt**

Claim: **goto**-programs can simulate any program.

By the claim: a problem is decidable if it is solved by some **goto**-program.

We prove the following two theorems:

Theorem: The halting problem for **goto**-programs is undecidable: There is no (**goto**-)program that takes as input a **goto**-program P and a valuation β of the variables of P and decides whether P initialized with β terminates.

Theorem: If the validity problem for predicate logic is decidable, then the halting problem for **goto**-programs is decidable.

Coding

Fact: Programs and valuations can be encoded as integers.

Notations:

- $P(a_1, \ldots, a_i)$ denotes the Program P initialized with $(a_1, \ldots, a_i, 0, \ldots, 0)$.
  I.e., variables $x_1, \ldots, x_i$ are initialized with $a_1, \ldots, a_i$ and variables $x_{i+1}, \ldots, x_n$ with 0.
- Π_n denotes the program with code number n (if the program exists).

Computable encodings

Fact: There exist computable encodings, i.e., encodings for which the following programs exist:

- Encoder.
  Input: a program P.
  Output: the code of P, i.e., the number n such that $P = \Pi_n$.
- Decoder.
  Input: a number n.
  Output: the program Π_n if n encodes a program, otherwise 'Not a program'.

Assumption: There is a program T such that for every pair $n, m \in \mathbb{N}$ the initialized program $T(n, m)$ halts and reports

Not a program	if n is not the code of a program
Yes	if n is the code of a program and $\Pi_n(m)$ halts
No	if n is the code of a program and $\Pi_n(m)$ does not halt

We show that this assumption leads to a contradiction.

The contradiction

Fact: The asumption implies the existence of a program T' such that for every $n \in \mathbb{N}$ the initialized program $T'(n)$

halts	if n is the code of a program and $\Pi_n(n)$ does not halt
does not halt	if n is not the code of a program or $\Pi_n(n)$ halts

Let k be the code of T', i.e., $\Pi_k = T'$. Either the initialized program $T'(k)$ halts, or it does not halt. But:

	$T'(k)$ halts	
$\Rightarrow$	k is the code of a program and	
	$\Pi_k(k)$ does not halt	(Def. of T')
$\Rightarrow$	$T'(k)$ does not halt	($\Pi_k = T'$)
	$T'(k)$ does not halt	
$\Rightarrow$	$\Pi_k(k)$ halts	(Def. von T', k is code)
$\Rightarrow$	$T'(k)$ halts	($\Pi_k = T'$)

So the assumption is false.

Undecidability of the validity problem

We assign to every program P and valuation β a formula $\phi_{P\beta}$ of predicate logic with equality such that

$\phi_{P\beta}$ is valid

if and only if

P with initialization β halts

There is a program that on input P, β outputs $\phi_{P\beta}$.

So no program can solve the validity problem.

Notations and definitions

Let k denote the number of instructions of P.
(The last instruction is always $\mathbf{halt}$.)

Let n denote the number of variables of P.
(I.e., the variables of P are $x_1, \ldots, x_n$.)

A configuration of P is a tuple $(pc, m_1, \ldots, m_n) \in \mathbb{N}^{n+1}$.
pc is the current value of the program counter and $m_1, \ldots m_n$ the current valuation of the variables.

Convention: the successor of a configuration $(\ell_k, m_1, \ldots, m_n)$ is again $(\ell_k, m_1, \ldots, m_n)$.

Symbols of the formula $\phi_{P\beta}$

- R, predicate symbol of arity $(n+2)$.
- $<$, predicate symbol of arity 2.
- f, function symbol of arity 1.
- $\mathbf{0}$, constant.

Canonical structure $\mathcal{A}$

- Universe: $\mathbb{N}$.
- $<^{\mathcal{A}}$ is the usual order on $\mathbb{N}$.
- $\mathbf{0}^{\mathcal{A}} = 0$.
- $f^{\mathcal{A}}$ is the successor function, i.e., $f^{\mathcal{A}}(i) = i + 1$.
- $R^{\mathcal{A}}(s, pc, m_1, \ldots, m_n) = 1$ if $(pc, m_1, \ldots, m_n)$ is the configuration of P after s steps (for the initialization β).

The auxiliary formula $\psi_{P\beta}$

$$\psi_{P\beta} = \quad \psi_0 \quad \wedge \quad R(\mathbf{0}, \beta) \quad \wedge \quad \psi_1 \quad \wedge \quad \ldots \quad \wedge \quad \psi_{k-1}$$

Meaning of $R(\mathbf{0}, \beta)$ in the structure $\mathcal{A}$: P is initialized with β

In the structure $\mathcal{A}$ the formula ψ_i describes the effect of the i-th instruction of P. For instance:

- If i: $x_j := x_j + 1$ then

$$\begin{aligned} \psi_i \quad = \quad & \forall x \forall y_1 \ldots \forall y_n \; (\\ & R(x, f^i(\mathbf{0}), y_1, \ldots y_n) \quad \rightarrow \\ & R(f(x), f^{(i+1)}(\mathbf{0}), y_1, \ldots y_{j-1}, f(y_j), y_{j+1}, \ldots, y_n) \\ &) \end{aligned}$$

- If i: $\textbf{if } x_j = 0 \textbf{ then goto } \ell$ then

$$\begin{array}{lll} \psi_i & = & \forall x \forall y_1 \ldots \forall y_n \ (\\ & & R(x, f^i(\mathbf{0}), y_1, \ldots y_n) \to \\ & & \quad (\quad y_j = \mathbf{0} \quad \wedge \quad R(f(x), f^{\ell}(\mathbf{0}), y_1, \ldots, y_n) \\ & & \qquad\qquad \vee \\ & & \qquad \neg(y_j = \mathbf{0}) \quad \wedge \quad R(f(x), f^{(i+1)}(\mathbf{0}), y_1, \ldots, y_n) \\ & & \quad) \\ & &) \end{array}$$

ψ_0 guarantees that in every model the symbol $<$ is interpreted as a total order, that $\mathbf{0}$ is its smallest element, that $x < f(x)$ holds, and that $f(x)$ is the $<$-successor of x:

$$
\begin{aligned}
\psi_0 \quad = \quad & \forall x \forall y ((x < y) \to \neg(y < x)) \quad \wedge \\
& \forall x \forall y \forall z ((x < y \wedge y < z) \to x < z) \quad \wedge \\
& \forall x (\mathbf{0} < x \vee \mathbf{0} = x) \quad \wedge \\
& \forall x (x < f(x)) \quad \wedge \\
& \forall x \forall z (x < z \to (f(x) < z \vee f(x) = z)
\end{aligned}
$$

The formula $\phi_{P\beta}$

We set

$$\phi_{P\beta} \quad = \quad \psi_{P\beta} \quad \longrightarrow \quad \exists x \exists y_1 \ldots \exists y_n \; R(x, f^k(\mathbf{0}), y_1, \ldots, y_n)$$

Theorem: $\phi_{P\beta}$ is valid iff program P with initialization β halts.

Proof: ($\Rightarrow$): If $\phi_{P\beta}$ is valid, then in particular the canonical structure $\mathcal{A}$ is a model of $\phi_{P\beta}$. Since $\mathcal{A} \models \psi_{P\beta}$ clearly holds, we get $\mathcal{A} \models \exists x \exists y_1 \ldots \exists y_n \; R(x, f^k(\mathbf{0}), y_1, \ldots, y_n)$. So P initialized with β halts.

($\Leftarrow$): (Sketch.) If $\phi_{P\beta}$ is not valid, then there is a structure $\mathcal{B} = (U_\mathcal{B}, I_\mathcal{B})$ such that

$$\mathcal{B} \models \psi_{P\beta} \text{ and } \mathcal{B} \not\models \exists x \exists y_1 \ldots \exists y_n \ R(x, f^k(\mathbf{0}), y_1, \ldots, y_n) \ .$$

For every $i \geq 0$ let d_i be the element of $U_\mathcal{B}$ such that $(f^i(\mathbf{0}))^\mathcal{B} = d_i$. Since $\mathcal{B} \models \psi_{P\beta}$ we have $\mathcal{B} \models \psi_0$, and so for $D = \{d_i \mid i \geq 0\}$ we have (why?):

$$<^\mathcal{B} \cap (D \times D) = \{(d_i, d_j) \mid i, j \in \mathbb{N}, i < j\}$$

Let $(pc^{(i)}, m_1^{(i)}, \ldots, m_n^{(i)})$ be the configuration of P after i steps (with initialization β). Since $\mathcal{B} \models \psi_{P\beta}$ for every $i \geq 0$ we have $(d_i, d_{pc^{(i)}}, d_{m_1^{(i)}}, \ldots, d_{m_n^{(i)}}) \in R^\mathcal{B}$. Since $\mathcal{B} \not\models \exists x \exists y_1 \ldots \exists y_n \ R(x, f^k(\mathbf{0}), y_1, \ldots, y_n)$, P does not terminate when initialized with β.

An alternative proof

The tiling problem:

Given: finite set of square tiles S, a horizontal relation $H \subseteq S \times S$ and a vertical relation $V \subseteq S \times S$.

Question: Can the plane ($\mathbb{N} \times \mathbb{N}$) be tiled with the given tiles in such a way, that neighboring tiles satisfy the horizontal or vertical relation? More precisely, does there exist a mapping $\chi : \mathbb{N} \times \mathbb{N} \to S$ such that for all $m, n \in \mathbb{N}$ we have

- if $\chi(m, n) = s$ and $\chi(m+1, n) = s'$, then $(s, s') \in H$ and
- if $\chi(m, n) = s$ and $\chi(m, n+1) = s'$, then $(s, s') \in V$?

Theorem (without proof): The tiling problem is undecidable.

The reduction

We define for each set S of tiles a formula $\phi_{S,H,V}$ that is satisfiable iff the plane can be tiled with S. Why does this prove the undecidability of the validity problem then?

Symbols: predicate symbol P_s of arity 2 for each tile $s \in S$,
function symbol f of arity 1.

Canonical structure $\mathcal{A}_\chi$ for each coloring χ:

- Universe: $\mathbb{N}$.
- $f^\mathcal{A}$ is the successor function, i.e., $f^\mathcal{A}(n) = n + 1$.
- $(m, n) \in P_s$ if and only if $\chi(m, n) = s$.

The formula $\phi_{S,H,V}$

We take $\phi_{S,H,V} = \forall x \forall y \ (F_1 \wedge F_2)$ where

$$
\begin{aligned}
F_1 &= \bigwedge_{s \neq s'} \neg (P_s(x,y) \wedge P_{s'}(x,y)) \\
F_2 &= \bigvee_{(s,s') \in H} (P_s(x,y) \wedge P_{s'}(f(x),y)) \ \wedge \\
&\quad \bigvee_{(s,s') \in V} (P_s(x,y) \wedge P_{s'}(x,f(y)))
\end{aligned}
$$

Consequences

Corollary: The satisfiability problem is undecidable for closed formulas of the form $F = \forall x \forall y \ F^*$.

Corollary: The satisfiability problem is undecidable for closed formulas of the form $F = \forall x \exists z \forall y \ F^*$, where F^* contains no function symbols.

Prefix classes

We consider formulas in prenex form without function symbols.

Undecidable classes:

- $\forall^*\exists^*$ (Skolem, 1920)
- $\forall\forall\forall\exists$ (Suranyi, 1959)
- $\forall\exists\forall$ (Kahr, Moore, Wang, 1962)

Decidable classes:

- $\exists^*\forall^*$ (Bernays, Schönfinkel, 1928)
- $\exists^*\forall\exists^*$ (Ackerman, 1928)
- $\exists^*\forall^2\exists^*$ (Gödel 1932, Kalmar 1933, Schütte 1934)