First-Order Logic
Undecidability

[Cutland, *Computability*, Section 6.5.]

- Aim:
  Show that validity of first-order formulas is undecidable
- Method:
  Reduce the halting problem for register machines to validity of formulas by expressing "program behaviour" as formulas

Logical formulas can talk about computations!

Register machine programs (RMPs)

A register machine program is a sequence of instructions $I_1, \ldots, I_t$. The instructions manipulate registers R_i ($i = 1, 2, \ldots, r$) that contain (unbounded!) natural numbers.

There are 4 types of instructions:

$$R_i := 0$$

$$R_i := R_i + 1$$

$$R_i := R_j$$

$$\text{IF } R_i = R_j \text{ GOTO } p$$

Assumption: all jumps in a program go to $1, \ldots, t+1$, and execution terminates when the PC (the number of the next instruction to be executed) is $t + 1$.

The state of P during execution can be described by a tuple of $r + 1$ natural numbers

$$(n_1, \ldots, n_r, k)$$

where n_i is the content of R_i and k is the value of the PC.

Undecidability

Theorem (Undecidability of the halting problem for RMPs)
It is undecidable if a given register machine program terminates when started in state $(0, \ldots, 0, 1)$.

We reduce the halting problem for RMPs to the validity problem for first-order formulas.

Notation:
$P(0) \downarrow =$ "RMP P started in state $(0, \ldots, 0, 1)$ terminates"

Theorem
Given an RMP P we can effectively construct a closed formula φ_P such that $P(0) \downarrow$ iff $\models \varphi_P$.

Proof by construction of φ_P from $P = I_1, \ldots, I_t$.

Funct. symb.: z, s. Abbr.: $\overline{0} = z$, $\overline{1} = s(z)$, $\overline{2} = s(s(z))$, ...
Pred. symb.: R (arity: $r+1$). (Think "reachable".)
Aim: if $R(\overline{n_1}, \ldots \overline{n_r}, \overline{k})$ then $(0, \ldots, 0, 1) \overset{P}{\rightsquigarrow} (n_1, \ldots, n_r, k)$.

1) For every I_i construct closed formula Ψ_i:

$I_i = (R_n := 0)$: $\Psi_i := \forall x_1 \ldots x_r \ (R(x_1, \ldots, x_n, \ldots, x_r, \overline{i}) \rightarrow$
$R(x_1, \ldots, z, \ldots, x_r, \overline{i+1}))$

$I_i = (R_n := R_n + 1)$: same except $s(x_n)$ instead of z

$I_i = (R_n := R_m)$: same except x_m instead of z

$I_i = (IF\ R_m = R_n\ GOTO\, p)$:
$\Psi_i := \forall x_1 \ldots x_r\, (R(x_1, \ldots, x_r, \overline{i}) \rightarrow (x_m = x_n \rightarrow R(x_1, \ldots, x_r, \overline{p})) \wedge$
$(x_m \neq x_n \rightarrow R(x_1, \ldots, x_r, \overline{i+1})))$

2) Define $\Psi_P := \Psi \wedge R(z, \ldots, z, s(z)) \wedge \Psi_1 \wedge \cdots \wedge \Psi_t$ where
$\Psi := \forall x \forall y (s(x) = s(y) \rightarrow x = y) \wedge \forall x (z \neq s(x))$.
Ψ enforces that every model is "similar enough" to $\mathbb{N}$.

3) Define $\varphi_P := \Psi_P \rightarrow \tau$ where $\tau := \exists x_1 \ldots x_r\, R(x_1, \ldots, x_r, s(\overline{t}))$.

Claim: $P(0)\downarrow$ iff $\models \varphi_P$, that is, $P(0)\downarrow$ iff $\models \Psi_P \to \tau$.

"$\Rightarrow$": Assume $P(0)\downarrow$. We show: for every $\mathcal{A}$, if $\mathcal{A} \models \Psi_P$ then $\mathcal{A} \models \tau$. Assume $\mathcal{A} \models \Psi_P$.

Lemma

If $(0,\ldots,0,1) \overset{P}{\rightsquigarrow} (n_1,\ldots,n_r,k)$ *then* $\mathcal{A} \models R(\overline{n_1},\ldots,\overline{n_r},\overline{k})$

Proof by induction on the length of the execution using $\mathcal{A} \models \Psi_P$.

Thus $\mathcal{A} \models \tau$ because $P(0)\downarrow$.

"$\Leftarrow$": Assume $\models \Psi_P \to \tau$. We show $P(0)\downarrow$.

We have $\mathcal{N} \models \Psi_P \to \tau$ for the structure $\mathcal{N}$ given by

$$U_{\mathcal{N}} := \mathbb{N} \quad z^{\mathcal{N}} := 0 \quad s^{\mathcal{N}}(n) := n+1\ .$$

In this structure $R^{\mathcal{N}} := \{s \mid (0,\ldots,0,1) \overset{P}{\rightsquigarrow} s\}$ and so $\mathcal{N} \models \Psi_P$.

From $\mathcal{N} \models \Psi_P$ and $\mathcal{N} \models \Psi_P \to \tau$ we get $\mathcal{N} \models \tau$, which implies $P(0)\downarrow$.