First-Order Logic
Undecidability

[Cutland, *Computability*, Section 6.5.]
Aim:
Show that validity of first-order formulas is undecidable

Method:
Reduce the halting problem 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 $R_i$ ($i = 1, 2, \ldots$) that contain (unbounded!) natural numbers. There are 4 instructions:

\begin{align*}
R_n & := 0 \\
R_n & := R_n + 1 \\
R_n & := R_m \\
\text{IF } R_m = R_n \text{ GOTO } p
\end{align*}

Assumption: all jumps in a program go to $1, \ldots, t + 1$; execution terminates when the PC is $t + 1$.

Let $r$ be the maximal index of any register used in a program $P$. Then the state of $P$ during execution can be described by a tuple of natural numbers

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

where $n_i$ is the contents of $R_i$ and $k$ is the PC (the number of the next instruction to be executed).
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:

$P(0) \downarrow = \text{“RMP } P \text{ started in state } (0, \ldots, 0, 1) \text{ terminates”}$

Theorem

*Given an RMP $P$ we can effectively construct a closed formula $\varphi_P$ such that $P(0) \downarrow$ iff $\models \varphi_P$.***
Proof by construction of $\varphi_P$ from $P = l_1, \ldots, l_t$.

Funct. symb.: $z$, $s$. Abbr.: $\bar{0} = z$, $\bar{1} = s(z)$, $\bar{2} = s(s(z))$, $\ldots$

Pred. symb.: $R$ (arity: $r + 1$) “reachable”

Aim: if $R(\bar{n}_1, \ldots, \bar{n}_r, \bar{k})$ then $(0, \ldots, 0, 1) \xrightarrow{P} (n_1, \ldots, n_r, k)$

For every $l_i$ construct closed formula $\Psi_i$:

$l_i = (R_n := 0)$: $\Psi_i := \forall x_1 \ldots x_r \left( R(x_1, \ldots, x_n, \ldots, x_r, \bar{i}) \rightarrow R(x_1, \ldots, z, \ldots, x_r, s(\bar{i})) \right)$

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

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

$l_i = (IF R_m = R_n GOTO p)$:

$\Psi_i := \forall x_1 \ldots x_r \left( R(x_1, \ldots, x_r, \bar{i}) \rightarrow (x_m = x_n \rightarrow R(x_1, \ldots, x_r, \bar{p})) \land (x_m \neq x_n \rightarrow R(x_1, \ldots, x_r, s(\bar{i}))) \right)$

$\Psi_P := \Psi \land R(z, \ldots, z, s(z)) \land \Psi_1 \land \cdots \land \Psi_t$

$\Psi$ enforces that every model is similar to $\mathbb{N}$:

$\Psi := \forall x \forall y (s(x) = s(y) \rightarrow x = y) \land \forall x (z \neq s(x))$

(How can models of $\Psi$ differ from $\mathbb{N}$?)
$\varphi_P := \Psi_P \rightarrow \tau$ where $\tau := \exists x_1 \ldots x_r R(x_1, \ldots, x_r, s(t))$

Claim: $P(0) \downarrow$ iff $\models \varphi_P$

“$\Rightarrow$”: Assume $P(0) \downarrow$, show $\models \varphi_P$. Assume $A \models \Psi_P$.

Lemma

If $(0, \ldots, 0, 1) \xrightarrow{P} (n_1, \ldots, n_r, k)$ then $A \models R(n_1, \ldots, n_r, k)$

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

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

“$\Leftarrow$”: $\models \varphi_P \Rightarrow \mathcal{N} \models \varphi_P \Rightarrow (\mathcal{N} \models \Psi_P \Rightarrow \mathcal{N} \models \tau) \Rightarrow P(0) \downarrow$

where $U_{\mathcal{N}} := \mathbb{N}$, $z^\mathcal{N} := 0$, $s^\mathcal{N}(n) := n + 1$, $R^\mathcal{N} := \{s \mid (0, \ldots, 0, 1) \xrightarrow{P} s\}$