First-Order Logic
Undecidability

[Cutland, *Computability*, Section 6.5.]

- Aim:
  Show that validity of first-order formulas is undecidability
- 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 $I_1, \ldots, I_s$.
The instructions manipulate registers R_i $(i = 1, 2, \ldots)$
that contain (unbounded!) natural numbers.
There are 4 instructions:

$$\begin{array}{l} R_n := 0 \\ R_n := R_n + 1 \\ R_n := R_m \\ \text{IF } R_m = R_n \text{ GOTO } p \end{array}$$

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

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:
$P(0) \downarrow =$ "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* $\models \varphi_P$.