The Halting Problem

Given the code for program `P` and input `i` for this program, decide whether `P` terminates when executed on input `i`.

The aim is to design a program `H` which determines this – returns 1 if `P` terminates on `i` and 0 otherwise.

Alan Turing proved in 1936 that a program solving the halting problem for all possible program-input pairs can't exist.

Real-time Languages

Real-time languages guarantee every program to be time-able with respect to the worst-case. They take out parts of the languages (e.g. while loops) to guarantee this.

These are used in safety-critical applications like e.g. plane autopilot.

Infinite Resources

We assume infinite resources for the halting problem – programs can take arbitrarily long or use arbitrarily much storage space.

Difficulty

The decision procedure must work for all possible programs and inputs.

Variants

- halts on a specific input?
- any input that causes it to halt?
    - very weak question
- ever outputs 5?

All of these questions can't be answered by a program, but these are simple questions that debuggers ask themselves frequently.

It is impossible to completely automate debugging.

- Though automated debuggers may become better than humans.

Gödel Encoding (page 31 of the notes)

Every program is made up of a sequence of symbols.

You can encode a program as a single natural number by:

1. Give each possible symbol a numeric value
2. Replace each symbol in the program with its numeric value
3. Raise each prime `P_n` to the power `x_n` where `x_n` is the numeric value of the nth symbol in the program
4. Sum the numbers

This gives every possible program a unique number. Taking the prime decomposition of the number gives the program it represents.

The Nutshell Approach

- Note: we're skipping the more detailed proof

Represent all programs as numbers using Gödel encoding.

- Assume you have a program H which solves the halting problem
- Create program G which uses H, and does the opposite:
  - If H says program P_e terminates, then G(e) loops forever
  - If H says program P_e doesn't terminate, then G(e) terminates
- G must be in the list of programs H can predict
- We can feed G to H

Contradiction

There are two cases:

1. [...]
2. [...]

[...]

For Studying

- Look at the solved exercises first, rather than the halting problem solution