Introduction to Algorithms and Data Structures

Lecture 30: Unsolvable Problems

John Longley

School of Informatics
University of Edinburgh

25 March 2025

The plan ...

Last time:

- Introduced register machines (RMs), due to Marvin Minsky.
- Used them to define the Church-Turing computable functions.
- Reviewed arguments for the Church-Turing thesis.
- Sketched construction of a universal machine, using the crucial idea of coding a machine as a natural number.

This time:

- Give an example of a function that's not CT-computable, based on the halting problem.
- Mention other algorithmically unsolvable problems in maths/CS.
- Hint at some philosophical teasers.

What is a 'function'? A culture divide ...

1. For 18th-century mathematicians (e.g. Euler), a function was something given by a formula. E.g. $f(n) = n! + n^2 + 5$.

2. In 19th century (Dirichlet onwards), a function was conceived as an abstract object: a sort of infinite lookup table, not necessarily describable by any finite formula or 'rule' at all.

n	0	1	2	3	4	.....
f(n)	42	578	2	10225464	999	.....

   This is what today's mathematicians typically have in mind.

   [Teaser: In what sense do all these functions 'exist'?]

3. For many CS-educated people today, the word 'function' naturally refers to an entity defined by a piece of code. E.g.

```
def factorial(n):
    if n==0: return 1
    else: return n * factorial(n-1)
```

   Understood in this way, the concept of 'non-computable function' makes little sense! We're thinking of 'functions' more in sense 2.

The halting problem

Say a RM computation halts if we eventually emerge at an exit.

Recall that both RM flowcharts and memory states can be 'coded' as natural numbers (m, n).

A halting tester, if it existed, would take numbers m, n, and tell us whether 'machine m halts when run on initial memory state n'.

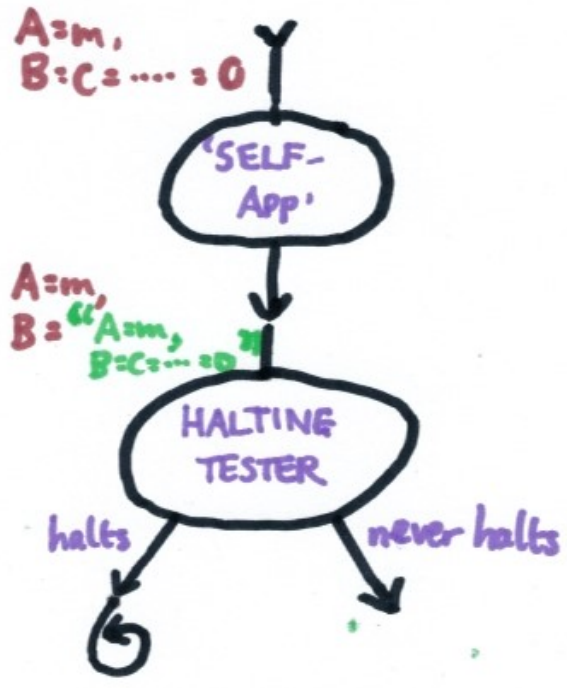
Theorem: There is no such register machine! In other words, the halting problem is (RM-)unsolvable.

Equivalently, the following function h isn't RM-computable:

$$h(m, n) = (0 \text{ if machine } m \text{ halts on input } n, 1 \text{ otherwise})$$

Unsolvability of the halting problem

Suppose a halting tester existed. Build the following machine P:



(NB. Running machine m on $\mathrm{A} = m$, $\mathrm{B} = \cdots = 0$ 'applies m to itself'.)

Let p be the numerical code for P.

Now run P on $\mathrm{A} = p$, $\mathrm{B} = \mathrm{C} = \cdots = 0$. What happens?

- We're testing if " P run on '$\mathrm{A} = p$, $\mathrm{B} = \mathrm{C} = \cdots = 0$' " halts!
- If answer is yes, we loop; if no, we halt. Contradiction!

Precursor: Russell's paradox (1901)

Define R to be the set of all sets that don't contain themselves:

$$R = \{S \mid S \notin S\}$$

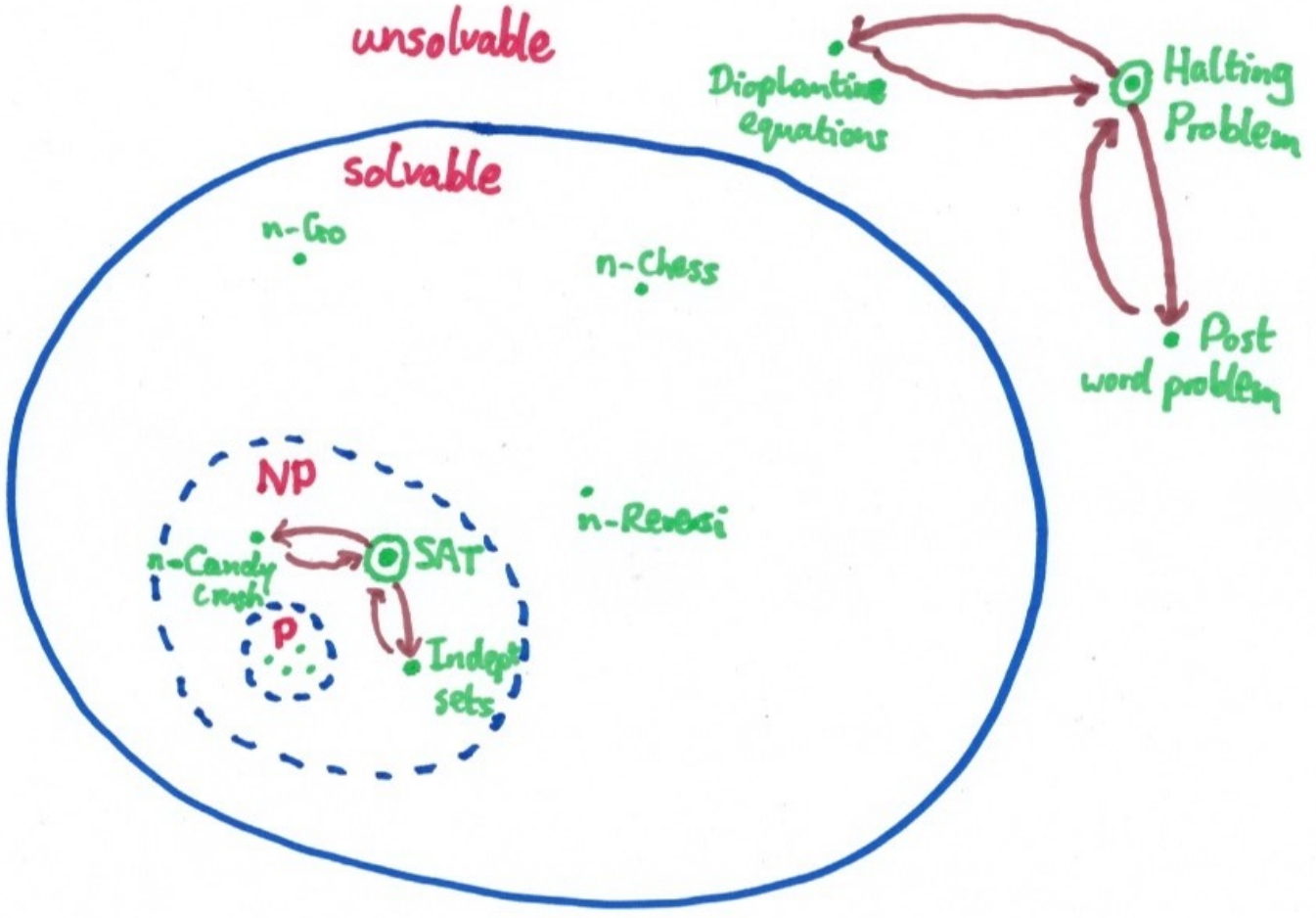
Does R contain itself, i.e. is $R \in R$?

Russell's analogy: The village barber (a man) shaves exactly those men in the village who don't shave themselves. Does the barber shave himself, or not?

Conclusion: No man exists in the village with the property proposed by Russell. (Not really a 'paradox' in this case.)

Recommended reading: *Scooping the Loop Snooper* by Geoffrey Pullum. (A proof that the Halting Problem is undecidable, written in verse in the style of Dr. Seuss).

The world of (un)solvability



- Dotted lines are conjectured boundaries.
- 'n-G' refers to the problem of choosing optimal moves in an 'n-ary generalization' of the game G.
- Halting problem is to unsolvable problems what SAT is to NP-hard ones.

Diophantine equations

Consider two equations. Want integer solutions ($x, y, z \in \mathbb{Z}$).

1. $x^2 + y^2 + z^2 = 42$.
   Clearly requires $|x|, |y|, |z| < 7$. So bounded search suffices.
   $5^2 + 4^2 + 1^2 = 42$.
2. $x^3 + y^3 + z^3 = 42$.
   Cubes may be $+$ or $-$. Seemingly requires unbounded search.
   Solved in 2019!
   $(-80538738812075974)^3 + 80435758145817515^3 + 12602123297335631^3 = 42$

More generally, a Diophantine equation is a multi-variable polynomial equation with integer coefficients for which we require integer solutions. E.g.

$$x^2y + 2yz^3 - 506zvw + w - v = 54321$$

Natural problem: Given such an equation, does it have a solution?

This turns out to be unsolvable: there's no program that takes a Dio eq as input, and gives a yes/no answer to this question.

Another example: Post's word problem

Given two finite sets S, T of strings, decide whether or not there's a string that can be formed both as a concatenation of strings in S and as a concatenation of strings in T.

E.g. suppose

$$S = \{a, ab, bba\}, \qquad T = \{baa, aa, bb\}$$

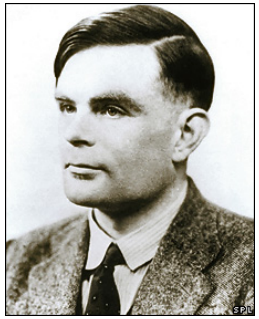
Then the answer is YES, because:

$$bba.ab.bba.a = bbaabbbaa = bb.aa.bb.baa$$

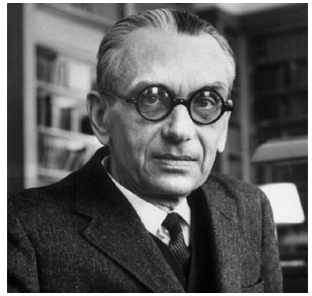
For general S and T, however, this is an unsolvable problem!

There are also examples in many other branches of CS/maths. E.g. given two context-free grammars G_1, G_2, it's an unsolvable problem whether $\mathbf{L}(G_1) \cap \mathbf{L}(G_2)$ is context-free.

Some big questions . . .



Alan Turing



Kurt Gödel

- Turing's theorem tells us that for any given machine M, there will be (yes/no) mathematical questions that M can't answer.
- Turing also thought that any aspect of human intelligence could in principle be imitated by a machine.
- If so, does that mean there are yes/no mathematical questions that *we humans* will never be able to answer? We seem to be led to . . .

Gödel's dichotomy:

> *"Either . . . the human mind (even within the realm of pure mathematics) infinitely surpasses the powers of any finite machine, or else there exist absolutely unsolvable diophantine problems."*
> *(Gödel 1952, emphasis mine)*

Unknowable truths?

- Gödel himself showed that any (suitable, consistent) formal system for math proofs will inevitably leave some questions unresolved.
- But *can* the power of the 'human proof engine' can be adequately encapsulated by some fixed formal system? If so, which system?
- Gödel himself believed deeply in the unlimited power of human reason, and thought that no mathematical questions were 'absolutely unsolvable'.

More questions . . .

- If some mathematical question were 'absolutely unsolvable', would we still say that question has a definite answer? If so, why?
- If not, which kinds of math statements have a definite 'meaning', and which don't? How much is just 'human convention'?

For much more in this area, see my book *Castles in the Air*.