A.M. Turing

ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

Proceedings of the London Mathematical Society, Second Series, vol. 42, Nov. 12, 1936, pp. 230–265.

ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM. A CORRECTION

Proceedings of the London Mathematical Society, Second Series, vol. 43, May 20, 1937, pp. 544–546.

Map

Not what it seems...

Ostensibly about "computable numbers" (which are interesting)

BUT more important:

- Turing machines
- Universal Turing machines
- (non-)computability
- (Turing) computability as an alternative to Church's "effective calculability"

A decade before any electronic computers had been built...

Difficulties

- Relatively difficult material
- First time ever these concepts presented
- Errors in 1936 paper—though of a technical nature
- Version in Lecture Notes is a later reprint:
  - Turing made extensive use of German ("Old English") letters
  - ɘ was originally ə
  - * missing target q for o/0 in §3.II
  - extra spacing in "working" of §3.II machine
  - page cross-references wrong

Entscheidungsproblem

Hilbert (1928):

Could proof be mechanized?

Is there any definite mechanical method or process by which all mathematical questions could be decided?

Before Gödel's Proof (1932), so truth and provability not distinguished.

Turing's Introduction

Stakes out claim:

- "...a number is computable if its decimal can be written down by a machine"
- "Although the class of computable numbers is so great, ... it is none the less enumerable"
- "Entscheidungsproblem can have no solution"
- "equivalence between [Turing's] 'computability' and [Church's] 'effective calculability"

1. Computing machines. & 2. Definitions.

Outlines idea of Turing machine. Note:

Binary fraction, $\mathbf{0}$, $\mathbf{1}$, plus other bookkeeping symbols

m-configuration	state
configuration	?
complete configuration	instantaneous description (configuration)
circular versus circle-free	analogous to halting

3. Examples of computing machines.

§3.I **010101...**

§3.II **001011011101111011111...**

Notational conveniences: multiple operations

Understand how they work

F-squares, E-squares, *marking*

4. Abbreviated tables.

Skeleton tables

Essentially a macro facility

Various utilities defined: "find", "erase", "print at end", "compare",. . .

5. Enumeration of computable sequences.

Reduce Turing-machine table to simple form

Standard form

Standard description

Standard number

"A number which a description number of a circle-free machine will be called a *satisfactory* number."

Undecidable

6. The universal computing machine.

Sketch of strategy for constructing a universal machine, via $\mathcal{M}'$

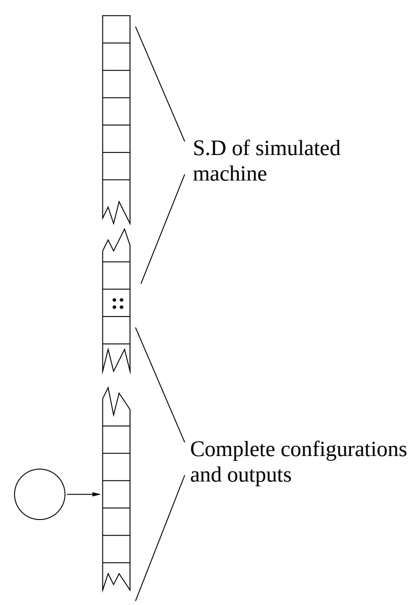
For a given machine $\mathcal{M}$, $\mathcal{M}'$ appends colon-separated successive complete configurations of $\mathcal{M}$ to the end of its tape.

$\mathcal{M}'$ does this by encoding rules of $\mathcal{M}$ internally

Hack for output **0**s and **1**s

Leap for universal machine is to encode machine description on front of tape, and interpret it

7. Detailed description of the universal machine.



8. Application of the diagonal process.

Demolishes a plausible but fallacious argument that computable numbers are not enumerable

Shows there can be no machine $\mathcal{D}$ that determines whether a given machine is circle-free

By constructing $\mathcal{H}$, which computes $\beta\prime$, whose n-th figure is $\phi_n(n)$, where $\phi_n(m)$ is the m-th figure in a_n, the n-th computable sequence

Diagonalization, leading to a contradiction

Corollary: There can be no machine $\mathcal{E}$ which will determine whether a given machine $\mathcal{M}$ ever prints a $\mathbf{0}$

9. The extent of the computable numbers.

(a) Intuition

(b) Can enumerate all provable formulae of Hilbert's functional calculus

(c) Examples

10. Examples of large classes of numbers which are computable.

Various examples of computable numbers, e.g.

- recursively defined function of intergral variables
- Dedekind-like cut
- convergence results: e, π, etc.

11. Application to the Entscheidungsproblem.

For each machine $\mathcal{M}$ construct a formula Un($\mathcal{M}$))

Show that:

If there is a general method for determining whether Un($\mathcal{M}$)) is provable,

then there is a general method for determining whether $\mathcal{M}$ ever prints $\mathbf{0}$

APPENDIX: Computability and effective calculability

Proof of the equivalence of Turing's computability and Church's effective calculability:

By showing that for every sequence computable by Church's λ-calculus, a corresponding Turing machine can be constructed, and vice-versa.