CDI

COMPUTABILITY
DECIDABILITY
INCOMPLETENESS

VIII

Eighth Lecture 6 June 2023

DECISION PROBLEMS

A **PROBLEM** is just a subset A of $\mathbb{N}^k$, interpreted as

"Given $\vec{w} \in \mathbb{N}^k$, can you decide whether $\vec{w} \in A$?"

From Lecture I:

"man soll ein Verfahren angeben"

Verfahren $\longrightarrow$ Algorithm

HILBERT'S PROBLEMS

A positive answer to H10 does not require to define "procedure".

A negative answer needs a definition of "VERFAHREN".

INTERPRETED as "algorithm".

David HILBERT
1862–1943

H10

D. Hilbert, Mathematische Probleme, Kgl. Ges. Wiss. Göttingen 1900

10. Entscheidung der Lösbarkeit einer Diophantischen Gleichung.

Eine Diophantische Gleichung mit irgend welchen Unbekannten und mit ganzen rationalen Zahlencoefficienten sei vorgelegt: *man soll ein Verfahren angeben, nach welchem sich mittelst einer endlichen Anzahl von Operationen entscheiden läßt, ob die Gleichung in ganzen rationalen Zahlen lösbar ist.*

From Lecture I:

Das ENTSCHEIDUNGSPROBLEM

Hilbert-Ackermann 1928

§ 12. Das Entscheidungsproblem.

Aus den Überlegungen des vorigen Paragraphen ergibt sich die grundsätzliche Wichtigkeit des Problems, bei einer vorgelegten Formel des Prädikatenkalküls zu erkennen, ob es sich um eine identische Formel handelt oder nicht. Nach der in § 5 gegebenen Definition bedeutet die Identität einer Formel dasselbe wie die Allgemeingültigkeit der Formel für jeden Individuenbereich. Man pflegt deswegen auch von dem *Problem der Allgemeingültigkeit* einer Formel zu sprechen. Genauer müßte man statt Allgemeingültigkeit Allgemeingültigkeit für jeden Individuenbereich sagen. Die identischen Formeln des Prädikatenkalküls sind nach den Ausführungen des § 10 gerade die Formeln, die aus dem Axiomensystem des § 1 sich ableiten lassen. Zu einer Lösung des Problems der Allgemeingültigkeit vermag uns diese Tatsache nicht zu helfen, da wir kein allgemeines Kriterium für die Ableitbarkeit einer Formel haben.

Q: Given a formula φ in predicate logic, is there an algorithm that determines whether φ is valid?

CONTRAST: For propositional logic, the method of truth tables produces an algorithm that checks validity.

Reminder: Distinction between EXISTENCE PROOFS and CONSTRUCTIVE PROOFS:

Subroutine Lemma

If f and g are computable, then $f \circ g$ is computable.

VERSUS

Used in the proofs of Software Principle & s-m-n like this:

There is an algorithm that transforms machines M, N s.t. $f = f_{M,1}$ & $g = f_{N,1}$ to a machine $\overline{M}$ s.t. $f \circ g = f_{\overline{M},1}$.

Analysing this:

We observe that we interpreted "there is an algorithm that transforms $\vec{w}$ into $\vec{v}$"

as

$$\boxed{\vec{w} \longmapsto \vec{v} \text{ can be performed by a RM}}$$

So, in our application, we identified "algorithm" with RM.

Q: Is that a reasonable interpretation in general?

Alan TURING
1912–1954

Alonzo CHURCH
1903–1995

Turing aimed to formalise the notion of COMPUTATION and transform it into a mathematical model of computation;
Church constructed a class of functions mathematically corresponding to usual computable processes.

<u>It turns out</u>: Any function f is computable iff it is recursive.

[We only proved $\Leftarrow$.]

<u>ROBUSTNESS</u> This is a nonmathematical, informal notion that a concept is <u>invariant</u> under small changes to the definition.

Remark As it turns out, it is the notion of computability rather than the notion of computation that is robust.

ILLUSTRATION Turing machines

Turing's 1936 paper defined computation in terms of Turing machines rather than RM.

A Turing machine has an infinite tape, indexed with $\mathbb{N}$, a read/write head that moves on the tape, each "cell" of the tape may contain $\sigma \in \Sigma$, the head reads it and acts according to what it sees.

Q : set of states

$\Sigma^{\mathbb{N}}$: "configuration" on the tape

$$\delta : Q \times \Sigma \longrightarrow Q \times \Sigma \times M$$

transition function

motion instruction either LEFT, RIGHT, STAY.

SNAPSHOT : $\underset{\text{current state}}{Q} \times \Sigma^{\mathbb{N}} \times \underset{\text{current head position}}{\mathbb{N}}$

As before, given a TM [Turing machine] and a configuration ["input"], we can define by recursion an (infinite) computation sequence of snapshots.

In precisely the same way as for RM, we can now define when a computation is halting and define a partial fn

$$f_M : \mathbb{N} \dashrightarrow \mathbb{N}$$

defined by the halting computations of M.

Def. A function $f : \mathbb{N} \dashrightarrow \mathbb{N}$ is partial TM computable if there is a TM M s.t. $f = f_M$.

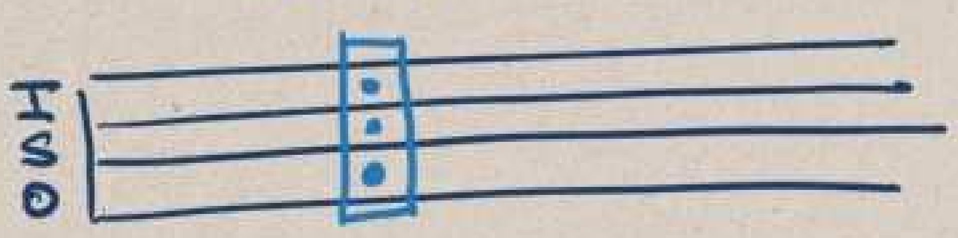
Theorem (no proof) For every $f : \mathbb{N} \dashrightarrow \mathbb{N}$, f is computable iff f is TM computable.

Variants of TM

(1) Number of tapes.

Have three tapes:

INPUT TAPE (read only)
SCRATCH TAPE (read/write)
OUTPUT TAPE (write only)



(1a) Or maybe output tape also read/write.

(1b) Or may three independent heads that move on the three tapes.

(2) Behaviour at cell 0.

If the head is on cell 0 and receives the motion instruction LEFT, what happens?

Options:

a Machine explodes
→ halting?
→ undefined?

b LEFT is interpreted as STAY.

NOTE The notion of COMPUTATION heavily depends on the choice of variant of TM.

Therefore: The notion of COMPUTATION (even the notion of TM COMPUTATION) is not robust in the above sense.

Theorem (no proof).
If I have two diff. variants of TM from the list on page 7, then their respective notions of TM computability are equivalent.

So: if M is a TM of variant 1 s.t.

$$f_M = f,$$

then there is some N TM of variant 2 s.t. $f_N = f$.

Thus: TM computability is robust.

This robustness goes much further: as observed,

TM computability

$\Longleftrightarrow$

RM computability

$\Longleftrightarrow$

recursiveness

and there are many other model of computation that yield eq. notions of COMPUTABILITY.

One more example:

WHILE computable functions

We design a very basic programming language that has the power of variable assignments and WHILE loops:

instruction

WHILE Φ DO .... ;

This can be used to define recursively a notion of WHILE COMPUTATION, as well as HALTING, and thus a notion of WHILE COMPUTABLE partial fn.

Theorem (no proof) $f: \mathbb{N} \dashrightarrow \mathbb{N}$ is WHILE computable $\Longleftrightarrow$ it is computable.

IMPORTANT This does not mean that RM computation, TM computation, and WHILE computation are the same; only the derived notion of computability.

Remark This is also relevant for the discussion about Quantum Computation. Quantum Computation is vastly different from classical computation, employing entanglement for parallel computing. But the corresponding notion of

Quantum Computability

is once more eq. to Turing's notion.

1937 Turing & Church observed (empirically) that computability is robust and formulated this in what is now known as the

Church-Turing thesis (= Church's thesis)

> Any reasonable attempt to formalise the intuitive notion of computation will lead to a notion of computability that is equivalent to TM / RM computability.

This is not a mathematical statement. It relies on "intuitive notion of computation" and "reasonable attempt", so it could be immune to counterexample and it is fundamentally unprovable.

It's more of an empirical modelling statement as is common in the natural science:

geometry of physical space is well represented by $\mathbb{R}^3$.

Therefore if we take the CT thesis on board, then the right interpretation of words such as

ALGORITHM

COMPUTATIONAL PROCEDURE

is "performed by a RM".

Definition $A \subseteq W^n$ is DECIDABLE if and only if A is a computable set.

[i.e., the question "$\vec{w} \in A$?" can be answered by a RM].

DECISION PROBLEM

1. Entscheidungsproblem.
   Formula φ and the problem is $\{\varphi ; \vdash \varphi\}$
2. Hilbert 10.
   Given $p \in \mathbb{Z}[X]$, is there $z \in \mathbb{Z}$ s.t. $p(z) = 0$.
   The problem is $\{p \in \mathbb{Z}[X] ; \exists z \; p(z) = 0\}$
3. Word problem:
   Given a language $L \subseteq W$ and $w \in W$, is $w \in L$?
   Normally, we are given a class of languages $\mathcal{C}$ parametrised by word
   $$\mathcal{C} = \{L_w ; w \in W\}.$$
   Then the problem is
   $$\{(w,v) ; w \in L_v\}$$
4. Emptiness problem: As before, $\mathcal{C} = \{L_w ; w \in W\}$ and $\{w ; L_w = \emptyset\}$
5. Equivalence problem As before, $\mathcal{C} = \{L_w ; w \in W\}$ and $\{(w,v) ; L_w = L_v\}$.

How are 1. & 2. <u>Decision Problems</u> ?

For 1., we need a representation of the formulas $\varphi \in \mathcal{L}$ by words $w \in \mathbb{W}$.
If we have this, we can w.l.o.g. assume $\mathcal{L} \subseteq \mathbb{W}$.

Then $\{\varphi ; \vdash \varphi\} \subseteq \mathbb{W}$, so it's a decision problem.

More about the ENTSCHEIDUNGS-PROBLEM in Lecture <u>IX</u>.

For 2., we need representations of $\mathbb{Z}$ and $\mathbb{Z}[X]$ in $\mathbb{W}$. Say w_p is the word representing $p \in \mathbb{Z}[X]$.

[Note that we encoded $\mathbb{N}$ in $\mathbb{W}$ by binary seq. Clearly elements of $\mathbb{Z}$ are just elements of $\mathbb{N}$ with a sign $\pm$; and elements of $\mathbb{Z}[X]$ are just finite seq. of elements of $\mathbb{Z}$.]

Then $\{w_p ; \exists z \in \mathbb{Z} \ p(z) = 0\}$ is the decision problem for H10.

Now talk about 3., 4., and 5.

They all have

$$\mathcal{C} = \{ L_w ; w \in \mathbb{W} \} ;$$

we need to decide which class $\mathcal{C}$ we use.
[This is really important for the answer, as we'll see in a moment.]

What about $\mathcal{C} = \{ A ; A \text{ is c.e.} \}$
$= \{ W_w ; w \in \mathbb{W} \}$?

For this choice of $\mathcal{C}$, we get

3. $\{ (w,v) ; w \in W_v \} = \{ (w,v) ; w \in \mathrm{dom}(f_{v,1}) \}$
$= \{ (w,v) ; f_{v,1}(w) \downarrow \}$
$= \mathbb{K}_0$.

So this is not computable, so not decidable

<u>In words</u>: The word problem for c.e. sets is not decidable.

4. $\{ w ; W_w = \emptyset \} = \mathbb{E}\mathrm{mp}$

This is not computable by Rice's Theorem.

<u>In words</u>: The emptiness problem for c.e. sets is not decidable.

5. Equivalence problem

$$\{(w,v);\ W_w = W_v\}. = \mathbb{E}q$$

We show that

$$\mathbb{E}mp \leq_m \mathbb{E}q.$$

Let e be s.t. $W_e = \emptyset$.

The function $h: w \longmapsto (w,e)$ can be performed by a register machine.

So if $\mathbb{E}q$ is computable, i.e., $\chi_{\mathbb{E}q}$ is a computable function, then so is

$$\chi_{\mathbb{E}q} \circ h = \chi_{\mathbb{E}mp},$$

so $\mathbb{E}mp$ is computable. Contradiction!

<u>Therefore</u>: The equivalence problem for c.e. sets is not decidable.