COMPUTABILITY, DECIDABILITY, INCOMPLETENESS

CDI #6 VI

Sixth Lecture 23 May 2023

Originally: index of f is a MACHINE, not a WORD (Cf. Remark on page 2)

Theorem 4.25 (The Software Principle)**.** There is a register machine U, called a *universal register machine* such that for every register machine M and sequence of words $\vec{w}$, we have that

$$f_{U,2}(v,u) = \begin{cases} f_{M,k}(\vec{w}) & \text{if } v = \text{code}(M) \text{ for a register machine } M \\ & \quad \text{and } u = \text{code}(\vec{w}) \text{ for a sequence of words of length } k, \\ \uparrow & \text{otherwise.} \end{cases}$$

Theorem 4.26 (The *s-m-n* Theorem)**.** Let $g : \mathbb{W}^{k+1} \dashrightarrow \mathbb{W}$ be any partial computable function. Then there is a total computable function $h : \mathbb{W} \to \mathbb{W}$ such that for all $v \in \mathbb{W}$ and all $\vec{w} \in \mathbb{W}^k$, we have $f_{h(v),k}(\vec{w}) = g(\vec{w}, v)$.

FOR THE NOTATION, SEE REMARK ON PAGE 2.

The curious name of this theorem derives from the notation S_n^m used for the function h in the original publication.[16] The *s-m-n* Theorem pulls one of the parameters of the function g into the index. This process is also called *Currying*, after the logician Haskell Curry (1900–1982).[17]

CURRYING

Process

$g : \mathbb{W}^{k+1} \dashrightarrow \mathbb{W}$

leading to

$h : \mathbb{W} \to \mathbb{W}$

total with

$f_{h(v),k}(\vec{w}) = g(\vec{w}, v)$

IMPORTANT

In the proof of T 4.25, we used the SUBROUTINE lemma and the CASE DISTINCTION LEMMA. It was important that their proofs are

CONSTRUCTIVE

i.e, that there is a concrete construction that produces the new machines from the old machines.

The Software Principle allows us to identify machines with the words coding them, i.e., write

$$f_{w,k}(\vec{w}) := f_{U,2}(w, \vec{w})$$

INTENDED MEANING

$[f_{M,k}(\vec{w})$ where w codes $M.]$

COMPUTABLY ENUMERABLE SETS

$$K := \{ w ; f_{w,1}(w)\downarrow \} \subseteq W$$

$$K_0 := \{ (w,v) ; f_{w,1}(v)\downarrow \} \subseteq W^2.$$

Both of these are usually called the **Halting Problem** or **Turing's Halting Problem**

If we do not specify which one we mean, we usually mean K.

Prop. Both K and K_0 are c.e.

Proof. Suppose U is a universal machine.

Then $f_{U,2}(w,v) = f_{w,1}(v)$ ..

by the Software Principle.

Thus: $\mathrm{dom}(f_{U,2}) = K_0$.

So K_0 is c.e.

For K, consider the operation

$w \longmapsto (w,w)$

This can be performed by a RM

[Copy content of register 0 to register 1 and halt.]

Note Not a computable fn, since $W \longrightarrow W^2$.

Therefore $f : W \longmapsto f_{U,2}(w,w)$

is computable.

But: $\mathrm{dom}(f) = K$.

q.e.d.

Theorem (Turing)

Neither K nor K_0 is computable.

Proof. In this proof it's enough to assume that one of them is computable, but we write it as a simultaneous proof of both implications.

Suppose either χ_K or χ_{K_0} is computable.

Then

$$\varphi(w) := \begin{cases} \uparrow & \text{if } \chi_{K_0}(w,w) = 1 \\ a & \text{if } \chi_{K_0}(w,w) = 0 \text{ and } \chi_K(w) = a \\ \varepsilon & \text{if } \chi_{K_0}(w,w) = 0 \text{ and } \chi_K(w) = \varepsilon \end{cases}$$

Cannot be replaced by ψ.

Think about why this construction cannot work with pseudo-characteristic functions!

Since φ is computable, there is $d \in W$ s.t. $\varphi = \varphi_{d,1}$.

Ask whether $\varphi(d)$ halts!

$$\begin{aligned} \varphi(d)\downarrow &\iff \chi_{K_0}(d,d) = \varepsilon \iff \chi_K(d) = \varepsilon \\ &\iff (d,d) \notin K_0 \iff d \notin K \\ &\iff \varphi_{d,1}(d)\uparrow \iff \varphi_{d,1}(d)\uparrow \\ &\iff \varphi(d)\uparrow \end{aligned}$$

CONTRADICTION! q.e.d

REMARK

(1) This is the classic diagonalisation argument that we also see in

(1) Russell's paradox

(2) the uncountability of $\mathbb{R}$

(3) Cantor's Theorem

[$\forall X$ there is no surjection from X onto $\mathcal{P}(X)$].

(2) The use of the characteristic fn in the proof is connected to our earlier observation that the computable sets are closed under

COMPLEMENTATION.

The c.e. sets will turn out not to be closed under complementation.

Definition A set $X \subseteq \mathbb{N}^k$ is called

Σ_1 if there is a computable set $Y \subseteq \mathbb{N}^{k+1}$ s.t.

for all $\vec{w} \in \mathbb{N}^k$

$$\vec{w} \in X \iff \exists v \, (\vec{w}, v) \in Y.$$

We say X is Π_1 if its complement is Σ_1.

We call X Δ_1 if it is both Σ_1 and Π_1.

Terminological remark.

Logicians consider $\vee, \exists$ to be analogues of SUMS / ADDITION and $\wedge, \forall$ to be analogous of PRODUCTS. So Σ_1 stands for "definable with one sum operator $\exists$" and Π_1 stands for "definable with one product operator $\forall$".

Δ stands for "Durchschnitt", since $\{A; A \text{ is } \Delta_1\} = \{A; A \text{ is } \Sigma_1\} \cap \{A; A \text{ is } \Pi_1\}$

Proposition If A is computable, then A is Δ_1.

Proof. By closure of computable sets under complement, we only need to show that A is Σ_1.

Idea Trivialize the quantifier.

Fix A computable, define

$$Y := \{(\vec{w}, v) ;\ \vec{w} \in A\}$$

This is clearly computable and

$$\vec{w} \in A \iff \exists v\ (\vec{w}, v) \in Y.$$

q.e.d.

Theorem TFAE

(i) A is c.e.

(ii) A is Σ_1

Proof. (i) $\Rightarrow$ (ii).

Fix f s.t. $A = \mathrm{dom}(f)$. Remember the truncated computation of f

$$t(\vec{w}, v) := \begin{cases} a & \text{the } f(\vec{w}) \text{ computation has halted after } \#v \text{ steps} \\ \varepsilon & \text{o/w} \end{cases}$$

Note that t is a characteristic function, viz. of

$$Y := \{(\vec{w}, v);\ t(\vec{w}, v) = a\}$$

By lecture V, t and thus Y are computable.

Observe

$$\vec{w} \in A \iff \vec{w} \in \mathrm{dom}(f)$$
$$\iff \exists v\ (\vec{w}, v) \in Y.$$

(ii) $\Rightarrow$ (i). Let $Y \subseteq W^{k+1}$ be computable such that $\vec{w} \in A \iff \exists v\ (\vec{w}, v) \in Y$.

Let $c := \chi_Y$; thus is computable, so we can apply minimisation to c to have the function h computable where $h(\vec{w})$ is the least v s.t. $(\vec{w}, v) \in Y$ if it exists

partial

Claim $A = \mathrm{dom}(h)$.

[if $h(\vec{w})\downarrow \Rightarrow \exists v\ (\vec{w}, v) \in Y \Rightarrow \vec{w} \in A$.

if $h(\vec{w})\uparrow \Rightarrow \forall v\ (\vec{w}, v) \notin Y \Rightarrow \vec{w} \notin A$.]

q.e.d.

Corollary A is computable $\iff$ A is Δ_1.

Proof. We already proved $\Longleftarrow$.

By Theorem, we know that if A is both Σ_1 and Π_1, we have Y, Z computable s.t.

$$\vec{w} \in A \longleftrightarrow \exists v\ (\vec{w}, v) \in Y \quad \chi_Y(\vec{w}, v) = a$$

$$\vec{w} \notin A \longleftrightarrow \exists v\ (\vec{w}, v) \in Z \quad \chi_Z(\vec{w}, v) = a$$

Define function f

$$f(\vec{w}, v) := \begin{cases} \chi_Y(\vec{w}, v_{(1)}) & \text{if } \#v_{(0)} \text{ is even} \\ \chi_Z(\vec{w}, v_{(1)}) & \text{if } \#v_{(0)} \text{ is odd} \end{cases}$$

Apply minimisation to f and have $h(\vec{w})$ be the least v s.t. $f(\vec{w}, v) = a$ if it exists.

Proof idea

PARALLELISING THE COMPUTATIONS OF χ_Y AND χ_Z.

Reminder Have function splitting a word into two and merging them into one.

$$v \longmapsto v_{(0)}$$
$$v \longmapsto v_{(1)}$$

s.t.

$$v_{(0)} * v_{(1)} = v.$$

Because each $\vec{w}$ is either in A or not, there is some v s.t. either

$$\chi_X(\vec{w}, v) = a \quad \text{or}$$

$$\chi_Z(\vec{w}, v) = a$$

Therefore k is a <u>total</u> computable function.

Here is the algorithm that computes χ_A:

(1) Take $\vec{w}$ and calculate $k(\vec{w})$.

(2) Check whether $(\vec{w}, k(\vec{w})) \in Y$ or in Z [it has to be in exactly one of the two].

(3) If it's in Y, output a.

(4) If it's in Z, output ε.

q.e.d.

<u>Corollary</u> $\Sigma_1 \neq \Pi_1$.

[K is in $\Sigma_1 \setminus \Delta_1$, but if $\Sigma_1 = \Pi_1$, then $\Delta_1 = \Sigma_1 \cap \Pi_1 = \Sigma_1$.]

Remark Some people claim that this is just Gödel's Incompleteness Theorem, i.e., $\Sigma_1 \neq \Pi_1$.

We'll see in the rest of the course why this is correct.

CLOSURE PROPERTIES

If A, B are computable, then

$A \cap B$
$A \cup B$
$\mathbb{N} \setminus A$

} are computable.

E.g. $\chi_{A \cap B}(\vec{n}) = \begin{cases} a & \text{if } \chi_A(\vec{n}) = a = \chi_B(\vec{n}) \\ \varepsilon & \text{o/w} \end{cases}$

$\chi_{A \cup B}(\vec{n}) = \begin{cases} a & \text{if } \chi_A(\vec{n}) = a \text{ or } \chi_B(\vec{n}) = b \\ \varepsilon & \text{o/w} \end{cases}$

This makes use of the fact that characteristic functions are total, and so their computations produce an answer after finitely many steps, no matter what the input is — so, after determining everything that matters, apply rules of propositional logic.

If A, B are c.e., then

$A \cap B$ is c.e.

$$\psi_{A\cap B}(\vec{w}) = \begin{cases} a & \text{if } \psi_A(\vec{w}) = a = \psi_B(\vec{w}) \\ \uparrow & \text{o/w} \end{cases}$$

This proof idea does not work for $A \cup B$.

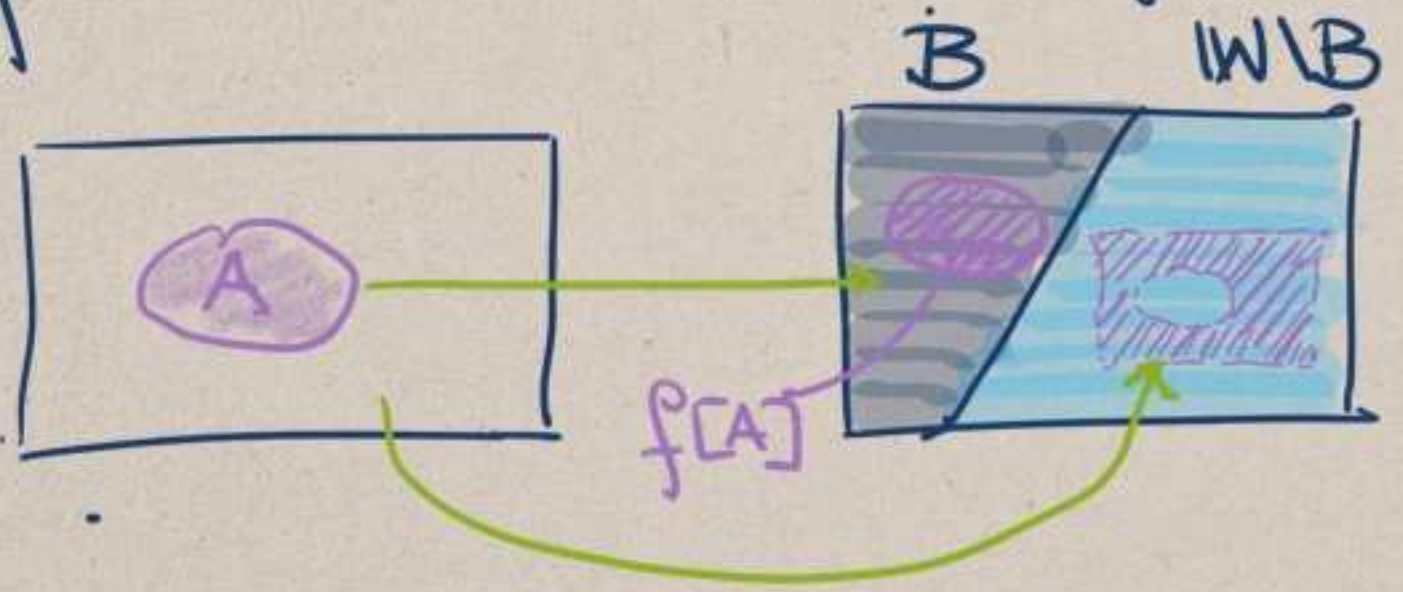
However, $A \cup B$ is c.e.

[Homework : HINT : Use the idea of the proof of $\Delta_1 \longrightarrow$ computable.]

Def. A function $f : W \longrightarrow W$ is called total

REDUCTION from A to B. if

for all w $\quad w \in A \iff f(w) \in B$.



Def. $A \leq_m B$ if there is a reduction f from A to B.

[The "m" stands for "many-one" reminding us that there is no requirement that f is injective.]

Prop. If $A \leq_m B$ and B is computable or c.e., then so is A.

Proof. Suppose f is the reduction.

Check that

$$\chi_A = \chi_B \circ f$$

$$\psi_A = \psi_B \circ f. \qquad \text{q.e.d.}$$

Note This justifies the word "reduction" since B is at least as complicated as A.

Note 2 There can be no computable A s.t. $\mathbb{K} \leq_m A$.

Note 3 This is a technique to show that a set is not computable: show that $\mathbb{K} \leq_m A$.

Def. Let $\mathcal{C}$ be any collection of subsets of $\mathbb{N}$. [E.g., $\Sigma_1, \Pi_1, \Delta_1$, c.e., computable ...]

We say A is $\mathcal{C}$-hard if for all $X \in \mathcal{C}$ $X \leq_m A$.

We say A is $\mathcal{C}$-complete if it is $\mathcal{C}$-hard & $A \in \mathcal{C}$.

Goal (Lecture VII): Prove that K is Σ_1-complete.