III

THIRD LECTURE

Computability, Decidability, Incompleteness CDI

18 APRIL 2023

Recap Register machines
→ computation sequence
→ halting, halting times, output

Fix an upper register index n and let $F : \mathbb{W}^{n+1} \dashrightarrow \mathbb{W}^{n+1}$ be any partial function. We say that a register machine *M performs the operation F* if upon input $\vec{w} \in \mathbb{W}^{n+1}$, if $F(\vec{w})\uparrow$, then M diverges on input $\vec{w}$ and if $F(\vec{w})\downarrow$, then M converges on input $\vec{w}$ with register content $F(\vec{w})$ at the time of halting. If F is a total function, we sometimes emphasise this by using the phrase "M performs the total operation F".

A *question with* $k+1$ *answers* is a partition of $\mathbb{W}^{n+1}$ into $k+1$ disjoint sets $A_0, ..., A_k$. E.g., the question "does the second register end with a?" is the partition $A_0 := \{\vec{w} ; \exists v(w_2 = va)\}$ and $A_1 := \mathbb{W}^{n+1} \backslash A_0$. A register machine *$M$ answers a question with $k+1$ answers* if it has $k+1$ designated *answer states* $\widehat{q}_0, ..., \widehat{q}_k$ and for each $\vec{w}$, the computation of M with input $\vec{w}$ produces in finitely many steps a configuration $(\widehat{q}_i, \vec{w})$ if and only if $\vec{w} \in A_i$.

Examples for performing operations

Example Operation NEVER HALT
represented by $F: \mathbb{W}^{n+1} \dashrightarrow \mathbb{W}^{n+1}$ s.t. $\mathrm{dom}(F) = \emptyset$.

E.g., $q_S \longmapsto +(0, a, q_S)$ produces an infinite loop.

From Lecture II

Example 2 ALWAYS HALT, DO NOT CHANGE INPUT

Represented by $F = \mathrm{id}: \mathbb{W}^{n+1} \dashrightarrow \mathbb{W}^{n+1}$ (total)

$q_S \longmapsto ?(0, \varepsilon, q_H, q_H)$

Examples for answering questions

Example 1 Is register i empty?

Possible answers: Yes / No

$A_0 := \{\vec{w} ; w_i = \varepsilon\}$, YES

$A_1 := \{\vec{w} ; w_i \neq \varepsilon\}$ NO

$q_s \longmapsto ?(i, \varepsilon, \hat{q}_0, \hat{q}_1)$

Example 2 Does register i end with letter a?

$A_0 := \{\vec{w} ; \exists w (w_i = wa)\}$

$A_1 := W^{n+1} \setminus A_0$.

$q_s \longmapsto ?(i, a, \hat{q}_0, \hat{q}_1)$.

Proposition (The concatenation lemma)

The subroutine lemma

If M performs F & M' performs F', then there is a RM performing $F' \circ F$.

Lemma 4.7 (Case Distinction Lemma). Let $Q = \{A_i ; i \leq k\}$ be a question with $k+1$ answers and $f_i : \mathbb{W}^{n+1} \dashrightarrow \mathbb{W}^{n+1}$ be operations for $i \leq k$. If Q is answered by a register machine $M = (\Sigma, Q, P)$ and f_i is performed by $M_i := (\Sigma, Q_i, P_i)$ (for $i \leq k$), then we can construct a register machine that performs the operation defined by $g(\vec{w}) := f_i(\vec{w})$ if and only of $\vec{w} \in A_i$.

Many more examples

1. DELETE THE FINAL LETTER IN REG. i (IF IT EXISTS).

$$q_S \longmapsto -(i, q_H, q_H)$$

2. DELETE THE CONTENT OF REG i ("empty reg. i")

$$q_S \longmapsto -(i, q_H, q_S).$$

3. ADD a TO THE END OF REG i

$$q_S \longmapsto +(i, a, q_H)$$

Remark: This also describes MAKE SURE THAT REG. i IS NON-EMPTY.

4. ADD $w \in W$ TO THE END OF REG. i

If $w = (a_0, \dots, a_k)$, then perform the operation "add a_j to the end of reg. i" by the subroutine lemma.

5. REPLACE REG. i WITH $w \in W$

First empty i (with 2.); then add $w \in W$ to the end of i (with 4.).

6. What is the final letter in reg. i (if any)?

Question with $|\Sigma|+1$ answers.

$$\Sigma = \{a_0, \dots, a_k\}$$

$\longrightarrow$ $k+2$ answers

$$A_\ell := \{\vec{w} ; \exists v (w_i = va_\ell)\} \quad \ell \leq k$$

$$A_{k+1} := \{\vec{w} ; w_i = \varepsilon\}.$$

Cascade $k+1$ many questions (Example from Lecture II)

Is the final letter of reg. i a_0?

If so $\longrightarrow$ $\hat{q}_0$

If not:

Is the final letter of reg. i a_1?

If so $\longrightarrow$ $\hat{q}_1$.

If not

⋮

Is the final letter of reg. i a_k?

If so $\longrightarrow$ $\hat{q}_k$

If not $\longrightarrow$ $\widehat{q_{k+1}}$.

7. Copy final letter of reg. i (if it exists) to reg. j.

First answer the question "what's the final letter of reg. i", get answer state $\hat{q}_\ell$.
Then add a_ℓ to reg. j or (if $\ell = k+1$) halt.

8. Move final letter of reg. i (if it exists) to reg. j.

Copy (as in 7.);
then remove final letter (as in 1.).

9. Move content of reg. i to reg. j in reverse order.

Repeat Example 8. until the register is empty.

10. Move content of reg. i to reg. j.

Take a register k UNUSED the the machine so far. Empty reg. k.
Move content of reg. i to k in reverse order (Ex. 9)
Move content of reg. k to j in reverse order.

11. Copy content of reg. i. to reg. j in reverse order.

Take unused reg. k; empty it.

Perform opers.

"copy final letter of reg i to reg j (7.)

& move final letter of reg. i to reg k"

until reg. i is empty.

After that move reg. content of k to i in reverse order.

12. Copy reg. content of i to j.

Take unused reg. k; empty it.

Copy i to k in reverse order. (11.)

Move k to j in reverse order. (9)

13. Is the reg. content of reg. i exactly $w \in W$?

If $w = (a_0, \dots, a_k) \in W$, checks from the back: Check for $l = k, k-1, \dots, 0$ whether

the final letter of reg i is a_l;

if not, restore reg. i from intermediate reg and answer No;

if yes, move a_l to intermediate reg. and continue.

if reg. i empty, restore i from intermediate reg. and YES.

C.3 Computable functions & sets

If M is a register machine, I define a partial fn

$$f_{M,k} : \mathbb{N}^k \dashrightarrow \mathbb{N} \text{ by}$$

$f_{M,k}(\vec{w}) \uparrow$ if M does not halt with input $\vec{w}$

$f_{M,k}(\vec{w}) = v$ if M halts with input $\vec{w}$ and the reg. content at halting time is $\vec{v}$ with $v_0 = v$.

$$\mathrm{dom}(f_{M,k}) = \{\vec{w} \,;\, M \text{ halts with input } \vec{w}\}$$

If $k = 1$, we write

$$W_M := \mathrm{dom}(f_{M,1})$$

<u>Def.</u> Two machines M, M' are called **weakly equivalent** if $W_M = W_{M'}$.

Clearly, if M, M' are strongly eq., they are weakly equivalent.

Remark We could also define a number of intermediate equivalence notion such as

M, M' $*$-equivalent if

$$f_{M,1} = f_{M',1}$$

M, M' $\circledast$-equivalent if

$$\forall k \ f_{M,k} = f_{M',k}.$$

For all of these, the analogue of the padding lemma holds: infinitely many eq. machines.

Definition A partial function

$$f : \mathbb{N}^k \dashrightarrow \mathbb{N}$$

is **computable** if there is a machine M s.t.

$$f = f_{M,k}.$$

REMARKS

1. The padding lemma implies that M is not unique.
2. The countability lemma from Lecture II implies that there are only countably many computable functions.

Note that this implies (in a very unsatisfactory way) our fast

LIMITATIVE THEOREM:

There are non-computable functions.

Examples (1.) $id : W \longmapsto W$

Halt w/o changing anything $\longrightarrow M$

then $f_{M,1} = id$.

(2.) Constant functions

$$c_{k,v} : \mathbb{W}^k \longrightarrow \mathbb{W}$$
$$\vec{w} \longmapsto v$$

Replace reg. 0 by v $\longrightarrow M$

$$f_{M,k} = c_{k,v}.$$

(3.) Projection function

$$\pi_{k;i} : \mathbb{W}^k \longrightarrow \mathbb{W}$$
$$\vec{w} \longmapsto w_i$$

[Delete reg. 0;
copy/move content of reg. i to reg. 0
halt.]

If $A \subseteq \mathbb{W}^k$, we call

Let's fix some $a \in \Sigma$.

$$\chi_A : \mathbb{W}^k \longrightarrow \mathbb{W}$$
$$\vec{w} \longmapsto \begin{cases} a & \text{if } \vec{w} \in A \\ \varepsilon & \text{if } \vec{w} \notin A \end{cases}$$

the characteristic function of A.

We call

$$\psi_A : \mathbb{W}^k \dashrightarrow \mathbb{W}$$
$$\vec{w} \longmapsto \begin{cases} a & \text{if } \vec{w} \in A \\ \uparrow & \text{if } \vec{w} \notin A \end{cases}.$$

the pseudocharacteristic function of A.

__Definition__ We call a set $A \subseteq \mathbb{W}^k$

__computable__ if χ_A is computable &

__computably enumerable__ (c.e.) if ψ_A is computable.

Proposition $A \subseteq \mathbb{W}^k$

1. If A is computable, then so is $\mathbb{W}^k \setminus A$.

2. A is c.e. $\Leftrightarrow$ there is some M s.t.
$$A = \mathrm{dom}(f_{M,k})$$

3. If A is computable, then A is c.e.

Proof. 1. Prove that
$$f(w) := \begin{cases} a & \text{if } w = \varepsilon \\ \varepsilon & \text{if } w \neq \varepsilon \end{cases}$$
is computable.

[Check whether reg. 0 is empty; if so, add a & halt; if not, empty it & halt.]

Then: $\chi_{\mathbb{W}^k \setminus A} = f \circ \chi_A$

So by the subroutine lemma, $\mathbb{W}^k \setminus A$ is computable.

(2.) A c.e. $\iff \exists M \; A = \mathrm{dom}(f_{M,k})$

"$\Rightarrow$" A c.e. $\Rightarrow \psi_A$ computable,
but $\mathrm{dom}(\psi_A) = A$, so done.

"$\Leftarrow$" We already saw that $c_{1,a}$
[constant fn with value a]
is computable, so
if $A = \mathrm{dom}(f)$, then

$$\psi_A = c_{1,a} \circ f.$$

Again, use subroutine lemma.

(3.) If χ_A is comp. $\Rightarrow \psi_A$ is comp.

So, need to find $g : \mathbb{W} \dashrightarrow \mathbb{W}$ s.t.

$g(\varepsilon)\uparrow$

$g(w) = w$ for all other words.

We have seen such a function at the end of Lecture II:

be explicitly given.

Example

$$g(\vec{w}) := F(\vec{w}) := \begin{cases} \vec{w} & w_i \neq \varepsilon \\ \uparrow & w_i = \varepsilon \end{cases}$$

This can be performed by a RM:

Check whether reg i is empty,
if so, halt without changing anything;
if not, don't halt.

IMPORTANT REMARK

While this looks like natural language, it is fully formalised, since CHECK WHETHER REG i IS EMPTY, HALT W/O CHANGING ANYTHING, & DO NOT HALT have formal definitions as RM.

$$\psi_A = g \circ \chi_A .$$

So, once more, the subroutine lemma yields the claim. q.e.d.