CHAPTER 5

What's a computer, anyway? (Cont'd)

3. Halt and catch fire (Cont'd)

3.2. Recursively Enumerable Sets. Just like we defined the notion of a primitive recursive set, we will say that $X \subseteq \mathbb{N}^n$ is a **recursive set** if its characteristic function is a (total) recursive function. More importantly, for now, we are interested in the following notion:

Definition 3.2.1. A set $X \subseteq \mathbb{N}^n$ is called *recursively enumerable* it is the domain of a partial recursive function.

Intuitively, $X \subseteq \mathbb{N}^n$ is recursive if there exists an algorithmic procedure which, for all $x \in \mathbb{N}$ allows us to decide if $x \in X$ or not. On the other hand, X is recursively enumerable if there is an algorithm which for all $x \in \mathbb{N}^n$ terminates if and only if $x \in X$. So given $x \in \mathbb{N}^n$ we can run our algorithm, but unless that algorithm terminates, we cannot know if $x \in X$ and we have no guarantee that our algorithm will terminate.

Our first big result about recursively enumerable sets will justify their name. Namely, we will prove that a non-empty $X \subseteq \mathbb{N}^n$ is recursively enumerable if and only if X is the range of a primitive recursive function, i.e. if there is an (easy) algorithm which eventually lists all elements of X.

The domain of the partial function whose index is x (i.e. of the partial function ϕ_x^p) will be denoted by W_x^p. Then, the set:

$$\mathcal{W} = \{W_x^p : x \in \mathbb{N}\}$$

is the set of *all* recursively enumerable sets. We overload our terminology to say that x is an **index** of $A \subseteq \mathbb{N}^p$ if $A = W_x^p$.

Lemma 3.2.2. *Every recursive set is recursively enumerable*

Proof. Let $X \subseteq \mathbb{N}^p$ be a recursive set. We have to show that there is some recursive (partial) function f such that $\mathsf{dom}(f) = X$.

By assumption, $\mathbb{1}_X$ is a recursive function, of course. To build our desired function f, consider the function:

$$g(x) = \mu y.(y + 1 \in \{x\}),$$

or, more simply put:

$$g(x) = \mu y.(y + 1 = x).$$

This is, of course, a recursive function, and $\mathsf{dom}(g) = \mathbb{N} \setminus \{0\}$. Now, the function:

$$f = g \circ \mathbb{1}_X$$

is the function we want, since:

$$f \text{ is defined at } x \iff \mathbb{1}_X(x) \neq 0 \iff x \in X. \qquad \square$$

The obvious question which we will eventually tackle is the following:

Is every recursively enumerable set recursive?

You can take a guess right now, but it may get more educated as we develop a somewhat deeper understanding of recursive and recursively enumerable sets.

As a warm-up, we have the following analogue of Lemma 2.2.6(3):

Lemma 3.2.3. *For all $p \in \mathbb{N}$, the set of recursive subsets of $\mathbb{N}^p$ is closed under Boolean combinations.*

PROOF. HW8 $\square$

The analogous result for recursively enumerable sets will eventually turn out to not be true, but we can at least prove the following:

Lemma 3.2.4. *For all $p \in \mathbb{N}$, the set $\mathcal{W}$ (which recall is the set of all recursively enumerable sets, i.e. $\{W_x^p : p \in \mathbb{N}\}$) is closed under unions and intersections.*

PROOF. Suppose we are given $W_x^p, W_y^p \in \mathcal{W}$ be the domains of partial functions with indices x and y, respectively. Consider the machine which computes $\phi_x^p + \phi_y^p$. Then, this machine halts on $n \in \mathbb{N}^p$ if and only if $n \in W_x^p \cap W_y^p$, so intersections are out of the way.

Unions are a slightly more delicate business. The intuition is the following, $\bar{n} \in W_x^p \cup W_y^p$ if one of the two computations $\phi_x^p(\bar{n})$ or $\phi_y^p(\bar{n})$ eventually halts. We thus take:

$$f(\bar{n}) := \mu t.[(t, n_1, \ldots, n_p) \in \mathsf{halted}^p(x) \cup \mathsf{halted}^p(y)].$$

Indeed, if $\bar{n} \in W_x^p \cup W_y^p$ then $f(\bar{n})$ is defined, and conversely, if. $f(\bar{n})$ is defined, then $\bar{n} \in \mathsf{halted}^{\mathsf{p}}(x)$ or $\bar{n} \in \mathsf{halted}^p(y)$ so one of $\phi_x^p(\bar{n})$ or $\phi_y^p(\bar{n})$ is defined, as required. □

The following theorem gives us what is probably one of the most important properties of recursively enumerable sets:

THEOREM 3.2.5 (Theorem of the Complement). *Let $A \subseteq \mathbb{N}^p$. Then, the following are equivalent:*

(1) A is recursive.

(2) Both A and $\mathbb{N}^p \setminus A$ are recursively enumerable.

PROOF. (1) $\implies$ (2) is immediate, since every recursive set is recursively enumerable, and recursive sets are closed under Boolean combinations. For (2) $\implies$ (1), suppose that both A and $\mathbb{N}^p \setminus A$ are recursively enumerable. To fix ideas, suppose that $A = W_x^p$ and $\mathbb{N}^p \setminus A = W_{x'}^p$. Then, the function:

$$h(n_1, \ldots, n_p) = \mu t.[(t, n_1, \ldots, n_p) \in \mathsf{halted}^p(x) \cup \mathsf{halted}^p(y)]$$

is a total recursive function. Observe that:

$$\bar{n} \in A \iff (h(\bar{n}), \bar{n}) \in \mathsf{halted}^p(x),$$

which means that:

$$\mathbb{1}_A(\bar{n}) = \mathbb{1}_{\mathsf{halted}^p(x)}(h(\bar{n}), \bar{n}).$$ □

Remark 3.2.6. Recursion theory has this way of making relatively easy statements sound very obscure. Here's what's happened in the proof above:

- If both A and its complement are recursively enumerable, then for every $\bar{n} \in \mathbb{N}^p$ either the machine that halts on every element of A or the machine that halts on every element not in A halts on $\bar{n}$.
- We write a new machine h that runs both these machines on a given input $\bar{n}$ at the same time, and stops when one of them stops. (At this point we don't know which!)
- Suppose that h runs for t steps. We can then check if the computation of the machine for A finishes in t steps. If it does, great, we're in A! If not, then also great, we're not in A!

THEOREM 3.2.7. *Let $A \subseteq \mathbb{N}^p$. Then, the following are equivalent:*

(1) A is recursively enumerable.

(2) *There is a primitive recursive set $B \subseteq \mathbb{N}^{p+1}$ such that $A = \pi(B)$, where π denotes the projection onto the last p coordinates.*

PROOF.

- For (1) $\implies$ (2), suppose that A is recursively enumerable, say $A = W_x^p$. Then A is precisely $\pi(\mathsf{halted}^p(x))$.
- For (2) $\implies$ (1), suppose that $A = \pi(B)$, where B is primitive recursive. Then A is the domain of the partial function:

$$\mu t.[(t, x_1, \ldots, x_p) \in B]$$

□

Exercise 3.2.8. Show that:

(1) The graph of a partial recursive function is recursively enumerable.

(2) The range of a partial recursive function is recursively enumerable.

THEOREM 3.2.9. *Every non-empty recursively enumerable subset of* $\mathbb{N}$ *is the range of a* primitive *recursive function in* $\mathcal{F}_1$.

PROOF. Let A be a non-empty recursively enumerable subset of $\mathbb{N}$. Suppose that $A = W_x^1$. Let $n_0 \in A$. Define the following function:

$$g(z) = \begin{cases} \mathsf{unpair}_2(z) & \text{if } (\mathsf{unpair}_1(z), \mathsf{unpair}_2(z)) \in \mathsf{halted}^p(x) \\ n_0 & \text{otherwise.} \end{cases}$$

Since $n \in A$ if and only if $(t, n) \in \mathsf{halted}^p(x)$, we have that:

- If $n \in \mathsf{ran}(g)$, then either $n = n_0 \in A$ or $n = \mathsf{unpair}_2(z)$, where $(\mathsf{unpair}_1(z), \mathsf{unpair}_2(z)) \in \mathsf{halted}^p(x)$.
- If $n \in A$, then there is some $t \in \mathbb{N}$ such that $(t, n) \in A$. Since pair is a bijection, there is some $z \in \mathbb{N}$ such that $z = \mathsf{pair}(t, n)$. Then, $n = g(z) \in \mathsf{ran}(g)$.

□

3.3. Back to diagonalisation. We will now show that there are recursively enumerable sets which are not recursive. This will be the key (and only step) in our (i.e. Alan Turing's) inevitable solution to the Halting problem.

Let's pull our first and best trick once again.

Let $g(x) = \phi^1(x, x)$ (g is the partial recursive function that halts on input x if the machine whose index is x halts on input x! Can you see how things are about to get diagonal?)

Let A be the domain of g. Since g is a recursive function, A is by definition recursively enumerable. The key property of A is that $x \in A$ if and only if $g(x)$ halts, if and only if $\phi^1(x, x)$ halts, i.e. if and only if $x \in W^1_x$.

By Theorem 3.2.5, A is recursive if and only if $\mathbb{N} \setminus A$ is also recursively enumerable. Suppose towards a contradiction that $\mathbb{N} \setminus A$ is recursively enumerable. Then. there is some $n \in \mathbb{N}$ such that $\mathbb{N} \setminus A = W^1_x$. In particular, for all $n \in \mathbb{N}$ we have that:

$$n \in \mathbb{N} \setminus A \text{ if and only if } n \in W^1_x$$

But then, we have that:

$$\begin{aligned} x \in \mathbb{N} \setminus A &\text{ if and only if } x \in W^1_x \\ &\text{ if and only if } x \in A, \end{aligned}$$

since, once again $x \in A$ if and only if $x \in W^1_x$. Thus we have written down something absurd, so $\mathbb{N} \setminus A$ is NOT recursively enumerable, and hence A is not recursive.

To put it all together:

Theorem 3.3.1 (The Halting Problem is Undecidable). *The set*

$$\{(m, x) : \phi^1(m, x) \textit{ is defined}\}$$

is not recursive.

Proof. If it were, then the set $\{x : \phi(x, x) \text{ is defined}\}$ would also be recursive, but we have just seen that it's not. □

Thus we have answered the **Halting Problem**:

Given a piece of computer code and a set of inputs for that code, there is no effective way of knowing whether the code will ever stop or not!

3.4. Two little highlights. One of the highlights of early (post-Turing) recursion theory is **Rice's theorem**, which essentially says that the only sets of partial recursive functions that we can effectively recognise are trivial. We'll not get to prove this here (although we are arguably very close to having all the tools to do so). This theorem, in particular, implies the following (intuitive) statements:

- There is no way of effectively describing all computer programs that compute a given function f.
- There is no effective way of recognising whether or not two pieces of code compute the same function.

Alright, that's enough recursion theory for a day.

Homework 8