11.1 Padding Lemma. Every p.r. function has infinitely many indices. In particular, there are strictly monotonic prim. functions f and g such that for every y, $\varphi_y = \varphi_{f(y)} = \varphi_{g(x,y)}$.

It is clear now how we must code configurations as numbers; and next, *finite sequences* of configurations. Moreover, we can check whether such a sequence codes a complete calculation of P_e.

11.2 Normal Form Theorem (*Kleene*). There exists a primitive recursive function U, and for every $n > 0$, there exists a primitive recursive predicate T_n, such that for all x,

$$\varphi_e^{(n)}(x_1,\dots, x_n) \simeq U(\mu y\, T_n(e, x_1,\dots, x_n, y)).$$

Comment. The definition of *complete equality* in terms of identity is

$$M \simeq N \text{ if and only if: if } M = M \text{ or } N = N, \text{ then } M = N.$$

Proof. U counts the primes that divide the multiplicity of the greatest prime factor (in other words, the exponent of that factor in the prime factorization) of its argument just once. ☒

Corollary. The Turing computable partial functions are μ-recursive.

The converse holds as well (see Kleene's book for a proof). We take this final piece of evidence for the Church-Turing thesis to be conclusive: we have captured a natural notion of computability that every student possesses.

12 The Enumeration and s-m-n Theorems

In fact we need only one, programmable, Turing machine.

12.1 Enumeration Theorem. For every $n > 0$, there is an index z_n such that

$$\varphi_{z_n}^{(n+1)}(e,x_1,\dots,x_n) \simeq \varphi_e^{(n)}(x_1,\dots,x_n).$$

By our prime product representation, not every number codes a pair. We will have use for a *surjective* pairing. Let $(x, y) \mapsto \langle x, y\rangle$ be one, with projections π_1 and π_2. Then we also have surjective tripling, with

$$\langle x, y, z\rangle = \langle\langle x, y\rangle, z\rangle,$$

and so on. We apply this as a reason to make light of arities.

12.2 s-m-n Theorem. For every m, $n > 0$, there exists an injective computable $(m + 1)$-ary function s_n^m such that for all $x, y_1,\dots, y_n$,

$$\varphi^{(n)}_{s_n^m(x,y_1,\dots,y_n)} \simeq \lambda(z_1,\dots,z_n).\varphi_x^{(m+n)}(y_1,\dots,y_m,z_1,\dots,z_n).$$

Example. There exists a computable function f such that $\varphi_{f(x)} = 2\varphi_x$.

12.3 Definition. By $\varphi_{e,s}(x) = y$ we express that e, x and y are less than s, and $\varphi_e(x)$ converges to y in fewer than s steps.

So $\varphi_{e,s}$ is a finite partial function; but if it diverges, we will know.

12.4 Theorem. The predicates $\varphi_{e,s}(x)\downarrow$ and $\varphi_{e,s}(x) = y$ are computable.

Every computable partial function φ_e is the union of a sequence of decidable finite partial functions.

13 Exercises

:1 (a) Convince yourself that the T-predicate is computable.
(b)* Prove that the T-predicate is primitive recursive.
:2* Prove that the s^m_n-function is primitive recursive.

14 Unsolvable problems

14.1 Definition. (i) A set is *computably enumerable* (*c.e.*) if it is the domain of a p.c. function.
(ii) $W_e := \mathrm{Dom}\,\varphi_e = \{x \mid \varphi_e(x)\downarrow\} = \{x \mid \exists y T(e, x, y)\}$.
(iii) $W_{e,s} = \mathrm{Dom}\,\varphi_{e,s}$.

So a computably enumerable set is a union of finite computable sets. Conversely, computable sets are computably enumerable. The 'enumerable' will be explained later.

14.2 Definition. $K := \{x \mid \varphi_x(x) \text{ converges}\} = \{x \mid x \in W_x\}$.

14.3 Theorem. K is c.e.

Proof. Let z_1 be as in the Enumeration Theorem (12.1); let e be an index of $\lambda x.\varphi_{z_1}(x, x)$. Then $K = W_e$. ☒

14.4 Theorem. K is not computable.

Proof. The function f defined by

$$f(x) = \varphi_x(x) + 1 \text{ if } x \in K, \quad 0 \text{ otherwise}$$

cannot be computable. ☒

14.5 Definition. $K_0 := \{\langle x, y\rangle \mid \varphi_x(y) \text{ converges}\}$.

Observe that K_0 is c.e.

14.6 Corollary (*unsolvability of the halting problem*). K_0 is not computable.

15 Reduction

15.1 Definition. Let A and B be sets (of natural numbers).
(i) A is *many-one reducible* (*m-reducible*) to B, notation $A \leq_m B$, if there exists a computable function f such that $x \in A \Leftrightarrow f(x) \in B$.
(ii) A is *one-one reducible* (*1-reducible*) to B, notation $A \leq_1 B$, if there exists a 1-1 computable function f such that $x \in A \Leftrightarrow f(x) \in B$.

For example, $K \leq_1 K_0$. Observe that $A \leq_m B$ implies $\overline{A} \leq_m \overline{B}$, by the same function. These reducibilities are easily seen to be reflexive and transitive, so $\leq_m \cap \geq_m$ and $\leq_1 \cap \geq_1$ are equivalence relations. We denote them by $\equiv_m$ and $\equiv_1$, respectively. The *m-degree* $\deg_m(A)$ is $A/\equiv_m$; the 1-*degree* $\deg_1(A)$ is $A/\equiv_1$.

15.2 Proposition. If $A \leq_m B$ and B is computable, then A is computable.

15.3 Theorem. $K \leq_1 \mathrm{Tot} := \{x \mid \mathrm{Dom}\, \varphi_x = \omega\}$.

Proof. There exists a 1-1 computable function f such that $\varphi_{f(x)}(y) \simeq \varphi_x(x)$. ☒

The proof shows that we cannot decide either whether a p.c. function is a constant function, or whether it is empty. Moreover, we can substitute any c.e. set for K.

16 Index sets

17 Complete sets, degrees and lattices

18 Exercises

:1 Suppose $B = A \oplus \overline{A}$ for some set $A \subset \omega$. Prove $B \leq_1 \overline{B}$.
:2 Prove that $\deg_m(A \oplus B) = \deg_m(A) \vee \deg_m(B)$.
:3 Prove that K_0, K_1 and K are 1-equivalent.