§1 (3.7) The recursion theorem

1.1. Definition. (i) $\mathbb{N}_* = \mathbb{N} \cup \{*\}$.

(ii) (Application: 'app') For $n, m \in \mathbb{N}_*$ define

$$\begin{array}{rcll} n.m & = & \varphi_n(m), & \text{if } n, m \in \mathbb{N} \text{ and } \varphi_n(m)\downarrow \\ & = & *, & \text{else} \end{array}$$

1.2. Notation (Currying).

Let $e, x_1, \ldots, x_n \in \mathbb{N}_*$. Write (application to the left)

$$ex_1 \ldots x_n = (..((e.x_1)x_2) \ldots x_n).$$

1.3. Proposition. (i) *For every $n, e \in \mathbb{N}$ there exists an $e' \in \mathbb{N}$ such that for all $\vec{x} = x_1, \ldots, x_n \in \mathbb{N}$*

$$\varphi_e^n(\vec{x}) \simeq e'\vec{x}.$$

(ii) $\forall n \in \mathbb{N}\ \exists s_n \in \mathbb{N}\ \forall e, \vec{x} = x_1, \ldots, x_n \in \mathbb{N}\ .\varphi_e^n(\vec{x}) \simeq s_n e\vec{x}$

Moreover, φ_{s_n} is primitive recursive and $\forall e, x_1, \ldots, x_{n-1}.s_n e x_1 \ldots x_{n-1}\downarrow$.

Moral: apart from 'app' we don't need functions with several variables.

This is simpler than $\forall n, e \exists e' \forall x_1, \ldots, x_n.\varphi_e^n(\vec{x}) = \varphi_{e'}^1(\langle\vec{x}\rangle)$.

Proof sketch of Prop 1.3

For $n = 3$ and e we construct e' such that $\varphi_e^3(x_1, x_2, x_3) = e' x_1 x_2 x_3$.

$$\begin{array}{rcl}
\varphi_e^3(x_1, x_2, x_3) & = & \varphi_{S(e,x_1,x_2)}(x_3) \\
 & = & \varphi_{\varphi_p(e,x_1,x_2)}(x_3) \\
 & = & \varphi_{\varphi_{S'(p,e,x_1)}(x_2)}(x_3) \\
 & = & \varphi_{\varphi_{\varphi_{S''(p',e)}(x_1)}(x_2)}(x_3) \\
 & = & \varphi_{\varphi_{\varphi_{e'}(x_1)}(x_2)}(x_3) \\
 & = & e' x_1 x_2 x_3. \blacksquare
\end{array}$$

1.4. Exercise. (i) Express this e' in terms of $e, 3$, using the S-m-n functions.

(ii) Give the complete proof of Proposition 1.3

(iii) Can you derive the S-n-m theorem from Proposition 1.3?

1.5. Definition. Let $n, n' \in \mathbb{N}$. Write (extensional equivalence)

$$n \sim n' \overset{\Delta}{\iff} \forall x. nx \simeq n'x.$$

1.6. Theorem (Recursion Theorem). *Let f be a (total) computable function. Then*

$$\exists e. f(e) \sim e.$$

1.7. Corollary. *There exists an e such that for all x*

$$\left.\begin{array}{rcll} \varphi_e(x) & = & x^2, & \textit{if } x \textit{ is even,} \\ & \simeq & \varphi_{e+1}(x), & \textit{else.} \end{array}\right\} \quad (1)$$

Proof. There exists a partially computable ψ such that for all e, x

$$\begin{array}{rcll} \psi(e,x) & = & x^2, & \text{if } x \text{ is even,} \\ & \simeq & \varphi_{e+1}(x), & \text{else.} \end{array}$$

From the S-m-n theorem $\psi(e,x) \simeq \varphi_{S(e)}(x)$. By the recursion theorem $S(e) \sim e$ for some e. But then this e satisfies (1)

$$\varphi_e(x) \simeq \varphi_{S(e)}(x) \simeq \psi(e,x). \blacksquare$$

In general

1.8. COROLLARY. *Let $\psi(e, \vec{x})$ be partial computable. There exists an e such that*

$$\varphi_e(\vec{x}) \simeq \psi(e, \vec{x}).$$

Sharper version of the recursion theorem (with similar proof).

1.9. PROPOSITION. *There exists a total computable function y of one argument such that if φ_p is total, then $e = y(p)$ satisfies*

$$\varphi_p(e) \sim e.$$

1.10. COROLLARY. *Given a computable function g with two arguments, there exists a unary computable function f such that*

$$g(n, f(n)) \sim f(n).$$

PROOF. Let $g(n, e) = \varphi_{s(n)}(e)$. Then take $f(n) = y(s(n))$:

$$f(n) = y(s(n)) \sim \varphi_{s(n)}(y(s(n))) = g(n, y(s(n))) = g(n, f(n)). \blacksquare$$

1.11. Corollary. *Let g, h be computable functions of two arguments. Then there exists e_1, e_2 such that*

$$\begin{aligned} e_1 &\sim g(e_1, e_2); \\ e_2 &\sim h(e_1, e_2). \end{aligned}$$

Proof. By Prop 1.10 there exists a computable function f such that

$$f(n) \sim h(n, f(n)). \tag{1}$$

By the recursion theorem there exists an a such that $a \sim g(a, f(a))$. Now take $b = f(a)$. Then $a \sim g(a, b)$, and $b \sim h(a, b)$ by (1). ■

1.12. Corollary. *Let $\psi_1(e, e', \vec{x}), \psi_2(e, e', \vec{y})$ be partial computable. There exist e_1, e_2 such that for all $\vec{x}, \vec{y}$*

$$\begin{aligned} \varphi_{e_1}(\vec{x}, \vec{y}) &\simeq \psi_1(e_1, e_2, \vec{x}), \\ \varphi_{e_2}(\vec{x}, \vec{y}) &\simeq \psi_2(e_1, e_2, \vec{y}). \end{aligned}$$

Proof. Exercise.

1.6. Theorem (Recursion Theorem). *Let f be a (total) computable function. Then*

$$\exists e. f(e) \sim e.$$

Proof. The partial computable function $\varphi_n(m)$ can be made total modulo $\sim$

$$\varphi_n(m) \sim s(n, m),$$

by considering

$$\varphi_{\varphi_n(m)}(x) \simeq \psi(n, m, x) \simeq \varphi_q(n, m, x) \simeq \varphi_{s(n,m)}(x).$$

Given $f = \varphi_p$ total recursive, define

$$\begin{array}{rcccl} h(n) & \triangleq & f(s(n, n)) & = & \varphi_r(n); \\ e & \triangleq & s(r, r) & & \end{array}$$

Then

$$e = s(r, r) \sim \varphi_r(r) = f(s(r, r)) = f(e).$$

Clearly e depends uniformly on the program p of f.

This gives the sharper version Proposition 1.9. ■

§2 (3.5) Many-one reducibility

2.1. Definition. Let $A, B \subseteq \mathbb{N}$. We say that A is (many-one) reducible to B, written $A \leq_m B$, if for some computable $f\colon \mathbb{N}\to\mathbb{N}$

$$\forall n \in \mathbb{N}\,[n \in A \;\Leftrightarrow\; f(n) \in B]$$

If we require f to be 1-1, then we write $A \leq_1 B$ (one reducibility).

2.2. Proposition. *Suppose $A \leq_m B$. Then*
(i) *B is computable $\Rightarrow$ A is computable.*
(ii) *A is non-computable $\Rightarrow$ B is non-computable.*

Proof. (i) $x \in A \;\Leftrightarrow\; f(x) \in B \;\Leftrightarrow\; \chi_B(f(x)) = 1$. So

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

Hence if B is computable, then by definition χ_B is computable, so is χ_A and hence A.
(ii) By (i). ■

Non-computable of a set B one can often proved by showing $B \leq_m K$.

2.3. Exercise. Show that A is c.e. iff $A \leq_m K$.