Recursive Functions

Gödel's Primitive Recursive Functions F_{pr}

A computable function represents a calculation that returns a result in finitely many well-defined steps.

Initial functions:

$$\begin{array}{lcll} Zero(x) & = & 0 & (Zero) \\ Proj^i_n(x_1, \ldots, x_n) & = & x_i \quad (1 \leq k \leq n) & (Projection) \end{array}$$

are in $\mathcal{F}_{pr}$.

Composition: If $g_1, \ldots, g_m : I\!N^k \to I\!N$, $h : I\!N^m \to I\!N \in \mathcal{F}_{pr}$, then $f : I\!N^k \to I\!N \in \mathcal{F}_{pr}$, where f is defined by

$$f(x_1, \ldots, x_k) \quad = \quad h(g_1(x_1, \ldots, x_k), \ldots, g_m(x_1, \ldots, x_k))$$

Primitive Recursion: If $g : I\!N^k \to I\!N$, $h : I\!N^{k+2} \to I\!N \in \mathcal{F}_{pr}$, then $f : I\!N^{k+1} \to I\!N \in \mathcal{F}_{pr}$, where f is defined by

$$\begin{array}{lcl} f(0, x_1, \ldots, x_k) & = & g(x_1, \ldots, x_k) \\ f(Succ(y), x_1, \ldots, x_k) & = & h(y, f(y, x_1, \ldots, x_k), x_1, \ldots, x_k) \end{array}$$

An Example

As an example, take the *factorial function*:

$$\begin{array}{lcl} fac\,(0) & = & 1 \\ fac\,(Succ(y)) & = & Succ(y) \times fac\,(y) \end{array}$$

(so here $k = 0$, $g() = 1$ (so we see constants as parameterless functions), and $h(y, z) = S(y) \times z$).

Thus, this function is **primitive recursive**.

But not all computable functions are primitive recursive.

The **Ackermann function** Ack : $I\!N^2 \to I\!N$ defined by

$$
\begin{array}{lcl}
\mathrm{Ack}\,(0, y) & = & \mathit{Succ}\,(y) \\
\mathrm{Ack}\,(\mathit{Succ}\,(x), 0) & = & \mathrm{Ack}\,(x, 1) \\
\mathrm{Ack}(\mathit{Succ}\,(x), \mathit{Succ}\,(y)) & = & \mathrm{Ack}\,(x, \mathrm{Ack}\,(\mathit{Succ}\,(x), y))
\end{array}
$$

is clearly computable (in fact, below we will show it always terminates) but does not belong to $\mathcal{F}_{pr}$.

Kleene's (Partial) Recursive Functions

The class of (partial) recursive functions F_{rec} in $N^k \to N$ (for any k) is defined as F_{pr}, but adding:

Minimisation : If $f : I\!N^{n+1} \to I\!N \in \mathcal{F}_{rec}$, then $h : I\!N^n \to I\!N \in \mathcal{F}_{rec}$, where h is defined by:

$$h(x_1, \ldots, x_n) \quad = \quad \mu y\,(\,f(y, x_1, \ldots, x_n) = 0\,)$$

Intuitively, minimisation 'μy' expresses the search for, beginning with 0 and proceeding upwards, the smallest argument y that causes $f(y, x_1, \ldots, x_n)$ to return 0. So it steps through

$$f(0, x_1, \ldots, x_n), \quad f(1, x_1, \ldots, x_n), \quad f(2, x_1, \ldots, x_n), \quad \ldots$$

This search might not terminate; then h is partial.

Partial recursive functions capture exactly the notion of computability.