Partial recursive functions

Aim

A more abstract, machine-independent description of the collection of computable partial functions than provided by register/Turing machines:

> *they form the smallest collection of partial functions containing some* *basic* *functions and closed under some fundamental operations for forming new functions from old—* *composition*, *primitive recursion* *and* *minimization*.

The characterization is due to Kleene (1936), building on work of Gödel and Herbrand.

Basic functions

- **Projection** functions, $\text{proj}_i^n \in \mathbb{N}^n{\to}\mathbb{N}$:

$$\text{proj}_i^n(x_1, \ldots, x_n) \triangleq x_i$$

- **Constant** functions with value 0, $\text{zero}^n \in \mathbb{N}^n{\to}\mathbb{N}$:

$$\text{zero}^n(x_1, \ldots, x_n) \triangleq 0$$

- **Successor** function, $\text{succ} \in \mathbb{N}{\to}\mathbb{N}$:

$$\text{succ}(x) \triangleq x + 1$$

Composition

Composition of $f \in \mathbb{N}^n \rightharpoonup \mathbb{N}$ with $g_1, \ldots, g_n \in \mathbb{N}^m \rightharpoonup \mathbb{N}$ is the partial function $f \circ [g_1, \ldots, g_n] \in \mathbb{N}^m \rightharpoonup \mathbb{N}$ satisfying for all $x_1, \ldots, x_m \in \mathbb{N}$

$$f \circ [g_1, \ldots, g_n](x_1, \ldots, x_m) \equiv$$
$$f(g_1(x_1, \ldots, x_m), \ldots, g_n(x_1, \ldots, x_m))$$

where $\equiv$ is "Kleene equivalence" of possibly-undefined expressions: LHS $\equiv$ RHS means "either both LHS and RHS are undefined, or they are both defined and are equal."

Primitive recursion

Theorem. Given $f \in \mathbb{N}^n \rightharpoonup \mathbb{N}$ and $g \in \mathbb{N}^{n+2} \rightharpoonup \mathbb{N}$, there is a unique $h \in \mathbb{N}^{n+1} \rightharpoonup \mathbb{N}$ satisfying

$$\begin{cases} h(\vec{x}, 0) & \equiv f(\vec{x}) \\ h(\vec{x}, x+1) & \equiv g(\vec{x}, x, h(\vec{x}, x)) \end{cases}$$

for all $\vec{x} \in \mathbb{N}^n$ and $x \in \mathbb{N}$.

We write $\rho^n(f, g)$ for h and call it the partial function defined by primitive recursion from f and g.

Example: addition

Addition $add \in \mathbb{N}^2 \rightarrow \mathbb{N}$ satisfies:

$$\begin{cases} add(x_1, 0) & \equiv x_1 \\ add(x_1, x+1) & \equiv add(x_1, x) + 1 \end{cases}$$

So $add = \rho^1(f, g)$ where $\begin{cases} f(x_1) & \triangleq x_1 \\ g(x_1, x_2, x_3) & \triangleq x_3 + 1 \end{cases}$

Note that $f = \mathrm{proj}_1^1$ and $g = \mathrm{succ} \circ \mathrm{proj}_3^3$; so add can be built up from basic functions using composition and primitive recursion:
$add = \rho^1(\mathrm{proj}_1^1, \mathrm{succ} \circ \mathrm{proj}_3^3)$.

Example: predecessor

Predecessor $pred \in \mathbb{N} \to \mathbb{N}$ satisfies:

$$\begin{cases} pred(0) & \equiv 0 \\ pred(x+1) & \equiv x \end{cases}$$

So $pred = \rho^0(f, g)$ where $\begin{cases} f() & \triangleq 0 \\ g(x_1, x_2) & \triangleq x_1 \end{cases}$

Thus $pred$ can be built up from basic functions using primitive recursion: $pred = \rho^0(\mathtt{zero}^0, \mathtt{proj}_1^2)$.

Example: multiplication

Multiplication $mult \in \mathbb{N}^2 \to \mathbb{N}$ satisfies:

$$\begin{cases} mult(x_1, 0) & \equiv 0 \\ mult(x_1, x+1) & \equiv mult(x_1, x) + x_1 \end{cases}$$

and thus $mult = \rho^1(\texttt{zero}^1, add \circ (\texttt{proj}_3^3, \texttt{proj}_1^3))$.
So $mult$ can be built up from basic functions using composition and primitive recursion (since add can be).

Definition. A [partial] function f is primitive recursive ($f \in \mathrm{PRIM}$) if it can be built up in finitely many steps from the basic functions by use of the operations of composition and primitive recursion.

In other words, the set PRIM of primitive recursive functions is the smallest set (with respect to subset inclusion) of partial functions containing the basic functions and closed under the operations of composition and primitive recursion.

Definition. A [partial] function f is primitive recursive ($f \in \mathrm{PRIM}$) if it can be built up in finitely many steps from the basic functions by use of the operations of composition and primitive recursion.

Every $f \in \mathrm{PRIM}$ is a total function, because:

- all the basic functions are total
- if $f, g_1, \ldots, g_n$ are total, then so is $f \circ (g_1, \ldots, g_n)$ [why?]
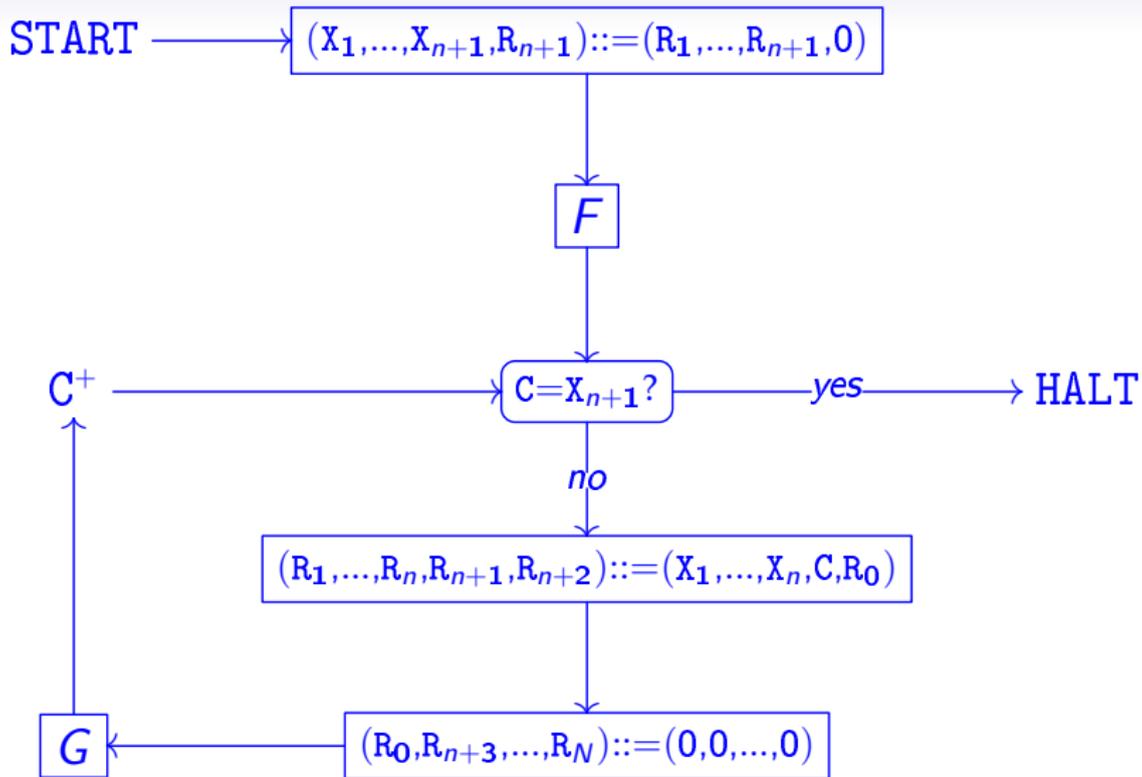- if f and g are total, then so is $\rho^n(f, g)$ [why?]

Definition. A [partial] function f is primitive recursive ($f \in \mathrm{PRIM}$) if it can be built up in finitely many steps from the basic functions by use of the operations of composition and primitive recursion.

Theorem. Every $f \in \mathrm{PRIM}$ is computable.
Proof. Already proved: basic functions are computable; composition preserves computability. So just have to show:

$\rho^n(f, g) \in \mathbb{N}^{n+1} \rightharpoonup \mathbb{N}$ computable if $f \in \mathbb{N}^n \rightharpoonup \mathbb{N}$ and $g \in \mathbb{N}^{n+2} \rightharpoonup \mathbb{N}$ are.

Suppose f and g are computed by RM programs F and G (with our usual I/O conventions). Then the RM specified on the next slide computes $\rho^n(f, g)$. (We assume $\mathtt{X}_1, \ldots, \mathtt{X}_{n+1}, \mathtt{C}$ are some registers not mentioned in F and G; and that the latter only use registers $\mathtt{R}_0, \ldots, \mathtt{R}_N$, where $N \geq n+2$.)

START $\longrightarrow$ $(X_1,...,X_{n+1},R_{n+1})::=(R_1,...,R_{n+1},0)$

$\boxed{F}$

C^+ $\longrightarrow$ $C=X_{n+1}?$ $\longrightarrow$ *yes* $\longrightarrow$ HALT

no

$(R_1,...,R_n,R_{n+1},R_{n+2})::=(X_1,...,X_n,C,R_0)$

$\boxed{G}$ $\longleftarrow$ $(R_0,R_{n+3},...,R_N)::=(0,0,...,0)$

Aim

A more abstract, machine-independent description of the collection of computable partial functions than provided by register/Turing machines:

> they form the smallest collection of partial functions containing some basic functions and closed under some fundamental operations for forming new functions from old—composition, primitive recursion and minimization.

The characterization is due to Kleene (1936), building on work of Gödel and Herbrand.

Minimization

Given a partial function $f \in \mathbb{N}^{n+1} \rightharpoonup \mathbb{N}$, define $\mu^n f \in \mathbb{N}^n \rightharpoonup \mathbb{N}$ by

$$\mu^n f(\vec{x}) \triangleq \text{ least } x \text{ such that } f(\vec{x}, x) = 0 \text{ and for}$$
each $i = 0, \ldots, x - 1$, $f(\vec{x}, i)$ is defined
and > 0
(undefined if there is no such x)

In other words

$$\mu^n f = \{(\vec{x}, x) \in \mathbb{N}^{n+1} \mid \exists y_0, \ldots, y_x$$

$$(\bigwedge_{i=0}^{x} f(\vec{x}, i) = y_i) \wedge (\bigwedge_{i=0}^{x-1} y_i > 0) \wedge y_x = 0\}$$

Example of minimization

$$\text{integer part of } x_1/x_2 \quad \equiv \quad \text{least } x_3 \text{ such that}$$
$$\text{(undefined if } x_2=0) \qquad \qquad x_1 < x_2(x_3+1)$$

Example of minimization

integer part of x_1/x_2 $\equiv$ least x_3 such that
(undefined if $x_2{=}0$) $\qquad x_1 < x_2(x_3+1)$

$$\equiv \quad \mu^2 f(x_1, x_2)$$

where $f \in \mathbb{N}^3 \to \mathbb{N}$ is

$$f(x_1, x_2, x_3) \triangleq \begin{cases} 1 & \text{if } x_1 \geq x_2(x_3+1) \\ 0 & \text{if } x_1 < x_2(x_3+1) \end{cases}$$

Example of minimization

$$\text{integer part of } x_1/x_2 \quad \equiv \quad \text{least } x_3 \text{ such that}$$

$$\text{(undefined if } x_2=0) \qquad\qquad x_1 < x_2(x_3+1)$$

$$\equiv \quad \mu^2 f(x_1, x_2)$$

where $f \in \mathbb{N}^3 \to \mathbb{N}$ is

$$f(x_1, x_2, x_3) \triangleq \begin{cases} 1 & \text{if } x_1 \geq x_2(x_3+1) \\ 0 & \text{if } x_1 < x_2(x_3+1) \end{cases}$$

(In fact, if we make the 'integer part of x_1/x_2' function total by defining it to be 0 when $x_2 = 0$, it can be shown to be in PRIM.)

Definition. A partial function f is partial recursive ($f \in \mathrm{PR}$) if it can be built up in finitely many steps from the basic functions by use of the operations of composition, primitive recursion and minimization.

In other words, the set PR of partial recursive functions is the smallest set (with respect to subset inclusion) of partial functions containing the basic functions and closed under the operations of composition, primitive recursion and minimization.

Definition. A partial function f is partial recursive ($f \in \mathrm{PR}$) if it can be built up in finitely many steps from the basic functions by use of the operations of composition, primitive recursion and minimization.

Theorem. Every $f \in \mathrm{PR}$ is computable.
Proof. Just have to show:

$\mu^n f \in \mathbb{N}^n \rightharpoonup \mathbb{N}$ is computable if $f \in \mathbb{N}^{n+1} \rightharpoonup \mathbb{N}$ is.

Suppose f is computed by RM program F (with our usual I/O conventions). Then the RM specified on the next slide computes $\mu^n f$. (We assume $\mathrm{X}_1, \ldots, \mathrm{X}_n, \mathrm{C}$ are some registers not mentioned in F; and that the latter only uses registers $\mathrm{R}_0, \ldots, \mathrm{R}_N$, where $N \geq n+1$.)

START

$(X_1,...,X_n)::=(R_1,...,R_n)$

$(R_1,...,R_n,R_{n+1})::=(X_1,...,X_n,C)$

C^+

$(R_0,R_{n+2},...,R_N)::=(0,0,...,0)$

F

R_0^- $\longrightarrow$ $R_0::=C$ $\longrightarrow$ HALT