rec.1 Primitive Recursion

cmp:rec:pre:sec

Suppose we specify that a certain function l from $\mathbb{N}$ to $\mathbb{N}$ satisfies the following two clauses: explanation

$$\begin{aligned} l(0) &= 1 \\ l(x+1) &= 2 \cdot l(x). \end{aligned}$$

It is pretty clear that there is only one function, l, that meets these two criteria. This is an instance of a *definition by primitive recursion*. We can define even more fundamental functions like addition and multiplication by

$$\begin{aligned} f(x,0) &= x \\ f(x,y+1) &= f(x,y)+1 \end{aligned}$$

and

$$\begin{aligned} g(x,0) &= 0 \\ g(x,y+1) &= f(g(x,y),x). \end{aligned}$$

Exponentiation can also be defined recursively, by

$$\begin{aligned} h(x,0) &= 1 \\ h(x,y+1) &= g(h(x,y),x). \end{aligned}$$

We can also compose functions to build more complex ones; for example,

$$\begin{aligned} k(x) &= x^x + (x+3) \cdot x \\ &= f(h(x,x), g(f(x,3),x)). \end{aligned}$$

Let $\mathrm{zero}(x)$ be the function that always returns 0, regardless of what x is, and let $\mathrm{succ}(x) = x+1$ be the successor function. The set of *primitive recursive functions* is the set of functions from $\mathbb{N}^n$ to $\mathbb{N}$ that you get if you start with zero and succ by iterating the two operations above, primitive recursion and composition. The idea is that primitive recursive functions are defined in a straightforward and explicit way, so that it is intuitively clear that each one can be computed using finite means.

Definition rec.1. If f is a k-place function and $g_0, \ldots, g_{k-1}$ are l-place functions on the natural numbers, the *composition* of f with $g_0, \ldots, g_{k-1}$ is the l-place function h defined by

$$h(x_0, \ldots, x_{l-1}) = f(g_0(x_0, \ldots, x_{l-1}), \ldots, g_{k-1}(x_0, \ldots, x_{l-1})).$$

Definition rec.2. If f is a k-place function and g is a $(k+2)$-place function, then the function defined by *primitive recursion from f and g* is the $(k+1)$-place function h defined by the equations

$$\begin{aligned} h(0, z_0, \ldots, z_{k-1}) &= f(z_0, \ldots, z_{k-1}) \\ h(x+1, z_0, \ldots, z_{k-1}) &= g(x, h(x, z_0, \ldots, z_{k-1}), z_0, \ldots, z_{k-1}) \end{aligned}$$

In addition to zero and succ, we will include among primitive recursive functions the projection functions,

$$P_i^n(x_0, \ldots, x_{n-1}) = x_i,$$

for each natural number n and $i < n$. These are not terribly exciting in themselves: P_i^n is simply the k-place function that always returns its ith argument. But the allow us to define new functions by disregarding arguments or switching arguments, as we'll see later.

In the end, we have the following:

Definition rec.3. The set of primitive recursive functions is the set of functions from $\mathbb{N}^n$ to $\mathbb{N}$, defined inductively by the following clauses:

1. zero is primitive recursive.
2. succ is primitive recursive.
3. Each projection function P_i^n is primitive recursive.
4. If f is a k-place primitive recursive function and $g_0, \ldots, g_{k-1}$ are l-place primitive recursive functions, then the composition of f with $g_0, \ldots, g_{k-1}$ is primitive recursive.
5. If f is a k-place primitive recursive function and g is a $k+2$-place primitive recursive function, then the function defined by primitive recursion from f and g is primitive recursive.

explanation

Put more concisely, the set of primitive recursive functions is the smallest set containing zero, succ, and the projection functions P_j^n, and which is closed under composition and primitive recursion.

Another way of describing the set of primitive recursive functions keeps track of the "stage" at which a function enters the set. Let S_0 denote the set of starting functions: zero, succ, and the projections. Once S_i has been defined, let S_{i+1} be the set of all functions you get by applying a single instance of composition or primitive recursion to functions in S_i. Then

$$S = \bigcup_{i \in \mathbb{N}} S_i$$

is the set of all primitive recursive functions

Our definition of composition may seem too rigid, since $g_0, \ldots, g_{k-1}$ are all required to have the same arity l. (Remember that the *arity* of a function is the number of arguments; an l-place function has arity l.) But adding the projection functions provides the desired flexibility. For example, suppose f and g are 3-place functions and h is the 2-place function defined by

$$h(x, y) = f(x, g(x, x, y), y).$$

The definition of h can be rewritten with the projection functions, as

$$h(x,y) = f(P_0^2(x,y), g(P_0^2(x,y), P_0^2(x,y), P_1^2(x,y)), P_1^2(x,y)).$$

Then h is the composition of f with P_0^2, l, and P_1^2, where

$$l(x,y) = g(P_0^2(x,y), P_0^2(x,y), P_1^2(x,y)),$$

i.e., l is the composition of g with P_0^2, P_0^2, and P_1^2.

For another example, let us again consider addition. This is described recursively by the following two equations:

$$\begin{aligned} x + 0 &= x \\ x + (y+1) &= \mathrm{succ}(x+y). \end{aligned}$$

In other words, addition is the function add defined recursively by the equations

$$\begin{aligned} \mathrm{add}(0,x) &= x \\ \mathrm{add}(y+1,x) &= \mathrm{succ}(\mathrm{add}(y,x)). \end{aligned}$$

But even this is not a strict primitive recursive definition; we need to put it in the form

$$\begin{aligned} \mathrm{add}(0,x) &= f(x) \\ \mathrm{add}(y+1,x) &= g(y, \mathrm{add}(y,x), x) \end{aligned}$$

for some 1-place primitive recursive function f and some 3-place primitive recursive function g. We can take f to be P_0^1, and we can define g using composition,

$$g(y,w,x) = \mathrm{succ}(P_1^3(y,w,x)).$$

The function g, being the composition of basic primitive recursive functions, is primitive recursive; and hence so is h. (Note that, strictly speaking, we have defined the function $g(y,x)$ meeting the recursive specification of $x+y$; in other words, the variables are in a different order. Luckily, addition is commutative, so here the difference is not important; otherwise, we could define the function g' by

$$g'(x,y) = g(P_1^2(y,x)), P_0^2(y,x)) = g(y,x),$$

using composition.

One advantage to having the precise description of the primitive recursive functions is that we can be systematic in describing them. For example, we can assign a "notation" to each such function, as follows. Use symbols zero, succ, and P_i^n for zero, successor, and the projections. Now suppose f is defined by composition from a k-place function h and l-place functions g_0, …, g_{k-1}, and we have assigned notations H, G_0, …, G_{k-1} to the latter functions. Then, using a new symbol $\mathrm{Comp}_{k,l}$, we can denote the function f by

explanation

$\mathrm{Comp}_{k,l}[H, G_0, \ldots, G_{k-1}]$. For the functions defined by primitive recursion, we can use analogous notations of the form $\mathrm{Rec}_k[G, H]$, where k denotes that arity of the function being defined. With this setup, we can denote the addition function by

$$\mathrm{Rec}_2[P_0^1, \mathrm{Comp}_{1,3}[\mathrm{succ}, P_1^3]].$$

Having these notations sometimes proves useful.

Problem rec.1. Multiplication satisfies the recursive equations

$$0 \cdot y = y$$
$$(x+1) \cdot y = (x \cdot y) + x$$

Give the explicit precise definition of the function $\mathrm{mult}(x, y) = x \cdot y$, assuming that $\mathrm{add}(x, y) = x + y$ is already defined. Give the complete notation for mult.

Photo Credits

Bibliography