

rec.1 Primitive Recursion Functions

cmp:rec:prf:
sec Let us record again how we can define new functions from existing ones using primitive recursion and composition.

cmp:rec:prf:
defn:primitive-recursion **Definition rec.1.** Suppose f is a k -place function ($k \geq 1$) and g is a $(k+2)$ -place function. The function defined by *primitive recursion from f and g* is the $(k+1)$ -place function h defined by the equations

$$\begin{aligned}h(x_0, \dots, x_{k-1}, 0) &= f(x_0, \dots, x_{k-1}) \\h(x_0, \dots, x_{k-1}, y+1) &= g(x_0, \dots, x_{k-1}, y, h(x_0, \dots, x_{k-1}, y))\end{aligned}$$

cmp:rec:prf:
defn:composition **Definition rec.2.** Suppose f is a k -place function, and g_0, \dots, g_{k-1} are k functions which are all n -place. The function defined by *composition from f and g_0, \dots, g_{k-1}* is the n -place function h defined by

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

In addition to succ and the projection functions

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

for each natural number n and $i < n$, we will include among the primitive recursive functions the function $\text{zero}(x) = 0$.

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, \dots, g_{k-1} are n -place primitive recursive functions, then the composition of f with g_0, \dots, 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.

Put more concisely, the set of primitive recursive functions is the smallest explanation 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 is by defining it in terms of “stages.” Let S_0 denote the set of starting functions: zero, succ, and the projections. These are the primitive recursive functions of stage 0. Once a stage 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 already in S_i . Then

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

is the set of all primitive recursive functions

Let us verify that add is a primitive recursive function.

Proposition rec.4. *The addition function $\text{add}(x, y) = x + y$ is primitive recursive.*

Proof. We already have a primitive recursive definition of add in terms of two functions f and g which matches the format of [Definition rec.1](#):

$$\begin{aligned} \text{add}(x_0, 0) &= f(x_0) = x_0 \\ \text{add}(x_0, y + 1) &= g(x_0, y, \text{add}(x_0, y)) = \text{succ}(\text{add}(x_0, y)) \end{aligned}$$

So add is primitive recursive provided f and g are as well. $f(x_0) = x_0 = P_0^1(x_0)$, and the projection functions count as primitive recursive, so f is primitive recursive. The function g is the three-place function $g(x_0, y, z)$ defined by

$$g(x_0, y, z) = \text{succ}(z).$$

This does not yet tell us that g is primitive recursive, since g and succ are not quite the same function: succ is one-place, and g has to be three-place. But we can define g “officially” by composition as

$$g(x_0, y, z) = \text{succ}(P_2^3(x_0, y, z))$$

Since succ and P_2^3 count as primitive recursive functions, g does as well, since it can be defined by composition from primitive recursive functions. \square

Proposition rec.5. *The multiplication function $\text{mult}(x, y) = x \cdot y$ is primitive recursive.* *cmp:rec:prf: prop:mult-pr*

Proof. Exercise. \square

Problem rec.1. Prove [Proposition rec.5](#) by showing that the primitive recursive definition of mult can be put into the form required by [Definition rec.1](#) and showing that the corresponding functions f and g are primitive recursive.

Example rec.6. Here’s our very first example of a primitive recursive definition:

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

This function cannot fit into the form required by [Definition rec.1](#), since $k = 0$. The definition also involves the constants 1 and 2. To get around the first problem, let's introduce a dummy argument and define the function h' :

$$\begin{aligned} h'(x_0, 0) &= f(x_0) = 1 \\ h'(x_0, y + 1) &= g(x_0, y, h'(x_0, y)) = 2 \cdot h'(x_0, y). \end{aligned}$$

The function $f(x_0) = 1$ can be defined from succ and zero by composition: $f(x_0) = \text{succ}(\text{zero}(x_0))$. The function g can be defined by composition from $g'(z) = 2 \cdot z$ and projections:

$$g(x_0, y, z) = g'(P_2^3(x_0, y, z))$$

and g' in turn can be defined by composition as

$$g'(z) = \text{mult}(g''(z), P_0^1(z))$$

and

$$g''(z) = \text{succ}(f(z)),$$

where f is as above: $f(z) = \text{succ}(\text{zero}(z))$. Now that we have h' , we can use composition again to let $h(y) = h'(P_0^1(y), P_0^1(y))$. This shows that h can be defined from the basic functions using a sequence of compositions and primitive recursions, so h is primitive recursive.

Photo Credits

Bibliography