req.1 Simulating Primitive Recursion

inc:req:pri: sec Now we can show that definition by primitive recursion can be "simulated" by regular minimization using the beta function. Suppose we have $f(\vec{x})$ and $g(\vec{x}, y, z)$. Then the function $h(x, \vec{z})$ defined from f and g by primitive recursion is

$$h(\vec{x}, 0) = f(\vec{x})$$
$$h(\vec{x}, y + 1) = g(\vec{x}, y, h(\vec{x}, y)).$$

We need to show that h can be defined from f and g using just composition and regular minimization, using the basic functions and functions defined from them using composition and regular minimization (such as β).

inc:req:pri: lem:prim-rec **Lemma req.1.** *If h can be defined from f and g using primitive recursion, it can be defined from f, g, the functions* zero, succ, P^n_i, add, mult, $\chi_=$, *using composition and regular minimization.*

Proof. First, define an auxiliary function $\hat{h}(\vec{x}, y)$ which returns the least number d such that d codes a sequence which satisfies

1. $(d)_0 = f(\vec{x})$, and
2. for each $i < y$, $(d)_{i+1} = g(\vec{x}, i, (d)_i)$,

where now $(d)_i$ is short for $\beta(d, i)$. In other words, $\hat{h}$ returns the sequence $\langle h(\vec{x}, 0), h(\vec{x}, 1), \dots, h(\vec{x}, y)\rangle$. We can write $\hat{h}$ as

$$\hat{h}(\vec{x}, y) = \mu d\ (\beta(d, 0) = f(\vec{x}) \land (\forall i < y)\ \beta(d, i + 1) = g(\vec{x}, i, \beta(d, i)).$$

Note: no primitive recursion is needed here, just minimization. The function we minimize is regular because of the beta function lemma **??**.

But now we have

$$h(\vec{x}, y) = \beta(\hat{h}(\vec{x}, y), y),$$

so h can be defined from the basic functions using just composition and regular minimization. □

Photo Credits

Bibliography