When it comes to primitive recursion, we finally need to do some work. We will have to proceed in stages. As before, on the assumption that we already have terms $G$ and $H$ that $\lambda$-define functions $g$ and $h$, respectively, we want a term $H'$ that $\lambda$-defines the function $f$ defined by

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

So, in general, given lambda terms $G'$ and $H'$, it suffices to find a term $F$ such that

$$F(\vec{0}, \vec{z}) \equiv G(\vec{z})$$
$$F(\vec{n} + 1, \vec{z}) \equiv H(\vec{\pi}, F(\vec{\pi}, \vec{z}), \vec{z})$$

for every natural number $n$; the fact that $G'$ and $H'$ $\lambda$-define $g$ and $h$ means that whenever we plug in numerals $\vec{m}$ for $\vec{z}$, $F(\vec{n} + 1, \vec{m})$ will normalize to the right answer.

But for this, it suffices to find a term $F$ satisfying

$$F(\vec{0}) \equiv G$$
$$F(\vec{n} + 1) \equiv H(\vec{\pi}, F(\vec{\pi}))$$

for every natural number $n$, where

$$G = \lambda \vec{z}. G'(\vec{z})$$
$$H(u, v) = \lambda \vec{z}. H'(u, v(u, \vec{z}), \vec{z}).$$

In other words, with lambda trickery, we can avoid having to worry about the extra parameters $\vec{z}$—they just get absorbed in the lambda notation.

Before we define the term $F$, we need a mechanism for handling ordered pairs. This is provided by the next lemma.

**Lemma int.1.** There is a lambda term $D$ such that for each pair of lambda terms $M$ and $N$, $D(M, N)(\vec{0}) \rightarrow M$ and $D(M, N)(\vec{1}) \rightarrow N$.

**Proof.** First, define the lambda term $K$ by

$$K(y) = \lambda x. y.$$

In other words, $K$ is the term $\lambda y. \lambda x. y$. Looking at it differently, for every $M$, $K(M)$ is a constant function that returns $M$ on any input.

Now define $D(x, y, z)$ by $D(x, y, z) = z(K(y))x$. Then we have

$$D(M, N, \vec{0}) \rightarrow \vec{0}(K(N))M \rightarrow M$$
$$D(M, N, \vec{1}) \rightarrow \vec{1}(K(N))M \rightarrow K(N)M \rightarrow N,$$

as required.

\[\square\]
The idea is that $D(M,N)$ represents the pair $\langle M,N \rangle$, and if $P$ is assumed to represent such a pair, $P(\bar{0})$ and $P(\bar{1})$ represent the left and right projections, $(P)_0$ and $(P)_1$. We will use the latter notations.

**Lemma int.2.** The $\lambda$-definable functions are closed under primitive recursion.

**Proof.** We need to show that given any terms, $G$ and $H$, we can find a term $F$ such that

$$F(\bar{0}) \equiv G$$

$$F(\bar{n} + 1) \equiv H(\bar{n}, F(\bar{n}))$$

for every natural number $n$. The idea is roughly to compute sequences of pairs $\langle \bar{0}, F(\bar{0}) \rangle, \langle \bar{1}, F(\bar{1}) \rangle, \ldots$, using numerals as iterators. Notice that the first pair is just $\langle \bar{0}, G \rangle$. Given a pair $\langle \bar{n}, F(\bar{n}) \rangle$, the next pair, $\langle \bar{n} + 1, F(\bar{n} + 1) \rangle$ is supposed to be equivalent to $\langle \bar{n} + 1, H(\bar{n}, F(\bar{n})) \rangle$. We will design a lambda term $T$ that makes this one-step transition.

The details are as follows. Define $T(u)$ by

$$T(u) = \langle S(u)_0, H((u)_0, (u)_{1}) \rangle.$$  

Now it is easy to verify that for any number $n$,

$$T(\langle \bar{n}, M \rangle) \rightarrow \langle \bar{n} + 1, H(\bar{n}, M) \rangle.$$  

As suggested above, given $G$ and $H$, define $F(u)$ by

$$F(u) = (u(T, (\bar{0}, G)))_{1}.$$  

In other words, on input $\bar{n}$, $F$ iterates $T$ $n$ times on $\langle \bar{0}, G \rangle$, and then returns the second component. To start with, we have

1. $\bar{0}(T, (\bar{0}, G)) \equiv \langle \bar{0}, G \rangle$

2. $F(\bar{0}) \equiv G$

By induction on $n$, we can show that for each natural number one has the following:

1. $\bar{n} + 1(T, (\bar{0}, G)) \equiv (\bar{n} + 1, F(\bar{n} + 1))$

2. $F(\bar{n} + 1) \equiv H(\bar{n}, F(\bar{n}))$

For the second clause, we have

$$F(\bar{n} + 1) \rightarrow (\bar{n} + 1(T, (\bar{0}, G)))_{1}$$

$$\equiv (T(\bar{n}, (\bar{0}, G)))_{1}$$

$$\equiv (T(\bar{n}, F(\bar{n})))_{1}$$

$$\equiv (\bar{n} + 1, H(\bar{n}, F(\bar{n})))_{1}$$

$$\equiv H(\bar{n}, F(\bar{n})).$$

primitive-recursion rev: 666b46f (2020-02-13) by OLP / CC–BY
Here we have used the induction hypothesis on the second-to-last line. For the first clause, we have

\[
\begin{align*}
n + 1 & \equiv T(n + 1) \\
& \equiv T(n, G) \\
& \equiv T(n, F(n)) \\
& \equiv \langle n + 1, H(n, F(n)) \rangle \\
& \equiv \langle n + 1, F(n + 1) \rangle.
\end{align*}
\]

Here we have used the second clause in the last line. So we have shown \( F(\overline{0}) \equiv G \) and, for every \( n \), \( F(\overline{n + 1}) \equiv H(\overline{n}, F(\overline{n})) \), which is exactly what we needed. □

**Photo Credits**

**Bibliography**