All Partial Recursive Functions are λ-Definable

"Lectures on the Curry-Howard Isomorphism," pp 19-21, 366-67.

We want to prove a correspondence between programs and proofs. We will use the λ-calculus, so we need to prove that it's a good model of computation. So in this talk, I'll prove that all partial recursive functions are λ-definable.

Outline:

① Examples of encodings in the λ-calculus.
② Partial recursive functions.
* ③ Church numerals and λ-definability.
④ Proposition 1.4.4 and Theorem 1.4.5.

① Examples of encodings in the λ-calculus

** **Definition:** (One of the key definitions)
β-equivalence, $=_\beta$, is the smallest equivalence relation on the set of λ-terms such that

i) $(\lambda x.P)Q =_\beta P[x:=Q]$
ii) if $M =_\beta N$ then $\lambda x.M =_\beta \lambda x.N$ for all variables x.
iii) if $M =_\beta N$ then $MZ =_\beta NZ$ and $ZM =_\beta ZN$

Equivalently, $=_\beta$ is the smallest equivalence relation containing $\to_\beta$ (see last week).

ordered pair: $\langle M, N\rangle = \lambda x.xMN$

projections: $\pi_1 = \lambda p.p(\lambda xy.x)$
$\pi_2 = \lambda p.p(\lambda xy.y)$

We ~~want~~ "π_1 to "project onto the first element of an ordered pair," so we check $\pi_1\langle M,N\rangle =_\beta M$:

$\pi_1\langle M,N\rangle = (\lambda p.p(\lambda xy.x))(\lambda x.xMN)$
$=_\beta (p(\lambda xy.x))[p:=\lambda x.xMN]$ i)

$= (\lambda x.xMN)(\lambda xy.x)$

$=_\beta (xMN)[x:=\lambda xy.x]$ i)

Recall that PQR abbreviates (PQ)R

$= ((\lambda xy.x)M)N$

Recall that $\lambda xy.P$ abbrev. $\lambda x(\lambda y.P)$

$=_\beta ((\lambda y.x)[x:=M])N$ i)

$= (\lambda y.M)N$

$=_\beta M[y:=N]$ i)

$= M$

can assume y does not occur in M (otherwise, replace y in π_1 with a variable not in M)

Similarly, $\pi_2\langle M,N\rangle =_\beta N$.

We can also encode ~~truth~~ "true", ~~and~~ "false" ~~these~~ and "if - then - else -" (see p19).

② Recursive Functions

Definition:

The initial functions are zero, successor and projections

$O: \mathbb{N} \to \mathbb{N}$, $a \mapsto 0$ $\quad S: \mathbb{N} \to \mathbb{N}$, $a \mapsto a+1$ $\quad p_k^m: \mathbb{N}^m \to \mathbb{N}$, $(a_1,\dots,a_m) \mapsto a_k$

Note that I define $\mathbb{N} = \{0,1,2,\dots\}$.

Definition:

Let $g: \mathbb{N}^m \to \mathbb{N}$, $h: \mathbb{N}^{m+2} \to \mathbb{N}$ be partial functions. The partial function $f: \mathbb{N}^{m+1} \to \mathbb{N}$ is defined from g and h by ~~recu~~ primitive recursion if, for all $a_1,\dots,a_m, n \in \mathbb{N}$,

$f(0,a_1,\dots,a_m) = g(a_1,\dots,a_m)$

$f(n+1,a_1,\dots,a_m) = h(f(n,a_1,\dots,a_m),n,a_1,\dots,a_m)$

It is understood that the left-hand side of ~~eq~~ each equation is defined if and only if the right-hand side is.

Definition:

Let $g: \mathbb{N}^{m+1} \to \mathbb{N}$ be a partial function. The partial function $f: \mathbb{N}^m \to \mathbb{N}$ is defined from g by minimization if

$$f(a_1,\dots,a_m)=\begin{cases} n & \text{if } n \text{ is the least } n\in\mathbb{N} \text{ such that } g(a_1,\dots,a_m,n)=0 \text{ and } g(a_1,\dots,a_m,n')\neq 0 \text{ for } n'<n \\ \text{undefined} & g(a_1,\dots,a_m,n)=0 \text{ and } g(a_1,\dots,a_m,n')\neq 0 \text{ for all } n'=0,\dots,n-1 \\ \text{undefined} & \text{if there is no such } n. \end{cases}$$

Definition:
The class of partial recursive functions is the smallest class of partial functions containing the initial functions and is closed under composition, primitive recursion and minimisation.
The class of primitive recursive functions is the smallest class of (total) functions containing the initial functions and is closed under composition and primitive recursion.

Here, composition has the usual meaning.

③ Church Numerals and λ-Definability

To encode partial recursive functions in the λ-calculus, we first need to be able to encode numbers.

For variables f, x, $f^n(x)$ abbreviates $\underbrace{f(f(\dots(f}_{n \text{ times}}x)\dots))$

** **Definition:** (Another of the key definitions)
The n^{th} Church numeral is $c_n = \underline{n} = \lambda f x.\, f^n(x)$

Examples:

$$\underline{0} = \lambda f x.\, x$$
$$\underline{1} = \lambda f x.\, f x$$
$$\underline{2} = \lambda f x.\, f(f x)$$

Definition:
A partial function $f: \mathbb{N}^m \to \mathbb{N}$ is **λ-definable** if there is a λ-term F such that
i) If $f(a_1, \dots, a_m) = n$ then $F\underline{a_1} \dots \underline{a_m} =_\beta \underline{n}$
ii) If $f(a_1, \dots, a_m)$ is undefined then $F\underline{a_1} \dots \underline{a_m}$ has no normal form.
The λ-term F **defines** f.

We don't need the definition of normal form for today, because all of our functions will be total. See p10 for the definition.

Examples:

1) $\underline{\text{Zero}} = \lambda m.\, \underline{0}$ defines $0 : a \mapsto 0$

$$\begin{aligned} \underline{\text{Zero}}\,\underline{a} &= (\lambda m.\, \underline{0})\,\underline{a} \\ &=_\beta \underline{0}[m := \underline{a}] \\ &= \underline{0} \end{aligned}$$

because m doesn't occur in $\underline{0}$.

2) $\underline{P_2^3} = \lambda x_1 x_2 x_3.\, x_2$ defines $p_2^3 : \mathbb{N}^3 \to \mathbb{N}$, $(a_1, a_2, a_3) \mapsto a_2$

$$\begin{aligned} P_2^3\,\underline{a_1}\,\underline{a_2}\,\underline{a_3} &= (\lambda x_1 x_2 x_3.\, x_2)\,\underline{a_1}\,\underline{a_2}\,\underline{a_3} \\ &=_\beta ((\lambda x_2 x_3.\, x_2)[x_1 := \underline{a_1}])\,\underline{a_2}\,\underline{a_3} \\ &= (\lambda x_2 x_3.\, x_2)\,\underline{a_2}\,\underline{a_3} \\ &=_\beta ((\lambda x_3.\, x_2)[x_2 := \underline{a_2}])\,\underline{a_3} \\ &= (\lambda x_3.\, \underline{a_2})\,\underline{a_3} \\ &=_\beta \underline{a_2} \end{aligned}$$

because x_3 doesn't occur in $\underline{a_2}$

3) $\underline{P_k^m} = \lambda x_1 \dots x_m.\, x_k$ defines $p_k^m : \mathbb{N}^m \to \mathbb{N}$, where $k \leq m$.

4) $\underline{\text{succ}} = \lambda n f x.\, f(n f x)$ defines the successor function.

$$\begin{aligned} \underline{\text{succ}}\,\underline{a} &= (\lambda n f x.\, f(n f x))\,\underline{a} \\ &=_\beta \lambda f x.\, f(\underline{a}\, f x) \\ &= \lambda f x.\, f(\underbrace{(\lambda g y.\, g(g(\dots(g y)\dots)))}_{a \text{ times}} f x) \end{aligned}$$

$=_\beta \lambda fx.f((\lambda y.f(f(\ldots(fy)\ldots)))x)$

$= \lambda fx.\underbrace{f(f(\ldots(fx)\ldots))}_{a+1 \text{ times}}$

$= \lambda fx.f^{a+1}(x)$

$= \underline{a+1}$.

5) $\underline{add} = \lambda mnfx.mf(nfx)$ defines addition

$(\lambda mnfx.mf(nfx))\,\underline{a}\,\underline{b}$

$=_\beta (\lambda nfx.\underline{a}f(nfx))\underline{b}$

$=_\beta \lambda fx.\underline{a}f(\underline{b}fx)$

$=_\beta \lambda fx.(\underline{a}f)(f^b(x))$ $\quad \underline{b}fx =_\beta f^b(x)$ as in succ

$= \lambda fx.((\lambda gy.g^a(y))f)(f^b(x))$

$=_\beta \lambda fx.((\lambda y.f^a(y))(f^b(x)))$

~~$=_\beta \lambda fx.$~~

$=_\beta \lambda fx.f^a(f^b(x))$

$= \lambda fx.f^{a+b}(x)$

$= \underline{a+b}$

④ ~~Pro~~ Proposition 1.7.4 and Theorem 1.7.5

Now we prove that all partial recursive functions are λ-definable.

Proposition 1.7.4:

All primitive recursive functions are λ-definable.

Proof:

Induce on the construction of the functions. We've seen that all of the initial functions are λ-definable.

Composition: Idea: Let $f:\mathbb{N}^2\to\mathbb{N}$ and $g_1,g_2:\mathbb{N}\to\mathbb{N}$ be (total) functions. Suppose F defines f and G_i defines g_i.

Claim: $\lambda x_1.F(G_1x_1)(G_2x_1)$ defines the composite of f with g_1 and g_2.

$(\lambda x.F(G_1x)(G_2x))\underline{a} =_\beta F(G_1\underline{a})(G_2\underline{a})$

$=_\beta F\ \underline{g_1(a)}\ \underline{g_2(a)}$ by hypothesis on G_i

$=_\beta \underline{f(g_1(a), g_2(a))}$ by hypothesis on F.

Primitive recursion:

Suppose $f: \mathbb{N}^{m+1} \to \mathbb{N}$ is defined from $g: \mathbb{N}^m \to \mathbb{N}$, $h: \mathbb{N}^{m+2} \to \mathbb{N}$ by primitive recursion, by the equations

$$f(0, a_1, \ldots, a_m) = g(a_1, \ldots, a_m)$$
$$f(n+1, a_1, \ldots, a_m) = h(f(n, a_1, \ldots, a_m), n, a_1, \ldots, a_m)$$

Suppose G, H define g, h respectively. Let

$\underline{Init} = \langle \underline{0}, G x_1 \ldots x_m \rangle$

$\underline{Step} = \lambda p. \langle \underline{succ}\ (\pi_1 p), H (\pi_2 p)(\pi_1 p) x_1 \ldots x_m \rangle$

Claim: $F = \lambda x\, x_1 \ldots x_m . \pi_2 (x\ \underline{Step}\ \underline{Init})$ defines f.

Induce on the first argument of F.

$F\,\underline{0} =_\beta \lambda x_1 \ldots x_m . \pi_2 (\underline{0}\ \underline{Step}\ \underline{Init})$

$= \lambda x_1 \ldots x_m . \pi_2 ((\lambda f x . x)\ \underline{Step}\ \underline{Init})$

$=_\beta \lambda x_1 \ldots x_m . \pi_2 ((\lambda x . x)\ \underline{Init})$

$=_\beta \lambda x_1 \ldots x_m . \pi_2\ \underline{Init}$

$=_\beta \lambda x_1 \ldots x_m . G x_1 \ldots x_m$

Hence $F\,\underline{0}\,\underline{a_1} \ldots \underline{a_m} =_\beta (\lambda x_1 \ldots x_m . G x_1 \ldots x_m)\,\underline{a_1} \ldots \underline{a_m}$

$=_\beta G\,\underline{a_1} \ldots \underline{a_m}$

$=_\beta \underline{g(a_1, \ldots, a_m)}$

$= \underline{f(0, a_1, \ldots, a_m)}$

Suppose $F\,\underline{n}\,\underline{a_1} \ldots \underline{a_m} =_\beta f(n, a_1, \ldots, a_m)$

$F\,\underline{n+1} =_\beta \lambda x_1 \ldots x_m . \pi_2 (\underline{n+1}\ \underline{Step}\ \underline{Init})$

$=_\beta \lambda x_1 \ldots x_m . \pi_2 (\underline{succ}\ \underline{n}\ \underline{Step}\ \underline{Init})$ because $\underline{succ}\ \underline{n} =_\beta \underline{n+1}$

$= \lambda x_1 \ldots x_m . \pi_2 ((\lambda y f x . f (y f x))\ \underline{n}\ \underline{Step}\ \underline{Init})$

$=_\beta \lambda x_1 \ldots x_m . \pi_2 ((\lambda f x . f (\underline{n} f x))\ \underline{Step}\ \underline{Init})$

$=_\beta \lambda x_1 \ldots x_m . \pi_2 ((\lambda x . \underline{Step}\ (\underline{n}\ \underline{Step}\ x))\ \underline{Init})$

$=_\beta \lambda x_1 \ldots x_m . \pi_2 (\underline{Step}\ (\underline{n}\ \underline{Step}\ \underline{Init}))$

Now, by definition, $\pi_2 (\underline{Step}\ (\underline{n}\ \underline{Step}\ \underline{Init}))$

$=_\beta \pi_2 \langle \underline{succ}\ (\pi_1 (\underline{n}\ \underline{Step}\ \underline{Init})), H (\pi_2 (\underline{n}\ \underline{Step}\ \underline{Init}))\ (\pi_1 (\underline{n}\ \underline{Step}\ \underline{Init}))\ x_1 \ldots x_m \rangle$

$=_\beta H(\pi_2(n\ \underline{Step}\ \underline{Init}))(\pi_1(n\ \underline{Step}\ \underline{Init}))x_1 \dots x_m$

Observe that $\underline{succ}(\pi_1\ \underline{Init}) =_\beta \underline{succ}\ \underline{0} =_\beta \underline{1}$, so

$\underline{Step}\ \underline{Init} =_\beta \langle \underline{1}, \text{some } \lambda\text{-term} \rangle$

Think of $n\ \underline{Step}\ \underline{Init}$ as applying $\underline{Step}$ to $\underline{Init}$ n times, so $\pi_1(n\ \underline{Step}\ \underline{Init}) =_\beta \underline{n}$ (can prove this by induction on n).

Hence $\underline{Fn+1}\ \underline{a_1} \dots \underline{a_m} =_\beta (\lambda x_1 \dots x_m . H(\pi_2(n\ \underline{Step}\ \underline{Init}))\ \underline{n}\ x_1 \dots x_m)\underline{a_1} \dots \underline{a_m}$

$=_\beta H(\pi_2(n\ \underline{Step}|_{x_i := a_i}\ \underline{Init}|_{x_i := a_i}))\ \underline{n}\ \underline{a_1} \dots \underline{a_m}$

Here $\underline{Step}|_{x_i := a_i}$ denotes $\underline{Step}$ with every occurrence of x_i replaced with $\underline{a_i}$, and similarly for $\underline{Init}|_{x_i := a_i}$.

Now, by the inductive hypothesis,

$\underline{f(n, a_1, \dots a_m)} =_\beta \underline{F\ n\ a_1 \dots a_m}$

$= (\lambda x x_1 \dots x_m . \pi_2(x\ \underline{Step}\ \underline{Init}))\ \underline{n}\ \underline{a_1} \dots \underline{a_m}$

$=_\beta \pi_2(n\ \underline{Step}|_{x_i := a_i}\ \underline{Init}|_{x_i := a_i})$

Hence $\underline{Fn+1}\ \underline{a_1} \dots \underline{a_m} =_\beta H\underline{f(n, a_1, \dots, a_m)}\ \underline{n}\ \underline{a_1} \dots \underline{a_m}$

$=_\beta \underline{h(f(n, a_1, \dots, a_m), n, a_1, \dots, a_m)}$

$= \underline{f(n+1, a_1, \dots, a_m)}$

□

Theorem 1.7.5:

All partial recursive functions are λ-definable.

Proof:

See L.C.H.I p 21.

□