Lambda-Definable Functions

Encoding data in λ-calculus

Computation in λ-calculus is given by β-reduction. To relate this to register/Turing-machine computation, or to partial recursive functions, we first have to see how to encode numbers, pairs, lists, … as λ-terms.

We will use the original encoding of numbers due to Church…

Church's numerals

Notation: $$\begin{cases} M^0 N & \triangleq N \\ M^1 N & \triangleq M\,N \\ M^{n+1} N & \triangleq M(M^n N) \end{cases}$$

Church's numerals

$$\begin{aligned}
\underline{0} &\triangleq \lambda f\, x.x \\
\underline{1} &\triangleq \lambda f\, x.f\, x \\
\underline{2} &\triangleq \lambda f\, x.f(f\, x) \\
&\vdots \\
\underline{n} &\triangleq \lambda f\, x.\underbrace{f(\cdots(f}_{n \text{ times}}\, x)\cdots)
\end{aligned}$$

Notation: $\begin{cases} M^0 N \triangleq N \\ M^1 N \triangleq M\, N \\ M^{n+1} N \triangleq M(M^n N) \end{cases}$

so we can write $\underline{n}$ as $\lambda f\, x.f^n x$ and we have $\boxed{\underline{n}\, M\, N =_\beta M^n\, N}$.

Church's numerals

$$\begin{aligned} \underline{0} &\triangleq \lambda f\,x.x \\ \underline{1} &\triangleq \lambda f\,x.f\,x \\ \underline{2} &\triangleq \lambda f\,x.\boxed{f(f\,x)} \\ &\vdots \\ \underline{n} &\triangleq \lambda f\,x.\underbrace{f(\cdots(f}_{n \text{ times}}\,x)\cdots) \end{aligned}$$

N.B. not ffx, which stands for $(ff)x$

Notation: $\begin{cases} M^0 N & \triangleq N \\ M^1 N & \triangleq M\,N \\ M^{n+1} N & \triangleq M(M^n N) \end{cases}$

so we can write $\underline{n}$ as $\lambda f\,x.f^n x$ and we have $\boxed{\underline{n}\,M\,N =_\beta M^n\,N}$.

λ-Definable functions

Definition. $f \in \mathbb{N}^n \rightharpoonup \mathbb{N}$ is λ-definable if there is a closed λ-term F that represents it: for all $(x_1, \ldots, x_n) \in \mathbb{N}^n$ and $y \in \mathbb{N}$

- if $f(x_1, \ldots, x_n) = y$, then $F\,\underline{x_1} \cdots \underline{x_n} =_\beta \underline{y}$
- if $f(x_1, \ldots, x_n)\uparrow$, then $F\,\underline{x_1} \cdots \underline{x_n}$ has no β-nf.

For example, addition is λ-definable because it is represented by $P \triangleq \lambda x_1\, x_2.\lambda f\, x.\, x_1\, f\,(x_2\, f\, x)$:

$$\begin{aligned} P\,\underline{m}\,\underline{n} &=_\beta \lambda f\, x.\, \underline{m}\, f\,(\underline{n}\, f\, x) \\ &=_\beta \lambda f\, x.\, \underline{m}\, f\,(f^n x) \\ &=_\beta \lambda f\, x.\, f^m(f^n x) \\ &= \lambda f\, x. f^{m+n} x \\ &= \underline{m+n} \end{aligned}$$

λ-Definable functions

Definition. $f \in \mathbb{N}^n \rightharpoonup \mathbb{N}$ is λ-definable if there is a closed λ-term F that represents it: for all $(x_1, \ldots, x_n) \in \mathbb{N}^n$ and $y \in \mathbb{N}$

- if $f(x_1, \ldots, x_n) = y$, then $F\,\underline{x_1} \cdots \underline{x_n} =_\beta \underline{y}$
- if $f(x_1, \ldots, x_n)\uparrow$, then $F\,\underline{x_1} \cdots \underline{x_n}$ has no β-nf.

For example, addition is λ-definable because it is represented by $P \triangleq \lambda x_1\,x_2.\lambda f\,x.\,x_1\,f\,(x_2\,f\,x)$:

$$\begin{aligned}
P\,\underline{m}\,\underline{n} &=_\beta \lambda f\,x.\,\underline{m}\,f\,(\underline{n}\,f\,x) \\
&=_\beta \lambda f\,x.\,\underline{m}\,f\,(f^n x) \\
&=_\beta \lambda f\,x.\,f^m(f^n x) \\
&\boxed{=} \lambda f\,x.f^{m+n}x \\
&= \underline{m+n}
\end{aligned}$$

can
prove
this equality
by induction
on n

Computable = λ-definable

Theorem. A partial function is computable if and only if it is λ-definable.

We already know that

$$\begin{array}{ll} & \text{Register Machine computable} \\ = & \text{Turing computable} \\ = & \text{partial recursive.} \end{array}$$

Using this, we break the theorem into two parts:

- every partial recursive function is λ-definable
- λ-definable functions are RM computable

λ-Definable functions

Definition. $f \in \mathbb{N}^n \rightharpoonup \mathbb{N}$ is λ-definable if there is a closed λ-term F that represents it: for all $(x_1, \ldots, x_n) \in \mathbb{N}^n$ and $y \in \mathbb{N}$

- if $f(x_1, \ldots, x_n) = y$, then $F\, \underline{x_1} \cdots \underline{x_n} =_\beta \underline{y}$
- if $f(x_1, \ldots, x_n)\uparrow$, then $F\, \underline{x_1} \cdots \underline{x_n}$ has no β-nf.

This condition can make it quite tricky to find a λ-term representing a non-total function.

For now, we concentrate on total functions. First, let us see why the elements of **PRIM** (primitive recursive functions) are λ-definable.

Recall :

Basic functions

- Projection functions, $\mathtt{proj}_i^n \in \mathbb{N}^n \to \mathbb{N}$:

$$\mathtt{proj}_i^n(x_1, \ldots, x_n) \triangleq x_i$$

- Constant functions with value $\mathbf{0}$, $\mathtt{zero}^n \in \mathbb{N}^n \to \mathbb{N}$:

$$\mathtt{zero}^n(x_1, \ldots, x_n) \triangleq \mathbf{0}$$

- Successor function, $\mathtt{succ} \in \mathbb{N} \to \mathbb{N}$:

$$\mathtt{succ}(x) \triangleq x + \mathbf{1}$$

Basic functions are representable

- $\mathtt{proj}_i^n \in \mathbb{N}^n \to \mathbb{N}$ is represented by $\lambda x_1 \ldots x_n . x_i$
- $\mathtt{zero}^n \in \mathbb{N}^n \to \mathbb{N}$ is represented by $\lambda x_1 \ldots x_n . \underline{0}$
- $\mathtt{succ} \in \mathbb{N} \to \mathbb{N}$ is represented by

$$\mathbf{Succ} \triangleq \lambda x_1\, f\, x . f\,(x_1\, f\, x)$$

since

$$\begin{aligned} \mathbf{Succ}\, \underline{n} &=_\beta \lambda f\, x .\, f\,(\underline{n}\, f\, x) \\ &=_\beta \lambda f\, x .\, f\,(f^n\, x) \\ &= \lambda f\, x .\, f^{n+1}\, x \\ &= \underline{n+1} \end{aligned}$$

Basic functions are representable

- $\mathtt{proj}_i^n \in \mathbb{N}^n \to \mathbb{N}$ is represented by $\lambda x_1 \ldots x_n . x_i$
- $\mathtt{zero}^n \in \mathbb{N}^n \to \mathbb{N}$ is represented by $\lambda x_1 \ldots x_n . \underline{0}$
- $\mathtt{succ} \in \mathbb{N} \to \mathbb{N}$ is represented by

$$\mathbf{Succ} \triangleq \lambda x_1\, f\, x . f\,(x_1\, f\, x)$$

since

$$\begin{aligned} \mathbf{Succ}\, \underline{n} &=_\beta \lambda f\, x .\, f\,(\underline{n}\, f\, x) \\ &=_\beta \lambda f\, x .\, f\,(f^n\, x) \\ &= \lambda f\, x .\, f^{n+1}\, x \\ &= \underline{n+1} \end{aligned}$$

($\lambda x_1 f x . x_1 f (f x)$ also represents Succ)

Representing composition

If total function $f \in \mathbb{N}^n \to \mathbb{N}$ is represented by $\boldsymbol{F}$ and total functions $g_1, \ldots, g_n \in \mathbb{N}^m \to \mathbb{N}$ are represented by $G_1, \ldots, G_n$, then their composition $f \circ (g_1, \ldots, g_n) \in \mathbb{N}^m \to \mathbb{N}$ is represented simply by

$$\lambda x_1 \ldots x_m . F\,(G_1\,x_1 \ldots x_m) \ldots (G_n\,x_1 \ldots x_m)$$

because

$$\begin{aligned}
& F\,(G_1\,\underline{a_1} \ldots \underline{a_m}) \ldots (G_n\,\underline{a_1} \ldots \underline{a_m}) \qquad .\\
=_\beta\; & F\,\underline{g_1(a_1, \ldots, a_m)} \ldots \underline{g_n(a_1, \ldots, a_m)}\\
=_\beta\; & \underline{f(g_1(a_1, \ldots, a_m), \ldots, g_n(a_1, \ldots, a_m))}\\
=\; & \underline{f \circ (g_1, \ldots, g_n)(a_1, \ldots, a_m)}
\end{aligned}$$

Representing composition

If total function $f \in \mathbb{N}^n \to \mathbb{N}$ is represented by F and total functions $g_1, \ldots, g_n \in \mathbb{N}^m \to \mathbb{N}$ are represented by $G_1, \ldots, G_n$, then their composition $f \circ (g_1, \ldots, g_n) \in \mathbb{N}^m \to \mathbb{N}$ is represented simply by

$$\lambda x_1 \ldots x_m . F\,(G_1\, x_1 \ldots x_m) \ldots (G_n\, x_1 \ldots x_m)$$

This does not necessarily work for <u>partial</u> functions. E.g. totally undefined function $u \in \mathbb{N} \rightharpoonup \mathbb{N}$ is represented by $U \triangleq \lambda x_1 . \Omega$ (why?) and $\mathtt{zero}^1 \in \mathbb{N} \to \mathbb{N}$ is represented by $Z \triangleq \lambda x_1 . \underline{0}$; but $\mathtt{zero}^1 \circ u$ is not represented by $\lambda x_1 . Z(U\, x_1)$, because $(\mathtt{zero}^1 \circ u)(n)\uparrow$ whereas $(\lambda x_1 . Z(U\, x_1))\, \underline{n} =_\beta Z\, \Omega =_\beta \underline{0}$. (What is $\mathtt{zero}^1 \circ u$ represented by?)

(see Ex. 12)

Recall:

Primitive recursion

Theorem. Given $f \in \mathbb{N}^n \rightharpoonup \mathbb{N}$ and $g \in \mathbb{N}^{n+2} \rightharpoonup \mathbb{N}$, there is a unique $h \in \mathbb{N}^{n+1} \rightharpoonup \mathbb{N}$ satisfying

$$\begin{cases} h(\vec{x}, 0) & \equiv f(\vec{x}) \\ h(\vec{x}, x+1) & \equiv g(\vec{x}, x, h(\vec{x}, x)) \end{cases}$$

for all $\vec{x} \in \mathbb{N}^n$ and $x \in \mathbb{N}$.

We write $\rho^n(f, g)$ for h and call it the partial function defined by primitive recursion from f and g.

Representing primitive recursion

If $f \in \mathbb{N}^n \to \mathbb{N}$ is represented by a λ-term F and

$g \in \mathbb{N}^{n+2} \to \mathbb{N}$ is represented by a λ-term G,

we want to show λ-definability of the unique
$h \in \mathbb{N}^{n+1} \to \mathbb{N}$ satisfying

$$\begin{cases} h(\vec{a}, 0) & = f(\vec{a}) \\ h(\vec{a}, a+1) & = g(\vec{a}, a, h(\vec{a}, a)) \end{cases}$$

or equivalently

$$\begin{aligned} h(\vec{a}, a) = & \ \mathit{if}\ a = 0\ \mathit{then}\ f(\vec{a}) \\ & \ \mathit{else}\ g(\vec{a}, a-1, h(\vec{a}, a-1)) \end{aligned}$$

Representing primitive recursion

If $f \in \mathbb{N}^n \to \mathbb{N}$ is represented by a λ-term F and

$g \in \mathbb{N}^{n+2} \to \mathbb{N}$ is represented by a λ-term G,

we want to show λ-definability of the unique $h \in \mathbb{N}^{n+1} \to \mathbb{N}$ satisfying $\boxed{h = \Phi_{f,g}(h)}$

where $\Phi_{f,g} \in (\mathbb{N}^{n+1} \to \mathbb{N}) \to (\mathbb{N}^{n+1} \to \mathbb{N})$ is given by

$$\begin{aligned} \Phi_{f,g}(h)(\vec{a}, a) \triangleq\ & \mathit{if}\ a = 0\ \mathit{then}\ f(\vec{a}) \\ & \mathit{else}\ g(\vec{a}, a-1, h(\vec{a}, a-1)) \end{aligned}$$

Representing primitive recursion

If $f \in \mathbb{N}^n \to \mathbb{N}$ is represented by a λ-term F and

$g \in \mathbb{N}^{n+2} \to \mathbb{N}$ is represented by a λ-term G,

we want to show λ-definability of the unique
$h \in \mathbb{N}^{n+1} \to \mathbb{N}$ satisfying $\boxed{h = \Phi_{f,g}(h)}$

where $\Phi_{f,g} \in (\mathbb{N}^{n+1} \to \mathbb{N}) \to (\mathbb{N}^{n+1} \to \mathbb{N})$ is given by…

Strategy:

- show that $\Phi_{f,g}$ is λ-definable;
- show that we can solve fixed point equations $\boxed{X = M\,X}$ up to β-conversion in the λ-calculus.

Representing booleans

$$\begin{aligned}\textbf{True} &\triangleq \lambda x\,y.\,x\\ \textbf{False} &\triangleq \lambda x\,y.\,y\\ \textbf{If} &\triangleq \lambda f\,x\,y.\,f\,x\,y\end{aligned}$$

satisfy

- $\textbf{If True}\ M\ N =_\beta \textbf{True}\ M\ N =_\beta M$
- $\textbf{If False}\ M\ N =_\beta \textbf{False}\ M\ N =_\beta N$

Representing test-for-zero

$$\mathsf{Eq}_0 \triangleq \lambda x.\, x(\lambda y.\, \mathsf{False})\, \mathsf{True}$$

satisfies

- $$\begin{aligned} \mathsf{Eq}_0\, \underline{0} &=_\beta \underline{0}\,(\lambda y.\, \mathsf{False})\, \mathsf{True} \\ &=_\beta \mathsf{True} \end{aligned}$$
- $$\begin{aligned} \mathsf{Eq}_0\, \underline{n+1} &=_\beta \underline{n+1}\,(\lambda y.\, \mathsf{False})\, \mathsf{True} \\ &=_\beta (\lambda y.\, \mathsf{False})^{n+1}\, \mathsf{True} \\ &=_\beta (\lambda y.\, \mathsf{False})((\lambda y.\, \mathsf{False})^{n}\, \mathsf{True}) \\ &=_\beta \mathsf{False} \end{aligned}$$

Representing predecessor

Want λ-term **Pred** satisfying

$$\begin{aligned} \mathbf{Pred}\,\underline{n+1} &=_\beta \underline{n} \\ \mathbf{Pred}\,\underline{0} &=_\beta \underline{0} \end{aligned}$$

Have to show how to reduce the "$n+1$-iterator" $\underline{n+1}$ to the "n-iterator" $\underline{n}$.

Idea: given f, iterating the function

$$g_f : (x,y) \mapsto (f(x), x)$$

$n+1$ times starting from (x,x) gives the pair $(f^{n+1}(x), f^n(x))$. So we can get $f^n(x)$ from $f^{n+1}(x)$ *parametrically in* f *and* x, by building g_f from f, iterating $n+1$ times from (x,x) and then taking the second component.

Hence. . .

Representing ordered pairs

$$\begin{aligned} \mathbf{Pair} &\triangleq \lambda x\, y\, f.\, f\, x\, y \\ \mathbf{Fst} &\triangleq \lambda f.\, f\, \mathbf{True} \\ \mathbf{Snd} &\triangleq \lambda f.\, f\, \mathbf{False} \end{aligned}$$

satisfy

- $$\begin{aligned} \mathbf{Fst}(\mathbf{Pair}\, M\, N) &=_\beta \mathbf{Fst}(\lambda f.\, f\, M\, N) \\ &=_\beta (\lambda f.\, f\, M\, N)\, \mathbf{True} \\ &=_\beta \mathbf{True}\, M\, N \\ &=_\beta M \end{aligned}$$
- $\mathbf{Snd}(\mathbf{Pair}\, M\, N) =_\beta \cdots =_\beta N$

Representing predecessor

Want $\boldsymbol{\lambda}$-term **Pred** satisfying

$$\begin{array}{lcl} \mathbf{Pred}\,\underline{n+1} & =_\beta & \underline{n} \\ \mathbf{Pred}\,\underline{0} & =_\beta & \underline{0} \end{array}$$

$$\mathbf{Pred} \triangleq \lambda y\, f\, x.\, \mathbf{Snd}(y\,(G\,f)\,(\mathbf{Pair}\,x\,x))$$

where

$$G \triangleq \lambda f\, p.\, \mathbf{Pair}(f\,(\mathbf{Fst}\,p))\,(\mathbf{Fst}\,p)$$

has the required $\boldsymbol{\beta}$-reduction properties.

Show

$$(\forall n \in \mathbb{N})\ \underline{n+1}\,(Gf)(\mathrm{Pair}\ x\,x) =_\beta \mathrm{Pair}\,(\underline{n+1}\ f x)(\underline{n}\ f x)$$

by induction on $n \in \mathbb{N}$:

Base case $n = 0$:

$$\begin{aligned} \underline{1}\,(Gf)(\mathrm{Pair}\ x x) &=_\beta Gf\,(\mathrm{Pair}\ x x) \\ &=_\beta \mathrm{Pair}\ (fx)\ x \\ &=_\beta \mathrm{Pair}\,(\underline{1}\ f x)(\underline{0}\ f x) \end{aligned}$$

✓

Show

$$(\forall n \in \mathbb{N})\ \underline{n+1}\,(Gf)(\mathrm{Pair}\,x\,x) =_\beta \mathrm{Pair}\,(\underline{n+1}\,f\,x)(\underline{n}\,f\,x)$$

by induction on $n \in \mathbb{N}$:

Induction step :

$$\underline{n+2}\,(Gf)(\mathrm{Pair}\,x\,x) =_\beta (Gf)\,\underline{n+1}\,(Gf)(\mathrm{Pair}\,x\,x)$$

by ind.hyp.

$$=_\beta (Gf)\,\mathrm{Pair}\,(\underline{n+1}\,f\,x)(\underline{n}\,f\,x)$$

Show

$$(\forall n \in \mathbb{N})\ \underline{n+1}\,(Gf)(\mathrm{Pair}\,x\,x) =_\beta \mathrm{Pair}\,(\underline{n+1}\,f\,x)(\underline{n}\,f\,x)$$

by induction on $n \in \mathbb{N}$:

Induction step:

$$\underline{n+2}\,(Gf)(\mathrm{Pair}\,x\,x) =_\beta (Gf)\,\underline{n+1}\,(Gf)(\mathrm{Pair}\,x\,x)$$

by ind. hyp.

$$=_\beta (Gf)\,\mathrm{Pair}\,(\underline{n+1}\,f\,x)(\underline{n}\,f\,x)$$

$$=_\beta \mathrm{Pair}\,(f(\underline{n+1}\,f\,x))(\underline{n+1}\,f\,x)$$

$$=_\beta \mathrm{Pair}\,(\underline{n+2}\,f\,x)(\underline{n+1}\,f\,x) \quad \checkmark$$

Show

$$(\forall n \in \mathbb{N})\ \underline{n+1}\,(Gf)(\mathrm{Pair}\,x\,x) =_\beta \mathrm{Pair}\,(\underline{n+1}\,f\,x)(\underline{n}\,f\,x)$$

So

$$\begin{aligned}\mathrm{Pred}\,\underline{n+1} &=_\beta \lambda f x.\,\mathrm{Snd}\big(\underline{n+1}\,(Gf)(\mathrm{Pair}\,x\,x)\big)\\ &=_\beta \lambda f x.\,\mathrm{Snd}\big(\mathrm{Pair}\,(\underline{n+1}\,f\,x)(\underline{n}\,f\,x)\big)\end{aligned}$$

$$\begin{aligned}
\mathrm{Pred}\ \underline{n+1} &=_\beta \lambda f x.\ \mathrm{Snd}(\underline{n+1}\,(G f)(\mathrm{Pair}\, x\, x)) \\
&=_\beta \lambda f x.\ \mathrm{Snd}\,(\mathrm{Pair}\,(\underline{n+1}\, f x)(\underline{n}\, f x)) \\
&=_\beta \lambda f x.\ \underline{n}\, f x \\
&=_\beta \lambda f x.\, f^n x \\
&= \underline{n}
\end{aligned}$$

Representing primitive recursion

If $f \in \mathbb{N}^n \to \mathbb{N}$ is represented by a λ-term F and

$g \in \mathbb{N}^{n+2} \to \mathbb{N}$ is represented by a λ-term G,

we want to show λ-definability of the unique $h \in \mathbb{N}^{n+1} \to \mathbb{N}$ satisfying $\boxed{h = \Phi_{f,g}(h)}$

where $\Phi_{f,g} \in (\mathbb{N}^{n+1} \to \mathbb{N}) \to (\mathbb{N}^{n+1} \to \mathbb{N})$ is given by

$$\begin{aligned} \Phi_{f,g}(h)(\vec{a}, a) \triangleq\ & \mathit{if}\ a = 0\ \mathit{then}\ f(\vec{a}) \\ & \mathit{else}\ g(\vec{a}, a-1, h(\vec{a}, a-1)) \end{aligned}$$

Representing primitive recursion

If $f \in \mathbb{N}^n \to \mathbb{N}$ is represented by a λ-term F and $g \in \mathbb{N}^{n+2} \to \mathbb{N}$ is represented by a λ-term G,

we want to show λ-definability of the unique $h \in \mathbb{N}^{n+1} \to \mathbb{N}$ satisfying $\boxed{h = \Phi_{f,g}(h)}$

where $\Phi_{f,g} \in (\mathbb{N}^{n+1} \to \mathbb{N}) \to (\mathbb{N}^{n+1} \to \mathbb{N})$ is given by…

Strategy:

- show that $\Phi_{f,g}$ is λ-definable;

$$\to \lambda z \vec{x} x . \mathrm{If}(\mathrm{Eq}_0 x)(F \vec{x})(G \vec{x} (\mathrm{Pred}\, x)(z \vec{x} (\mathrm{Pred}\, x)))$$