The λ-Calculus

Kian Dianati

March 2, 2022

Alonzo Church

What is the λ-calculus?

A way to formally express computable functions. Kinda like a Turing machine.

Unlike in a Turing machine, you can “express” functions in the λ-calculus.

So it's programming before programming existed.

It is an alternative to Zermelo-Fraenkel Set Theory for the foundations of mathematics.

How to λ-calculus?

- Expressions
- Functions
- Applying expressions

$$\lambda xy.(\lambda f.x(yf))$$

$$\lambda xy.(\lambda f.x(yf))$$

decleration $\rightarrow$ $\lambda xy.$

$$\lambda xy.(\lambda f.x(yf))$$

$$\begin{aligned} \text{decleration} &\rightarrow \lambda xy. \\ \text{expression} &\rightarrow \lambda f.x(yf) \end{aligned}$$

$$(\lambda xy.x)(\lambda xy.y)$$

$$(\lambda xy.x)(\lambda xy.y) =_\alpha (\lambda xy.x)(\lambda pq.q)$$

$$
\begin{aligned}
(\lambda xy.x)(\lambda xy.y) &=_\alpha (\lambda xy.x)(\lambda pq.q) \\
&\to_\beta (\lambda xy.x)[(\lambda pq.q)/x]
\end{aligned}
$$

$$
\begin{aligned}
(\lambda xy.x)(\lambda xy.y) &=_\alpha (\lambda xy.x)(\lambda pq.q) \\
&\rightarrow_\beta (\lambda xy.x)[(\lambda pq.q)/x] \\
&\rightarrow_\beta \lambda pq.q
\end{aligned}
$$

And we can also name expressions, for easier writing.

Logic

$$\mathbf{t} = \lambda xy.x$$
$$\mathbf{f} = \lambda xy.y$$

In **not**, we want the opposite of the input:

$$\mathbf{not} = \lambda a.a$$

In **not**, we want the opposite of the input:

$$\mathbf{not} = \lambda a.a\mathbf{f}$$

In **not**, we want the opposite of the input:

$$\mathbf{not} = \lambda a.a\mathbf{f}\,\mathbf{t}$$

$$\mathbf{and} = \lambda ab.ab\mathbf{f}$$

$$\textbf{and} = \lambda ab.ab\textbf{f}$$

$$\textbf{or} = \lambda ab.a\textbf{t}b$$

$$\textbf{and} = \lambda ab.ab\mathbf{f}$$

$$\textbf{or} = \lambda ab.a\mathbf{t}b$$

$$\textbf{if} = \lambda cte.cte$$

$$
\begin{aligned}
\textbf{and} &= \lambda ab.ab\mathbf{f} \\
\textbf{or} &= \lambda ab.a\mathbf{t}b \\
\textbf{if} &= \lambda cte.cte \\
&\rightarrow_\beta \lambda x.x
\end{aligned}
$$

```
(define true (lambda (x y) x))
(define false (lambda (x y) y))
(define and (lambda (a b) (a b false)))

(and true false)
#<procedure false>
```

Numbers

In the λ-calculus we encode numbers using higher order functions

$$\mathbf{0} = \lambda f x . x$$

$$\mathbf{0} = \lambda f x . x$$

$$\mathbf{1} = \lambda f x . f(x)$$

$$\mathbf{0} = \lambda fx.x$$

$$\mathbf{1} = \lambda fx.f(x)$$

$$\mathbf{2} = \lambda fx.f(f(x))$$

$$\mathbf{0} = \lambda fx.x$$

$$\mathbf{1} = \lambda fx.f(x)$$

$$\mathbf{2} = \lambda fx.f(f(x))$$

$$\mathbf{3} = \lambda fx.f(f(f(x)))$$

$$
\begin{aligned}
\mathbf{0} &= \lambda f x . x \\
\mathbf{1} &= \lambda f x . f(x) \\
\mathbf{2} &= \lambda f x . f(f(x)) \\
\mathbf{3} &= \lambda f x . f(f(f(x))) \\
&\vdots \\
\mathbf{n} &=_{\alpha} \lambda f x . f^{n}(x)
\end{aligned}
$$

To $m \times n$:

To $m \times n$:

$$\lambda fx.f^n(x)$$

To $m \times n$:

$$\lambda fx.f^n(x)$$

$$\downarrow$$

To $m \times n$:

$$\lambda fx.f^n(x)$$

$$\downarrow$$

$$\lambda fx.(f^m)^n(x)$$

nm

$$\mathbf{nm} =_\alpha (\lambda fx.f^n(x))(\lambda pq.p^m(q))$$

$$\mathbf{nm} =_\alpha (\lambda fx.f^n(x))(\lambda pq.p^m(q))$$
$$\to_\beta \lambda fx.f^n(x)[\lambda pq.p^m(q)/f]$$

$$\begin{aligned}
\mathbf{nm} &=_\alpha (\lambda fx.f^n(x))(\lambda pq.p^m(q)) \\
&\rightarrow_\beta \lambda fx.f^n(x)[\lambda pq.p^m(q)/f] \\
&\rightarrow_\beta \lambda x.(\lambda pq.p^m(q))^n(x)
\end{aligned}$$

$$\begin{aligned}\mathbf{nm} &=_\alpha (\lambda fx.f^n(x))(\lambda pq.p^m(q)) \\ &\to_\beta \lambda fx.f^n(x)[\lambda pq.p^m(q)/f] \\ &\to_\beta \lambda x.(\lambda pq.p^m(q))^n(x) \\ &\to_\beta \lambda xq.(x^m)^n(q)\end{aligned}$$

$$
\begin{aligned}
\mathbf{nm} &=_\alpha (\lambda fx.f^n(x))(\lambda pq.p^m(q)) \\
&\to_\beta \lambda fx.f^n(x)[\lambda pq.p^m(q)/f] \\
&\to_\beta \lambda x.(\lambda pq.p^m(q))^n(x) \\
&\to_\beta \lambda xq.(x^m)^n(q) \\
&\to_\beta \lambda xq.x^{nm}(q)
\end{aligned}
$$

Two interesting things to think about.

Let $\lambda x.\mathbf{not}(xx) = \mathbf{R}$

Let $\lambda x.\mathbf{not}(xx) = \mathbf{R}$

$$\mathbf{RR}$$

Let $\lambda x.\mathbf{not}(xx) = \mathbf{R}$

$$\mathbf{RR} \rightarrow_{\beta} \lambda x.\mathbf{not}(xx)[\mathbf{R}/\mathrm{x}]$$

Let $\lambda x.\mathbf{not}(xx) = \mathbf{R}$

$$\begin{aligned}\mathbf{RR} &\rightarrow_\beta \lambda x.\mathbf{not}(xx)[\mathbf{R}/\mathrm{x}] \\ &\rightarrow_\beta \mathbf{not}(\mathbf{RR})\end{aligned}$$

$$\mathbf{Y} = \lambda f x.(\lambda x.f(xx))(\lambda x.f(xx))$$

And so $\mathbf{Y}g \rightarrow_\beta g(g(g(\ldots)))$

Could this mean transfinite cardinals in the λ-calculus?

Thanks for listening!