Notes on Lambda Calculus

Foundations of Computer Science

Fall 2017

Terms. A term of the λ-calculus is either:

- a variable x, y, z, ...
- an abstraction $\langle x \to M \rangle$ where x is a variable and M a term
- an application $M\ N$ where M, N are terms

Examples: $x \quad \langle x \to x \rangle \quad \langle y \to \langle x \to x \rangle \rangle \quad \langle x \to x\ \langle y \to y \rangle \rangle$

Intuitively, $\langle x \to M \rangle$ represents a function with parameter x and returning M, while $M\ N$ represents an application of function M to argument N. The simplification rules below will enforce this interpretation.[1]

Just like elsewhere in mathematics, we will use parentheses freely to group terms together to affect or just clarify the order of applications. Application $M\ N$ is a binary operation that associates to the left, so that writing $M\ N\ P$ is the same as writing $(M\ N)\ P$. If you want $M\ (N\ P)$ (which means something different) then you need to use parentheses explicitly.

The *scope* of x in $\langle x \to M \rangle$ is all of M. An occurrence of a variable is said to be *bound* if it occurs in the scope of an abstraction with that variable as a parameter. More precisely, it is bound to the nearest enclosing abstraction. An occurrence of a variable is said to be *free* if it is not bound.

Examples: y is free in $\langle x \to y \rangle$; the first occurrence of x is free in $\langle y \to x\ \langle x \to x \rangle \rangle$ while the second is not; z is bound in $\langle z \to \langle x \to z \rangle \rangle$.

Bound variables can be renamed without affecting the meaning of term. Intuitively, $\langle x \to x \rangle$ and $\langle y \to y \rangle$ represent the same function, the identity function. That we happen to call the parameter x in the first and y in the second is pretty irrelevant. Two terms are α-equivalent when they are equal up to renaming of some bound variables. Thus, $\langle x \to x\ z \rangle$ and $\langle y \to y\ z \rangle$ are α-equivalent. Be careful that your renaming does not *capture* a free occurrence of a variable. For example, $\langle x \to x\ z \rangle$ and $\langle z \to z\ z \rangle$ are *not* α-equivalent. They represent different functions.

[1] The standard presentation of the λ-calculus uses notation $\lambda x.M$ for $\langle x \to M \rangle$, hence the name.

We will generally identify α-equivalent terms.

Substitution. An important operation is that of substituting a term N for a variable x inside another term M, written $M\{N/x\}$. It is defined formally as

$$x\{N/x\} = N$$

$$y\{N/x\} = y \quad \text{if } x \neq y$$

$$(M_1\ M_2)\{N/x\} = M_1\{N/x\}\ M_2\{N/x\}$$

$$\langle y \to M\rangle\{N/x\} = \langle y \to M\{N/x\}\rangle \quad \text{if } y \text{ is not free in } N$$

In the last case, if $x = y$ or if y *is* free in N, we can always find a term $\langle z \to M'\rangle$ that is α-equivalent to $\langle y \to M\rangle$ and such that $x \neq z$ and z is not free in N to perform the sustitution.

(Because we avoid capturing free variables, this form of substitution is called a *capture-avoiding substitution.*)

Simplification Rules. The main simplification rule is:

$$\langle x \to M\rangle\ N = M\{N/x\}$$

A term of the form $\langle x \to M\rangle\ N$ is called a *redex*.

Simplification can occur within the context of a larger term, of course, leading to the following three additional simplification rules:

$$M\ P = N\ P \quad \text{if } M = N$$

$$P\ M = P\ N \quad \text{if } M = N$$

$$\langle x \to M\rangle = \langle x \to N\rangle \quad \text{if } M = N$$

Examples:

$$\begin{aligned} \langle x \to x\rangle\ \langle y \to y\rangle &= x\{\langle y \to y\rangle/x\} \\ &= \langle y \to y\rangle \end{aligned}$$

$$(\langle x \to \langle y \to x\rangle\rangle\ z_1)\ z_2 = (\langle y \to x\rangle\{z_1/x\})\ z_2$$

$$
\begin{aligned}
&= \langle y \to z_1 \rangle \; z_2 \\
&= z_1\{z_2/y\} \\
&= z_1
\end{aligned}
$$

$$
\begin{aligned}
(\langle x \to \langle y \to y \rangle\rangle \; \langle z \to z \rangle) \; \langle x \to \langle y \to x \rangle\rangle &= \langle y \to y \rangle\{\langle z \to z \rangle/x\} \; \langle x \to \langle y \to x \rangle\rangle \\
&= \langle y \to y \rangle \; \langle x \to \langle y \to x \rangle\rangle \\
&= y\{\langle x \to \langle y \to x \rangle\rangle/y\} \\
&= \langle x \to \langle y \to x \rangle\rangle
\end{aligned}
$$

From now on, I will skip the explicit substitution step when showing simplifications.

A term is in *normal form* if it has no redex (and thus cannot be simplified any further).

Not every term can be simplified to a normal form:

$$
\begin{aligned}
\langle x \to x \; x \rangle \; \langle x \to x \; x \rangle &= \langle x \to x \; x \rangle \; \langle x \to x \; x \rangle \\
&= \langle x \to x \; x \rangle \; \langle x \to x \; x \rangle \\
&= \quad \ldots
\end{aligned}
$$

There can be more than one redex in a term, meaning that there may be more than one applicable simplification. For instance, in the term $(\langle x \to x \rangle \; \langle y \to x \rangle) \; (\langle x \to \langle y \to x \rangle\rangle \; z_1 \; z_2)$. A property of the λ-calculus is that all the ways to simplify a term down to a normal form yield the same normal form (up to renaming of bound variables). This is called the *Church-Rosser property*. It says that the order in which we perform simplifications to reach a normal form is not important.

In practice, one often imposes an order in which to apply simplifications to avoid nondeterminisn. The *normal-order strategy*, which always simplifies the leftmost and outermost redex, is guaranteed to find a normal form if one exists.

To simplify the presentation of more complex terms, we introduce a convenient abbreviation. We write

$$
\begin{aligned}
\langle x_1 \; x_2 \to M \rangle &= \langle x_1 \to \langle x_2 \to M \rangle\rangle \\
\langle x_1 \; x_2 \; x_3 \to M \rangle &= \langle x_1 \to \langle x_2 \to \langle x_3 \to M \rangle\rangle\rangle \\
\langle x_1 \; x_2 \; x_3 \; x_4 \to M \rangle &= \langle x_1 \to \langle x_2 \to \langle x_3 \to \langle x_4 \to M \rangle\rangle\rangle\rangle \\
&\vdots
\end{aligned}
$$

Working through the abbreviations, this means that we have simplifications:

$$\langle x_1 \; x_2 \to M \rangle \; N = \langle x_2 \to M\{N/x_1\} \rangle$$

$$\langle x_1\ x_2\ x_3 \to M\rangle\ N = \langle x_2\ x_3 \to M\{N/x_1\}\rangle$$
$$\vdots$$

Encoding Booleans. Even though the λ-calculus only has variables and functions, that's enough to encode all traditional data types.

Here's one way to encode Boolean values (due to Church):

$$\begin{aligned}\textbf{true} &= \langle x\ y \to x\rangle \\ \textbf{false} &= \langle x\ y \to y\rangle\end{aligned}$$

In what sense are these encodings of Boolean values? Booleans are useful because they allow you to select one branch or the other of a conditional expression.

$$\textbf{if} = \langle c\ x\ y \to c\ x\ y\rangle$$

The trick is that when B simplifies to either **true** or **false**, then **if** $B\ M\ N$ simplifies either to M or to N, respectively:

If $B = \textbf{true}$, then

$$\begin{aligned}\textbf{if}\ B\ M\ N =&\ \ B\ M\ N \\ =&\ \ \textbf{true}\ M\ N \\ =&\ \ \langle x\ y \to x\rangle\ M\ N \\ =&\ \ \langle y \to M\rangle\ N \\ =&\ \ M\end{aligned}$$

while if $B = \textbf{false}$, then

$$\begin{aligned}\textbf{if}\ B\ M\ N =&\ \ B\ M\ N \\ =&\ \ \textbf{false}\ M\ N \\ =&\ \ \langle x\ y \to y\rangle\ M\ N \\ =&\ \ \langle y \to y\rangle\ N \\ =&\ \ N\end{aligned}$$

Of course, these show that **if** is not strictly necessary. You should convince yourself that **true** $M\ N = M$ and that **false** $M\ N = N$.

We can define logical operators:

$$\begin{aligned}\textbf{and} &= \langle m\ n \to m\ n\ m\rangle \\ \textbf{or} &= \langle m\ n \to m\ m\ n\rangle \\ \textbf{not} &= \langle m \to \langle x\ y \to m\ y\ x\rangle\rangle\end{aligned}$$

Thus, for example:

$$\begin{aligned}
\textbf{and true false} &= \langle m\ n \to m\ n\ m\rangle\ \textbf{true false} \\
&= \langle n \to \textbf{true}\ n\ \textbf{true}\rangle\ \textbf{false} \\
&= \textbf{true false true} \\
&= \langle x\ y \to x\rangle\ \textbf{false true} \\
&= \langle y \to \textbf{false}\rangle\ \textbf{true} \\
&= \textbf{false}
\end{aligned}$$

$$\begin{aligned}
\textbf{not false} &= \langle m \to \langle x\ y \to m\ y\ x\rangle\rangle\ \textbf{false} \\
&= \langle x\ y \to \textbf{false}\ y\ x\rangle \\
&= \langle x\ y \to \langle u\ v \to v\rangle\ y\ x\rangle \\
&= \langle x\ y \to \langle v \to v\rangle\ x\rangle \\
&= \langle x\ y \to x\rangle \\
&= \textbf{true}
\end{aligned}$$

Encoding Natural Numbers. Here is an encoding of natural numbers, again due to Church (hence the name, Church numerals):

$$\begin{aligned}
\mathbf{0} &= \langle f\ x \to x\rangle \\
\mathbf{1} &= \langle f\ x \to f\ x\rangle \\
\mathbf{2} &= \langle f\ x \to f\ (f\ x)\rangle \\
\mathbf{3} &= \langle f\ x \to f\ (f\ (f\ x))\rangle \\
\mathbf{4} &= \quad \ldots
\end{aligned}$$

In general, natural number n is encoded as $\langle f\ x \to \underbrace{f\ (f\ (f\ (\ldots (f}_{n \text{ times}}\ x)))))\rangle$.

Successor operation:

$$\textbf{succ} = \langle n \to \langle f\ x \to n\ f\ (f\ x)\rangle\rangle$$

$$\begin{aligned}
\textbf{succ 1} &= \langle n \to \langle f\ x \to n\ f\ (f\ x)\rangle\rangle\ \langle f\ x \to f\ x\rangle \\
&= \langle f\ x \to \langle f\ x \to f\ x\rangle\ f\ (f\ x)\rangle \\
&= \langle f\ x \to \langle x \to f\ x\rangle\ (f\ x)\rangle \\
&= \langle f\ x \to f\ (f\ x)\rangle
\end{aligned}$$

$$= \mathbf{2}$$

Other operations:

$$\begin{aligned}
\mathbf{plus} &= \langle m\ n \to m\ \mathbf{succ}\ n \rangle \\
\mathbf{times} &= \langle m\ n \to \langle f\ x \to m\ (n\ f)\ x \rangle\rangle \\
\mathbf{iszero}? &= \langle n \to n\ \langle x \to \mathbf{false} \rangle\ \mathbf{true} \rangle
\end{aligned}$$

$$\begin{aligned}
\mathbf{plus\ 2\ 1} &= \langle m\ n \to m\ \mathbf{succ}\ n \rangle\ \mathbf{2\ 1} \\
&= \langle n \to \mathbf{2\ succ}\ n \rangle\ \mathbf{1} \\
&= \mathbf{2\ succ\ 1} \\
&= \langle f\ x \to f\ (f\ x) \rangle\ \mathbf{succ\ 1} \\
&= \langle x \to \mathbf{succ}\ (\mathbf{succ}\ x) \rangle\ \mathbf{1} \\
&= \mathbf{succ}\ (\mathbf{succ\ 1}) \\
&= \mathbf{succ}\ (\langle n \to \langle f\ x \to n\ f\ (f\ x) \rangle\rangle\ \mathbf{1}) \\
&= \mathbf{succ}\ \langle f\ x \to \mathbf{1}\ f\ (f\ x) \rangle \\
&= \mathbf{succ}\ \langle f\ x \to \langle f\ x \to f\ x \rangle\ f\ (f\ x) \rangle \\
&= \mathbf{succ}\ \langle f\ x \to \langle x \to f\ x \rangle\ (f\ x) \rangle \\
&= \mathbf{succ}\ \langle f\ x \to f\ (f\ x) \rangle \\
&= \langle n \to \langle f\ x \to n\ f\ (f\ x) \rangle\rangle\ \langle f\ x \to f\ (f\ x) \rangle \\
&= \langle f\ x \to \langle f\ x \to f\ (f\ x) \rangle\ f\ (f\ x) \rangle \\
&= \langle f\ x \to \langle x \to f\ (f\ x) \rangle\ (f\ x) \rangle \\
&= \langle f\ x \to f\ (f\ (f\ x)) \rangle \\
&= \mathbf{3}
\end{aligned}$$

$$\begin{aligned}
\mathbf{times\ 2\ 3} &= \langle m\ n \to \langle f\ x \to m\ (n\ f)\ x \rangle\rangle\ \mathbf{2\ 3} \\
&= \langle n \to \langle f\ x \to \mathbf{2}\ (n\ f)\ x \rangle\rangle\ \mathbf{3} \\
&= \langle f\ x \to \mathbf{2}\ (\mathbf{3}\ f)\ x \rangle \\
&= \langle f\ x \to \mathbf{2}\ (\langle f\ x \to f\ (f\ (f\ x)) \rangle\ f)\ x \rangle \\
&= \langle f\ x \to \mathbf{2}\ \langle x \to f\ (f\ (f\ x)) \rangle\ x \rangle \\
&= \langle f\ x \to \langle f\ x \to f\ (f\ x) \rangle\ \langle x \to f\ (f\ (f\ x)) \rangle\ x \rangle \\
&= \langle f\ x \to \langle x \to \langle x \to f\ (f\ (f\ x)) \rangle\ (\langle x \to f\ (f\ (f\ x)) \rangle\ x) \rangle\ x \rangle \\
&= \langle f\ x \to \langle x \to \langle x \to f\ (f\ (f\ x)) \rangle\ (f\ (f\ (f\ x))) \rangle\ x \rangle \\
&= \langle f\ x \to \langle x \to f\ (f\ (f\ (f\ (f\ (f\ x))))) \rangle\ x \rangle
\end{aligned}$$

$$
\begin{aligned}
&= \langle f\ x \to f\ (f\ (f\ (f\ (f\ (f\ x)))))\rangle \\
&= \mathbf{6}
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{iszero?}\ \mathbf{0} &= \langle n \to n\ \langle x \to \mathbf{false}\rangle\ \mathbf{true}\rangle\ \langle f\ x \to x\rangle \\
&= \langle f\ x \to x\rangle\ \langle x \to \mathbf{false}\rangle\ \mathbf{true} \\
&= \langle x \to x\rangle\ \mathbf{true} \\
&= \mathbf{true}
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{iszero?}\ \mathbf{2} &= \langle n \to n\ \langle x \to \mathbf{false}\rangle\ \mathbf{true}\rangle\ \langle f\ x \to f\ (f\ x)\rangle \\
&= \langle f\ x \to f\ (f\ x)\rangle\ \langle x \to \mathbf{false}\rangle\ \mathbf{true} \\
&= \langle x \to \langle x \to \mathbf{false}\rangle\ (\langle x \to \mathbf{false}\rangle\ x)\rangle\ \mathbf{true} \\
&= \langle x \to \langle x \to \mathbf{false}\rangle\ \mathbf{false}\rangle\ \mathbf{true} \\
&= \langle x \to \mathbf{false}\rangle\ \mathbf{true} \\
&= \mathbf{false}
\end{aligned}
$$

(An alternative way to define **times** is as $\langle m\ n \to m\ (\mathbf{plus}\ n)\ \mathbf{0}\rangle$. Check that $\mathbf{times}\ \mathbf{2}\ \mathbf{3} = \mathbf{6}$ with this definition.)

Defining a predecessor function is a bit more challenging. Predecessor takes a nonzero natural number n and returning $n-1$. There are several ways of defining such a function; here is probably the simplest:

$$
\mathbf{pred} = \langle n \to \langle f\ x \to n\ \langle g\ h \to h\ (g\ f)\rangle\ \langle u \to x\rangle\ \langle u \to u\rangle\rangle\rangle
$$

$$
\begin{aligned}
\mathbf{pred}\ \mathbf{2} &= \langle n \to \langle f\ x \to n\ \langle g\ h \to h\ (g\ f)\rangle\ \langle u \to x\rangle\ \langle u \to u\rangle\rangle\rangle\ \langle f\ x \to f\ (f\ x)\rangle \\
&= \langle f\ x \to \langle f\ x \to f\ (f\ x)\rangle\ \langle g\ h \to h\ (g\ f)\rangle\ \langle u \to x\rangle\ \langle u \to u\rangle\rangle \\
&= \langle f\ x \to \langle x \to \langle g\ h \to h\ (g\ f)\rangle\ (\langle g\ h \to h\ (g\ f)\rangle\ x)\rangle\ \langle u \to x\rangle\ \langle u \to u\rangle\rangle \\
&= \langle f\ x \to \langle g\ h \to h\ (g\ f)\rangle\ (\langle g\ h \to h\ (g\ f)\rangle\ \langle u \to x\rangle)\rangle\ \langle u \to u\rangle \\
&= \langle f\ x \to \langle g\ h \to h\ (g\ f)\rangle\ (\langle h \to h\ (\langle u \to x\rangle\ f)\rangle)\rangle\ \langle u \to u\rangle \\
&= \langle f\ x \to \langle g\ h \to h\ (g\ f)\rangle\ \langle h \to h\ x\rangle\rangle\ \langle u \to u\rangle \\
&= \langle f\ x \to \langle h \to h\ (\langle h \to h\ x\rangle\ f)\rangle\rangle\ \langle u \to u\rangle \\
&= \langle f\ x \to \langle h \to h\ (f\ x)\rangle\rangle\ \langle u \to u\rangle \\
&= \langle f\ x \to \langle u \to u\rangle\ (f\ x)\rangle \\
&= \langle f\ x \to f\ x\rangle
\end{aligned}
$$

$$= \mathbf{1}$$

Note that **pred 0** is just **0**:

$$\begin{aligned}
\mathbf{pred}\ \mathbf{0} = &\ \langle n \to \langle f\ x \to n\ \langle g\ h \to h\ (g\ f)\rangle\ \langle u \to x\rangle\ \langle u \to u\rangle\rangle\rangle\ \langle f\ x \to x\rangle \\
= &\ \langle f\ x \to \langle f\ x \to x\rangle\ \langle g\ h \to h\ (g\ f)\rangle\ \langle u \to x\rangle\ \langle u \to u\rangle\rangle \\
= &\ \langle f\ x \to \langle x \to x\rangle\ \langle u \to x\rangle\ \langle u \to u\rangle\rangle \\
= &\ \langle f\ x \to \langle u \to x\rangle\ \langle u \to u\rangle\rangle \\
= &\ \langle f\ x \to x\rangle \\
= &\ \mathbf{0}
\end{aligned}$$

Encoding pairs. A pair is just a packaging up of two terms in such a way that we can recover the two terms later on.

$$\begin{aligned}
\mathbf{pair} = &\ \langle x\ y \to \langle s \to s\ x\ y\rangle\rangle \\
\mathbf{first} = &\ \langle p \to p\ \langle x\ y \to x\rangle\rangle \\
\mathbf{second} = &\ \langle p \to p\ \langle x\ y \to y\rangle\rangle
\end{aligned}$$

It is easy to check that this works as advertised:

$$\begin{aligned}
\mathbf{first}\ (\mathbf{pair}\ a\ b) = &\ \langle p \to p\ \langle x\ y \to x\rangle\rangle\ (\langle x\ y \to \langle s \to s\ x\ y\rangle\rangle\ a\ b) \\
= &\ \langle p \to p\ \langle x\ y \to x\rangle\rangle\ (\langle y \to \langle s \to s\ a\ y\rangle\rangle\ b) \\
= &\ \langle p \to p\ \langle x\ y \to x\rangle\rangle\ \langle s \to s\ a\ b\rangle \\
= &\ \langle s \to s\ a\ b\rangle\ \langle x\ y \to x\rangle \\
= &\ \langle x\ y \to x\rangle\ a\ b \\
= &\ \langle y \to a\rangle\ b \\
= &\ a
\end{aligned}$$

$$\begin{aligned}
\mathbf{second}\ (\mathbf{pair}\ a\ b) = &\ \langle p \to p\ \langle x\ y \to y\rangle\rangle\ (\langle x\ y \to \langle s \to s\ x\ y\rangle\rangle\ a\ b) \\
= &\ \langle p \to p\ \langle x\ y \to y\rangle\rangle\ (\langle y \to \langle s \to s\ a\ y\rangle\rangle\ b) \\
= &\ \langle p \to p\ \langle x\ y \to y\rangle\rangle\ \langle s \to s\ a\ b\rangle \\
= &\ \langle s \to s\ a\ b\rangle\ \langle x\ y \to y\rangle \\
= &\ \langle x\ y \to y\rangle\ a\ b \\
= &\ \langle y \to y\rangle\ b \\
= &\ b
\end{aligned}$$

Recursion. With conditionals and basic data types, we are very close to having a Turing-complete programming language (that is, one that can simulate Turing machines). All that is missing is a way to do loops. It turns out we can write recursive functions in the λ-calculus, which is sufficient to give us loops.

Consider factorial. Intuitively, we would like to define **fact** by

$$\mathbf{fact} = \langle n \to (\mathbf{iszero?}\ n)\ \mathbf{1}\ (\mathbf{times}\ n\ (\mathbf{fact}\ (\mathbf{pred}\ n)))\rangle$$

but this is not a valid definition, since the right-hand side refers to the term being defined. It is really an *equation*, the same way $x = 3x$ is an equation. Consider that equation, $x = 3x$. Define $F(x) = 3x$. Then, a solution of $x = 3x$ is really a fixed-point of F, namely, a value x_0 for which $F(x_0) = x_0$. And F has only one fixed-point, namely $x_0 = 0$, which gives us the one solution to $x = 3x$, namely $x = 0$.

Similarly, if we define

$$F_{\mathit{fact}} = \langle f \to \langle n \to (\mathbf{iszero?}\ n)\ \mathbf{1}\ (\mathbf{times}\ n\ (f\ (\mathbf{pred}\ n)))\rangle\rangle$$

then we see that the definition that we're looking for is a fixed-point of F_{fact}, namely, a term $\mathbf{f}$ such that $F_{\mathit{fact}}\ \mathbf{f} = \mathbf{f}$. Indeed, if we have such a term, then:

$$\begin{aligned}
\mathbf{f}\ \mathbf{3} = {} & F_{\mathit{fact}}\ \mathbf{f}\ \mathbf{3} \\
= {} & \langle f \to \langle n \to (\mathbf{iszero?}\ n)\ \mathbf{1}\ (\mathbf{times}\ n\ (f\ (\mathbf{pred}\ n)))\rangle\rangle\ \mathbf{f}\ \mathbf{3} \\
= {} & \langle n \to (\mathbf{iszero?}\ n)\ \mathbf{1}\ (\mathbf{times}\ n\ (\mathbf{f}\ (\mathbf{pred}\ n)))\rangle\ \mathbf{3} \\
= {} & (\mathbf{iszero?}\ \mathbf{3})\ \mathbf{1}\ (\mathbf{times}\ \mathbf{3}\ (\mathbf{f}\ (\mathbf{pred}\ \mathbf{3})) \\
= {} & \mathbf{times}\ \mathbf{3}\ (\mathbf{f}\ (\mathbf{pred}\ \mathbf{3})) \\
= {} & \mathbf{times}\ \mathbf{3}\ (\mathbf{f}\ \mathbf{2}) \\
= {} & \mathbf{times}\ \mathbf{3}\ (F_{\mathit{fact}}\ \mathbf{f}\ \mathbf{2}) \\
= {} & \mathbf{times}\ \mathbf{3}\ (\mathbf{times}\ \mathbf{2}\ (\mathbf{f}\ \mathbf{1})) \\
= {} & \mathbf{times}\ \mathbf{3}\ (\mathbf{times}\ \mathbf{2}\ (F_{\mathit{fact}}\ \mathbf{f}\ \mathbf{1})) \\
= {} & \mathbf{times}\ \mathbf{3}\ (\mathbf{times}\ \mathbf{2}\ (\mathbf{times}\ \mathbf{1}\ (\mathbf{f}\ \mathbf{1}))) \\
= {} & \mathbf{times}\ \mathbf{3}\ (\mathbf{times}\ \mathbf{2}\ (\mathbf{times}\ \mathbf{1}\ (F_{\mathit{fact}}\ \mathbf{f}\ \mathbf{1}))) \\
= {} & \mathbf{times}\ \mathbf{3}\ (\mathbf{times}\ \mathbf{2}\ (\mathbf{times}\ \mathbf{1}\ \mathbf{1})) \\
= {} & \mathbf{6}
\end{aligned}$$

(I coalesced together quite a few simplification steps in the above, for the sake of space.)

Thus, what we need is a way to find fixed-points in the λ-calculus. The following function does just that, for *any* term of the λ-calculus:

$$Y = \langle f \to \langle x \to f\ (x\ x)\rangle\ \langle x \to f\ (x\ x)\rangle\rangle$$

YG gives us a fixed-point of G:

First, note that

$$\begin{aligned}
YG = \quad & \langle f \to \langle x \to f\ (x\ x)\rangle\ \langle x \to f\ (x\ x)\rangle\rangle\ G \\
= \quad & \langle x \to G\ (x\ x)\rangle\ \langle x \to G\ (x\ x)\rangle \\
= \quad & G\ (\langle x \to G\ (x\ x)\rangle\ \langle x \to G\ (x\ )\rangle) \\
= \quad & G\ (G\ (\langle x \to G\ (x\ x)\rangle\ \langle x \to G\ (x\ )\rangle))
\end{aligned}$$

and that

$$\begin{aligned}
G(YG) = \quad & G(\langle f \to \langle x \to f\ (x\ x)\rangle\ \langle x \to f\ (x\ x)\rangle\rangle\ G) \\
= \quad & G(\langle x \to G\ (x\ x)\rangle\ \langle x \to G\ (x\ x)\rangle) \\
= \quad & G\ (G\ (\langle x \to G\ (x\ x)\rangle\ \langle x \to G\ (x\ )\rangle))
\end{aligned}$$

Therefore, $G(YG) = YG$, showing that $\langle x \to G\ (x\ x)\rangle\ \langle x \to G\ (x\ x)\rangle$ is a fixed-point of G.

We can use Y to define our factorial function:

$$\mathbf{fact} = Y\ F_{\mathit{fact}}$$

By the above derivation, we know that

$$\begin{aligned}
&\mathbf{fact} = \langle x \to F_{\mathit{fact}}\ (x\ x)\rangle\ \langle x \to F_{\mathit{fact}}\ (x\ x)\rangle \\
&\mathbf{fact} = F_{\mathit{fact}}\ \mathbf{fact}
\end{aligned}$$

and thus:

$$\begin{aligned}
\mathbf{fact\ 3} = \quad & Y\ F_{\mathit{fact}}\ \mathbf{3} \\
= \quad & \langle f \to \langle x \to f\ (x\ x)\rangle\ \langle x \to f\ (x\ x)\rangle\rangle\ F_{\mathit{fact}}\ \mathbf{3} \\
= \quad & \langle x \to F_{\mathit{fact}}\ (x\ x)\rangle\ \langle x \to F_{\mathit{fact}}\ (x\ x)\rangle\ \mathbf{3} \\
= \quad & F_{\mathit{fact}}\ (\langle x \to F_{\mathit{fact}}\ (x\ x)\rangle\ \langle x \to F_{\mathit{fact}}\ (x\ x)\rangle)\ \mathbf{3} \\
= \quad & F_{\mathit{fact}}\ \mathbf{fact\ 3} \\
= \quad & \langle f \to \langle n \to (\mathbf{iszero?}\ n)\ \mathbf{1}\ (\mathbf{times}\ n\ (f\ (\mathbf{pred}\ n)))\rangle\rangle\ \mathbf{fact\ 3} \\
= \quad & \langle n \to (\mathbf{iszero?}\ n)\ \mathbf{1}\ (\mathbf{times}\ n\ (\mathbf{fact}\ (\mathbf{pred}\ n)))\rangle\ \mathbf{3} \\
= \quad & (\mathbf{iszero?}\ \mathbf{3})\ \mathbf{1}\ (\mathbf{times\ 3}\ (\mathbf{fact}\ (\mathbf{pred\ 3}))) \\
= \quad & \mathbf{times\ 3}\ (\mathbf{fact}\ (\mathbf{pred\ 3})) \\
= \quad & \mathbf{times\ 3}\ (\mathbf{fact\ 2}) \\
= \quad & \mathbf{times\ 3}\ (F_{\mathit{fact}}\ \mathbf{fact\ 2}) \\
= \quad & \mathbf{times\ 3}\ (\mathbf{times\ 2}\ (\mathbf{fact\ 1})) \\
= \quad & \mathbf{times\ 3}\ (\mathbf{times\ 2}\ (F_{\mathit{fact}}\ \mathbf{fact\ 1})) \\
= \quad & \mathbf{times\ 3}\ (\mathbf{times\ 2}\ (\mathbf{times\ 1}\ (\mathbf{fact\ 1})))
\end{aligned}$$

$$
\begin{aligned}
&= \textbf{times 3 } (\textbf{times 2 } (\textbf{times 1 } (F_{fact} \textbf{ fact 1}))) \\
&= \textbf{times 3 } (\textbf{times 2 } (\textbf{times 1 1})) \\
&= \textbf{6}
\end{aligned}
$$