Typed λ-calculus: Substitution and Equations

P. B. Levy
adapted for G53POP by T.Altenkirch

Universities of Birmingham and Nottingham

1 Renaming and Substitution

Suppose we have a term $\Gamma \vdash M : B$, and we want to turn it into a term in context Δ, by replacing the identifiers. For example, we're given the term

$$\mathtt{x} : \mathtt{int}, \mathtt{y} : \mathtt{bool}, \mathtt{z} : \mathtt{int} \vdash \mathtt{z}{+}\mathtt{case}\ \mathtt{y}\ \mathtt{of}\ \{\mathtt{true} \rightarrow\ \mathtt{x}{+}\mathtt{z} \mid \mathtt{false} \rightarrow\ \mathtt{x}{+}1\} : \mathtt{int}$$

and we want to change it to something in the context $\mathtt{u} : \mathtt{bool}, \mathtt{x} : \mathtt{int}, \mathtt{y} : \mathtt{bool}$.

1.1 Replacing Identifiers With Identifiers

One way is to replace identifiers in Γ with *identifiers* in Δ. A *renaming* from Γ to Δ (beware the direction here) is a function θ taking each identifier $\mathtt{x} : A$ in Γ to an identifier $\theta(\mathtt{x}) : A$ in Δ.

For example, using the above Γ and Δ, one renaming from Γ to Δ is

$$\begin{aligned} \mathtt{x} &\mapsto \mathtt{x} \\ \mathtt{y} &\mapsto \mathtt{u} \\ \mathtt{z} &\mapsto \mathtt{x} \end{aligned}$$

We write $\theta^* M$ for the result of changing all the free identifiers in M according to θ. In the above example, we obtain

$$\mathtt{u} : \mathtt{bool}, \mathtt{x} : \mathtt{int}, \mathtt{y} : \mathtt{bool} \vdash \mathtt{x}{+}\mathtt{case}\ \mathtt{u}\ \mathtt{of}\ \{\mathtt{true} \rightarrow\ \mathtt{x}{+}\mathtt{x} \mid \mathtt{false} \rightarrow \mathtt{x}{+}1\} : \mathtt{int}$$

Exercise 1. Apply to the term

$$\mathtt{x} : \mathtt{int} \to \mathtt{int}, \mathtt{y} : \mathtt{int} \vdash \mathtt{let}\ \mathtt{w} = 5\ \mathtt{in}\ (\mathtt{x}\,\mathtt{y}) + (\mathtt{x}\,\mathtt{w}) : \mathtt{int}$$

the renaming

$$\begin{aligned} \mathtt{x} &\mapsto \mathtt{y} \\ \mathtt{y} &\mapsto \mathtt{w} \end{aligned}$$

to obtain a term in context

$$\mathtt{w} : \mathtt{int}, \mathtt{y} : \mathtt{int} \to \mathtt{int}, \mathtt{z} : \mathtt{int}$$

1.2 Replacing Identifiers With Terms

The second example is called *substitution*, where we replace each identifier in Γ with a *term* in context Δ. A *substitution* from Γ to Δ is a function k taking each identifier $\mathtt{x} : A$ in Γ to a term $\Delta \vdash k(\mathtt{x}) : A$.

For example, using the above Γ and Δ, a substitution from Γ to Δ is

$$\begin{aligned} &\mathtt{x} \mapsto 3 + \mathtt{x} \\ &\mathtt{y} \mapsto \mathtt{u} \\ &\mathtt{z} \mapsto \mathtt{case}\ \mathtt{y}\ \mathtt{of}\ \{\mathtt{true} \to\ \mathtt{x} + 2 \mid \mathtt{false} \to\ \mathtt{x}\} \end{aligned}$$

We write k^*M for the result of replacing all the free identifiers in M according to k (avoiding capture, of course). In the above example, we obtain

$$\begin{aligned} &\mathtt{u} : \mathtt{bool}, \mathtt{x} : \mathtt{int}, \mathtt{y} : \mathtt{bool} \vdash \\ &\mathtt{case}\ \mathtt{y}\ \mathtt{of}\ \{\mathtt{true} \to\ \mathtt{x} + 2 \mid \mathtt{false} \to\ \mathtt{x}\} + \\ &\mathtt{case}\ \mathtt{u}\ \mathtt{of}\ \{\mathtt{true} \to\ (3 + \mathtt{x}) + \mathtt{case}\ \mathtt{y}\ \mathtt{of}\ \{\mathtt{true} \to\ \mathtt{x} + 2 \mid \mathtt{false} \to\ \mathtt{x}\} \\ &\quad \mid \mathtt{false} \to\ (3 + \mathtt{x}) + 1\} : \mathtt{int} \end{aligned}$$

Exercise 2. Apply to the term

$$\mathtt{x} : \mathtt{int} \to \mathtt{int}, \mathtt{y} : \mathtt{int} \vdash \mathtt{let}\ \mathtt{w} = 5\ \mathtt{in}\ (\mathtt{x}\,\mathtt{y}) + (\mathtt{x}\,\mathtt{w}) : \mathtt{int}$$

the substitution

$$\begin{aligned} \mathtt{x} &\mapsto \mathtt{y} \\ \mathtt{y} &\mapsto \mathtt{w} + 1 \end{aligned}$$

to obtain a term in context

$$\mathtt{w} : \mathtt{int}, \mathtt{y} : \mathtt{int} \to \mathtt{int}, \mathtt{z} : \mathtt{int}$$

1.3 Substitution Uses Renaming

It is clear that renaming is a special case of substitution. So why is it important to consider both? The reason appears when we wish to define k^*M by induction on M. Some of the inductive clauses are easy:

$$\begin{aligned} k^*3 &= 3 \\ k^*(M+N) &= k^*M + k^*N \\ k^*\mathtt{x} &= k(\mathtt{x}) \end{aligned}$$

But what about substituting into a `let` expression? Let's first remember the typing rule for `let` :

$$\frac{\Gamma \vdash M : A \quad \Gamma, \mathtt{x} : A \vdash N : B}{\Gamma \vdash \mathtt{let\ x} = M \ \mathtt{in}\ N : B}$$

(I'm going to assume that x doesn't appear in Γ or Δ. Otherwise, you can α-convert it to something else.)

We want to define

$$k^*(\mathtt{let\ x} = M \ \mathtt{in}\ N) = \mathtt{let\ x\ in}\ k^*M \ \mathtt{in}\ (k, \mathtt{x} : A)^*N$$

where the substitution $\Gamma, \mathtt{x} : A \xrightarrow{k,\mathtt{x}:A} \Delta, \mathtt{x} : A$ is ... what? Remember that it has to map each identifier in $\Gamma, \mathtt{x} : A$ to a term (of the same type) in context $\Delta, \mathtt{x} : A$. Clearly it maps x to x. And it maps $(\mathtt{y} : B) \in \Gamma$ to $k(\mathtt{y})$—which is in context Δ—renamed along the renaming from Δ to $\Delta, \mathtt{x} : A$.

So we have to define renaming before we can define $k, \mathtt{x} : A$, and we have to define $k, \mathtt{x} : A$ before we can define substitution.

How do we define renaming inductively? Again, some of the inductive clauses are easy:

$$\begin{aligned} \theta^*3 &= 3 \\ \theta^*(M+N) &= \theta^*M + \theta^*N \\ \theta^*\mathtt{x} &= \theta(\mathtt{x}) \end{aligned}$$

For `let` , we want to define

$$\theta^*(\mathtt{let}\ M\ \mathtt{be\ x\ in}\ N) = \mathtt{let\ x} = \theta^*M \ \mathtt{in}\ (\theta, \mathtt{x} : A)^*N$$

where the renaming morphism $\Gamma, \mathtt{x} : A \xrightarrow{\theta,\mathtt{x}:A} \Delta, \mathtt{x} : A$ maps $\mathtt{x}$ to $\mathtt{x}$, and otherwise is the same as θ.

In summary, the definition of substitution goes in 4 stages:

- define $\theta, \mathtt{x} : A$
- define renaming by induction
- define $k, \mathtt{x} : A$
- define substitution by induction.

A consequence of this is that if you want to prove a theorem about substitution, you'll first have to prove it for renaming.

Proposition 1. *1. Contexts and substitutions form a category—composition is defined by subsitution. This means*

$$k; \mathrm{id} = k$$
$$\mathrm{id}; k = k$$
$$(k; l); m = k; (l; m)$$

Renamings form a subcategory, i.e. every renaming is a substitution and renamings have the same set of laws.

2. $(k; l)^*M$ *is the same as* k^*l^*M*, and* id^*M *is the same as* M*.*

2 Evaluation Through β-reduction

Intuitively, a β-reduction means simplification. I'll write $M \leadsto N$ to mean that M can be simplified to N. For example, there are β-reduction rules for all the arithmetic operations:

$$\underline{m} + \underline{n} \leadsto \underline{m+n}$$
$$\underline{m} \times \underline{n} \leadsto \underline{m \times n}$$
$$\underline{m} > \underline{n} \leadsto \mathtt{true} \text{ if } m > n$$
$$\underline{m} > \underline{n} \leadsto \mathtt{false} \text{ if } m \leqslant n$$

There is a β-reduction rule for local definitions:

$$\mathtt{let}\ \mathtt{x} = M\ \mathtt{in}\ N \leadsto N[M/\mathtt{x}]$$

But the most interesting are the β-reductions for all the types. The rough idea is: if you use an introduction rule and then, immediately, use an elimination rule, then they can be simplified.

For the boolean type, the β-reduction rule is

$$\texttt{case true of } \{\texttt{true} \to N \mid \texttt{false} \to N'\} \rightsquigarrow N$$
$$\texttt{case false of } \{\texttt{true} \to N \mid \texttt{false} \to N'\} \rightsquigarrow N'$$

For the type $A \times B$, if we use projections the β-reduction rule is

$$\texttt{fst } (M, M') \rightsquigarrow M$$
$$\texttt{snd } (M, M') \rightsquigarrow M'$$

If we use pattern-matching, the β-reduction rule is

$$\texttt{case } (M, M') \texttt{ of } (\texttt{x}, \texttt{y}) \to N \rightsquigarrow N[M/\texttt{x}, M'/\texttt{y}]$$

For the type $A + B$, the β-reduction rule is

$$\texttt{case } (\#\text{left}, M) \texttt{ of } \{(\#\text{left}, \texttt{x}) \texttt{ in } N, (\#\text{right}, \texttt{y}) \texttt{ in } N'\} \rightsquigarrow N[M/\texttt{x}]$$
$$\texttt{case } (\#\text{right}, M) \texttt{ of } \{(\#\text{left}, \texttt{x}) \texttt{ in } N, (\#\text{right}, \texttt{y}) \texttt{ in } N'\} \rightsquigarrow N'[M/\texttt{y}]$$

For the type $A \to B$, the β-reduction rule is

$$(\lambda \texttt{x}.M)N \rightsquigarrow M[N/\texttt{x}]$$

A term which is the left-hand-side of a β-reduction is called a *β-redex.*

You can simplify any term M by picking a subterm that's a β-redex, and reduce it. Do this again and again until you get a β-*normal* term, i.e. one that doesn't contain any β-redex. It can be shown that this process has to terminate (the *strong normalization theorem*).

Proposition 2. *A closed term M that is β-normal must have an introduction rule at the root. (Remember that we consider $\underline{n}$ to be an introduction rule, but not $+ \times >$.) Hence, if M has type* `int`*, then it must be $\underline{n}$ for some $n \in \mathbb{Z}$.*

We prove the first part by induction on M.

Exercise 3. All the sums that we did can be turned into expressions and evaluated using β-reduction. Try:

1. $\texttt{let x} = (5, (2, \texttt{true}))\ \texttt{in fst x} + \texttt{fst}\ (\texttt{case x of}\ (\texttt{y}, \texttt{z}) \to \texttt{z})$
2. $$\texttt{case}\ (\texttt{case}\ (3 < 7)\ \texttt{of}\ \{\texttt{true} \to\ (\#\text{right}, 8+1) \mid \texttt{false} \to\ (\#\text{left}, 2)\})\ \texttt{of}$$
   $$\{(\#\text{left}, \texttt{u}) \to\ \texttt{u} + 8 \mid (\#\text{right}, \texttt{u}) \to\ \texttt{u} + 3\}$$
3. $(\lambda \texttt{f} : \texttt{int} \to \texttt{int}.\lambda \texttt{x} : \texttt{int}.\texttt{f}(\texttt{fx})(\lambda \texttt{x} : \texttt{int}.\texttt{x} + 3)2$

3 η-expansion

The η-expansion laws express the idea that

- everything of type `bool` is `true` or `false`
- everything of type $A \times B$ is a pair (x, y)
- everything of type $A + B$ is a pair $(\#\text{left}, x)$ or a pair $(\#\text{right}, x)$
- everything of type $A \to B$ is a function.

They are given by first applying an elimination, then an introduction (the opposite of β-reduction).

Let's begin with the type `bool`. If we have a term $\Gamma, \texttt{z} : \texttt{bool} \vdash N : B$, it can be η-expanded to

$$\texttt{case z of}\ \{\texttt{true} \to\ N[\texttt{true}/\texttt{z}] \mid \texttt{false} \to\ N[\texttt{false}/\texttt{z}]\}$$

The reason this ought to be true is that, whatever we define the identifiers in Γ to be, `z` will be either `true` or `false`. Either way, both sides should be the same.

What about $A \times B$? If we're using projections, then any $\Gamma \vdash M : A \times B$ can be η-expanded to $(\texttt{fst}\ M, \texttt{snd}\ M)$.

And if we're using pattern-match, suppose $\Gamma, \texttt{z} : A \times B \vdash N : C$. Then N can be expanded into

$$\texttt{case z of}\ (\texttt{x}, \texttt{y}) N[(\texttt{x}, \texttt{y})/\texttt{z}]$$

(I'm supposing the `x` and `y` we use here don't appear in $\Gamma, \texttt{z} : A \times B$.)

For $A + B$, it's similar. Suppose $\Gamma, \mathtt{z} : A + B \vdash N : C$. Then N can be expanded into

$$\mathtt{case}\,\mathtt{z}\,\mathtt{of}\ \{(\#\text{left}, \mathtt{x}) \to N[(\#\text{left}, \mathtt{x})/\mathtt{z}] \mid (\#\text{right}, \mathtt{y}) \to N[(\#\text{right}, \mathtt{y})/\mathtt{z}]\}$$

(Again, I'm supposing the $\mathtt{x}$ and $\mathtt{y}$ don't appear in $\Gamma, \mathtt{z} : A + B$.)

And finally, $A \to B$. Any term $\Gamma \vdash M : A \to B$ can be expanded as $\lambda\mathtt{x}.(Mx)$.

(Again, I'm supposing the $\mathtt{x}$ doesn't appear in Γ.)

Exercise 4. Take the term

$$\mathtt{f} : (\mathtt{int} + \mathtt{bool}) \to (\mathtt{int} + \mathtt{bool}) \vdash \mathtt{f} : (\mathtt{int} + \mathtt{bool}) \to (\mathtt{int} + \mathtt{bool})$$

Apply an η-expansion for $\to$, then for $+$, then for $\mathtt{bool}$.

4 Equality

λ-calculus isn't just a set of terms; it comes with an equational theory. If $\Gamma \vdash M : B$ and $\Gamma \vdash N : B$, we write $\Gamma \vdash M = N : B$ to express the intuitive idea that, no matter what we define the identifiers in Γ to be, M and N have the same "meaning" (even though they're different expressions).

First of all we need rules to say that this is an equivalence relation:

$$\frac{\Gamma \vdash M : B}{\Gamma \vdash M = M : B} \qquad \frac{\Gamma \vdash M = N : B}{\Gamma \vdash N = M : B}$$

$$\frac{\Gamma \vdash M = N : B \quad \Gamma \vdash N = P : B}{\Gamma \vdash M = P : B}$$

Secondly, we need rules to say that this is *compatible*—preserved by every construct:

$$\frac{\Gamma \vdash M = M' : A \quad \Gamma, \mathtt{x} : A \vdash N = N' : B}{\Gamma \vdash \mathtt{let}\ \mathtt{x} = M\ \mathtt{in}\ N = \mathtt{let}\ \mathtt{x} = M'\ \mathtt{in}\ N' : B}$$

and so forth. A compatible equivalence relation is often called a *congruence*.

Thirdly, each of the β-reductions that we've seen is an axiom of this theory.

$$\frac{\Gamma \vdash N : B \quad \Gamma \vdash N' : B}{\Gamma \vdash \texttt{case true of } \{\texttt{true} \to \ N \mid \texttt{false} \to N'\} = N : B}$$

$$\frac{\Gamma, \texttt{x} : A \vdash M : B \quad \Gamma \vdash N : A}{\Gamma \vdash (\lambda \texttt{x}.M)N = M[N/\texttt{x}] : B}$$

Fourthly, each of the η-expansions is an axiom of the theory, e.g.

$$\frac{\Gamma \vdash M : A \to B}{\Gamma \vdash M = \lambda \texttt{x}.(M\texttt{x}) : A \to B}$$

But in the case of the η-expansions involving pattern-matching, we need to generalize them slightly. The reason is that we want to prove

Proposition 3. *If* $\Gamma \vdash M = N : B$ *and* $\Gamma \xrightarrow{k} \Delta$ *is a substitution, then* $\Delta \vdash k^*M = k^*N : B$

Consequently, the η-law for `bool` looks like this:

$$\frac{\Gamma \vdash M : \texttt{bool} \quad \Gamma, \texttt{z} : \texttt{bool} \vdash N : C}{\begin{array}{l}\Gamma \vdash N[M/\texttt{z}] = \\ \quad \texttt{case } M \texttt{ of } \{\texttt{true} \to N[\texttt{true}/\texttt{z}] \mid \texttt{false} \to N[\texttt{false}/\texttt{z}]\} \\ \quad : C\end{array}}$$

and similarly for the other pattern-matching laws. We can then prove Prop. 3, first for renamings, then for substitution.

5 Exercises

1. Suppose that $\Gamma \vdash M : \texttt{bool}$ and $\Gamma \vdash N_0, N_1, N_2, N_3 : C$. Show that

$$\begin{array}{l}\Gamma \vdash \texttt{case } M \texttt{ of } \{ \\ \quad \texttt{true} \to \ \texttt{case } M \texttt{ of } \{\texttt{true}.N_0 \mid \texttt{false}.N_1\}, \\ \quad \mid \texttt{false} \to \ \texttt{case } M \texttt{ of } \{\texttt{true} \to N_2 \mid \texttt{false} \to N_3\} \\ \} \\ = \texttt{case } M \texttt{ of } \{\texttt{true} \to N_0 \mid \texttt{false} \to N_3\} : C\end{array}$$

2. Show that $(\#\text{left}, -)$ is injective, i.e. if $\Gamma \vdash M, M' : A$ and $\Gamma \vdash (\#\text{left}, M) = (\#\text{left}, M') : A + B$ then $\Gamma \vdash M = M' : A$.
3. Write down the η-law for the 0 type.
4. Given a term $\Gamma, \mathtt{x} : A \vdash M : 0$, show that it is an "isomorphism" in the sense that there is a term $\Gamma, \mathtt{y} : 0 \vdash N : A$ satisfying

$$\Gamma, \mathtt{y} : 0 \vdash M[N/\mathtt{x}] = \mathtt{y} : 0$$
$$\Gamma, \mathtt{x} : A \vdash N[M/\mathtt{x}] = \mathtt{x} : A$$

5. Give β and η laws for $\alpha(A, B, C, D, E)$ and for $\beta(A, B, C, D, E, F, G)$. (See yesterday's exercises for a description of these types.)