TSPL: Notes on Subtyping

Philip Wadler

Tuesday 22 March 2016

1 Simply-typed lambda calculus with records

Write $\Gamma \vdash_{\text{STLC}} M : A$ if term M has type A in the simply-typed lambda calculus with functions and records.

$\boxed{\Gamma \vdash_{\text{STLC}} M : A}$

$$\text{id}\,\frac{(x : A) \in \Gamma}{\Gamma \vdash x : A}$$

$$\text{abstract}\,\frac{\Gamma, x : A \vdash N : B}{\Gamma \vdash (\lambda x{:}A.\, N) : A \to B} \qquad \text{apply}\,\frac{\Gamma \vdash L : A \to B \qquad \Gamma \vdash M : A}{\Gamma \vdash (L\, M) : B}$$

$$\text{record}\,\frac{\Gamma \vdash \vec{M} : \vec{A}}{\Gamma \vdash \{\vec{\ell} = \vec{M}\} : \{\vec{\ell} : \vec{A}\}} \qquad \text{select}\,\frac{\Gamma \vdash L : \{\vec{\ell} : \vec{A}\} \qquad 1 \leq i \leq |\ell|}{\Gamma \vdash L.\ell_i : A_i}$$

We write $\{\vec{\ell} : \vec{A}\}$ to stand for $\{\ell_1 : A_1, \ldots, \ell_n : A_n\}$ where $|\vec{\ell}|$ stands for n, and similarly for $\{\vec{M} : \vec{A}\}$ and $\{\vec{\ell} = \vec{M}\}$. For example, an instance of the last rule is:

$$\text{select}\,\frac{\Gamma \vdash L : \{\ell_1 : A_1, \ell_2 : A_2, \ell_3 : A_3\} \qquad 1 \leq i \leq 3}{\Gamma \vdash L.\ell_i : A_i}$$

2 Subtyping

Write $A <: B$ if every value of type A is also of type B.

$\boxed{A <: B}$

$$\text{refl}\,\frac{}{A <: A} \qquad \text{tran}\,\frac{A <: B \quad B <: C}{A <: C}$$

$$\text{fun}\,\frac{C <: A \quad B <: D}{A \to B <: C \to D}$$

$$\text{depth}\,\frac{\vec{A} <: \vec{B}}{\{\vec{\ell} : \vec{A}\} <: \{\vec{\ell} : \vec{B}\}} \qquad \text{width}\,\frac{}{\{\vec{\ell}, \vec{m} : \vec{A}, \vec{B}\} <: \{\vec{\ell} : \vec{A}\}}$$

The first line says subtyping is reflexive and transitive. The second line says subtyping of functions is *contravariant* in the domain and *covariant* in the range. The third line says records are covariant in the field types (depth), and a record with more fields is a subtype of a record with fewer (width).

For example, using depth and width subtyping we have

$$\frac{A_1 <: B_1 \quad A_2 <: B_2}{\{\ell_1 : A_1,\, \ell_2 : A_2,\, \ell_3 : A_3\} <: \{\ell_1 : B_1,\, \ell_2 : B_2\}}$$

Write $\Gamma \vdash_{\text{SUB}} M : A$ if term M has type A in the lambda calculus with subtyping. The rules are identical to those of simply-typed lambda calculus, augmented with a rule for *subsumption*.

$\boxed{\Gamma \vdash_{\text{SUB}} M : A}$

$$\text{id}\,\frac{(x : A) \in \Gamma}{\Gamma \vdash x : A}$$

$$\text{abstract}\,\frac{\Gamma,\, x : A \vdash N : B}{\Gamma \vdash (\lambda x{:}A.\, N) : A \to B} \qquad \text{apply}\,\frac{\Gamma \vdash L : A \to B \quad \Gamma \vdash M : A}{\Gamma \vdash (L\, M) : B}$$

$$\text{record}\,\frac{\Gamma \vdash \vec{M} : \vec{A}}{\Gamma \vdash \{\vec{\ell} = \vec{M}\} : \{\vec{\ell} : \vec{A}\}} \qquad \text{select}\,\frac{\Gamma \vdash L : \{\vec{\ell} : \vec{A}\} \quad 1 \le i \le |\ell|}{\Gamma \vdash L.\ell_i : A_i}$$

$$\text{sub}\,\frac{\Gamma \vdash M : A \quad A <: B}{\Gamma \vdash M : B}$$

Simply-typed lambda calculus has the nice property that there is exactly one typing derivation possible for each well-typed term. With subtyping, that is no longer the case. Here are two different derivations showing term $L\, M$ has type D, given that $L : A \to B$, $M : C$, $C <: A$, and $B <: D$.

$$\text{apply}\frac{\text{sub}\dfrac{\Gamma \vdash L : A \to B \qquad \text{fun}\dfrac{C <: A \quad B <: D}{A \to B <: C \to D}}{\Gamma \vdash L : C \to D} \qquad \Gamma \vdash M : C}{\Gamma \vdash (L\,M) : D}$$

$$\text{sub}\frac{\text{apply}\dfrac{\Gamma \vdash L : A \to B \qquad \text{sub}\dfrac{\Gamma \vdash M : C \quad C <: A}{\Gamma \vdash M : A}}{\Gamma \vdash (L\,M) : B} \qquad B <: D}{\Gamma \vdash (L\,M) : D}$$

3 Translation

One way to think of subtyping is that if $A <: B$ then there is a coercion function $c : A \to B$. We can think of the type rules as performing a *type-directed translation*. Write $A <: B \rightsquigarrow c$ to indicate that $c : A \to B$ is a coercion from A to B.

$\boxed{A <: B \rightsquigarrow c}$

$$\text{refl}\frac{}{A <: A \rightsquigarrow (\lambda x{:}A.\, x)} \qquad \text{tran}\frac{A <: B \rightsquigarrow c \quad B <: C \rightsquigarrow d}{A <: C \rightsquigarrow (\lambda x{:}A.\, d(c(x)))}$$

$$\text{fun}\frac{C <: A \rightsquigarrow c \quad B <: D \rightsquigarrow d}{A \to B <: C \to D \rightsquigarrow (\lambda f{:}A{\to}B.\, \lambda x{:}C.\, d(f(c(x))))}$$

$$\text{depth}\frac{\vec{A} <: \vec{B} \rightsquigarrow \vec{c}}{\{\vec{\ell} : \vec{A}\} <: \{\vec{\ell} : \vec{B}\} \rightsquigarrow (\lambda z{:}\{\vec{\ell} : \vec{A}\}.\, \{\vec{\ell} = \vec{c}(z.\vec{\ell})\})}$$

$$\text{width}\frac{}{\{\vec{\ell}, \vec{m} : \vec{A}, \vec{B}\} <: \{\vec{\ell} : \vec{A}\} \rightsquigarrow (\lambda z{:}\{\vec{\ell}, \vec{m} : \vec{A}, \vec{B}\}.\, \{\vec{\ell} = z.\vec{\ell}\})}$$

For example, using depth and width subtyping we have

$$\frac{A_1 <: B_1 \rightsquigarrow c_1 \qquad A_2 <: B_2 \rightsquigarrow c_2}{\begin{array}{l}\{\ell_1 : A_1,\, \ell_2 : A_2,\, \ell_3 : A_3\} <: \{\ell_1 : B_1,\, \ell_2 : B_2\} \rightsquigarrow \\ \quad (\lambda z : \{\ell_1 : A_1,\, \ell_2 : A_2,\, \ell_3 : A_3\}.\, \{\ell_1 = c_1(z.\ell_1),\, \ell_2 = c_2(z.\ell_2)\})\end{array}}$$

The translation of a subtyping relation is a coercion function of the appropriate type.

Proposition 1 *If $A <: B \rightsquigarrow c$ then $\emptyset \vdash_{\text{STLC}} c : A \to B$.*

Write $\Gamma \vdash M : A \rightsquigarrow M'$ to indicate that term M of type A translates to term M'. Each rule simply translates the parts of the term, with the exception of the subsumption rule, which applies the coercion corresponding to the subtype.

$\boxed{\Gamma \vdash_{\text{TRAN}} M : A \rightsquigarrow M'}$

$$\text{id}\,\frac{(x : A) \in \Gamma}{\Gamma \vdash x : A \rightsquigarrow x}$$

$$\text{abstract}\,\frac{\Gamma,\, x : A \vdash N : B \rightsquigarrow N'}{\Gamma \vdash (\lambda x{:}A.\, N) : A \to B \rightsquigarrow \lambda x{:}A.\, N'}$$

$$\text{apply}\,\frac{\Gamma \vdash L : A \to B \rightsquigarrow L' \qquad \Gamma \vdash M : A \rightsquigarrow M'}{\Gamma \vdash (L\, M) : B \rightsquigarrow (L'\, M')}$$

$$\text{record}\,\frac{\Gamma \vdash \vec{M} : \vec{A} \rightsquigarrow \vec{M}'}{\Gamma \vdash \{\vec{\ell} = \vec{M}\} : \{\vec{\ell} : \vec{A}\} \rightsquigarrow \{\vec{\ell} = \vec{M}'\}}$$

$$\text{select}\,\frac{\Gamma \vdash L : \{\vec{\ell} : \vec{A}\} \rightsquigarrow L' \qquad 1 \le i \le |\ell|}{\Gamma \vdash L.\ell_i : A_i \rightsquigarrow L'.\ell_i}$$

$$\text{sub}\,\frac{\Gamma \vdash M : A \rightsquigarrow M' \qquad A <: B \rightsquigarrow c}{\Gamma \vdash M : B \rightsquigarrow c(M')}$$

A term has a translation if it is well-typed in the lambda calculus with subtyping, and its translation is well-typed in the simply-typed lambda calculus.

Proposition 2 (Translation preserves types) *If* $\Gamma \vdash_{\text{TRAN}} M : A \rightsquigarrow M'$ *then* $\Gamma \vdash_{\text{SUB}} M : A$ *and* $\Gamma \vdash_{\text{STLC}} M' : A$.

Indeed, a term is well-typed with subtyping exactly when it has a translation.

Proposition 3 (Subtyping and translation) $\Gamma \vdash_{\text{SUB}} M : A$ *if and only if* $\Gamma \vdash_{\text{TRAN}} M : A \rightsquigarrow M'$ *for some* M'.

If there is more than one derivation of a typing judgement, then correspondingly there will be more than one translation. Corresponding to the two derivations of term $L\,M$ of type D given earlier, here are two translations of that term, given that $L : A \to B \rightsquigarrow L'$, $M : C \rightsquigarrow M'$, $C <: A \rightsquigarrow c$, and $B <: D \rightsquigarrow d$.

$$\frac{\dfrac{\Gamma \vdash L : A \to B \rightsquigarrow L' \qquad \dfrac{C <: A \rightsquigarrow c \qquad B <: D \rightsquigarrow d}{A \to B <: C \to D \rightsquigarrow F}}{\Gamma \vdash L : C \to D \rightsquigarrow F\, L'} \qquad \Gamma \vdash M : C \rightsquigarrow M'}{\Gamma \vdash (L\, M) : D \rightsquigarrow F\, L'\, M'}$$

$$\frac{\dfrac{\Gamma \vdash L : A \to B \rightsquigarrow L' \qquad \dfrac{\Gamma \vdash M : C \rightsquigarrow M' \qquad C <: A \rightsquigarrow c}{\Gamma \vdash M : A \rightsquigarrow c\, M'}}{\Gamma \vdash (L\, M) : B \rightsquigarrow L'\,(c\, M')} \qquad B <: D \rightsquigarrow d}{\Gamma \vdash (L\, M) : D \rightsquigarrow d\,(L'\,(c\, M'))}$$

where $F = (\lambda f{:}A \to B.\, \lambda x{:}C.\, d(f(c(x))))$. The two derived terms are equivalent since

$$(\lambda f{:}A \to B.\, \lambda x{:}C.\, d(f(c(x))))\; L'\; M' \longrightarrow^{*} d\,(L'\,(c\,M'))$$

under call-by-name.

Whenever we use type derivations to guide translation, it is desirable that the system to be *coherent*, meaning that if there is more than one possible type derivation then all of the corresponding derivations are equivalent. For terms M and N of simply-typed lambda calculus, write $M =_{\text{STLC}} N$ iff $M \longrightarrow^{*} P$ and $N \longrightarrow^{*} P$ under call-by-name for some term P.

Proposition 4 (Translation is coherent) *If* $\Gamma \vdash_{\text{TRAN}} L : A \leadsto M$ *and* $\Gamma \vdash_{\text{TRAN}} L : A \leadsto N$ *then* $M =_{\text{STLC}} N$.