Formalization in Coq of the Properties of Subtyping in System-F with a Locally Nameless Representation

Arthur Charguéraud - July 2006

1 Definitions

1.1 Definition of Types and Substitutions

$$T \quad := \quad \text{Top} \quad | \quad i \quad | \quad X \quad | \quad T_1 \to T_2 \quad | \quad \forall <: T_1 . T_2$$

A type is either the maximal type, a bound variable (built on an integer), a free variable (built on a name), an arrow, or an universal type. As in informal practice, we will make the confusion in the notations between a name X and the free variable X.

Bound variables are represented using de-Bruijn indices, and thus asbstractions do not introduce a name. This avoid the introduction of α-equivalence classes and α-renaming. Free variables are represented using names. This has the advantage that there is no need for shifting and also we can remain very close to informal practice. This mixed representation is usually being referred to as the *locally nameless* approach.

Notice that some terms may be ill-formed, for instance $(\forall <: \text{Top}.\, 2)$ is not a valid term in any environment, and so we will later introduce a relation capturing well-formation of terms.

1.2 Substitution for Indices and Names

Let $\{k \to U\}\, T$ be the substitution of type U for the index k in the type T.

$$\begin{array}{lcll}
\{k \to U\}\, \text{Top} & = & \text{Top} & \\
\{k \to U\}\, k & = & U & \\
\{k \to U\}\, i & = & i & (k \neq i) \\
\{k \to U\}\, X & = & X & \\
\{k \to U\}\, (T_1 \to T_2) & = & (\{k \to U\}\, T_1) \to (\{k \to U\}\, T_2) & \\
\{k \to U\}\, (\forall <: T_1 . T_2) & = & \forall <: (\{k \to U\}\, T_1) . (\{k+1 \to U\}\, T_2) &
\end{array}$$

From a user point of view, only the substitution for index zero is interesting, and thus we will have a more friendly notation for it:

$$T^U \qquad \equiv \qquad \{0 \to U\}\, T$$

Next, let $[Z \to U]\,T$ be the substitution of type U for the name Z in the type T.

$$\begin{array}{lcll}
[Z \to U]\,\mathrm{Top} & = & \mathrm{Top} & \\
[Z \to U]\,i & = & i & \\
[Z \to U]\,Z & = & U & \\
[Z \to U]\,X & = & X & (X \neq Z) \\
[Z \to U]\,(T_1 \to T_2) & = & ([Z \to U]\,T_1) \to ([Z \to U]\,T_2) & \\
[Z \to U]\,(\forall \mathsf{<:} T_1.\,T_2) & = & \forall \mathsf{<:} ([Z \to U]\,T_1).\,([Z \to U]\,T_2) &
\end{array}$$

With the locally nameless approach, we have to define two substitutions. However, because free variables are not indices, there is no need for any shifting of indices in the type U that we are substituting in. Furthermore, because bound variables are not names, there is no problem of variable capture nor shadowing. In the end we have two substitutions, but both are very simple. And this is nice because the simplicity of the substitutions is critical for the simplicity of the entire formalization.

1.3 Definition of Environments

$$E \quad := \quad \varnothing \quad | \quad E,\, X\mathsf{<:}T$$

An environment is represented as a list of bindings, a binding being a pair of a variable name and a type.

Now that we have defined environments as lists, we are going to use them as sets. We define $(X\mathsf{<:}T) \in E$ to be the proposition capturing that the environment E contains the binding $(X\mathsf{<:}T)$. Then we build inclusion of environments on the top of this belonging relation (this is the standard set inclusion). Here is the definition of "F extends E":

$$E \subset F \qquad \equiv \qquad \forall\, X\ T, \quad (X\mathsf{<:}T) \in E \quad \Rightarrow \quad (X\mathsf{<:}T) \in F$$

We use a similar definition for capturing subsitution of types in enviroments. Consider a substitution from name Z to type P. We want to relate two environment E and F so as to capture the idea that for any binding $(X\mathsf{<:}T)$ which can be found in E and which does not bind the variable being substituted for, the binding $(X\mathsf{<:}\,[Z \to P]\,T)$ can be found in F. More precisely, we define the relation "F extends E in which P has been substituted for Z" as

$$[Z \to P]\,E \ \subset F \qquad \equiv \qquad \forall\, X\ T, \quad (X\mathsf{<:}T) \in E \quad \wedge \quad X \neq Z \quad \Rightarrow \quad (X\mathsf{<:}\,[Z \to P]\,T) \in F$$

Moreover, let the domain of an environment, written $\mathrm{dom}(E)$, to be the set of variable names bound by this environment (the list built taking the first projection of all the pairs contained in the environment). As usual, we say that a name X is fresh from an environment E and write $X \,\#\, E$ if it is not included in the domain of E. More formally:

$$X \,\#\, E \qquad \equiv \qquad X \notin \mathrm{dom}(E)$$

Now we want to keep only the well-formed environments. First we want environments to be functional: any name must be bound at most once. Second, types pushed in the environment must be well-formed in the environment into which they are pushed. We use the following inductive predicate to capture this well-formation relation $\vdash E\ ok$.

$$\frac{}{\vdash \varnothing\ ok}\ \textsc{ok-empty} \qquad \frac{\vdash E\ ok \qquad X \,\#\, E \qquad E \vdash T\ wf}{\vdash (E,\, X\mathsf{<:}T)\ ok}\ \textsc{ok-push}$$

1.4 Well-formed Types

A type T is well-formed in an environment E if two conditions are satisfied. First, all the free variables of T must be mapped in E. Second, all the bound variables occuring in T must have an index which makes them point to some abstraction inside T. In our formalization, we define an equivalent notion of well-formedness using the inductive predicate $E \vdash T\ wf$ defined by the rules below.

$$\frac{}{E \vdash \text{Top}\ wf}\ \text{WF-TOP} \qquad \frac{(X{<:}U) \in E}{E \vdash X\ wf}\ \text{WF-FVAR} \qquad \frac{E \vdash T_1\ wf \qquad E \vdash T_2\ wf}{E \vdash (T_1 \to T_2)\ wf}\ \text{WF-ARROW}$$

$$\frac{E \vdash T_1\ wf \qquad \forall X \notin L,\ \forall U,\quad (E,\, X{<:}U) \vdash {T_2}^X\ wf}{E \vdash (\forall{<:}T_1.\, T_2)\ wf}\ \text{WF-ALL}$$

The L in WF-ALL stands for an arbitrary finite list of names. We will come back on the role of the list L when we will introduce the subtyping relation. Also, remember that ${T_2}^X$ is the substitution replacing by X all the variables bound by the universal type from which T_2 is the body.

The difficult rule is WF-ALL. It says that if T_1 is well-formed in E, and if for any type U and for any name X which does not belong to some finite list of names L it can be shown that the term ${T_2}^X$ is well-formed in the environment E extended with a binding $(X{<:}U)$, then the universal type $(\forall{<:}T_1.\, T_2)$ is well-formed in the environment E.

Technical remark: the type U which is quantified in the WF-ALL rule does is not assumed to be well-formed. Indeed, the well-formation of types relation does not require the environment to be well-formation. Anyway, we could not require U to be well-formed in E because this would use a negative occurence of the relation that we are defining inductively, and this can't be done in Coq.

The interest of defining well-formation inductively is that it gives us an induction principle over well-formed terms, corresponding to the "induction on the structure of a term" used in informal practice.

1.5 Subtyping Relation

Here are the rules that we would like to use to define subtyping:

$$\frac{}{E \vdash S {<:} \text{Top}}\ \text{SA-TOP} \qquad \frac{E \vdash T_1 {<:} S_1 \qquad E \vdash S_2 {<:} T_2}{E \vdash (S_1 \to S_2) {<:} (T_1 \to T_2)}\ \text{SA-ARROW}$$

$$\frac{}{E \vdash X {<:} X}\ \text{SA-REFL-TVAR} \qquad \frac{(X{<:}U) \in E \qquad E \vdash U {<:} T}{E \vdash X {<:} T}\ \text{SA-TRANS-TVAR}$$

$$\frac{E \vdash T_1 {<:} S_1 \qquad \forall X \notin L,\ (E,\, X{<:}T_1) \vdash {S_2}^X {<:} {T_2}^X}{E \vdash (\forall{<:}S_1.\, S_2) {<:} (\forall{<:}T_1.\, T_2)}\ \text{SA-ALL}$$

Unfortunatly, we need to introduce some well-formation premises for the definition to be correct. Rather than trying to minimize this number of premises to add, we will introduce them in a systematic way. For the result to be readable, we use a special notation (specialized for the subtyping relation):

$$E\ oks\ S\ and\ T \qquad \equiv \qquad \vdash E\ ok \quad \wedge \quad E \vdash S\ wf \quad \wedge \quad E \vdash T\ wf$$

Then we define the correct subtyping relation $E \vdash S <: T$ inductively.

$$\frac{E \text{ oks } S \text{ and } \mathrm{Top}}{E \vdash S <: \mathrm{Top}}\ \textsc{sa-top} \qquad \frac{E \text{ oks } (S_1 \to S_2) \text{ and } (T_1 \to T_2) \quad E \vdash T_1 <: S_1 \quad E \vdash S_2 <: T_2}{E \vdash (S_1 \to S_2) <: (T_1 \to T_2)}\ \textsc{sa-arrow}$$

$$\frac{E \text{ oks } X \text{ and } X}{E \vdash X <: X}\ \textsc{sa-refl-tvar} \qquad \frac{E \text{ oks } X \text{ and } T \quad (X<:U) \in E \quad E \vdash U <: T}{E \vdash X <: T}\ \textsc{sa-trans-tvar}$$

$$\frac{E \text{ oks } (\forall <: S_1.\, S_2) \text{ and } (\forall <: T_1.\, T_2) \quad E \vdash T_1 <: S_1 \quad \forall X \notin L,\ (E, X<:T_1) \vdash {S_2}^X <: {T_2}^X}{E \vdash (\forall <: S_1.\, S_2) <: (\forall <: T_1.\, T_2)}\ \textsc{sa-all}$$

In the SA-ALL rule, L is an arbitrary finite list of names. In short, we do this to avoid the need of a formal proof of equivariance of the subtyping relation. We only need equivariance to show that our relation is adequate with respect to the standard definition of subtyping (which uses a $\exists X \# E$ quantification in rule SA-ALL).

At first sight, it may seem unintuitive and artificial to take such a list L and not simply the domain of the environment E. Let us explain where this comes from. When we do the proof that type substitution preserves the subtyping relation, we have a name Z on which to perform the substitution. And we need to avoid conflicts between this name Z and the name X universally quantify in the SA-ALL rule. In others words, we would like a quantification:

$$\forall X \# E, \text{ such as } X \neq Z$$

If we are to have one arbitrary name Z, let's just have an entire finite arbitrary list L, it is more general. This is why we consider

$$\forall X \# E, \text{ such as } X \notin L$$

The quantification above works perfectly fine. Nevertheless we found that it made the presentation a little more compact and the proof no more complicated if we include $\mathrm{dom}(E)$ into the list L. Finally, it leaves simply the quantification

$$\forall X \notin L$$

Justification of why this works: in SA-ALL, if X was already in $\mathrm{dom}(E)$, then $(E, X<:T_1)$ would be ill-formed, and the premise $(E, X<:T_1) \vdash {S_2}^X <: {T_2}^X$ could not hold.

2 Proofs

2.1 Statements of the Results

★ SUBTYPING-REFLEXIVITY:

$$\vdash E \text{ ok} \quad \wedge \quad E \vdash T \text{ wf} \quad \Rightarrow \quad E \vdash T <: T$$

★ SUBTYPING-EXTENSION:

$$E \vdash S <: T \quad \wedge \quad E \subset F \quad \wedge \quad \vdash F \text{ ok} \quad \Rightarrow \quad F \vdash S <: T$$

★ SUBTYPING-NARROWING:

$$E \vdash P <: Q \quad \wedge \quad (E, Z<:Q) \vdash S <: T \quad \Rightarrow \quad (E, Z<:P) \vdash S <: T$$

★ SUBTYPING-TRANSITIVITY:

$$E \vdash S <: Q \quad \wedge \quad E \vdash Q <: T \quad \Rightarrow \quad E \vdash S <: T$$

★ SUBTYPING-SUBSTITUTION:

$$(E, Z<:Q) \vdash S <: T \quad \wedge \quad E \vdash P <: Q \quad \Rightarrow \quad E \vdash [Z \to P]S <: [Z \to P]T$$

Couple of remarks. First, lemma SUBTYPING-EXTENSION shows that subtyping is preserved by weakening and by permutation of the environment. Second, lemmas SUBTYPING-NARROWING and SUBTYPING-SUBSTITUTION are not presented in their generalized form (starting from an environment of the form $E, (Z<:Q), F$). The general form is required for the induction principle to be strong enough when environment are viewed as lists, but this is not needed when environments are viewed as sets.

2.2 Sketch of the Main Proofs

Reflexivity. By induction on the derivation of $E \vdash T \ wf$, we prove the statement

$$\vdash E \ ok \quad \wedge \quad E \vdash T \ wf \quad \Rightarrow \quad E \vdash T <: T$$

In the case WF-ALL, we are given a list L of names, and we apply the constructor SA-ALL with the list of names $L ++ \mathrm{dom}(E)$.

Extension. Preservation of subtyping through extension relies on an induction on the derivation of $E \vdash S <: T$ in the following statement:

$$E \vdash S <: T \quad \Rightarrow \quad \forall F, \quad E \subset F \quad \Rightarrow \quad F \vdash S <: T$$

In case SA-ALL, we use $L ++ \mathrm{dom}(F)$ as list of names.

Transitivity. We will say that transitivity holds for a type Q well-formed in an environment E if

$$\forall F, \quad E \subset F \quad \Rightarrow \quad \forall S, \quad \forall T, \ E \vdash S <: Q \quad \Rightarrow \quad E \vdash Q <: T \quad \Rightarrow \quad E \vdash S <: T$$

And we will write this proposition as *sub_trans_prop* $(E \vdash Q \ wf)$. Then we show by induction on the well-formation judgment $E \vdash Q \ wf$ that

$$\forall E, \ \forall Q, \quad E \vdash Q \ wf \quad \Rightarrow \quad \textit{sub\_trans\_prop} \ (E \vdash Q \ wf)$$

Transitivity of subtyping is then an immediate consequence of this result.

Generalized Substitution. From this lemma, we will be able to derive that both narrowing and type substitution preserves subtyping. Here is its statement.

$$\begin{array}{l} F_0 \vdash P <: Q \quad \wedge \quad [Z \to P]Q = Q \quad \wedge \quad (Z<:Q) \in E \quad \wedge \quad E \vdash S <: T \\ \quad \Rightarrow \quad \forall F, \quad \vdash F \ ok \quad \wedge \quad F_0 \subset F \quad \wedge \quad [Z \to P]E \ \subset F \quad \Rightarrow \quad F \vdash [Z \to P]S <: [Z \to P]T \end{array}$$

It is not as bad as it looks. Skipping the details and identifying F and F_0, it leaves

$$E \vdash S <: T \quad \wedge \quad (Z<:Q) \in E \quad \wedge \quad F \vdash P <: Q \quad \Rightarrow \quad F \vdash [Z \to P]S <: [Z \to P]T$$

where F extends a copy of the environment E in which P has been substituted Z.

The proof of that lemma goes by induction on $E \vdash S <: T$, and relies on the very same arguments as the paper proof of the lemma type subsitution preserves subtyping. Technicality: we prove transitivity and generalized substitution together, because the lemmas rely on each other. More precisely, we choosed to have the generalized substitution lemma to take as extra hypothesis the proposition *sub_trans_prop* ($F_0 \vdash Q$ *wf*).

Narrowing. The lemma states that

$$E \vdash P <: Q \quad \wedge \quad (E, Z<:Q) \vdash S <: T \quad \Rightarrow \quad (E, Z<:P) \vdash S <: T$$

and we want to show that this is an instance of the generalized substitution. By instanciating the subtyping rule SA-TRANS-TVAR on $E \vdash P <: Q$, we can transform the statement of the lemma into

$$(E, Z<:P) \vdash Z <: Q \quad \wedge \quad (E, Z<:Q) \vdash S <: T \quad \Rightarrow \quad (E, Z<:P) \vdash [Z \to Z]\,S <: [Z \to Z]\,T$$

To conclude from the generalized substitution lemma, take $E = (E, Z<:Q)$ and $F0 = F = (E, Z<:P)$, and instanciate P as the free variable Z.

Substitution. This one is immediate from the generalized substitution lemma.

$$(E, Z<:Q) \vdash S <: T \quad \wedge \quad E \vdash P <: Q \quad \Rightarrow \quad E \vdash [Z \to P]\,S <: [Z \to P]\,T$$

Acknowledgments

I wish to thanks my two advisors Benjamin Pierce and Stephanie Weirich. Also, feedback and comments from the other members of the Programming Language Group at UPenn were very useful.