1 Overview

Recall the subtyping rules extension for the typed lambda calculus:

\[
\begin{align*}
S &< S \\
S &< \text{Top} \\
\frac{T_3 < T_1 \quad T_2 < T_4}{T_1 \rightarrow T_2 < T_3 \rightarrow T_4} \quad \text{(Arrow)} \\
\frac{\Gamma \vdash e : t_1 \quad \Gamma \vdash t_1 < t_2}{\Gamma \vdash e : t_2} \quad \text{(Sub)} \\
\frac{A < B \quad B < C}{A < C} \quad \text{(Trans)}
\end{align*}
\]

We discussed some of the problems these new rules introduce.

2 Unicity of Typing

Our original statement for unicity of typing is:

\[
\forall t. \forall t'. \Gamma \vdash e : t \land \Gamma \vdash e : t' \Rightarrow t = t'
\]

We lose unicity of typing in the new system. For example, given a term \(\{x:1, y:1\}\), the type of it can be either \(\{x:\text{int}, y:\text{int}\}\), \(\{x:\text{int}\}\), or \(\{y:\text{int}\}\).

As a quick fix, we might want to state the following:

\[
\forall e. \forall \Gamma. \exists t_1, \Gamma \vdash e : t_1 \land \Gamma \vdash e : t_2 \Rightarrow t_1 < t_2 \lor t_2 < t_1
\]

This does not work either. A counterexample would be \(e = \{x:1, y:1\}\), \(t_1 = \{x:\text{int}\}\), \(t_2 = \{y:\text{int}\}\). The correct fix for it is to throw in an extra \(t_3\):

\[
\forall e. \forall \Gamma. \exists t_1, \Gamma \vdash e : t_1 \land \Gamma \vdash e : t_2 \Rightarrow \exists t_3, t_1 > t_3 \land t_2 > t_3
\]
3 Typechecking process

With the new rules, how do we typecheck? Before we had the new rules, if we had a derivation tree, we knew the proof would always terminate. The two rules add uncertainty to our derivation process. If we follow the tree “bottom up”, the typechecker would not know how to find $B$ in the transitivity rule. There is an indefinite number of steps we could possibly perform.

One of the solutions is merging the rules: if there exists a tree that has a chain of rules of arbitrary length, they can be compacted into one rule. For example, by transitivity, two or more adjacent subsumption rules can be merged into one:

$$\frac{\Gamma \vdash e : t_1 \quad t_1 < t_2}{\Gamma \vdash e : t_2} \quad \frac{\Gamma \vdash e : t_2 \quad t_2 < t_3}{\Gamma \vdash e : t_3} \quad \frac{\Gamma \vdash e : t_3}{\Gamma \vdash e : t_4}$$

We can also try to move the steps around. To reorder a proof tree, we still want the new proof to start with the exact same thing, and end with the exact same thing. In the lambda calculus, there are three kinds of expressions: variable, lambda abstraction, and application. There is nothing interesting for the variable case. Consider the lambda case:

$$\frac{\Gamma, x : t_1 \vdash e : t_2}{\Gamma \vdash \lambda x.e : t_1 \rightarrow t_2} \quad (\text{Lam}) \quad \frac{t_1 \rightarrow t_2 < t_3 \rightarrow t_4}{t_3 \rightarrow t_4 \rightarrow t_4} \quad (\text{Sub})$$

To reorder it, we would need a lemma like:

$$\frac{\Gamma, x : t_1 \vdash e : t \land t_2 < t_1}{\Gamma, x : t_2 \vdash e : t}$$

Also, from the arrow rule, we should be able to show that if $t_1 \rightarrow t_2 < t_3 \rightarrow t_4$, then $t_3 < t_1$. Therefore, we have $t_2 < t_4$ and can perform our reordering:

$$\frac{\Gamma, x : t_3 \vdash e : t_2 \rightarrow t_4}{\Gamma \vdash \lambda x.e : t_3 \rightarrow t_4} \quad (\text{Lam})$$

But by using this lemma, we longer have guarantees on the size of the tree, and we cannot do induction on the trees anymore. However, to move things around in the other direction, we don’t need a lemma anymore. Since we already have $t_2 < t_4$ and $t_3 < t_3$, by the arrow rule, we know that $t_3 \rightarrow t_2 < t_3 \rightarrow t_4$. So

$$\frac{\Gamma, x : t_3 \vdash t : t_2}{\Gamma \vdash \lambda x.e : t_3 \rightarrow t_2} \quad \frac{t_3 \rightarrow t_2 < t_3 \rightarrow t_4}{t_3 \rightarrow t_4 \rightarrow t_4} \quad (\text{Sub})$$

This is an important part of the reason why we want to focus on moving things down the tree.

Now consider the application case:
\[
\frac{
\Gamma \vdash e_1 : t_1' \rightarrow t_2' \quad t_1' \rightarrow t_2' < t_1 \rightarrow t_2}
{\text{(Sub)}}
\quad \frac{
\Gamma \vdash e_2 : t_1'' \quad t_1'' < t_1}
{\text{(Sub)}}
\]
\[
\Gamma \vdash e_1 e_2 : t_2
\]

From \(t_1' \rightarrow t_2' < t_1 \rightarrow t_2\) we know that \(t_2' < t_2\) and \(t_1 < t_1'\). With \(t_1'' < t_1\) (from the premise), by transitivity, we have \(t_1'' < t_1 \Rightarrow t_1'' < t_1'\). We may now construct our derivation:

\[
\frac{
\Gamma \vdash e_1 : t_1' \rightarrow t_2' \quad \Gamma \vdash e_2 : t_1'' \quad t_1'' < t_1' \quad \text{(Sub)}}
{\text{(App)}}
\]
\[
\frac{
\Gamma \vdash e_1 e_2 : t_2'
}{\Gamma \vdash e_1 e_2 : t_2}
\]

\[
\frac{
\Gamma \vdash e_1 : t' \rightarrow t_2'
}{\Gamma \vdash e_1 e_2 : t_2}
\quad \frac{
\Gamma \vdash e_2 : t'' \quad t'' < t'
}{\Gamma \vdash e_1 e_2 : t_2'}
\]

Note the extra subsumption rule on the top: our derivation does not start with the application rule anymore. This suggests a new application rule that incorporates the subsumption rule: