CST Part II Types: Exercise Sheet

ML Polymorphism

Exercise 1. Here are some type checking problems, in the sense of Slide 7. Prove the following typings hold for the Mini-ML type system:

$$\begin{aligned}
&\vdash \lambda x(x :: \texttt{nil}) : \forall\,\alpha\,(\alpha \to \alpha\ \mathit{list})\\
&\vdash \lambda x(\texttt{case}\ x\ \texttt{of}\ \texttt{nil} \Rightarrow \texttt{true}\ |\ x_1 :: x_2 \Rightarrow \texttt{false}) : \forall\,\alpha\,(\alpha\ \mathit{list} \to \mathit{bool})\\
&\vdash \lambda x_1(\lambda x_2(x_1)) : \forall\,\alpha_1, \alpha_2\,(\alpha_1 \to (\alpha_2 \to \alpha_1))\\
&\vdash \texttt{let}\ f = \lambda x_1(\lambda x_2(x_1))\ \texttt{in}\ f\ f : \forall\,\alpha_1, \alpha_2, \alpha_3\,(\alpha_1 \to (\alpha_2 \to (\alpha_3 \to \alpha_2))).
\end{aligned}$$

Exercise 2. Show that if $\{\,\} \vdash M : \sigma$ is provable, then M must be *closed*, i.e. have no free variables. [Hint: use rule induction for the rules on Slides 16–19 to show that the provable typing judgements, $\Gamma \vdash M : \tau$, all have the property that $\mathit{fv}(M) \subseteq \mathit{dom}(\Gamma)$.]

Exercise 3. Let σ and σ' be Mini-ML type schemes. Show that the relation $\sigma \succ \sigma'$ defined on Slide 27 holds if and only if

$$\forall\,\tau\,(\sigma' \succ \tau \ \Rightarrow\ \sigma \succ \tau).$$

[Hint: use the following property of simultaneous substitution:

$$(\tau[\tau_1/\alpha_1, \ldots, \tau_n/\alpha_n])[\vec{\tau}'/\vec{\alpha}'] = \tau[\tau_1[\vec{\tau}'/\vec{\alpha}']/\alpha_1, \ldots, \tau_n[\vec{\tau}'/\vec{\alpha}']/\alpha_n]$$

which holds provided the type variables $\vec{\alpha}'$ do not occur in τ.]

Exercise 4. Try to augment the definition of pt on Slide 30 and in Figure 3 with clauses for `nil`, cons, and `case`-expressions.

Exercise 5. Suppose M is a closed expression and that (S, σ) is a principal solution for the typing problem $\{\,\} \vdash M : ?$ in the sense of Slide 27. Show that σ must be a principal type scheme for M in the sense of Slide 23.

Exercise 6. Show that if $\Gamma \vdash M : \sigma$ is provable and $S \in \mathrm{Sub}$ is a type substitution, then $S\,\Gamma \vdash M : S\,\sigma$ is also provable.

Polymorphic Reference Types

Exercise 7. Letting M denote the expression on Slide 33 and $\{\,\}$ the empty state, show that $\langle M, \{\,\}\rangle \to^* \mathit{FAIL}$ is provable in the transition system defined in Figure 4.

Exercise 8. Give an example of a Mini-ML `let`-expression which is typeable in the type system of Section 2.1, but not in the type system of Section 3.2 for Midi-ML with the value-restricted rule (letv).

Polymorphic Lambda Calculus

Exercise 9. Give a proof inference tree for (8) in Example 4.1.1. Show that

$$\forall\,\alpha_1\,(\alpha_1 \to \forall\,\alpha_2\,(\alpha_2)) \to \mathit{bool\ list}$$

is another possible polymorphic type for $\lambda f((f\,\texttt{true}) :: (f\,\texttt{nil}))$.

Exercise 10. Show that if $\Gamma \vdash M : \tau$ and $\Gamma \vdash M : \tau'$ are both provable in the PLC type system, then $\tau = \tau'$ (equality up to α-conversion). [Hint: show that $H \stackrel{\text{def}}{=} \{(\Gamma, M, \tau) \mid \Gamma \vdash M : \tau \;\;\&\;\; \forall\,\tau'\,(\Gamma \vdash M : \tau' \;\Rightarrow\; \tau = \tau')\}$ is closed under the axioms and rules on Slide 45.]

Exercise 11. In PLC, defining the expression $\texttt{let}\,x = M_1 : \tau\,\texttt{in}\,M_2$ to be an abbreviation for $(\lambda\,x : \tau\,(M_2))\,M_1$, show that the typing rule

$$\frac{\Gamma \vdash M_1 : \tau_1 \quad \Gamma, x : \tau_1 \vdash M_2 : \tau_2}{\Gamma \vdash (\texttt{let}\,x = M_1 : \tau_1\,\texttt{in}\,M_2) : \tau_2} \quad \text{if } x \notin \mathit{dom}(\Gamma)$$

is admissible—in the sense that the conclusion is provable if the hypotheses are.

Exercise 12. The *erasure*, $\mathit{erase}(M)$, of a PLC expression M is the expression of the untyped lambda calculus obtained by deleting all type information from M:

$$\begin{aligned}
\mathit{erase}(x) &\stackrel{\text{def}}{=} x\\
\mathit{erase}(\lambda\,x : \tau\,(M)) &\stackrel{\text{def}}{=} \lambda\,x\,(\mathit{erase}(M))\\
\mathit{erase}(M_1\,M_2) &\stackrel{\text{def}}{=} \mathit{erase}(M_1)\;\mathit{erase}(M_2)\\
\mathit{erase}(\Lambda\,\alpha\,(M)) &\stackrel{\text{def}}{=} \mathit{erase}(M)\\
\mathit{erase}(M\,\tau) &\stackrel{\text{def}}{=} \mathit{erase}(M).
\end{aligned}$$

(i) Find PLC expressions M_1 and M_2 satisfying $\mathit{erase}(M_1) = \lambda\,x\,(x) = \mathit{erase}(M_2)$ such that $\vdash M_1 : \forall\,\alpha\,(\alpha \to \alpha)$ and $\vdash M_2 : \forall\,\alpha_1\,(\alpha_1 \to \forall\,\alpha_2\,(\alpha_1))$ are provable PLC typings.

(ii) We saw in Example 4.2.6 that there is a closed PLC expression M of type $\forall\,\alpha\,(\alpha) \to \forall\,\alpha\,(\alpha)$ satisfying $\mathit{erase}(M) = \lambda\,f\,(f\,f)$. Find some other closed, typeable PLC expressions with this property.

(iii) [For this part you will need to recall, from the CST Part IB *Foundations of Functional Programming* course, some properties of beta reduction of expressions in the untyped lambda calculus.] A theorem of Girard says that if $\vdash M : \tau$ is provable in the PLC type system, then $\mathit{erase}(M)$ is strongly normalisable in the untyped lambda calculus, i.e. there are no infinite chains of beta-reductions starting from $\mathit{erase}(M)$. Assuming this result, exhibit an expression of the untyped lambda calculus which is not equal to $\mathit{erase}(M)$ for any closed, typeable PLC expression M.

Exercise 13. Prove the various typings and beta-reductions asserted in Example 4.4.4.

Exercise 14. Prove the various typings asserted in Example 4.4.5 and the beta-conversions on Slide 56.

Exercise 15. For the polymorphic product type $\alpha_1 * \alpha_2$ defined in the right-hand column of Figure 5, show that there are PLC expressions $Pair$, fst, and snd satisfying:

$$
\begin{aligned}
&\{\,\} \vdash Pair : \forall\, \alpha_1, \alpha_2\, (\alpha_1 \to \alpha_2 \to (\alpha_1 * \alpha_2))\\
&\{\,\} \vdash fst : \forall\, \alpha_1, \alpha_2\, ((\alpha_1 * \alpha_2) \to \alpha_1)\\
&\{\,\} \vdash snd : \forall\, \alpha_1, \alpha_2\, ((\alpha_1 * \alpha_2) \to \alpha_2)\\
&fst\, \alpha_1\, \alpha_2 (Pair\, \alpha_1\, \alpha_2\, x_1\, x_2) =_\beta x_1\\
&snd\, \alpha_1\, \alpha_2 (Pair\, \alpha_1\, \alpha_2\, x_1\, x_2) =_\beta x_2.
\end{aligned}
$$

Exercise 16. [hard] Suppose that τ is a PLC type with a single free type variable, α. Suppose also that T is a closed PLC expression satisfying

$$\{\,\} \vdash T : \forall\, \alpha_1, \alpha_2\, ((\alpha_1 \to \alpha_2) \to (\tau[\alpha_1/\alpha] \to \tau[\alpha_2/\alpha])).$$

Define ι to be the closed PLC type

$$\iota \stackrel{\text{def}}{=} \forall\, \alpha\, ((\tau \to \alpha) \to \alpha).$$

Show how to define PLC expressions R and I satisfying

$$
\begin{aligned}
&\{\,\} \vdash R : \forall\, \alpha\, ((\tau \to \alpha) \to \iota \to \alpha)\\
&\{\,\} \vdash I : \tau[\iota/\alpha] \to \iota\\
&(R\, \alpha\, f)(I\, x) \to^* f\, (T\, \iota\, \alpha\, (R\, \alpha\, f)\, x).
\end{aligned}
$$