Homework 4: System F

15-814: Types and Programming Languages
TA: Carlo Angiuli (cangiuli@cs.cmu.edu)

Out: 11/7/13
Due: 11/21/13 (12 PM)

Please submit your work by the start of lecture on the due date, as a PDF to `cangiuli@cs.cmu.edu`. Include the phrase "15-814 Homework 4" in the subject line of your email.

1 Type Safety for System F

Recall the grammar for System F:

$$\tau := t \mid \tau \to \tau \mid \forall t.\tau$$
$$M := x \mid \lambda x\!:\!\tau.\, M \mid M\,N \mid \Lambda t.M \mid M[\tau]$$

and the statics and dynamics:

$$\frac{}{\Delta;\Gamma, x : \tau \vdash x : \tau} \qquad \frac{\Delta \vdash \tau_1\ \mathsf{type} \quad \Delta;\Gamma, x : \tau_1 \vdash M : \tau_2}{\Delta;\Gamma \vdash \lambda x\!:\!\tau_1.\, M : \tau_1 \to \tau_2} \qquad \frac{\Delta;\Gamma \vdash M : \tau_1 \to \tau_2 \quad \Delta;\Gamma \vdash N : \tau_1}{\Delta;\Gamma \vdash M\,N : \tau_2}$$

$$\frac{\Delta, t\ \mathsf{type};\Gamma \vdash M : \tau}{\Delta;\Gamma \vdash \Lambda t.M : \forall t.\tau} \qquad \frac{\Delta;\Gamma \vdash M : \forall t.\tau_1 \quad \Delta \vdash \tau\ \mathsf{type}}{\Delta;\Gamma \vdash M[\tau] : [\tau/t]\tau_1} \qquad \frac{\mathsf{FV}(\tau) \subseteq \mathsf{dom}(\Delta)}{\Delta \vdash \tau\ \mathsf{type}}$$

$$\frac{M \mapsto M'}{M\,N \mapsto M'\,N} \qquad \frac{M\ \mathsf{val} \quad N \mapsto N'}{M\,N \mapsto M\,N'} \qquad \frac{N\ \mathsf{val}}{(\lambda x\!:\!\tau.\, M)\,N \mapsto [N/x]M}$$

$$\frac{M \mapsto M'}{M[\tau] \mapsto M'[\tau]} \qquad \frac{}{(\Lambda t.M)[\tau] \mapsto [\tau/t]M} \qquad \frac{}{\Lambda t.M\ \mathsf{val}} \qquad \frac{}{\lambda x\!:\!\tau_1.\, M\ \mathsf{val}}$$

Before we can prove preservation and progress for System F, we need to know that substitution of types and terms preserves both typing and well-formedness of types. In the following lemma, $[\tau/t]\Gamma$ stands for the context obtained by substituting τ for t in each type in Γ.

Lemma 1.1 (Substitution)**.**

1. *If* $\Delta, t\ \mathsf{type} \vdash \tau\ \mathsf{type}$ *and* $\Delta \vdash \tau'\ \mathsf{type}$*, then* $\Delta \vdash [\tau'/t]\tau\ \mathsf{type}$.
2. *If* $\Delta, t\ \mathsf{type};\Gamma \vdash M : A$ *and* $\Delta \vdash \tau\ \mathsf{type}$*, then* $\Delta;[\tau/t]\Gamma \vdash [\tau/t]M : [\tau/t]A$.
3. *If* $\Delta;\Gamma, x : A \vdash M : B$ *and* $\Delta;\Gamma \vdash N : A$*, then* $\Delta;\Gamma \vdash [N/x]M : B$.

Now we are prepared to prove preservation and progress. Recall from the previous homework that type safety is an easy consequence of these theorems.

Task 1 (Preservation)**.** *If* $\cdot;\cdot \vdash M : \tau$ *and* $M \mapsto M'$ *then* $\cdot;\cdot \vdash M' : \tau$.

Task 2 (Progress)**.** *If* $\cdot;\cdot \vdash M : \tau$ *then either* $M\ \mathsf{val}$ *or* $M \mapsto M'$*, for some* M'.

2 Polymorphic Church Encodings

Polymorphism gives us the power to Church-encode data types like we could in the untyped λ-calculus, but now we have the power of types to ensure that we don't pass illegal terms to our constructors and recursors.

In this task, we will Church-encode typed sums, where the left and right branches of the sum are required to contain a term of the correct type. In particular, for any two types τ_1, τ_2, we will define a type $\mathsf{Sum}(\tau_1, \tau_2)$ and terms $\mathsf{inl}, \mathsf{inr}, \mathsf{rec}_{\mathsf{Sum}}$ with the following properties:

$$
\begin{aligned}
\mathsf{inl} &: \tau_1 \to \mathsf{Sum}(\tau_1, \tau_2) \\
\mathsf{inr} &: \tau_2 \to \mathsf{Sum}(\tau_1, \tau_2) \\
\mathsf{rec}_{\mathsf{Sum}} &: \forall t. \mathsf{Sum}(\tau_1, \tau_2) \to (\tau_1 \to t) \to (\tau_2 \to t) \to t
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{rec}_{\mathsf{Sum}}[\tau]\,(\mathsf{inl}\, M)\, N_1\, N_2 &\mapsto^* N_1\, M \\
\mathsf{rec}_{\mathsf{Sum}}[\tau]\,(\mathsf{inr}\, M)\, N_1\, N_2 &\mapsto^* N_2\, M
\end{aligned}
$$

Task 3. *Define* $\mathsf{Sum}(\tau_1, \tau_2)$*,* inl*,* inr*, and* $\mathsf{rec}_{\mathsf{Sum}}$*.*

3 Contextual and Logical Equivalence

Now, extend System F to include an observable type of booleans $\mathbf{2}$ with terms tt and ff. In class, we defined the grammar and typing rules for expression contexts $\mathcal{C}$. Recall the following definitions:

Definition 3.1 (Kleene equivalence)**.** *Two closed terms* $\cdot;\cdot \vdash M, N : \mathbf{2}$ *are Kleene equivalent,* $M \simeq N$*, if* $(M \mapsto^* \mathsf{tt}) \iff (N \mapsto^* \mathsf{tt})$*, and* $(M \mapsto^* \mathsf{ff}) \iff (N \mapsto^* \mathsf{ff})$*.*

Definition 3.2 (Contextual equivalence)**.** *Two terms* $\Delta; \Gamma \vdash M, N : \tau$ *are contextually equivalent,* $M \cong N$*, if, for any expression context* $\mathcal{C} : (\Delta, \Gamma \triangleright \tau) \to (\cdot, \cdot \triangleright \mathbf{2})$*,* $\mathcal{C}[M] \simeq \mathcal{C}[N]$*.*

Because contextual equivalence quantifies over all expression contexts, it requires some machinery to prove terms are contextually equivalent. However, it is straightforward to show two terms are *not* contextually equivalent.

Task 4. *Show that the closed terms* $(\Lambda t. \lambda x : t.\, \lambda y : t.\, x)$ *and* $(\Lambda t. \lambda x : t.\, \lambda y : t.\, y)$ *are not contextually equivalent.*

A clever choice of expression contexts can also let us prove some basic results about contextually equivalent terms.

Task 5. *For any two terms* $\cdot;\cdot \vdash M, N : \tau$*, if* M *terminates and* $M \cong N$*, then* N *terminates.*

To get a better handle on contextual equivalence, we then defined a type-directed notion of logical equivalence, which it turns out coincides with contextual equivalence.

Definition 3.3 (Logical equivalence of closed terms)**.** *To each closed type* τ *we associate a binary relation* $[\![\tau]\!]$ *on closed terms of that type:*

- $(M, N) \in [\![\mathbf{2}]\!]$ *if* M, N *terminate, and* $M \simeq N$*.*
- $(M, N) \in [\![\tau_1 \to \tau_2]\!]$ *if* M, N *terminate, and for any* $(M', N') \in [\![\tau_1]\!]$*,* $(M\, M', N\, N') \in [\![\tau_2]\!]$*.*

Task 6. *Show that the two closed terms* $\lambda x : \mathbf{2}.\, x$ *and* $\lambda x : \mathbf{2}.\, ((\lambda y : \mathbf{2}.\, x)\, \mathsf{tt})$ *are logically equivalent at type* $\mathbf{2} \to \mathbf{2}$*.*

Finally, we extended this definition to open terms $\Delta; \Gamma \vdash M, N : \tau$ by saying that these terms are logically equivalent if, for any assignment of splendid binary relations to each type variable in Δ, and two closing substitutions γ, γ' for Γ which are elementwise related at their types—interpreting the type variables as the chosen relations—M and N are related by the semantics of τ.

Task 7. *Show that the two open terms*

$$\cdot; f : \mathbf{2} \to \mathbf{2} \vdash f : \mathbf{2} \to \mathbf{2}$$

$$\cdot; f : \mathbf{2} \to \mathbf{2} \vdash \lambda x : \mathbf{2}.\, f\, x : \mathbf{2} \to \mathbf{2}$$

are logically equivalent.