Call-By-Name Normalization for System F

Aaron Stump

November 10, 2014

1 Introduction

This note gives a proof that call-by-name reduction is normalizing for unannotated System F (polymorphic lambda calculus), and considers a few consequences. System F is defined with annotated terms, where λ-bound variables must be declared with their types. So we have \( \lambda x : T.t \) instead of just \( \lambda x.t \). For metatheoretic analysis, I prefer to work with unannotated terms. This system (with unannotated terms) is also called \( \lambda 2 \).

2 Syntax

\[
\begin{align*}
term\;variables\;x \\
type\;variables\;X
\end{align*}
\]
\[
\begin{align*}
terms\;t & ::= x | \lambda x.t | t t' \\
types\;T & ::= X | T \rightarrow T' | \forall X.T
\end{align*}
\]

3 Typing

A typing context \( \Gamma \) declares free term and type variables:

\[
\text{Typing context } \Gamma ::= \cdot | \Gamma, x : T | \Gamma, X : \star
\]

We treat \( \Gamma \) as a function, and write \( \Gamma(x) = T \) to mean that \( \Gamma \) contains a declaration \( x : T \). We will implicitly require that \( \Gamma \) does not declare any variable \( x \) twice. Variables can be implicitly renamed in \( \lambda \)-terms to make it possible to enforce this requirement. The typing rules are in Figure 1. To ensure that types are well-formed, we use some extra rules, called \textit{kinding} rules, in Figure 2.

\[
\begin{align*}
\Gamma(x) = T & \quad \Gamma, x : T \vdash t : T' \\
\Gamma \vdash x : T & \quad \Gamma \vdash \lambda x.t : T \rightarrow T' \\
\Gamma \vdash t : T_1 \rightarrow T_2 & \quad \Gamma \vdash t' : T_1 \\
\Gamma \vdash t : T_2 & \quad \Gamma \vdash t' : T_2
\end{align*}
\]

\[
\begin{align*}
\Gamma, X : \star \vdash t : T & \quad \Gamma \vdash t : \forall X.T \quad \Gamma \vdash T' : \star \\
\Gamma \vdash t : \forall X.T & \quad \Gamma \vdash T' : \star
\end{align*}
\]

Figure 1: Typing rules for unannotated System F
\[
\begin{align*}
\Gamma(X) &= * \\
\Gamma \vdash X : * & \quad \implies & \quad \Gamma \vdash T_1 : * & \quad \implies & \quad \Gamma \vdash T_2 : * & \quad \implies & \quad \Gamma \vdash \forall X. T : *
\end{align*}
\]

Figure 2: Kinding rules for unannotated System F

\[
\begin{align*}
\llbracket X \rrbracket_\rho &= \rho(X) \\
\llbracket T_1 \to T_2 \rrbracket_\rho &= \{ t \in \mathcal{N} \mid \forall t' \in \llbracket T_1 \rrbracket_\rho, t \ t' \in \llbracket T_2 \rrbracket_\rho \} \\
\llbracket \forall X. T \rrbracket_\rho &= \bigcap_{R \in \mathcal{R}} \llbracket T \rrbracket_\rho \downarrow \rho \downarrow X \to R
\end{align*}
\]

Figure 3: Reducibility semantics for types

4 Semantics for types

Figure 3 gives a compositional semantics \( \llbracket T \rrbracket_\rho \) for types. The function \( \rho \) gives the interpretations of free type variables in \( T \). Each free type variable is interpreted as a reducibility candidate, and write \( \rho \) only for functions mapping type variables \( X \) to reducibility candidates. To define what a reducibility candidate is: let us denote the set of closed terms which normalize using call-by-name reduction as \( \mathcal{N} \). We will write \( \rightsquigarrow \) for call-by-name reduction. Then a reducibility candidate \( R \) is a set of terms satisfying the following requirements:

- \( R \subseteq \mathcal{N} \)
- If \( t \in R \) and \( t' \rightsquigarrow t \), then \( t' \in R \)

The set of all reducibility candidates is denoted \( \mathcal{R} \).

Lemma 1 (\( \mathcal{R} \) is a cpo). The set \( \mathcal{R} \) ordered by subset forms a complete partial order, with greatest element \( \mathcal{N} \) and greatest lower bound of a nonempty set of elements of \( \mathcal{R} \) given by intersection.

Proof. \( \mathcal{N} \) satisfies both requirements for a reducibility candidate, and since one of those requirements is being a subset of \( \mathcal{N} \), it is clearly the largest such set to do so. Let us prove that the intersection of a nonempty set \( S \) of reducibility candidates is still a reducibility candidate. Certainly if the members of \( S \) are subsets of \( \mathcal{N} \) then so is \( \bigcap S \). For the second property: assume an arbitrary \( t \in \bigcap S \) with \( t' \rightsquigarrow t \), and show \( t' \in \bigcap S \). For the latter, it suffices to show \( t' \in R \) for every \( R \in S \). Consider an arbitrary such \( R \). From \( t \in \bigcap S \) and \( R \in S \), we have \( t \in R \). Then since \( R \) is a reducibility candidate, \( t \in R \) and \( t' \rightsquigarrow t \) implies \( t' \in R \).

Lemma 2 (The semantics of types computes reducibility candidates). If \( \rho(X) \) is defined for every free type variable of \( T \), then \( \llbracket T \rrbracket_\rho \in \mathcal{R} \).

Proof. The proof is by induction on the structure of the type. If \( T \) is a type variable \( X \), then by assumption, \( \rho(X) \) is a reducibility candidate, and this is the value of \( \llbracket T \rrbracket_\rho \).

If \( T \) is an arrow type \( T_1 \to T_2 \), we must prove the two properties listed above for being a reducibility candidate. Certainly \( \llbracket T \rrbracket_\rho \subseteq \mathcal{N} \), because the semantics of arrow types requires this explicitly. Now suppose that \( t \in \llbracket T_1 \to T_2 \rrbracket_\rho \) and \( t' \rightsquigarrow t \). We must show \( t' \in \llbracket T_1 \to T_2 \rrbracket_\rho \). Since \( t \) is normalizing and \( t' \rightsquigarrow t \), we know that \( t' \) is also normalizing (there is a reduction sequence from \( t' \) to \( t \) and from \( t \) to a normal form). So let us assume an arbitrary \( t'' \in \llbracket T_1 \rrbracket_\rho \), and show that \( t' \ t'' \in \llbracket T_2 \rrbracket_\rho \). Since \( t' \rightsquigarrow t \), by the definition of call-by-name reduction, we have

\[
t' \ t'' \rightsquigarrow t \ t''
\]
Since \( t \in \llbracket T_1 \to T_2 \rrbracket_\rho \), we know by the semantics of types that \( t, t'' \in \llbracket T_2 \rrbracket_\rho \), and \( t'' \in \llbracket T_1 \rrbracket_\rho \). By the IH, \( \llbracket T_2 \rrbracket_\rho \) is a reducibility candidate. So since \( t', t'' \leadsto t, t'' \in \llbracket T_2 \rrbracket_\rho \), we also have \( t', t'' \in \llbracket T_2 \rrbracket_\rho \). This was all we had to prove in this case.

Finally, if \( T \) is a universal type \( \forall X.T' \), then by IH, the set \( \llbracket T' \rrbracket_{\rho[X\to R]} \) is a reducibility candidate for all \( R \in \mathcal{R} \). Since \( \mathcal{R} \) is a complete partial order, \( \bigcap_{R \in \mathcal{R}} \llbracket T' \rrbracket_{\rho[X\to R]} \) is then also a reducibility candidate.

\[ \Box \]

5 Soundness of Typing Rules

The goal of this section is to prove that terms which can be assigned a type using the rules of Figure 1 are normalizing. We will actually prove a stronger statement, based on an interpretation of typing judgments. First, we must define an interpretation \( \llbracket \cdot \rrbracket \) for typing contexts \( \Gamma \). This interpretation will be a set of pairs \((\sigma, \rho)\), where \( \rho \) is, as above, a function mapping type variables to reducibility candidates; and \( \sigma \) maps term variables to terms. The definition is by recursion on the structure of \( \Gamma \):

\[
(\sigma, \rho) \in \llbracket x : T, \Gamma \rrbracket \iff \sigma(x) \in \llbracket T \rrbracket_\rho \land (\sigma, \rho) \in \llbracket \Gamma \rrbracket \\
(\sigma, \rho) \in \llbracket X : *, \Gamma \rrbracket \iff \rho(x) \in \mathcal{R} \land (\sigma, \rho) \in \llbracket \Gamma \rrbracket \\
(\sigma, \rho) \in \llbracket \rrbracket 
\]

In the statement of the theorem below, we write \( \sigma t \) to mean the result of simultaneously substituting \( \sigma(x) \) for \( x \) in \( t \), for all \( x \) in the domain of \( \sigma \).

**Lemma 3.** Suppose \((\sigma, \rho) \in \llbracket \Gamma \rrbracket \). If \( t \in \llbracket T \rrbracket_\rho \), then \( (\sigma[x \mapsto t], \rho) \in \llbracket \Gamma, x : T \rrbracket \). Also, if \( R \in \mathcal{R} \), then \( (\sigma, \rho[x \mapsto R]) \in \llbracket \Gamma, X : * \rrbracket \).

**Proof.** The proof of the first part is by induction on \( \Gamma \). If \( \Gamma = \cdot \), then to show \( (\sigma[x \mapsto t], \rho) \in \llbracket \cdot, x : T \rrbracket \), it suffices to show \( t \in \llbracket T \rrbracket_\rho \), which holds by assumption. If \( \Gamma = y : T, \Gamma' \), then we have \( (\sigma, \rho) \in \llbracket \Gamma' \rrbracket \) by the definition of \( \llbracket \Gamma \rrbracket \), and we may apply the IH to conclude \( (\sigma[x \mapsto t], \rho) \in \llbracket \Gamma', x : T \rrbracket \), from which we can conclude the desired \( (\sigma[x \mapsto t], \rho) \in \llbracket \Gamma, x : T \rrbracket \), again by the definition of \( \llbracket \Gamma \rrbracket \). Similar reasoning applies if \( \Gamma = X : *, \Gamma' \). The proof of the second part of the lemma is exactly analogous.

**Theorem 4** (Soundness of typing rules with respect to the semantics). If \( \Gamma \vdash t : T \), then for all \((\sigma, \rho) \in \llbracket \Gamma \rrbracket \), we have \( \sigma t \in \llbracket T \rrbracket_\rho \).

**Proof.** The proof is by induction on the structure of the assumed typing derivation. In each case, we will implicitly assume an arbitrary \((\sigma, \rho) \in \llbracket \Gamma \rrbracket \).

Case: 

\[
\Gamma(x) = T \\
\Gamma \vdash x : T
\]

We proceed by inner induction on \( \Gamma \). If \( \Gamma \) is empty, then \( \Gamma(x) = T \) is false, and this case cannot arise. Suppose \( \Gamma \) is of the form \( x : T, \Gamma' \). Then \( \sigma(x) \in \llbracket T \rrbracket_\rho \) by definition of \( \llbracket \Gamma \rrbracket \), which suffices to prove the conclusion. Suppose \( \Gamma \) is of the form \( y : T, \Gamma' \), where \( y \neq x \), or of the form \( X : *, \Gamma' \). Then \( \Gamma'(x) = T \) and \((\sigma, \rho) \in \llbracket \Gamma' \rrbracket \), and we use the induction hypothesis to conclude \( \sigma x \in \llbracket T \rrbracket_\rho \).

Case: 

\[
\Gamma, x : T \vdash t : T' \\
\Gamma \vdash \lambda x. t : T \to T'
\]

To prove \( (\lambda x. \sigma t) \in \llbracket T \to T' \rrbracket_\rho \), it suffices to assume an arbitrary \( t' \in \llbracket T' \rrbracket_\rho \) and prove \( (\lambda x. \sigma t) t' \in \llbracket T' \rrbracket_\rho \). Since \( \llbracket T' \rrbracket_\rho \) is a reducibility candidate, it suffices to prove \( [t'/x] \sigma t \in \llbracket T' \rrbracket_\rho \), since \( (\lambda x. \sigma t) t' \leadsto [t'/x](\sigma t) \). But if
we let $\sigma' = \sigma[x \mapsto t']$, then we have $(\sigma', \rho) \in \llbracket \Gamma, x : T \rrbracket$ by Lemma 3, so we may apply the IH to conclude $\sigma t \in \llbracket T \rrbracket_{\rho}$, as required.

Case:

\[
\frac{\Gamma \vdash t : T_1 \to T_2 \quad \Gamma' \vdash t' : T_1}{\Gamma \vdash t' : T_2}
\]

By the IH, $\sigma t \in \llbracket T_1 \to T_2 \rrbracket_{\rho}$ and $\sigma t' \in \llbracket T_1 \rrbracket_{\rho}$. By the semantics of arrow types, this immediately implies $(\sigma t) (\sigma t') \in \llbracket T_2 \rrbracket_{\rho}$, as required.

Case:

\[
\frac{\Gamma, X : * \vdash t : T}{\Gamma \vdash t : \forall X. T}
\]

We must prove $\sigma t \in \llbracket \forall X. T \rrbracket_{\rho}$. By the semantics of universal types, it suffices to assume an arbitrary $R \in \mathcal{R}$, and prove $\sigma t \in \llbracket T \rrbracket_{\rho[X \mapsto R]}$. But this follows by the IH, which we can apply because $(\sigma, \rho[X \mapsto R]) \in \llbracket \Gamma, X : * \rrbracket$, by Lemma 3.

Case:

\[
\frac{\Gamma \vdash t : \forall X. T \quad \Gamma' \vdash T' : *}{\Gamma \vdash t : [T'/X]T}
\]

By the IH, we know $\sigma t \in \llbracket \forall X. T \rrbracket_{\rho}$, which by the semantics of universal types is equivalent to

\[
\sigma t \in \bigcap_{R \in \mathcal{R}} T_{\rho[X \mapsto R]}
\]  

(1)

Since $(\sigma, \rho) \in \llbracket \Gamma \rrbracket$, we may easily observe that $\rho$ is defined for all the free type variables of $T'$. So by Lemma 2, $\llbracket T' \rrbracket_{\rho} \in \mathcal{R}$. From the displayed formula above (1), we can conclude $\sigma t \in \llbracket T \rrbracket_{\rho[X \mapsto [T']_{\rho}]}. \text{ Now we must apply the following lemma, whose easy proof by induction on } T \text{ we omit, to conclude } \sigma t \in \llbracket [T'/X]T \rrbracket_{\rho}.$

Lemma 5. $\llbracket [T'/X]T \rrbracket_{\rho} = \llbracket T \rrbracket_{\rho[X \mapsto T']}$. 

\[\square\]