Hereditary Substitution for Stratified System F

Harley Eades and Aaron Stump

Computer Science
The University of Iowa

PSTT'10

Introduction

- Preliminaries.
  - Stratified System F.
  - Well-founded ordering on types.
  - Hereditary substitution function.
  - Hereditary substitution function for Stratified System F.
- Normalization of Stratified System F.
  - The interpretation of types.
  - Substitution for the interpretation of types.
  - Type-soundness with respect to the interpretation of types.

Stratified System F [Leivant, 1991]

- The Language.

$$\begin{array}{lcl} \text{Terms: } t & := & x \mid \lambda x : \phi.t \mid t\ t \mid \Lambda X : K.t \mid t[\phi] \\ \text{Types: } \phi & := & X \mid \phi \to \phi \mid \forall X : K.\phi \\ \text{Kinds: } K & := & *_0 \mid *_1 \mid \ldots \end{array}$$

- The Reduction Rules (Full β-Reduction).

$$\begin{array}{lcl} (\Lambda X : *_p.t)[\phi] & \rightsquigarrow & [\phi/X]t \\ (\lambda x : \phi.t)t' & \rightsquigarrow & [t'/x]t \end{array}$$

- All contexts are well-formed.

$$\frac{}{\cdot\ Ok} \qquad \frac{\Gamma\ Ok}{\Gamma, X : *_p\ Ok} \qquad \frac{\Gamma \vdash \phi : *_p \quad \Gamma\ Ok}{\Gamma, x : \phi\ Ok}$$

Kind checking rules - Stratified System F [Leivant, 1991]

- Our rules.

$$\frac{\Gamma(X) = *_p \quad p \leq q \quad \Gamma\ Ok}{\Gamma \vdash X : *_q} \qquad \frac{\Gamma \vdash \phi_1 : *_p \quad \Gamma \vdash \phi_2 : *_q}{\Gamma \vdash \phi_1 \to \phi_2 : *_{max(p,q)}}$$

$$\frac{\Gamma, X : *_q \vdash \phi : *_p}{\Gamma \vdash \forall X : *_q.\phi : *_{max(p,q)+1}}$$

- Leivant's Rules.

$$\frac{\Gamma(X) = *_p}{\Gamma \vdash X : *_p} \qquad \frac{\Gamma \vdash \phi_1 : *_p \quad \Gamma \vdash \phi_2 : *_q}{\Gamma \vdash \phi_1 \to \phi_2 : *_{max(p,q)}}$$

$$\frac{\Gamma, X : *_q \vdash \phi : *_p}{\Gamma \vdash \forall X : *_q.\phi : *_{max(p,q+1)}}$$

Type checking rules - Stratified System F [Leivant, 1991]

$$\frac{\Gamma(x) = \phi \quad \Gamma\ Ok}{\Gamma \vdash x : \phi} \qquad \frac{\Gamma, x : \phi_1 \vdash t : \phi_2}{\Gamma \vdash \lambda x : \phi_1 . t : \phi_1 \to \phi_2}$$

$$\frac{\Gamma \vdash t_1 : \phi_1 \to \phi_2 \quad \Gamma \vdash t_2 : \phi_1}{\Gamma \vdash t_1 t_2 : \phi_2} \qquad \frac{\Gamma, X : *_p \vdash t : \phi}{\Gamma \vdash \Lambda X : *_p . t : \forall X : *_p . \phi}$$

$$\frac{\Gamma \vdash t : \forall X : *_l . \phi_1 \quad \Gamma \vdash \phi_2 : *_l}{\Gamma \vdash t[\phi_2] : [\phi_2 / X]\phi_1}$$

Well-founded ordering on types

Definition (well-founded ordering on types)

The ordering $>_\Gamma$ is defined as the least relation satisfying the universal closures of the following formulas:

$$\begin{array}{lll} \phi_1 \to \phi_2 & >_\Gamma & \phi_1 \\ \phi_1 \to \phi_2 & >_\Gamma & \phi_2 \\ \forall X : *_l.\phi & >_\Gamma & [\phi'/X]\phi \text{ where } \Gamma \vdash \phi' : *_l. \end{array}$$

Theorem ($>_\Gamma$ is well-founded)

The ordering $>_\Gamma$ is well-founded on types ϕ such that $\Gamma \vdash \phi : *_l$ for some l.

Hereditary substitution function [Watkins et al., 2004]

- Syntax: $[t/x]^{\phi} t' = t''$.
- Like ordinary capture avoiding substitution.
- Except, if the substitution introduces a redex, then that redex is recursively reduced.
  - Example: $[(\lambda z : b.z)/x]^{b \to b} xy \; (\rightsquigarrow (\lambda z : b.z)y \rightsquigarrow [y/z]^{b} z) = y$.

Hereditary substitution function [Watkins et al., 2004]

Four main properties of the hereditary substitution function.

- It is a total function.
- If t'' is an introduction form and t' is not then $\phi \geq_\Gamma \phi'$.
- When a new redex is recursively reduced the type of the free-variable gets smaller.
- Given normal forms the hereditary substitution function will return a normal form. I.e. $[n/x]^\phi n' = n''$.

Lemma (Termination of the hereditary substitution function.)

If $\Gamma \vdash t : \phi$, and $\Gamma, x : \phi \vdash t' : \phi'$ then $[t/x]^\phi t' = t''$ and if t' is not a λ-abstraction or a Λ-abstraction and t'' is then $\phi \geq_\Gamma \phi'$.

Hereditary substitution function

$$
\begin{array}{lcl}
[t/x]^{\phi} x & = & t \\
[t/x]^{\phi} y & = & y, \text{ if } y \text{ is a variable distinct from } x. \\
[t/x]^{\phi} \lambda y : \phi'.t' & = & \lambda y : \phi'.([t/x]^{\phi} t') \\
[t/x]^{\phi} \Lambda X : *_l.t' & = & \Lambda X : *_l.([t/x]^{\phi} t') \\
[t/x]^{\phi} (t_1\ t_2) & = & \text{let } s_1 = ([t/x]^{\phi} t_1) \text{ in} \\
 & & \text{let } s_2 = ([t/x]^{\phi} t_2) \text{ in} \\
 & & \text{if } s_1 \equiv \lambda y : \phi'.s_1' \text{ for some } y \text{ and } s_1' \text{ and} \\
 & & t_1 \text{ is not a } \lambda\text{-abstraction then} \\
 & & \quad [s_2/y]^{\phi'} s_1' \quad \text{Note: } \phi >_{\Gamma} \phi' \\
 & & \text{else} \\
 & & \quad (s_1\ s_2) \\
[t/x]^{\phi} t'[\phi'] & = & \text{let } s_1 = [t/x]^{\phi} t' \text{ in} \\
 & & \text{if } s_1 \equiv \Lambda X : *_l.s_1', \text{ for some } X, s_1' \text{ and } \Gamma \vdash \phi' : *_q, \\
 & & \text{such that, } q \leq l \text{ and } t' \text{ is not a } \Lambda\text{-abstraction then} \\
 & & \quad [\phi'/X]s_1' \\
 & & \text{else} \\
 & & \quad s_1[\phi']
\end{array}
$$

The interpretation of types [Prawitz, 2006]

- The definition of the interpretation of types is a modified version of the interpretation defined by Prawitz.
- We define the meaning of open terms directly, where Prawitz defines the meaning of open terms via the meaning of all their ground instances.

The interpretation of types [Prawitz, 2006]

- The definition of the interpretation of types procedes in two parts.
- First, we define the interpretation of kindable types for normal terms. Which is defined as $n \in [\![\phi]\!]_\Gamma \Leftrightarrow \Gamma \vdash n : \phi$.
- Second, we extend the first part to non-normal terms. Which is defined as $t \in [\![\phi]\!]_\Gamma \Leftrightarrow t \leadsto^! n \in [\![\phi]\!]_\Gamma$.

Substitution for interpretation of types

Lemma (Substitution for the Interpretation of Types)

If $n' \in [\![\phi']\!]_{\Gamma,x:\phi,\Gamma'}$, $n \in [\![\phi]\!]_{\Gamma}$, then $[n/x]n' \rightsquigarrow^! \hat{n} \in [\![\phi']\!]_{\Gamma,\Gamma'}$ and if n' is not a λ-abstraction or a Λ-abstraction and $\hat{n}$ is, then $\phi \geq_{\Gamma,\Gamma'} \phi'$.

- The proof of the substitution lemma requires:
  - induction on the measure (ϕ, n') and
  - the central idea of hereditary substitution.

Lemma (Context Strengthening for Kinding,Context-Ok)

If $\Gamma, x : \phi', \Gamma' \vdash \phi : *_p$ with a derivation of depth d, then $\Gamma, \Gamma' \vdash \phi : *_p$. Also, if $\Gamma, x : \phi, \Gamma'$ Ok with a derivation of depth d, then Γ, Γ' Ok.

Proof - Substitution for interpretation of types

Throughout this proof we will refer to "if n' is not a λ-abstraction or a Λ-abstraction and $\hat{n}$ is then $\phi \geq_{\Gamma,\Gamma'} \phi'$" as **A**.

We abbreviate “definition of the interpretation of types” by DIT.

1.	$n' \equiv n'_1\ n'_2$	Assumption.
2.	$n'_1 \in [\![\phi'' \to \phi']\!]_{\Gamma,x:\phi,\Gamma'}$	By DIT,1.
3.	$n'_2 \in [\![\phi'']\!]_{\Gamma,x:\phi,\Gamma'}$	By DIT,1.
4.	$\Gamma, x : \phi, \Gamma' \vdash \phi'' \to \phi' : *_r$	By DIT,2.
5.	$\Gamma, \Gamma' \vdash \phi'' \to \phi' : *_r$	By Lemma 4,4.
6.	$(\phi, n'_1\ n'_2) > (\phi, n'_1)$	
7.	$(\phi, n'_1\ n'_2) > (\phi, n'_2)$	
8.	$[n/x]n'_1 \rightsquigarrow^! \hat{n}_1 \in [\![\phi'' \to \phi']\!]_{\Gamma,\Gamma'}$	By IH,6.
9.	$[n/x]n'_2 \rightsquigarrow^! \hat{n}_2 \in [\![\phi'']\!]_{\Gamma,\Gamma'}$	By IH,7.

We case split on whether or not $\hat{n}_1$ is a λ-abstraction.

Proof - Substitution for interpretation of types

10.	$\hat{n}_1 \not\equiv \lambda y : \phi''.z.$	Assumption.
11.	$\quad \hat{n}_1\,\hat{n}_2 \in [\![\phi']\!]_{\Gamma,\Gamma'}.$	By DIT,8,9.
12.	$\quad$ Take $\hat{n}_1\ \hat{n}_2$ for $\hat{n}$.	
13.	$\quad$ **A**	By 10.
14.	$\hat{n}_1 \equiv \lambda y : \phi''.z.$	Assumption.
15.	$\quad z \in [\![\phi']\!]_{\Gamma,\Gamma',y:\phi''}$	By DIT,14.
16.	$\quad \phi \geq_{\Gamma,\Gamma'} \phi'' \to \phi'$	By 8.
17.	$\quad \phi >_{\Gamma,\Gamma'} \phi''$	By 16,transitivity.
18.	$\quad \phi >_{\Gamma,\Gamma'} \phi'$	By 16,transitivity.
19.	$\quad [n/x]n' \rightsquigarrow^* (\lambda y : \phi''.z)\hat{n}_2 \rightsquigarrow [\hat{n}_2/y]z$	By 8,9,14.
20.	$\quad (\phi, \hat{n}_1) > (\phi'', [\hat{n}_2/y]z)$	By 17.
21.	$\quad [\hat{n}_2/y]z \rightsquigarrow^! \hat{z} \in [\![\phi']\!]_{\Gamma,\Gamma'}$	By 17,IH.
22.	$\quad$ **A**	By 18.

Concluding normalization

- We are now in a position to conclude normalization for stratified system F.

Theorem (Type Soundness for the Interpretation of Types)

If $\Gamma \vdash t : \phi$ *then* $t \in [\![\phi]\!]_\Gamma$.

Corollary

If $\Gamma \vdash t : \phi$ *then there exists a* n *such that* $t \rightsquigarrow^! n$ *and* $\Gamma \vdash n : \phi$.

Hereditary substitution and reducibility

- Type soudness for the interpretation of types seems simplier then type soundness for reducibility.

Theorem (Type Soundness for Reducibility)

For all $\sigma \in [\![\Gamma]\!]$, *if* $\Gamma \vdash t : \phi$ *then* $\sigma t \in [\![\phi]\!]$..

Theorem (Type Soundness for the Interpretation of Types)

If $\Gamma \vdash t : \phi$ *then* $t \in [\![\phi]\!]_\Gamma$.

Concluding remarks

- Future work.
  - Extend hereditary substitution to other type theories:
    - SSF with sum-types and commuting conversions.
    - SSF with equality types.
    - SSF with universe polymorphism.
    - Dependently typed lang. with large eliminations.
  - Extend to higher ordinals. Goal: System T.
- Thank you,
  - anonymous PSTT reviewers, and all of you for listening.
- Some revisions have been made to our paper since submitting our proceedings version.

  Please see:
  http://www.cs.uiowa.edu/∼heades/papers/pstt10_submission.pdf.