Type Preservation as a Confluence Problem

Aaron Stump, Garrin Kimmell, Ruba El Haj Omar
Computational Logic Center
The University of Iowa
Iowa City, Iowa, USA

Funding from U.S. National Science Foundation
TRELLYS project (Tim Sheard, Stephanie Weirich)

The Standard Approach to Typing

Syntax.

$$\begin{array}{lcl} \textit{types } T & ::= & A \mid T_1 \Rightarrow T_2 \\ \textit{terms } t & ::= & x \mid t_1 \; t_2 \mid \lambda x : T.t \end{array}$$

Operational semantics.

$t \rightarrow_c t'$, small-step reduction relation (concrete).

Typing relation.

$\Gamma \vdash t : T$, with Γ a *context*.

Type Preservation.

$$\Gamma \vdash t : T \;\; \wedge \;\; t \rightarrow_c t' \;\; \Longrightarrow \;\; \Gamma \vdash t' : T$$

Alternative: The Rewriting Approach to Typing

Proposed by Kuan, MacQueen, and Findler.

"A Rewriting Semantics for Type Inference", ESOP 2007.

Define typing as small-step <u>abstract</u> reduction $\rightarrow_a$.

Terms rewrite to their types.

$$\begin{array}{lcl} \lambda y : A.y & \rightarrow_a^* & A \Rightarrow A \\ (\lambda x : A \Rightarrow A.x)\ \lambda y : A.y & \rightarrow_a^* & A \Rightarrow A \\ \lambda x : A.\lambda y : A.x & \rightarrow_a^* & A \Rightarrow A \Rightarrow A \end{array}$$

Formulate metatheory in term-rewriting terms.

This Work

Build on Kuan et al.'s work by applying term-rewriting methods.

Prove confluence of combined reduction $\rightarrow_{ca} = \rightarrow_c \cup \rightarrow_a$.

Use decreasing diagrams.

Type Preservation is a corollary.

Ultimate goal : simplify metatheory of programming languages.

The Rewriting Approach to STLC

Syntax

To rewrite a term to type, need intermediate mixed terms.

$$
\begin{array}{lll}
\textit{types } T & ::= & A \mid T_1 \Rightarrow T_2 \\
\textit{mixed terms } m & ::= & x \mid m\, m' \mid \lambda x : T.\, m \mid A \mid T \Rightarrow m \\
\textit{mixed values } u & ::= & \lambda x : T.m \mid T \Rightarrow m \mid A
\end{array}
$$

$T \Rightarrow m$ is viewed as an abstract function.

Can be applied to arguments (concrete or abstract).

Example mixed terms:

$$
\begin{array}{l}
\lambda x : A.x \\
A \Rightarrow A \\
A \Rightarrow \lambda x : A.x \\
(\lambda x : A.x)\; A \\
(A \Rightarrow A)\; A
\end{array}
$$

Concrete and Abstract Reduction Relations

Concrete reduction ($\rightarrow_c$) is call-by-value.

Abstract reduction ($\rightarrow_a$) is nondeterministic.

$$\frac{}{E_c[(\lambda x : T.\, m)\ u] \ \rightarrow_c \ E_c[[u/x]m]}\ c(\beta)$$

$$\frac{}{E_a[\lambda x : T.\, m] \ \rightarrow_a \ E_a[T \Rightarrow [T/x]m]}\ a(\lambda)$$

$$\frac{}{E_a[(T \Rightarrow m)\ T] \ \rightarrow_a \ E_a[m]}\ a(\beta)$$

where:

$$\begin{array}{lcl} E_c & ::= & * \mid (E_c\ t) \mid (u\ E_c) \\ E_a & ::= & * \mid (E_a\ m) \mid (m\ E_a) \mid \lambda x : T.\, E_a \mid T \Rightarrow E_a \end{array}$$

An Example Abstract Reduction Sequence

Reduce a term to its type (redexes underlined):

$$\begin{array}{l}
\lambda x : (A \Rightarrow A).\, \underline{\lambda y : A.\, (x\ (x\ y))} \ \rightarrow_a \\
\underline{\lambda x : (A \Rightarrow A).\, A \ \Rightarrow \ (x\ (x\ A))} \ \rightarrow_a \\
(A \Rightarrow A) \ \Rightarrow \ A \ \Rightarrow ((A \Rightarrow A)\ \underline{((A \Rightarrow A)\ A)}) \ \rightarrow_a \\
(A \Rightarrow A) \ \Rightarrow \ A \ \Rightarrow \underline{((A \Rightarrow A)\ A)} \ \rightarrow_a \\
(A \Rightarrow A) \ \Rightarrow \ A \ \Rightarrow A
\end{array}$$

Use $a(\lambda)$ rule to turn λ to $\Rightarrow$.

$$\lambda y : A.\, (x\ (x\ y)) \ \rightarrow_a \ A \ \Rightarrow \ (x\ (x\ A))$$

Use $a(\beta)$ rule to reduce applications of abstract functions.

$$((A \Rightarrow A)\ A) \ \rightarrow_a \ A$$

Metatheory: Relation to Standard Typing

Define standard terms t.

$$t ::= x \mid \lambda x : T.\, t \mid t\ t'$$

Theorem. $x_1 : T_1, \cdots, x_n : T_n \vdash t : T$ iff $[T_1/x_1, \cdots, T_n/x_n]t \to_a^* T$.

Proof is similar to relating big-step and small-step concrete reduction.

Usual typing rules define big-step abstract reduction:

$$\frac{\Gamma \vdash t : T' \Rightarrow T \quad \Gamma \vdash t' : T'}{\Gamma \vdash t\ t' : T}$$

Metatheory: Type Preservation

Standard formulation:

$$\Gamma \vdash t : T \ \wedge\ t \rightarrow_c t' \implies \Gamma \vdash t' : T$$

Rewriting formulation:

$$m \rightarrow_a^* T \wedge m \rightarrow_c m' \implies m' \rightarrow_a^* T$$

Follows from confluence of $\rightarrow_{ca}$ for typable terms (prove now).

Confluence of Combined Reduction on Typable Terms

Initial Observations

Call m typable iff $m \rightarrow_a^* T$, for some T.

Confluence fails for non-typable terms.

$$
\begin{tikzcd}
 & (\lambda x : A.\lambda y : A.y)\ x \arrow[dl, "a"'] \arrow[dr, "c"] & \\
(A \Rightarrow \lambda y : A.y)\ x \arrow[d, "a"'] & & \lambda y : A.y \arrow[d, "a"'] \\
(A \Rightarrow A \Rightarrow A)\ x & & A \Rightarrow A
\end{tikzcd}
$$

We will prove confluence for typable terms.

Basic Properties of Abstract Reduction

Theorem. $\rightarrow_a$ is terminating.

Theorem. $\rightarrow_a$ has the diamond property.

Proof:

No critical overlap of $\rightarrow_a$-redexes.

No duplication or deletion of $\rightarrow_a$-redexes.

$$\frac{}{E_a[\lambda x : T.\, m] \;\rightarrow_a\; E_a[T \Rightarrow [T/x]m]}\; a(\lambda)$$

$$\frac{}{E_a[(T \Rightarrow m)\; T] \;\rightarrow_a\; E_a[m]}\; a(\beta)$$

Recall: Decreasing Diagrams [van Oostrom]

Assign labels to steps $e \to e'$.

Order labels using some well-founded ordering $<$.

Confluent if all local peaks completable to locally decreasing diagrams.

Local peak has form

$$s_1 \leftarrow_\alpha t \rightarrow_\beta s_2$$

Valley has form

$$s_1 \rightarrow^*_{\curlyvee\alpha} \rightarrow^=_\beta \rightarrow^*_{(\curlyvee\alpha)\cup(\curlyvee\beta)} \hat{t} \leftarrow^*_{(\curlyvee\alpha)\cup(\curlyvee\beta)} \leftarrow^=_\alpha \leftarrow^*_{\curlyvee\beta} s_2$$

<u>where:</u>

$$\curlyvee\alpha = \{\alpha' | \alpha' < \alpha\}$$
$$\rightarrow_A = \bigcup_{\alpha \in A} \rightarrow_\alpha$$

Confluence of Combined Reduction

Main Theorem. $\to_{ca}$ is confluent on typable terms.

To prove, apply decreasing diagrams with $a < c$.

- *aa*-peaks completable since $\to_a$ has diamond property.
- No non-trivial *cc*-peaks, by determinism of $\to_c$.
- Two simple forms for *ac*-peaks, where $m \to_a m'$, $u \to_a u'$.

$$\begin{array}{ccccc} E_c[(\lambda x : T.\, m')\ u] & \leftarrow_a & E_c[(\lambda x : T.\, m)\ u] & \to_c & E_c[[u/x]m] \\ E_c[(\lambda x : T.\, m)\ u'] & \leftarrow_a & E_c[(\lambda x : T.\, m)\ u] & \to_c & E_c[[u/x]m] \end{array}$$

- One more complex peak (next).

Crucial Decreasing Diagram for Combined Reduction

$$\begin{tikzcd}
& E_c[(\lambda x : T_1.m)\ u] \arrow[dl, "a"'] \arrow[dr, "c"] & \\
E_c[(T_1 \Rightarrow [T_1/x]m)\ u] \arrow[d, "a"', "*"] & & E_c[[u/x]m] \arrow[ddl, "a"', "*"] \\
E_c[(T_1 \Rightarrow [T_1/x]m)\ T_1] \arrow[dr, "a"'] & & \\
& E_c[[T_1/x]m] &
\end{tikzcd}$$

$u \rightarrow_a^* T_1$, since redex typable.

Locally decreasing, since $c > a$.

Concluding Confluence

All local peaks completable to locally decreasing diagrams.

Therefore, $\to_{ca}$ is confluent for typable terms.

Type Preservation for STLC established by rewriting techniques.

New proof method.

In the paper, apply to other example systems:

STLC+fix, STLC+$\forall$, Uniform-STC

For Uniform-STC, we apply APROVE and ACP for $\to_a$.

Quick Look: STLC with Type Inference

Extend types to include meta-variables α_i.

Now $a(\beta)$ rule uses narrowing to instantiate α_i:

$$\frac{\sigma \textit{ is mgu}(T_1, T_2)}{E_a[(T_1 \Rightarrow m) T_2] \ \rightarrow_a \ \sigma(E_a[m])} \ a(\beta)$$

Again establish confluence for typable terms.

Conclude Type Preservation.

Conclusion: A New Approach to Type Systems

Can use rewriting techniques to prove Type Preservation.

Proofs are different, qualitatively simpler, partly automatable.

A new agenda: redevelop Type Theory using rewriting approach.

1. Dependent types.
2. Curry-style systems.
    - Confluence fails.
    - Prove Type Preservation (rewriting form) directly.
3. Normalization proofs.
4. Soundness of symbolic simulation.
    - Natural operational formulation.
    - Soundness often not proved.