On Typed Lambda Definability and Normalization by Evaluation

Andreas Abel[1]

[1]Department of Computer Science and Engineering
Chalmers and Gothenburg University, Sweden

27th Agda Implementor's Meeting (AIM XXVII)
Chalmers Teknikparken, Gothenburg, Sweden
7 June 2018

Introduction

- *Given a function f of some type, is it definable in STLC?* (Replace simply-typed lambda calculus (STLC) by your favorite type theory.)
- Extended question: Can we decide whether f is STLC-definable?
- Trivial answer to original question:

$$f \text{ STLC-definable} \iff \exists t.\ (\!|t|\!) = f$$

- Modified question: Can we characterize the STLC-definable functions without referencing STLC-syntax?

A Universe of Types

- To talk about typed functions, we need a language of types.

ι	:	Base			base type
S, T, U	:	Ty	$::=$	$\iota \mid U \Rightarrow T$	simple type hierarchy

- Interpretation $(\!|\_|\!) : \mathsf{Ty} \to \mathsf{Set}$.

$(\!|\iota|\!)$	$=$	*parameter*	
$(\!|U \Rightarrow T|\!)$	$=$	$(\!|U|\!) \to (\!|T|\!)$	full (meta-theoretic) function space

- Our type language is parametrized by Base types and their interpretation.

Contexts

- Types of argument lists (contexts).

$$\begin{array}{lclll} \Gamma, \Delta & : & \mathsf{Cxt} & ::= & \emptyset & \text{empty context} \\ & & & \mid & \Gamma.U & \text{context extension} \end{array}$$

- Interpretation $(\!|\_|\!) : \mathsf{Cxt} \to \mathsf{Set}$.

$$\begin{array}{lcll} (\!|\emptyset|\!) & = & 1 & \text{unit set} \\ (\!|\Gamma.U|\!) & = & (\!|\Gamma|\!) \times (\!|U|\!) & \text{cartesian product} \end{array}$$

Contexts as Worlds

- Context thinning $\Gamma \leq \Delta$.
- For the sake of consistency with (record) subtyping (and to confuse the audience) I consider longer contexts as *smaller*.

$$\frac{}{\mathrm{id}_\Gamma : \Gamma \leq \Gamma} \qquad \frac{\tau : \Gamma \leq \Delta}{\mathrm{weak}_U\, \tau : \Gamma.U \leq \Delta} \qquad \frac{\tau : \Gamma \leq \Delta}{\mathrm{lift}_U\, \tau : \Gamma.U \leq \Delta.U}$$

- Makes the *category of contexts and order-preserving embeddings*.
- Interpretation $(\!|\_|\!) : \Gamma \leq \Delta \to (\!|\Gamma|\!) \to (\!|\Delta|\!)$ as sublist projection.

$$\begin{array}{lcllcl} (\!|\mathrm{id}_\Gamma|\!) & = & \mathrm{id}_{(\!|\Gamma|\!)} & : & (\!|\Gamma|\!) \to (\!|\Gamma|\!) \\ (\!|\mathrm{weak}_U\, \tau|\!) & = & (\!|\tau|\!) \circ \pi_1 & : & (\!|\Gamma.U|\!) \to (\!|\Delta|\!) \\ (\!|\mathrm{lift}_U\, \tau|\!) & = & (\!|\tau|\!) \times \mathrm{id}_{(\!|U|\!)} & : & (\!|\Gamma.U|\!) \to (\!|\Delta.U|\!) \end{array}$$

Kripke predicates in the world of contexts

- We define predicate $f \in [\![T]\!]_\Gamma$ on $f : (\!|\Gamma|\!) \to (\!|T|\!)$ such that
  1. (Monotonicity:) If $f \in [\![T]\!]_\Gamma$ and $\tau : \Delta \leq \Gamma$ then $f \circ (\!|\tau|\!) \in [\![T]\!]_\Delta$.
  2. (Kripke function space:) $f \in [\![U \to T]\!]_\Gamma$ iff $f \overset{\tau}{\cdot} d \in [\![T]\!]_\Delta$ for all $\tau : \Delta \leq \Gamma$ and $d \in [\![U]\!]_\Delta$.
     Herein: $(f \overset{\tau}{\cdot} d)\,\delta = f\,((\!|\tau|\!)\,\delta)\,(d\,\delta)$.
- Base case $[\![\iota]\!]_\Gamma$ is parameter (must be monotone!).

Theorem

A function $f : (\!|\Gamma|\!) \to (\!|T|\!)$ is STLC-definable iff it satisfies all Kripke predicates, i.e., $f \in [\![T]\!]_\Gamma$ no matter how $[\![\iota]\!]$ is chosen.

$\Rightarrow$ If $t : (\Gamma \vdash T)$ then $(\!|t|\!) \in [\![T]\!]_\Gamma$ (fundamental theorem of LR).
$\Leftarrow$ $\Sigma\, t : (\Gamma \vdash T).\ (\!|t|\!) = f$ is a Kripke predicate $f \in [\![T]\!]_\Gamma$ (term model).

Application: Refuting STLC-definability

Theorem

Boolean negation is not definable in STLC equipped with $\mathsf{true}, \mathsf{false} : \mathsf{Bool}$.

- Proof 1: Look at possible normal forms of type $\mathsf{Bool} \to \mathsf{Bool}$.
- Proof 2: Construct a Kripke countermodel.
  - Let $f \in [\![\mathsf{Bool}]\!]_\Gamma$ iff f is constant $\mathsf{true}/\mathsf{false}$ or a projection from $(\!|\Gamma|\!)$.
  - This is a Kripke model for STLC with $\mathsf{true}, \mathsf{false} : \mathsf{Bool}$.
  - Negation is neither constant nor a projection.
- By the connection between STLC-definability and normalization, these two proofs are somewhat "the same".

Theorem (Peirce not inhabited)

There is not closed STLC-term of type $((A \Rightarrow B) \Rightarrow A) \Rightarrow A$ *for some types* A, B.

Proof: Exercise!

Syntax and Interpretation of STLC

- Variables: index $x : \mathsf{Var}\ \Gamma\ T$ into the context.

$$\frac{}{\mathsf{v}_0 : \mathsf{Var}\ \Gamma.T\ T} \qquad \frac{\mathsf{v}_i : \mathsf{Var}\ \Gamma\ T}{\mathsf{v}_{i+1} : \mathsf{Var}\ \Gamma.U\ T}$$

- Interpretation $(\!|\_|\!) : \mathsf{Var}\ \Gamma\ T \to (\!|\Gamma|\!) \to (\!|T|\!)$ as projections.

$$\begin{array}{rcl} (\!|\mathsf{v}_0|\!) & = & \pi_2 \\ (\!|\mathsf{v}_{i+1}|\!) & = & (\!|\mathsf{v}_i|\!) \circ \pi_1 \end{array}$$

- Terms $t : \Gamma \vdash T$.

$$\frac{x : \mathsf{Var}\ \Gamma\ T}{x : \Gamma \vdash T} \qquad \frac{t : \Gamma.U \vdash T}{\lambda t : \Gamma \vdash U \Rightarrow T} \qquad \frac{t : \Gamma \vdash U \Rightarrow T \qquad u : \Gamma \vdash U}{t\,u : \Gamma \vdash T}$$

- Interpretation $(\!|\_|\!) : (\Gamma \vdash T) \to (\!|\Gamma|\!) \to (\!|T|\!)$.

$$\begin{array}{rcl} (\!|\lambda t|\!) & = & \mathsf{curry}\ (\!|t|\!) \\ (\!|t\,u|\!) & = & \mathsf{S}\ (\!|t|\!)\ (\!|u|\!) \end{array} \qquad \begin{array}{ll} \mathsf{curry}\ f\ (\gamma, d) & = f\ \gamma\ d \\ \mathsf{S}\ f\ g\ \gamma & = f\ \gamma\ (g\ \gamma) \end{array}$$

Fundamental theorem

- Extension to environments: $\rho \in [\![\Gamma]\!]_\Delta$ for $\rho : (\!|\Delta|\!) \to (\!|\Gamma|\!)$.

$$\begin{array}{lcl} \rho \in [\![\emptyset]\!]_\Delta & \iff & \text{true} \\ \rho \in [\![\Gamma.U]\!]_\Delta & \iff & \pi_1 \circ \rho \in [\![\Gamma]\!]_\Delta \text{ and } \pi_2 \circ \rho \in [\![U]\!]_\Delta \end{array}$$

Monotonicity: If $\rho \in [\![\Gamma]\!]_\Delta$ and $\tau : \Delta' \leq \Delta$ then $\rho \circ (\!|\tau|\!) \in [\![\Gamma]\!]_{\Delta'}$.

Theorem (Fundamental theorem of logical relations)

If $t : (\Gamma \vdash T)$ and $\rho \in [\![\Gamma]\!]_\Delta$ then $(\!|t|\!) \circ \rho \in [\![T]\!]_\Delta$.

- Prove this first for $x : \mathsf{Var}\ \Gamma\ T$ (easy).
- Then prove by induction on $t : \Gamma \vdash T$.
- Case $\lambda t : \Gamma \vdash U \Rightarrow T$: Show $\mathsf{curry}(\!|t|\!) \circ \rho \in [\![U \Rightarrow T]\!]_\Delta$. (Needs monotonicity!)
- Case $t\ u : \Gamma \vdash T$: Show $(\mathsf{S}\ (\!|t|\!)\ (\!|u|\!)) \circ \rho \in [\![T]\!]_\Delta$.

Term model

- Define $f \in [\![\iota]\!]_\Gamma$ as $\Sigma t : (\Gamma \vdash \iota).\ (\!|t|\!) = f$.

Theorem (Reflect/reify)

1. *If* $t : \Gamma \vdash T$ *then* $(\!|t|\!) \in [\![T]\!]_\Gamma$ *(reflect).*
2. *If* $f \in [\![T]\!]_\Gamma$ *then* $(\!|t|\!) = f$ *for some* $t : \Gamma \vdash T$ *(reify).*

- Prove simulateneously by induction on T.
- Discovery: does not introduce β-redexes!

Normal forms

- Define simultaneously $t : \mathsf{Ne}\ \Gamma\ T$ (neutral) and $t : \mathsf{Nf}\ \Gamma\ T$ (normal).

$$\frac{x : \mathsf{Var}\ \Gamma\ T}{x : \mathsf{Ne}\ \Gamma\ T} \qquad \frac{t : \mathsf{Ne}\ \Gamma\ (U \Rightarrow T) \qquad u : \mathsf{Nf}\ \Gamma\ U}{t\ u : \mathsf{Ne}\ \Gamma\ T}$$

$$\frac{t : \mathsf{Ne}\ \Gamma\ T}{t : \mathsf{Nf}\ \Gamma\ T} \qquad \frac{t : \mathsf{Nf}\ \Gamma.U\ T}{\lambda t : \mathsf{Nf}\ \Gamma\ (U \Rightarrow T)}$$

- Define $f \in [\![\iota]\!]_\Gamma$ as $\Sigma(t : \mathsf{Ne}\ \Gamma\ \iota).\ (\!|t|\!) = f$.

Theorem (Reflect/reify)

1. *If* $t : \mathsf{Ne}\ \Gamma\ T$ *then* $(\!|t|\!) \in [\![T]\!]_\Gamma$ *(reflect).*
2. *If* $f \in [\![T]\!]_\Gamma$ *then* $(\!|t|\!) = f$ *for some* $t : \mathsf{Nf}\ \Gamma\ T$ *(reify).*

Normalization by Evaluation

- Show $id_{(\!|\Gamma|\!)} \in [\![\Gamma]\!]_\Gamma$ (reflection!).
- Assume $t : \Gamma \vdash T$.
- By the fundamental theorem, $(\!|t|\!) \circ id : [\![T]\!]_\Gamma$.
- By reification, $(\!|t|\!) = (\!|v|\!)$ for some $v : \mathrm{Nf}\ \Gamma\ T$.

Conclusions

- Proof-relevant version of completeness proof of IPL.
- Implemented in Agda with a tiny bit of `--rewriting`.
  https://github.com/andreasabel/lambda-definability/tree/master/src-stlc
- Extension to sum types in progress:
  - Need Beth models to represent case trees.
  - Need lots of `--rewriting`.
- Extension to dependent types: still figuring out stuff.
  Related to McBride's *Outrageous But Meaningful Coincidences*?!

Related Work

- None of this is originally by me!
- Friedman / Plotkin: Logical relations.
- Jung, Tiuryn (TLCA 1993): More or less this formulation.
- Fiore et al. / Altenkirch, Dybjer, Hofmann, Scott: Extension to disjoint sum types.
- Altenkirch Kaposi 2016: Extension to Π-types.