Recursion

Homework

Piotr Polesiuk

May 7, 2014

You have to prove that during evaluation of a program with equi-recursive types no stuck-terms will occur, i.e., you should prove the following theorem:

Theorem (Progress). *If $\varnothing \vdash e \ : \ \tau$ then either e is a value or there exists an expression e' such that $e \ \to \ e'$.*

There are two variants of the task: informal and formal. You have to pick one of them, solve it, and send the solution to `Piotr.Polesiuk@cs.uni.wroc.pl`. Deadline is June 10, 2014.

1. Informal variant

- Add equi-recursive types to the simply-typed ς-calculus in the analogous way as they are defined in appendix A.
- Define a reduction semantics in call-by-value order.
- Give an informal proof of the progress theorem.
- Why is the *syntactic requirement* from appendix A important?

Hint Look at the hint for the formal variant.

2. Formal variant

Formalize the proof of the progress theorem for the calculus defined in appendix A in your favorite proof-assistant (preferred are: Coq and Agda). If you are using Coq, you may use definitions from the file `homework.v`.

Hint Two following lemmas would be useful:

Lemma. *If $\varnothing \vdash \tau_1 \ \equiv \ \tau_2$ then τ_1 has the form $\mu\alpha_1.\dots.\mu\alpha_n.\tau_1' \to \tau_1''$ iff τ_2 has the form $\mu\beta_1.\dots.\mu\beta_m.\tau_2' \to \tau_2''$.*

Lemma (Inversion). *If $\varnothing \vdash v \ : \ \mu\alpha_1.\dots.\mu\alpha_n.\tau_1' \to \tau_1''$ then the value v is a λ-abstraction.*

A. λ-calculus with recursive types and unit type

Syntax

e	$::=$	$x \mid \lambda x.e \mid e\ e \mid 1$	(expressions)
v	$::=$	$\lambda x.e \mid 1$	(values)
τ	$::=$	$\alpha \mid \tau \to \tau \mid \mathbb{1} \mid \mu\alpha.\tau$	(types)

Syntactic requirement: No type variable occurs directly under the μ construct, i.e., types of the form $\mu\alpha.\beta$ are forbidden.

Type-equating rules

$$\frac{}{\Delta \vdash \tau \ \equiv\ \tau}(\textsc{Eq-Refl}) \qquad \frac{\Delta \vdash \tau_2 \ \equiv\ \tau_1}{\Delta \vdash \tau_1 \ \equiv\ \tau_2}(\textsc{Eq-Symm}) \qquad \frac{(\tau_1 \equiv \tau_2) \in \Delta}{\Delta \vdash \tau_1 \ \equiv\ \tau_2}(\textsc{Eq-Ax})$$

$$\frac{\Delta \vdash \tau_1 \ \equiv\ \tau_2 \quad \Delta \vdash \tau_1' \ \equiv\ \tau_2'}{\Delta \vdash \tau_1 \to \tau_1' \ \equiv\ \tau_2 \to \tau_2'}(\textsc{Eq-Arrow}) \qquad \frac{\Delta, \mu\alpha.\tau \equiv \tau' \vdash \tau\{\!|\alpha \leftarrow \mu\alpha.\tau|\!\} \ \equiv\ \tau'}{\Delta \vdash \mu\alpha.\tau \ \equiv\ \tau'}(\textsc{Eq-Fix})$$

Typing rules

$$\frac{(x : \tau) \in \Gamma}{\Gamma \vdash x \ :\ \tau}(\textsc{T-Var}) \qquad \frac{\Gamma, x : \tau_1 \vdash e \ :\ \tau_2}{\Gamma \vdash \lambda x.e \ :\ \tau_1 \to \tau_2}(\textsc{T-Abs})$$

$$\frac{\Gamma \vdash e_1 \ :\ \tau_2 \to \tau_1 \quad \Gamma \vdash e_2 \ :\ \tau_2}{\Gamma \vdash e_1\ e_2 \ :\ \tau_1}(\textsc{T-App}) \qquad \frac{}{\Gamma \vdash 1 \ :\ \mathbb{1}}(\textsc{T-Unit})$$

$$\frac{\Gamma \vdash e \ :\ \tau_1 \quad \varnothing \vdash \tau_1 \ \equiv\ \tau_2}{\Gamma \vdash e \ :\ \tau_2}(\textsc{T-Conv})$$

Reduction semantics

$$\frac{}{(\lambda x.e)\ v \ \to\ e\{\!|x \leftarrow v|\!\}}(\textsc{E-Beta}) \qquad \frac{e_1 \ \to\ e_1'}{e_1\ e_2 \ \to\ e_1'\ e_2}(\textsc{E-Func}) \qquad \frac{e \ \to\ e'}{v\ e \ \to\ v\ e'}(\textsc{E-Arg})$$