CS 4110

Programming Languages & Logics

Lecture 20
Normalization

17 October 2016

Announcements

- Proof-writing workshop tomorrow night, 7pm, in Gates 310!

Type “Completeness”?

Are all well-behaved programs well-typed?

Normalization

The simply-typed lambda calculus enjoys a remarkable property:

Every well-typed program terminates.

Simply-Typed Lambda Calculus

Syntax

expressions	$e ::= x \mid \lambda x\!:\!\tau.\, e \mid e_1\, e_2 \mid ()$
values	$v ::= \lambda x\!:\!\tau.\, e \mid ()$
types	$\tau ::= \textbf{unit} \mid \tau_1 \to \tau_2$

Simply-Typed Lambda Calculus

Syntax

expressions	$e ::= x \mid \lambda x\!:\!\tau.\, e \mid e_1\, e_2 \mid ()$
values	$v ::= \lambda x\!:\!\tau.\, e \mid ()$
types	$\tau ::= \textbf{unit} \mid \tau_1 \to \tau_2$

Dynamic Semantics

$$E ::= [\cdot] \mid E\, e \mid v\, E$$

$$\frac{e \to e'}{E[e] \to E[e']} \qquad \frac{}{(\lambda x\!:\!\tau.\, e)\, v \to e\{v/x\}}$$

Simply-Typed Lambda Calculus

Static Semantics

$$\frac{}{\Gamma \vdash () : \mathbf{unit}}\ \textsc{T-Unit}$$

$$\frac{\Gamma(x) = \tau}{\Gamma \vdash x : \tau}\ \textsc{T-Var}$$

$$\frac{\Gamma, x : \tau \vdash e : \tau'}{\Gamma \vdash \lambda x : \tau.\, e : \tau \to \tau'}\ \textsc{T-Abs}$$

$$\frac{\Gamma \vdash e_1 : \tau \to \tau' \quad \Gamma \vdash e_2 : \tau}{\Gamma \vdash e_1\, e_2 : \tau'}\ \textsc{T-App}$$

Supporting Lemmas

Lemma (Inversion)

- *If* $\Gamma \vdash x : \tau$ *then* $\Gamma(x) = \tau$
- *If* $\Gamma \vdash \lambda x : \tau_1 . e : \tau$ *then* $\tau = \tau_1 \rightarrow \tau_2$ *and* $\Gamma, x : \tau_1 \vdash e : \tau_2$*.*
- *If* $\Gamma \vdash e_1\, e_2 : \tau$ *then* $\Gamma \vdash e_1 : \tau' \rightarrow \tau$ *and* $\Gamma \vdash e_2 ty \tau'$*.*

Supporting Lemmas

Lemma (Inversion)

- *If* $\Gamma \vdash x : \tau$ *then* $\Gamma(x) = \tau$
- *If* $\Gamma \vdash \lambda x : \tau_1 . e : \tau$ *then* $\tau = \tau_1 \rightarrow \tau_2$ *and* $\Gamma, x : \tau_1 \vdash e : \tau_2$*.*
- *If* $\Gamma \vdash e_1\, e_2 : \tau$ *then* $\Gamma \vdash e_1 : \tau' \rightarrow \tau$ *and* $\Gamma \vdash e_2 ty \tau'$*.*

Lemma (Canonical Forms)

- *If* $\Gamma \vdash v : \textbf{\textit{unit}}$ *then* $v = ()$
- *If* $\Gamma \vdash v : \tau_1 \rightarrow \tau_2$ *then* $v = \lambda x : \tau_1 . e$ *and* $\Gamma, x : \tau_1 \vdash e : \tau_2$*.*

First Attempt

Theorem (Normalization)

If $\vdash e : \tau$ *then there exists a value* v *such that* $e \rightarrow^* v$.

Logical Relations

Idea: define a set with the following properties:

- At base types the set contains all expressions satisfying some property.
- At function types, the set contains all expressions such that the property is preserved whenever we apply the function to an argument of appropriate type that is also in the set.

Logical Relations

Idea: define a set with the following properties:

- At base types the set contains all expressions satisfying some property.
- At function types, the set contains all expressions such that the property is preserved whenever we apply the function to an argument of appropriate type that is also in the set.

In our setting, the property will concern normalization...

Logical Relation

Definition (Logical Relation)

- $R_{\textbf{unit}}(e)$ iff $\vdash e : \textbf{unit}$ and e halts.
- $R_{\tau_1 \to \tau_2}(e)$ iff $\vdash e : \tau_1 \to \tau_2$ and e halts, and for every e' such that $R_{\tau_1}(e')$ we have $R_{\tau_2}(e\, e')$.

Supporting Lemmas

Lemma

If $R_\tau(e)$ then e halts.

Supporting Lemmas

Lemma

If $R_\tau(e)$ then e halts.

Lemma

If $\vdash e : \tau$ and $e \to e'$ then $R_\tau(e)$ iff $R_\tau(e')$.

Supporting Lemmas

Lemma

If $R_\tau(e)$ *then e halts.*

Lemma

If $\vdash e : \tau$ *and* $e \rightarrow e'$ *then* $R_\tau(e)$ *iff* $R_\tau(e')$.

Lemma (Goal)

If $\vdash e : \tau$ *then* $R_\tau(e)$

Main Lemma

Lemma (Goal – Strengthened)

If

- $x_1 : \tau_1, \ldots, x_k : \tau_k \vdash e : \tau$,
- v_1 *through* v_k *are values such that* $\vdash v_1 : \tau_1$ *through* $\vdash v_k : \tau_k$*, and*
- $R_{\tau_1}(v_1)$ *through* $R_{\tau_k}(v_k)$,

then $R_\tau(e\{v_1/x_1\} \ldots \{v_k/x_k\})$.