CS 4110

Programming Languages & Logics

Lecture 21
Advanced Types

19 October 2016

Announcements

- HW #5 due today at 11:59pm
- HW #6 out
- No Adrian office hours on Friday
- Guest lecture on Friday: Michael Roberts
- I'll be back on Monday, but another guest lecture: Yaron Minsky, Jane Street

Review

We've developed a type system for the λ-calculus and mathematical tools for proving its type soundness.

We also know how to extend the λ-calculus with new language features.

Today, we'll extend our *type system* with features commonly found in real-world languages: products, sums, references, and exceptions.

Products (Pairs)

Syntax

$$
\begin{aligned}
e &::= \cdots \mid (e_1, e_2) \mid \#1\, e \mid \#2\, e \\
v &::= \cdots \mid (v_1, v_2)
\end{aligned}
$$

Products (Pairs)

Syntax

$$
\begin{aligned}
e &::= \cdots \mid (e_1, e_2) \mid \#1\, e \mid \#2\, e \\
v &::= \cdots \mid (v_1, v_2)
\end{aligned}
$$

Semantics

$$E ::= \cdots \mid (E, e) \mid (v, E) \mid \#1\, E \mid \#2\, E$$

$$\frac{}{\#1\,(v_1, v_2) \rightarrow v_1} \qquad \frac{}{\#2\,(v_1, v_2) \rightarrow v_2}$$

Product Types

$$\tau_1 \times \tau_2$$

Product Types

$$\tau_1 \times \tau_2$$

$$\frac{\Gamma \vdash e_1 : \tau_1 \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash (e_1, e_2) : \tau_1 \times \tau_2}$$

Product Types

$$\tau_1 \times \tau_2$$

$$\frac{\Gamma \vdash e_1 : \tau_1 \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash (e_1, e_2) : \tau_1 \times \tau_2}$$

$$\frac{\Gamma \vdash e : \tau_1 \times \tau_2}{\Gamma \vdash \#1\, e : \tau_1}$$

$$\frac{\Gamma \vdash e : \tau_1 \times \tau_2}{\Gamma \vdash \#2\, e : \tau_2}$$

Product Types

$$\tau_1 \times \tau_2$$

$$\frac{\Gamma \vdash e_1 : \tau_1 \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash (e_1, e_2) : \tau_1 \times \tau_2}$$

$$\frac{\Gamma \vdash e : \tau_1 \times \tau_2}{\Gamma \vdash \#1\, e : \tau_1}$$

$$\frac{\Gamma \vdash e : \tau_1 \times \tau_2}{\Gamma \vdash \#2\, e : \tau_2}$$

Note the similarities to the natural deduction rules for conjunction (logical *and*).

Sums (Tagged Unions)

Syntax

$$e ::= \cdots \mid \mathsf{inl}_{\tau_1+\tau_2}\ e \mid \mathsf{inr}_{\tau_1+\tau_2}\ e \mid (\mathsf{case}\ e_1\ \mathsf{of}\ e_2 \mid e_3)$$

$$v ::= \cdots \mid \mathsf{inl}_{\tau_1+\tau_2}\ v \mid \mathsf{inr}_{\tau_1+\tau_2}\ v$$

Sums (Tagged Unions)

Syntax

$$e ::= \cdots \mid \mathsf{inl}_{\tau_1+\tau_2}\ e \mid \mathsf{inr}_{\tau_1+\tau_2}\ e \mid (\mathsf{case}\ e_1\ \mathsf{of}\ e_2 \mid e_3)$$
$$v ::= \cdots \mid \mathsf{inl}_{\tau_1+\tau_2}\ v \mid \mathsf{inr}_{\tau_1+\tau_2}\ v$$

Semantics

$$E ::= \cdots \mid \mathsf{inl}_{\tau_1+\tau_2}\ E \mid \mathsf{inr}_{\tau_1+\tau_2}\ E \mid (\mathsf{case}\ E\ \mathsf{of}\ e_2 \mid e_3)$$

$$\frac{}{\mathsf{case}\ \mathsf{inl}_{\tau_1+\tau_2}\ v\ \mathsf{of}\ e_2 \mid e_3 \rightarrow e_2\ v}$$

$$\frac{}{\mathsf{case}\ \mathsf{inr}_{\tau_1+\tau_2}\ v\ \mathsf{of}\ e_2 \mid e_3 \rightarrow e_3\ v}$$

Sum Types

$$\tau ::= \cdots \mid \tau_1 + \tau_2$$

Sum Types

$$\tau ::= \cdots \mid \tau_1 + \tau_2$$

$$\frac{\Gamma \vdash e : \tau_1}{\Gamma \vdash \mathsf{inl}_{\tau_1+\tau_2}\ e : \tau_1 + \tau_2}$$

$$\frac{\Gamma \vdash e : \tau_2}{\Gamma \vdash \mathsf{inr}_{\tau_1+\tau_2}\ e : \tau_1 + \tau_2}$$

Sum Types

$$\tau ::= \cdots \mid \tau_1 + \tau_2$$

$$\frac{\Gamma \vdash e : \tau_1}{\Gamma \vdash \mathsf{inl}_{\tau_1 + \tau_2}\ e : \tau_1 + \tau_2}$$

$$\frac{\Gamma \vdash e : \tau_2}{\Gamma \vdash \mathsf{inr}_{\tau_1 + \tau_2}\ e : \tau_1 + \tau_2}$$

$$\frac{\Gamma \vdash e : \tau_1 + \tau_2 \quad \Gamma \vdash e_1 : \tau_1 \to \tau \quad \Gamma \vdash e_2 : \tau_2 \to \tau}{\Gamma \vdash \mathsf{case}\ e\ \mathsf{of}\ e_1 \mid e_2 : \tau}$$

Example

$\text{let } f = \lambda a : \mathbf{int} + (\mathbf{int} \to \mathbf{int}).\ \text{case } a \text{ of } (\lambda y.\, y + 1) \mid (\lambda g.\, g\ 35) \text{ in}$

$\text{let } h = \lambda x : \mathbf{int}.\, x + 7 \text{ in}$

$f\,(\text{inr}_{\mathbf{int}+(\mathbf{int}\to\mathbf{int})}\ h)$

References

Syntax

$$
\begin{aligned}
e &::= \cdots \mid \mathsf{ref}\, e \mid\, !e \mid e_1 := e_2 \mid \ell \\
v &::= \cdots \mid \ell
\end{aligned}
$$

References

Syntax

$$
\begin{aligned}
e &::= \cdots \mid \mathsf{ref}\, e \mid !e \mid e_1 := e_2 \mid \ell \\
v &::= \cdots \mid \ell
\end{aligned}
$$

Semantics

$$
E ::= \cdots \mid \mathsf{ref}\, E \mid !E \mid E := e \mid v := E
$$

$$
\frac{\ell \notin dom(\sigma)}{\langle \sigma, \mathsf{ref}\, v \rangle \to \langle \sigma[\ell \mapsto v], \ell \rangle}
\qquad
\frac{\sigma(\ell) = v}{\langle \sigma,\ !\ell \rangle \to \langle \sigma, v \rangle}
$$

$$
\frac{}{\langle \sigma, \ell := v \rangle \to \langle \sigma[\ell \mapsto v], v \rangle}
$$

Reference Types

$$\tau ::= \cdots \mid \tau\ \textbf{ref}$$

Reference Types

$$\tau ::= \cdots \mid \tau\ \textbf{ref}$$

$$\frac{\Gamma \vdash e : \tau}{\Gamma \vdash \mathsf{ref}\ e : \tau\ \textbf{ref}}$$

Reference Types

$$\tau ::= \cdots \mid \tau\ \mathbf{ref}$$

$$\frac{\Gamma \vdash e : \tau}{\Gamma \vdash \mathsf{ref}\ e : \tau\ \mathbf{ref}}$$

$$\frac{\Gamma \vdash e : \tau\ \mathbf{ref}}{\Gamma \vdash\ !e : \tau}$$

Reference Types

$$\tau ::= \cdots \mid \tau\ \textbf{ref}$$

$$\frac{\Gamma \vdash e : \tau}{\Gamma \vdash \mathsf{ref}\, e : \tau\ \textbf{ref}}$$

$$\frac{\Gamma \vdash e : \tau\ \textbf{ref}}{\Gamma \vdash !e : \tau}$$

$$\frac{\Gamma \vdash e_1 : \tau\ \textbf{ref} \quad \Gamma \vdash e_2 : \tau}{\Gamma \vdash e_1 := e_2 : \tau}$$

Question

Is this type system sound?

Question

Is this type system sound?

Well... what is the type of a location ℓ?

Question

Is this type system sound?

Well... what is the type of a location ℓ? (Oops!)

Store Typings

Let Σ range over maps from locations to types

Store Typings

Let Σ range over maps from locations to types

$$\frac{\Gamma, \Sigma \vdash e : \tau}{\Gamma, \Sigma \vdash \mathsf{ref}\, e : \tau\ \textbf{ref}}$$

Store Typings

Let Σ range over maps from locations to types

$$\frac{\Gamma, \Sigma \vdash e : \tau}{\Gamma, \Sigma \vdash \mathsf{ref}\, e : \tau\ \textbf{ref}}$$

$$\frac{\Gamma, \Sigma \vdash e : \tau\ \textbf{ref}}{\Gamma, \Sigma \vdash\ !e : \tau}$$

Store Typings

Let Σ range over maps from locations to types

$$\frac{\Gamma, \Sigma \vdash e : \tau}{\Gamma, \Sigma \vdash \mathsf{ref}\, e : \tau\ \textbf{ref}}$$

$$\frac{\Gamma, \Sigma \vdash e : \tau\ \textbf{ref}}{\Gamma, \Sigma \vdash\ !e : \tau}$$

$$\frac{\Gamma, \Sigma \vdash e_1 : \tau\ \textbf{ref} \qquad \Gamma, \Sigma \vdash e_2 : \tau}{\Gamma, \Sigma \vdash e_1 := e_2 : \tau}$$

Store Typings

Let Σ range over maps from locations to types

$$\frac{\Gamma, \Sigma \vdash e : \tau}{\Gamma, \Sigma \vdash \mathsf{ref}\, e : \tau\ \textbf{ref}}$$

$$\frac{\Gamma, \Sigma \vdash e : \tau\ \textbf{ref}}{\Gamma, \Sigma \vdash !e : \tau}$$

$$\frac{\Gamma, \Sigma \vdash e_1 : \tau\ \textbf{ref} \quad \Gamma, \Sigma \vdash e_2 : \tau}{\Gamma, \Sigma \vdash e_1 := e_2 : \tau}$$

$$\frac{\Sigma(\ell) = \tau}{\Gamma, \Sigma \vdash \ell : \tau\ \textbf{ref}}$$

Reference Types Metatheory

Definition

Store σ is *well-typed* with respect to typing context Γ and store typing Σ, written $\Gamma, \Sigma \vdash \sigma$, if $dom(\sigma) = dom(\Sigma)$ and for all $\ell \in dom(\sigma)$ we have $\Gamma, \Sigma \vdash \sigma(\ell) : \Sigma(\ell)$.

Reference Types Metatheory

Definition

Store σ is *well-typed* with respect to typing context Γ and store typing Σ, written $\Gamma, \Sigma \vdash \sigma$, if $dom(\sigma) = dom(\Sigma)$ and for all $\ell \in dom(\sigma)$ we have $\Gamma, \Sigma \vdash \sigma(\ell) : \Sigma(\ell)$.

Theorem (Type soundness)

If $\cdot, \Sigma \vdash e : \tau$ and $\cdot, \Sigma \vdash \sigma$ and $\langle e, \sigma\rangle \rightarrow^* \langle e', \sigma'\rangle$ then either e' is a value, or there exists e'' and σ'' such that $\langle e', \sigma'\rangle \rightarrow \langle e'', \sigma''\rangle$.

Reference Types Metatheory

Definition

Store σ is *well-typed* with respect to typing context Γ and store typing Σ, written $\Gamma, \Sigma \vdash \sigma$, if $dom(\sigma) = dom(\Sigma)$ and for all $\ell \in dom(\sigma)$ we have $\Gamma, \Sigma \vdash \sigma(\ell) : \Sigma(\ell)$.

Theorem (Type soundness)

If $\cdot, \Sigma \vdash e : \tau$ and $\cdot, \Sigma \vdash \sigma$ and $\langle e, \sigma \rangle \rightarrow^* \langle e', \sigma' \rangle$ then either e' is a value, or there exists e'' and σ'' such that $\langle e', \sigma' \rangle \rightarrow \langle e'', \sigma'' \rangle$.

Lemma (Preservation)

If $\Gamma, \Sigma \vdash e : \tau$ and $\Gamma, \Sigma \vdash \sigma$ and $\langle e, \sigma \rangle \rightarrow \langle e', \sigma' \rangle$ then there exists some $\Sigma' \supseteq \Sigma$ such that $\Gamma, \Sigma' \vdash e' : \tau$ and $\Gamma, \Sigma' \vdash \sigma'$.

Landin's Knot

Using references, we (re)gain the ability define recursive functions!

Landin's Knot

Using references, we (re)gain the ability define recursive functions!

$$\begin{array}{l}\textbf{let}\ r = \text{ref}\ \lambda x.\,0\ \textbf{in}\\ r := \lambda x\!:\!\textbf{int}.\ \textbf{if}\ x = 0\ \textbf{then}\ 1\ \textbf{else}\ x \times (!r)\,(x-1)\end{array}$$

Fixed Points

Syntax

$$e ::= \cdots \mid \mathsf{fix}\, e$$

Fixed Points

Syntax

$$e ::= \cdots \mid \mathsf{fix}\; e$$

Semantics

$$E ::= \cdots \mid \mathsf{fix}\; E$$

$$\frac{}{\mathsf{fix}\; \lambda x\!:\!\tau.\, e \rightarrow e\{(\mathsf{fix}\; \lambda x\!:\!\tau.\, e)/x\}}$$

Fixed Points

Syntax

$$e ::= \cdots \mid \mathsf{fix}\ e$$

Semantics

$$E ::= \cdots \mid \mathsf{fix}\ E$$

$$\frac{}{\mathsf{fix}\ \lambda x{:}\tau.\, e \rightarrow e\{(\mathsf{fix}\ \lambda x{:}\tau.\, e)/x\}}$$

The typing rule for fix is on the homework...

Exceptions

Syntax

$$e ::= \cdots \mid \textbf{error} \mid \textbf{try}\ e\ \textbf{with}\ e$$

Exceptions

Syntax

$$e ::= \cdots \mid \textbf{error} \mid \textbf{try}\ e\ \textbf{with}\ e$$

Semantics

$$\frac{e_1 \to e_1'}{\textbf{try}\ e_1\ \textbf{with}\ e_2 \to \textbf{try}\ e_1'\ \textbf{with}\ e_2} \qquad \frac{}{E[\textbf{error}] \to \textbf{error}}$$

Exceptions

Syntax

$$e ::= \cdots \mid \textbf{error} \mid \textbf{try}\ e\ \textbf{with}\ e$$

Semantics

$$\frac{e_1 \rightarrow e_1'}{\textbf{try}\ e_1\ \textbf{with}\ e_2 \rightarrow \textbf{try}\ e_1'\ \textbf{with}\ e_2} \qquad \frac{}{E[\textbf{error}] \rightarrow \textbf{error}}$$

$$\frac{}{\textbf{try error with}\ e \rightarrow e}$$

Exceptions

Syntax

$$e ::= \cdots \mid \textbf{error} \mid \textbf{try}\ e\ \textbf{with}\ e$$

Semantics

$$\frac{e_1 \to e_1'}{\textbf{try}\ e_1\ \textbf{with}\ e_2 \to \textbf{try}\ e_1'\ \textbf{with}\ e_2} \qquad \frac{}{E[\textbf{error}] \to \textbf{error}}$$

$$\frac{}{\textbf{try error with}\ e \to e} \qquad \frac{}{\textbf{try}\ v\ \textbf{with}\ e \to v}$$

Exception Types

We don't need to add any new types...

$$\frac{}{\Gamma \vdash \textbf{error} : \tau}$$

$$\frac{\Gamma \vdash e_1 : \tau \qquad \Gamma \vdash e_2 : \tau}{\Gamma \vdash \textbf{try}\ e_1\ \textbf{with}\ e_2 : \tau}$$

Exception Metatheory

Lemma (Progress)

If $\vdash e : \tau$ *then either*

- e *is a value or*
- e *is* ***error*** *or*
- *there exists* e' *such that* $e \to e'$.

Exception Metatheory

Lemma (Progress)

If $\vdash e : \tau$ then either

- *e is a value or*
- *e is **error** or*
- *there exists e' such that $e \to e'$.*

Theorem (Soundness)

If $\vdash e : \tau$ and $e \to^* e'$ and $e' \not\to$ then either

- *e is a value or*
- *e is **error***