Effects for lazy languages

Dylan McDermott Alan Mycroft

Computer Laboratory, University of Cambridge
first.last@cl.cam.ac.uk

BCTCS 2017

Motivation

- Effect systems allow us to reason about program behaviour
- Previous work has been for call-by-value languages
- Call-by-need languages also have effects: nontermination, resource usage, unsafePerformIO, ...
- Can we design an effect system for a call-by-need language?

Effect systems

- Traditional effect systems: add a set of operations to the typing judgement

$$\Gamma \vdash 0 : \mathsf{int} \,\&\, \emptyset$$

$$\Gamma \vdash \mathsf{write}\, 0 : \mathsf{unit} \,\&\, \{\mathsf{write}\}$$

- Function types $A \xrightarrow{f} B$ have latent effects f.

$$\Gamma \vdash \lambda x.\, x : A \xrightarrow{\emptyset} A \,\&\, \emptyset$$

$$\Gamma \vdash \lambda x.\, x + \mathsf{read}\,() : A \xrightarrow{\{\mathsf{read}\}} A \,\&\, \emptyset$$

Effect systems

- Have a *preordered monoid* [Katsumata '14]

Set of *effects* Sequencing operation

$$\langle \mathcal{F}, \leq, \bullet, 1 \rangle$$

Subeffecting relation *Pure* effect

- Example (traditional effect systems): $\langle \mathcal{P}\,\Sigma, \subseteq, \cup, \emptyset \rangle$
- Example: $\mathcal{F} = \{\mathsf{writesFirst}, 1, \mathsf{readsFirst}\}$ where

$$\mathsf{writesFirst} \leq 1 \leq \mathsf{readsFirst}$$

$$\mathsf{readsFirst} \bullet f = \mathsf{readsFirst} \qquad 1 \bullet f = f \qquad \mathsf{writesFirst} \bullet f = \mathsf{writesFirst}$$

Call-by-value effect system

$$
\begin{array}{lrcl}
\text{Types} & A, B & ::= & b \mid A \xrightarrow{f} B \\
\text{Contexts} & \Gamma & ::= & \overline{x : A}
\end{array}
$$

$$
(\text{var})\ \frac{}{\Gamma \vdash x : A \,\&\, 1} \text{ if } (x : A) \in \Gamma
\qquad
(\text{sub})\ \frac{\Gamma \vdash e : A \,\&\, f}{\Gamma \vdash e : A \,\&\, f'} \text{ if } f \leq f'
$$

$$
(\text{abs})\ \frac{\Gamma, x : A \vdash e : B \,\&\, f}{\Gamma \vdash \lambda x.\, e : A \xrightarrow{f} B \,\&\, 1} \text{ if } x \notin \operatorname{dom} \Gamma
$$

$$
(\text{app})\ \frac{\Gamma \vdash e_1 : A \xrightarrow{f''} B \,\&\, f \qquad \Gamma \vdash e_2 : A \,\&\, f'}{\Gamma \vdash e_1\, e_2 : B \,\&\, f \bullet f' \bullet f''}
$$

Call-by-name effect system

$$
\begin{array}{llcl}
\text{Types} & A, B & ::= & b \mid A \xrightarrow{f', f} B \\
\text{Contexts} & \Gamma & ::= & \overline{x : \langle A \rangle_f}
\end{array}
$$

$$
(\text{var})\ \frac{}{\Gamma \vdash x : A \,\&\, f} \text{ if } (x : \langle A \rangle_f) \in \Gamma
\qquad
(\text{sub})\ \frac{\Gamma \vdash e : A \,\&\, f}{\Gamma \vdash e : A \,\&\, f'} \text{ if } f \leq f'
$$

$$
(\text{abs})\ \frac{\Gamma, x : \langle A \rangle_{f'} \vdash e : B \,\&\, f}{\Gamma \vdash \lambda x.\, e : A \xrightarrow{f', f} B \,\&\, 1} \text{ if } x \notin \mathrm{dom}\,\Gamma
$$

$$
(\text{app})\ \frac{\Gamma \vdash e_1 : A \xrightarrow{f', f''} B \,\&\, f \qquad \Gamma \vdash e_2 : A \,\&\, f'}{\Gamma \vdash e_1\, e_2 : B \,\&\, f \bullet f''}
$$

Call-by-need?

- Have to know where arguments are evaluated
- Easy for call-by-value and call-by-name, hard for call-by-need
- This is a contextual property: use a *coeffect* system [Petricek et al. '13]

Tracking uses of variables

- Want to know where each variable is used first
- First attempt: add a set of variables to the judgement
  - Doesn't provide enough information
- Need to use a set of traces

$$\boxed{\Gamma \,@\, R \vdash e : A}$$

where R is a set of lists of variables

$$x : \mathsf{int}, y : \mathsf{int} \,@\, \{xy\} \vdash x + y : \mathsf{int}$$

$$x : \mathsf{int} \,@\, \{\varepsilon\} \vdash \lambda y.\, x + y : (y : \mathsf{int}) \xrightarrow{\{xy\}} \mathsf{int}$$

$$x : \mathsf{int}, y : \mathsf{int} \,@\, \{x, xy\} \vdash \text{if } x \text{ then } x \text{ else } y : \mathsf{int}$$

Tracking uses of variables

$$\boxed{\Gamma \mathbin{@} R \vdash e : A}$$

$$(\text{var})\ \frac{}{\Gamma \mathbin{@} \{x\} \vdash x : A}\ \text{if } (x : A) \in \Gamma \qquad\qquad (\text{sub})\ \frac{\Gamma \mathbin{@} R \vdash e : A}{\Gamma \mathbin{@} R' \vdash e : A}\ \text{if } R \subseteq R'$$

$$(\text{abs})\ \frac{\Gamma, x : A \mathbin{@} R \vdash e : B}{\Gamma \mathbin{@} \{\varepsilon\} \vdash \lambda x.\, e : (x : A) \xrightarrow{R} B}\ \text{if } x \notin \operatorname{dom} \Gamma$$

Tracking uses of variables

$$\boxed{\Gamma \mathbin{@} R \vdash e : A}$$

$$\text{(var)}\ \frac{}{\Gamma \mathbin{@} \{x\} \vdash x : A}\ \text{if } (x : A) \in \Gamma \qquad \text{(sub)}\ \frac{\Gamma \mathbin{@} R \vdash e : A}{\Gamma \mathbin{@} R' \vdash e : A}\ \text{if } R \subseteq R'$$

$$\text{(abs)}\ \frac{\Gamma, x : A \mathbin{@} R \vdash e : B}{\Gamma \mathbin{@} \{\varepsilon\} \vdash \lambda x.\, e : (x : A) \xrightarrow{R} B}\ \text{if } x \notin \mathrm{dom}\,\Gamma$$

$$\text{(app)}\ \frac{\Gamma \mathbin{@} R \vdash e_1 : (x : A) \xrightarrow{R'} B \qquad \Gamma \mathbin{@} R'' \vdash e_2 : A}{\Gamma \mathbin{@} R \mathbin{+\!\!+} (R'[R''/x]) \vdash e_1\, e_2 : B[R''/x]}$$

Tracking uses of variables

$$\boxed{\Gamma \mathbin{@} R \vdash e : A}$$

$$\text{(var)}\ \frac{}{\Gamma \mathbin{@} \{x\} \vdash x : A}\ \text{if } (x : A) \in \Gamma \qquad \text{(sub)}\ \frac{\Gamma \mathbin{@} R \vdash e : A}{\Gamma \mathbin{@} R' \vdash e : A}\ \text{if } R \subseteq R'$$

$$\text{(abs)}\ \frac{\Gamma, x : A \mathbin{@} R \vdash e : B}{\Gamma \mathbin{@} \{\varepsilon\} \vdash \lambda x.\, e : (x : A) \xrightarrow{R} B}\ \text{if } x \notin \operatorname{dom} \Gamma$$

$$\text{(app)}\ \frac{\Gamma \mathbin{@} R \vdash e_1 : (x : A) \xrightarrow{R'} B \qquad \Gamma \mathbin{@} R'' \vdash e_2 : A}{\Gamma \mathbin{@} R \mathbin{+\!\!+} (R'[R''/x]) \vdash e_1\, e_2 : B[R''/x]}$$

Sound for call-by-need (but soundness is hard to define)

Call-by-need effect system

$$\boxed{\Gamma \mathbin{@} R \vdash e : A \,\&\, f}$$

- Similar to call-by-name effect system
- Change the effects of variables that are used
  - If a variable must have been used, it has effect 1
  - If it must not, it has effect f
  - Otherwise, it has effect $f \sqcup 1$

$$(\text{app})\ \frac{\Gamma' \mathbin{@} R \vdash e_1 : (x : A) \xrightarrow{f', R', f''} B \,\&\, f \qquad \Gamma'' \mathbin{@} R'' \vdash e_2 : A \,\&\, f'}{\Gamma \mathbin{@} R \mathbin{+\!\!+} (R'[R''/x]) \vdash e_1\, e_2 : B[R''/x] \,\&\, f \bullet f''}$$

Recursion

- Adding fix:

$$(\text{fix})\ \frac{\Gamma \mathbin{@} R \vdash e : (x : A[R/x]) \xrightarrow{R'} A}{\Gamma \mathbin{@} R + (R'[R/x]) \vdash \text{fix}\, e : A[R/x]}$$

- Enough information to do strictness analysis
- Adding effects is more complicated
  - The algebra of effects needs to have fixpoints
  - But this is also true for call-by-value

Conclusion

- Can give a type system that tracks uses of variables
- Use this information to track effects
- Can apply previous work on effect systems

Operational semantics

- Heaps ρ are ordered lists of pairs

$$x \mapsto \mathsf{val}\,\mathsf{true}, y \mapsto \mathsf{expr}\, y$$

- Judgement for well typed heaps

$$\Gamma \mathbin{@} \overline{R} \vdash \rho$$

Typing of expressions with heaps

$$\boxed{\Gamma \mid \rho \mathbin{@} R \vdash e : A}$$

$$\begin{aligned}
& x \mid y \mapsto \mathsf{expr}\,\mathsf{true}, x \mapsto \mathsf{expr}\,y \\
\overset{y}{\rightsquigarrow}\ & x \mid y \mapsto \mathsf{val}\,\mathsf{true}, x \mapsto \mathsf{expr}\,\mathsf{true} \\
\overset{x}{\rightsquigarrow}\ & \mathsf{true} \mid y \mapsto \mathsf{val}\,\mathsf{true}, x \mapsto \mathsf{val}\,\mathsf{true}
\end{aligned}$$

$$y : \mathsf{bool}, x : \mathsf{bool} \mathbin{@} \{x\} \vdash x : \mathsf{bool}$$

$$y : \mathsf{bool}, x : \mathsf{bool} \mid y \mapsto \mathsf{expr}\,\mathsf{true}, x \mapsto \mathsf{expr}\,y \mathbin{@} \{y \bullet x\} \vdash x : \mathsf{bool}$$