Coeffects
Types for tracking context-dependence

Tomas Petricek, University of Cambridge
tomas.petricek@cl.cam.ac.uk | @tomaspetricek

Joint work with Dominic Orchard and Alan Mycroft
Properties of computations

**Effects** on the environment

\[ \Gamma \vdash e : M^\sigma \tau \]

Memory, network, I/O, ...

Effect systems and monads

**Requirements** from the environment

\[ \mathcal{C}^\sigma \Gamma \vdash e : \tau \]

Resources, meta-data, contexts...

Coeffect systems and comonads
Effects ≠ Coeffects

Lambda abstraction rule

“In the rule for abstraction, the effect is empty because evaluation immediately returns the function, with no side effects. The effect on the function arrow is the same as the effect for the function body, because applying the function will have the same side effects as evaluating the body.” [Wadler and Thiemann, 2003]

Not true for context-dependent properties!
Motivating examples
Implicit parameters are coeffects

Dynamically scoped parameters \(?\text{param}\)

let ?\text{culture} = "en-US"

let print = fun (num) →
  printNumber num ?\text{culture} ?\text{format}

Abstraction splits requirements

\[
\begin{align*}
\Gamma, x : \tau_1 & \vdash e : \tau_2 \\
\Gamma & \vdash \lambda x. e : C^s \tau_1 \rightarrow \tau_2
\end{align*}
\]
Implicit parameters are coeffects

Dynamically scoped parameters ?param

let ?culture = "en-US"
let print = fun (num) →
  printNumber num ?culture ?format

Different typing for different uses

\[ C^{?\text{culture}} \Gamma \vdash \text{print} : C^{?\text{format}} \text{int} \rightarrow \text{unit} \]
\[ C^{\emptyset} \Gamma \vdash \text{print} : C^{?\text{format}, ?\text{culture}} \text{int} \rightarrow \text{unit} \]
Liveness is a coeffect

Is the entire variable context live or dead?

\[
\frac{x: \tau \in \Gamma}{C^1 \Gamma \vdash x: \tau}
\]

\[
\frac{n \in \{0,1,2, \ldots \}}{C^0 \Gamma \vdash n: \iota}
\]

Application propagates contexts

\[
\frac{C^r \Gamma \vdash e_1: C^t \tau_1 \rightarrow \tau_2 \quad C^s \Gamma \vdash e_2: \tau_1}{C^{r \lor (s \land t)} \Gamma \vdash e_1 \ e_2: \tau_2}
\]

foo 42  \quad (\textbf{fun} \ x \rightarrow \ x) \ 42 \quad (\textbf{fun} \ x \rightarrow 42) \ y
Semantics of liveness

Context with liveness annotation

\[ [C^r \Gamma \vdash e : \tau] = C^r (\tau_1 \times \cdots \times \tau_n) \to \tau \]

Modeled using indexed **Maybe** type

- Definitely dead: **Nothing**\[ C^0 \tau = 1 \]
- Maybe live: **Just** of context\[ C^1 \tau = \tau \]

Impossible using a monad!

\[ [C^r \Gamma \vdash e : \tau] = (\tau_1 \times \cdots \times \tau_n) \to M^\sigma \tau \]
Structural coeffects for **liveness**

Track liveness of individual variables

Use **structural** type system

Product $\times$ that mirrors variable structure

$$
\frac{C^{r\times s}(\Gamma, x: \tau_1) \vdash e: \tau_2}{C^r \Gamma \vdash \lambda x. e: C^s \tau_1 \rightarrow \tau_2}
$$

Add contraction, exchange, weakening etc.

$$
\frac{C^{r\times s}(x : \tau, y: \tau) \vdash e: \tau'}{C^{r\times s}(\tau) \vdash \{z/x\}\{z/y\}e: \tau'}
\frac{C^{r\times s}(\Gamma_1, \Gamma_2) \vdash e: \tau}{C^{s\times r}(\Gamma_2, \Gamma_1) \vdash e: \tau}
$$
Efficient data-flow language

Discrete-time computations over streams

Access past value using `prev` keyword

How many past values need to cache?

\[
\frac{\mathcal{C}^r \Gamma \vdash e : \tau}{\mathcal{C}^{r+1} \Gamma \vdash \text{prev } e : \tau}
\]

\[
\frac{\mathcal{C}^{\min(r,s)} (\Gamma, x : \tau_1) \vdash e : \tau_2}{\mathcal{C}^r \Gamma \vdash \lambda x. e : \mathcal{C}^s \tau_1 \rightarrow \tau_2}
\]

\[
\frac{\mathcal{C}^r \Gamma \vdash e_1 : \mathcal{C}^t \tau_1 \rightarrow \tau_2 \quad \mathcal{C}^s \Gamma \vdash e_2 : \tau_1}{\mathcal{C}^{\max(r,s+t)} \Gamma \vdash e_1 \ e_2 : \tau_2}
\]
Semantics of data-flow language

Comonadic semantics

Non-empty list
Over the domain

Non-empty list

\[ \text{NEList} \tau = \tau \times (1 + \text{NEList} \tau) \]

Indexed comonadic semantics

Indexed List type
Index specifies length

Indexed List type

\[
\begin{align*}
C^0 \tau &= \tau \\
C^1 \tau &= \tau \times (\tau) \\
C^2 \tau &= \tau \times (\tau \times \tau) \\
&\cdots
\end{align*}
\]
Some more theory
Coeffect systems

Flat and structural variants

Both are useful for some applications
Flat: Implicits, dataflow, cross-compilation
Structural: Liveness, security, taintedness

Semantics using indexed comonads

Generalize ordinary comonads
With more structure for context passing
Flat coeffect system

**Coeffect algebra** \((S, \oplus, \cap, \cup, \varepsilon)\) formed by a monoid \((S, \oplus, \varepsilon)\), semi-lattice \((S, \cup)\) and a binary operation \(\cap\).

\[\frac{x: \tau \in \Gamma}{C^\varepsilon \Gamma \vdash x: \tau}\]

\[\frac{C^r \Gamma \vdash e_1: C^t \tau_1 \to \tau_2}{C^{r \cup (s \oplus t)} \Gamma \vdash e_1 \ e_2: \tau_2}\]

\[\frac{C^r \Gamma \vdash e_1: C^t \tau_1 \to \tau_2}{C^r \Gamma \vdash \lambda x. e: C^s \tau_1 \to \tau_2}\]

\[\frac{C^r \Gamma \vdash e: \tau}{C^s \Gamma \vdash e: \tau} (r \leq s)\]
Semantics using **indexed comonads**

Family \( C^r A \) of mappings (data types) with

\[
\varepsilon : C^\varepsilon A \to A
\]

\[
\circ : (C^r A \to B) \to (C^s B \to C) \to (C^{r \oplus s} A \to C)
\]

and for context propagation

\[
m_{r,s} : C^r A \times C^s B \to C^{(r \sqcap s)} (A \times B)
\]

\[
n_{r,s} : C^{(r \sqcup s)} (A \times B) \to C^r A \times C^s B
\]
Structural coeffect system

Different operations for context passing

\[ m_{r,s} : \mathcal{C}^r A \times \mathcal{C}^s B \rightarrow \mathcal{C}^{(r\times s)}(\text{Node}(A, B)) \]
\[ n_{r,s} : \mathcal{C}^{(r\times s)}(\text{Node}(A, B)) \rightarrow \mathcal{C}^r A \times \mathcal{C}^s B \]
\[ \Delta_{r,s} : \mathcal{C}^{(r\sqcup s)} A \rightarrow \mathcal{C}^r A \times \mathcal{C}^s A \]

Structural application & abstraction

\[ \mathcal{C}^r \Gamma_1 \vdash e_1 : \mathcal{C}^t \tau_1 \rightarrow \tau_2 \]
\[ \mathcal{C}^s \Gamma_2 \vdash e_2 : \tau_1 \]
\[ \mathcal{C}^{r\times(s\oplus t)}(\Gamma_1, \Gamma_2) \vdash e_1 \; e_2 : \tau_2 \]
\[ \mathcal{C}^r \Gamma \vdash \lambda x. e : \mathcal{C}^s \tau_1 \rightarrow \tau_2 \]
Conclusions
Why **coeffects** matter

**Important properties** not connected before

Liveness, security tainting, provenance
Implicits or resource usage, cross-compilation
Efficient data-flow

What **coeffects** are the right way

Types for standard **lambda calculus**
Indexed comonads for wide-spread use?
BACKUP SLIDES
Syntactic equational theory

Using substitution \((\lambda x.e_2)e_1 \rightarrow e_2[x \leftarrow e_1]\)
Function “values” may still need context
So consider call-by-name evaluation

If \(C \cdot \Gamma \vdash e: \tau\) and \(e \rightarrow e'\) then \(C \cdot \Gamma \vdash e: \tau\) holds
  – If \(\varepsilon = T\) (for example, liveness)
  – If \(\cup = \cap = \oplus\) (for example, implicit params)

But dataflow does not use substitution