Scoped effects as parameterized algebraic theories

Cristina Matache, University of Edinburgh

Joint work with: Sam Lindley, Sean Moss, Sam Staton, Zhixuan Yang, Nick Wu

Algebraic effects

- Adding impure computation to the simply typed λ-calculus using:
  - operations
  - program equations

[Plotkin & Power]

- λ-calc. + operations is modelled using strong monads on cartesian closed categories [Moggi '91]

Algebraic vs. non-algebraic effects

throw — throws an exception — ~ algebraic

catch(x, y) — handles exceptions in x with y — ~ non-algebraic

Catch is not algebraic:

$$\mathrm{catch}(x, y) \gg\!= k \neq \mathrm{catch}(x \gg\!= k,\ y \gg\!= k)$$

catch is a handler for throw [Plotkin & Pretnar '09, '13]

Question: how do we treat catch as an operation?

Outline

1. Algebraic effects
2. Scoped effects
3. Parameterized algebraic theories
4. Scoped effects as parameterized theories (Contribution)

Algebraic Effects: Explicit nondeterminism (backtracking)

[Plotkin & Pretnar '09, '13]

Operations:

$\mathtt{or}(x, y)$	choice
$\mathtt{fail}$	failure

Equations:

$$\mathtt{or}(\mathtt{or}(x, y), z) = \mathtt{or}(x, \mathtt{or}(y, z))$$

$$\mathtt{or}(x, \mathtt{fail}) = \mathtt{or}(\mathtt{fail}, x) = x$$

Generic effects:

$\underline{\mathtt{or}} : \mathtt{unit} \to \mathtt{bool}$

$\underline{\mathtt{fail}} : \mathtt{unit} \to 0$

$$\begin{cases} \mathtt{or}(x, y) = \mathtt{if}\ \underline{\mathtt{or}}()\ \mathtt{then}\ x\ \mathtt{else}\ y \\ \underline{\mathtt{or}}() = \mathtt{or}(\mathtt{true}, \mathtt{false}) \end{cases}$$

Algebraic Effects: Explicit nondeterminism (backtracking)

Fix a set A. The intended model is:

Carrier: the set $List(A)$

Operations: $[\![or]\!] : List(A)^2 \longrightarrow List(A)$, $[\![or]\!](x, y) = x +\!\!+ y$

$[\![fail]\!] : 1 \longrightarrow List(A)$, $[\![fail]\!]() = []$

- $List(A)$ is a free model on A
- List extends to a strong monad

implementation / denotational semantics

Algebraic effects have:

- An equational reasoning system i.e. algebraic theories
- with semantic models
- s.t. equality in the theory is sound and complete
- Correspondence between theories and monads on Set

Question:
Equational reasoning for non-algebraic effects, like scoped effects?

E.g. $\mathsf{catch}\ (x, y) \gg\!= k \neq \mathsf{catch}\ (x \gg\!= k, y \gg\!= k)$

Outline

1. Algebraic effects
2. Scoped effects
3. Parameterized algebraic theories
4. Scoped effects as parameterized theories (Contribution)

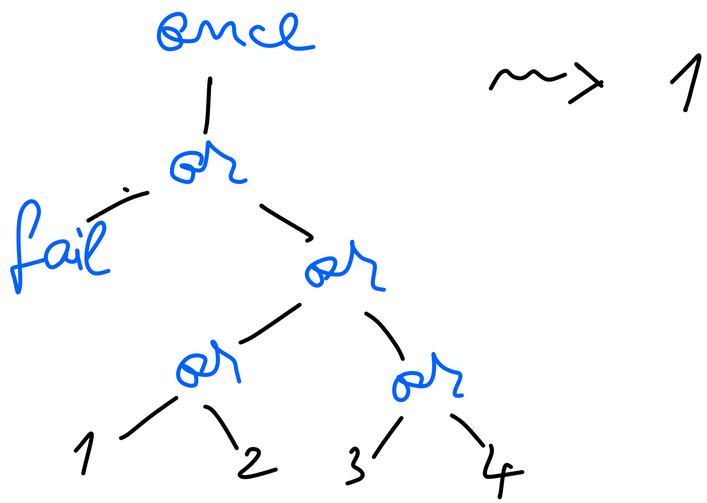
Scoped effects: Nondeterminism with once [Wu et al. '14]

Operations: `or(x, y)`, `fail`

`once(x)` chooses first non-failing branch of `x`

Example:

`once(or(fail, or(or(1, 2), or(3, 4))))`



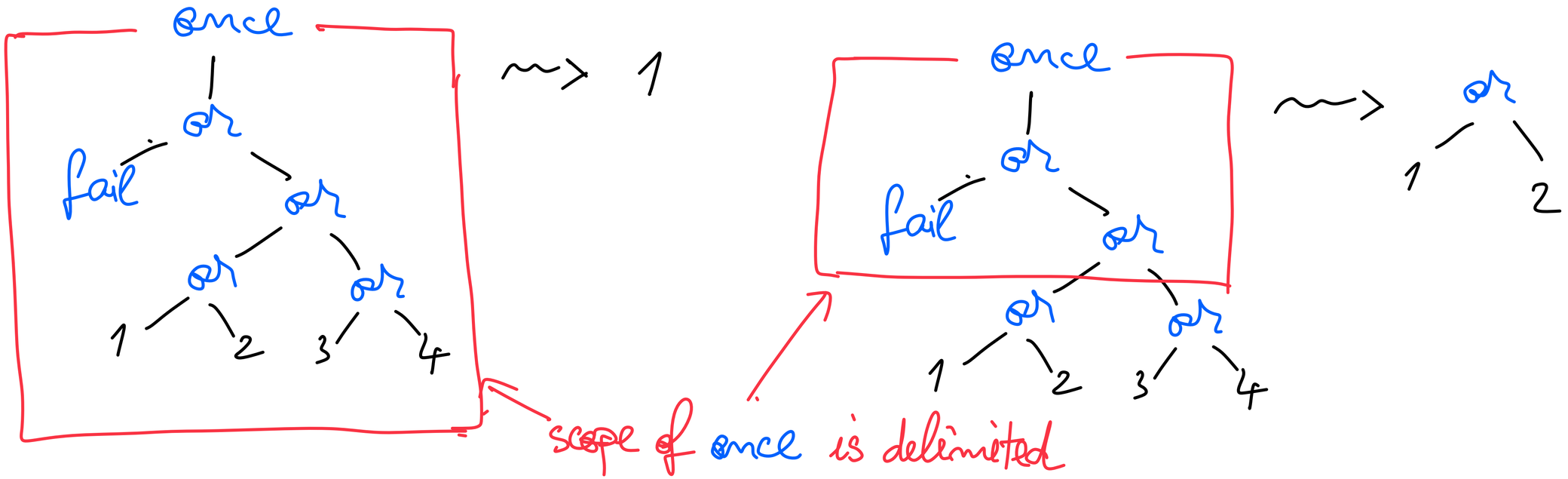
⇝ 1

Another scoped operation: `catch(x, y)`

Scoped effects: Nondeterminism with once

Once is not algebraic:

$$once(or(fail, or(1,3)) \gg\!= \lambda x.\, or(x, x+1)) \neq once(or(fail, or(1,3))) \gg\!= \lambda x.\, or(x, x+1)$$



Scoped effects : background

- Scoped effects look like handlers of algebraic effects

- Handling scoped effects ⇝ free monads from a signature
  E.g. [Wu et.al.'14], [Piróg et.al LICS'18], [Yang et.al. ESOP'22, ICFP'23]

Our contribution:

Equational theories for scoped effects, that generate monads.

Outline

1. Algebraic effects
2. Scoped effects
3. Parameterized algebraic theories
4. Scoped effects as parameterized theories (Contribution)

Parameterized algebraic theories [Staton FoSSaCS'13, LiCS'13, PoPL'15]

- Extend algebraic theories with binding of abstract parameters
- Uniform syntax for axiomatizing e.g.:

Example	Parameters
local state	location names

$new_0(a.x(a))$ create new location a, containing 0
↑ fresh parameter, bound

$read(a, x, y)$ read the bit stored in a
↑ free parameter

+ other operations and equations

Parameterized algebraic theories [Staton FoSSaCS'13, LiCS'13, PoPL'15]

- Extend algebraic theories with binding of abstract parameters
- Uniform syntax for axiomatizing e.g.:

Example	Parameters
local state	location names
π-calculus	channels
quantum computation	qubits

- Have canonical semantic status, similar to algebraic theories

Outline

1. Algebraic effects
2. Scoped effects
3. Parameterized algebraic theories
4. Scoped effects as parameterized theories (Contribution)

Scoped effects as parameterized algebraic theories

Contribution: Equational theories for scoped effects

Idea:

- A scoped effect = a parameterized theory
  where parameters are names of scopes

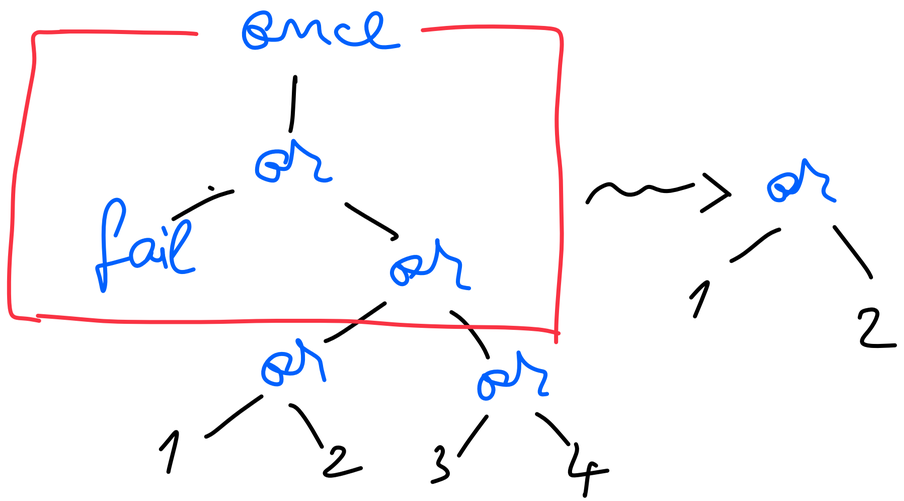
- Opening/closing scopes is explicit

Nondeterminism with once as a parameterized theory

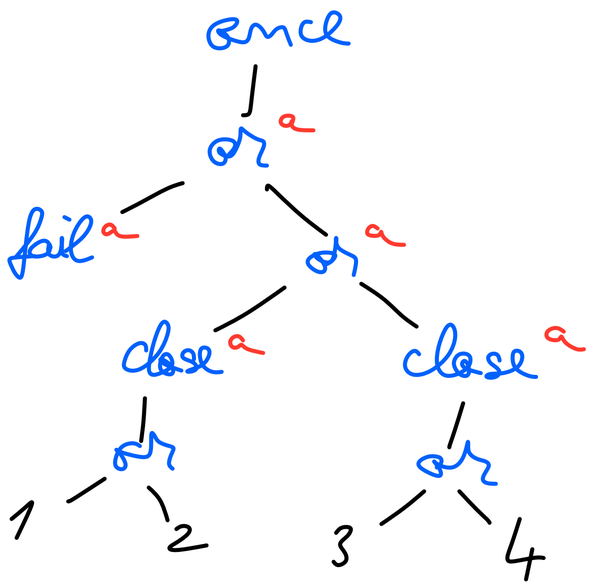
Operations: $or(x, y)$, $fail$, $once(a.x(a))$, $close(a, y)$

$once(or(fail, or(1, 3))) \gg= \lambda x.\, or(x, x+1)$

$once(a.\, or(fail, or(close(a, or(1, 2)), close(a, or(3, 4)))))$



becomes



Equations for nondeterminism with once

Explicit nondeterminism

$$or(or(x, y), z) = or(x, or(y, z))$$

$$or(x, fail) = x$$

$$or(fail, x) = x$$

Once / close

$$once(a.close(a, x)) = x$$

$$once(a.fail) = fail$$

$$once(a.or(x(a), x(a))) = once(a.x(a))$$

$$once(a.or(close(a, x), y(a))) = x$$

Example:

$$once(a.or(fail, or(close(a, or(1, 2)), close(a, or(3, 4))))) = or(1, 2)$$

Free model for nondeterminism with once

Fix a sequence of sets $X = (X_0, \emptyset, \dots \emptyset, \dots) \in \mathrm{Set}^{\mathbb{N}}$. The free model on X:

Carrier: the sequence $TX(n) = \mathrm{List}^{n+1}(X_0)$, for $n \in \mathbb{N}$

Operations: $\mathrm{once}(a.x(a))$, $\mathrm{close}(a, y)$, $\mathrm{or}(x, y)$, fail

$[\![\mathrm{once}]\!]_n : TX(n+1) \to TX(n)$ $\quad [\![\mathrm{once}]\!]_n([x, \dots]) = x$, $\quad [\![\mathrm{once}]\!]_n([]) = []$

$[\![\mathrm{close}]\!]_n : TX(n) \to TX(n+1)$ $\quad [\![\mathrm{close}]\!]_n(x) = [x]$

$[\![\mathrm{or}]\!]_n : TX(n)^2 \to TX(n)$ $\quad [\![\mathrm{or}]\!]_n(x_1, x_2) = x_1 + + x_2$

$[\![\mathrm{fail}]\!]_n : 1 \to TX(n)$ $\quad [\![\mathrm{fail}]\!]_n() = []$

— T extends to a monad on $\mathrm{Set}^{\mathbb{N}}$.

Theory of nondeterminism with once

Explicit nondeterminism

$or(or(x, y), z) = or(x, or(y, z))$

$or(x, fail) = x$

$or(fail, x) = x$

Once / close

$once(a.\, close(a, x)) = x$

$once(a.\, fail) = fail$

$once(a.\, or(x(a), x(a))) = once(a.\, x(a))$

$once(a.\, or(close(a, x), y(a))) = x$

Theorem. The model of [Lics'18] for nondeterminism with once,
(described there as an algebra for a monad)
is a free model for the parameterized theory above
(described on previous slide)

Other scoped effects we studied: [To appear]

- Exception catching
- Mutable state with local values

} Equational characterizations using parameterized theories

Future work

- More examples: backtracking with cut
- Programming in generic effect style for scoped effects
- Monad-theory correspondence for scoped effects (by restricting the correspondence for parameterized theories)