Algebraic operations and λ-calculus

Renato Neves

Universidade do Minho

HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

Table of Contents

Recalling λ-Calculus

$$\mathbb{A} \ni 1 \mid \mathbb{A} \times \mathbb{A} \mid \mathbb{A} \to \mathbb{A}$$

$$\frac{x : \mathbb{A} \in \Gamma}{\Gamma \vdash x : \mathbb{A}} \qquad \frac{}{\Gamma \vdash * : 1} \qquad \frac{\Gamma \vdash V : \mathbb{A} \times \mathbb{B}}{\Gamma \vdash \pi_1 V : \mathbb{A}}$$

$$\frac{\Gamma \vdash V : \mathbb{A} \qquad \Gamma \vdash U : \mathbb{B}}{\Gamma \vdash \langle V, U \rangle : \mathbb{A} \times \mathbb{B}} \qquad \frac{\Gamma, x : \mathbb{A} \vdash V : \mathbb{B}}{\Gamma \vdash \lambda x : \mathbb{A}.\, V : \mathbb{A} \to \mathbb{B}}$$

$$\frac{\Gamma \vdash V : \mathbb{A} \to \mathbb{B} \quad \Gamma \vdash U : \mathbb{A}}{\Gamma \vdash V\, U : \mathbb{B}}$$

Sequential Composition

A "new" deductive rule

$$\frac{\Gamma \vdash V : \mathbb{A} \qquad x : \mathbb{A} \vdash U : \mathbb{B}}{\Gamma \vdash x \leftarrow V; U : \mathbb{B}}$$

It reads as "bind the computation V to x and then run U"

Interpretation defined as

$$\frac{[\![\Gamma \vdash V : \mathbb{A}]\!] = f \qquad [\![x : \mathbb{A} \vdash U : \mathbb{B}]\!] = g}{[\![\Gamma \vdash x \leftarrow V; U : \mathbb{B}]\!] = g \cdot f}$$

Table of Contents

Signatures

Signature

A set $\Sigma = \{(\sigma_1, n_1), (\sigma_2, n_2), \dots\}$ of operations σ_i paired with the number of inputs n_i they are supposed to receive

Signatures will later be integrated in λ-calculus

They constitute the aforementioned algebraic operations

Examples

- Exceptions: $\{(\mathrm{e}, 0)\}$
- Read a bit from the environment: $\{(\mathrm{read}, 2)\}$
- Wait calls: $\{(\mathrm{wait}_n, 1) \mid n \in \mathbb{N}\}$
- Non-deterministic choice: $\{(+, 2)\}$

Algebraic operations in λ-calculus

We choose a signature Σ of algebraic operations and introduce a new deductive rule

$$\frac{(\sigma, n) \in \Sigma \qquad \forall i \leq n.\, \Gamma \vdash M_i : \mathbb{A}}{\Gamma \vdash \sigma(M_1, \ldots, M_n) : \mathbb{A}}$$

Examples of effectful λ-terms

- $x : \mathbb{A} \vdash \mathrm{wait}_1(x) : \mathbb{A}$ – adds delay of one second to returning x
- $\Gamma \vdash \mathrm{e}() : \mathbb{A}$ – raises an exception e
- $\Gamma \vdash \mathrm{write}_v(M) : \mathbb{A}$ – writes v in memory and then runs M
- $x : \mathbb{A} \times \mathbb{A} \vdash \mathrm{read}(\pi_1\, x, \pi_2\, x) : \mathbb{A}$ – receives a bit: if the bit is 0 it returns $\pi_1\, x$ otherwise it returns $\pi_2\, x$

Examples of effectful λ-terms

- $x : \mathbb{A} \vdash \mathrm{wait}_1(x) : \mathbb{A}$ – adds delay of one second to returning x
- $\Gamma \vdash \mathrm{e}() : \mathbb{A}$ – raises an exception e
- $\Gamma \vdash \mathrm{write}_v(M) : \mathbb{A}$ – writes v in memory and then runs M
- $x : \mathbb{A} \times \mathbb{A} \vdash \mathrm{read}(\pi_1\, x, \pi_2\, x) : \mathbb{A}$ – receives a bit: if the bit is 0 it returns $\pi_1\, x$ otherwise it returns $\pi_2\, x$

Exercise

Define a λ-term $x : \mathbb{A} \vdash ? : \mathbb{A}$ that requests a bit from the user and depending on the value read it returns x with either one or two seconds of delay.

Table of Contents

Semantics of λ-Calculus with algebraic Operations

How to provide semantics to these programming languages?

Short answer: via monads

Long answer: see the next slides ...

The core idea

Programs $\Gamma \vdash V : \mathbb{A}$ interpreted as functions

$$[\![\Gamma \vdash V : \mathbb{A}]\!] : [\![\Gamma]\!] \longrightarrow [\![\mathbb{A}]\!]$$

... and there exists only one function of type

$$[\![\Gamma]\!] \longrightarrow [\![1]\!]$$

Problem: it is then necessarily the case that

$$[\![\Gamma \vdash x : 1]\!] = [\![\Gamma \vdash \mathrm{wait}_1(x) : 1]\!]$$

despite these programs having different execution times

The core idea pt. II

Interpreted a program $\Gamma \vdash V : \mathbb{A}$ as a function

$$[\![\Gamma \vdash V : \mathbb{A}]\!] : [\![\Gamma]\!] \longrightarrow [\![\mathbb{A}]\!]$$

which returns values in $[\![\mathbb{A}]\!]$. But values now come with effects ...

Instead of having $[\![\mathbb{A}]\!]$ as set of outputs, we will have a set $T[\![\mathbb{A}]\!]$ of effectful values

$$[\![\Gamma \vdash M : \mathbb{A}]\!] : [\![\Gamma]\!] \longrightarrow T[\![\mathbb{A}]\!]$$

T should thus be a set-constructor: given a set of outputs X it returns a set of effectful values TX over X

The core idea pt. III

For wait calls, the corresponding set-constructor T is defined as

$$X \mapsto \mathbb{N} \times X$$

i.e. values in X paired with an execution time

For exceptions, the corresponding set-constructor T is defined as

$$X \mapsto X + \{e\}$$

i.e. values in X plus an element e representing the exception

Another problem

This idea of a set-constructor T seems good, but it breaks sequential composition

$$\begin{aligned} [\![\Gamma \vdash M : \mathbb{A}]\!] &: [\![\Gamma]\!] \to T[\![\mathbb{A}]\!] \\ [\![x : \mathbb{A} \vdash N : \mathbb{B}]\!] &: [\![\mathbb{A}]\!] \to T[\![\mathbb{B}]\!] \end{aligned}$$

We need a way to convert a function $h : X \to TY$ into a function of the type

$$h^{\star} : TX \to TY$$

Another problem pt. II

There are set-constructors T for which this is possible

In the case of wait-calls

$$\frac{f : X \to TY = \mathbb{N} \times Y}{f^\star(n, x) = (n + m, y) \text{ where } f(x) = (m, y)}$$

In the case of exceptions

$$\frac{f : X \to TY = Y + \{e\}}{f^\star(x) = f(y) \qquad f^\star(e) = e}$$

Testing the idea...

$$
\begin{aligned}
&[\![x : 1 \vdash y \leftarrow \mathrm{wait}_1(x); \mathrm{wait}_2(y) : 1]\!] \\
= \;& [\![y : 1 \vdash \mathrm{wait}_2(y) : 1]\!]^* \cdot [\![x : 1 \vdash \mathrm{wait}_1(x) : 1]\!] \\
= \;& (v \mapsto (2, v))^* \;\cdot\; (v \mapsto (1, v)) \\
= \;& v \mapsto (3, v)
\end{aligned}
$$

Yet another problem

Idea of interpreting λ-terms $\Gamma \vdash M : \mathbb{A}$ as functions

$$[\![\Gamma \vdash M : \mathbb{A}]\!] : [\![\Gamma]\!] \longrightarrow T[\![\mathbb{A}]\!]$$

looks good but it presupposes that all terms invoke effects

Some terms do not do this, *e.g.*

$$[\![x : \mathbb{A} \vdash x : \mathbb{A}]\!] : [\![\mathbb{A}]\!] \longrightarrow [\![\mathbb{A}]\!]$$

Solution

$T[\![\mathbb{A}]\!]$ should include effect-free values, and we should have

$$\eta_{[\![\mathbb{A}]\!]} : [\![\mathbb{A}]\!] \longrightarrow T[\![\mathbb{A}]\!]$$

which maps a value to its effect-free representation

Yet another problem pt. II

Again there are set-constructors T for which this is possible:

In the case of wait-calls

$$\frac{TX = \mathbb{N} \times X}{\eta_X(x) = (0, x)}$$

(*i.e.* no wait call was invoked)

In the case of exceptions

$$\frac{TX = X + \{e\}}{\eta_X(x) = x}$$

(*i.e.* the exception e was never raised)

Monads unlocked!!

Our previous analysis naturally leads to the notion of a monad

Monad

A triple $(T, \eta, (-)^\star)$ where T is a set-constructor, η a function $\eta_X : X \to TX$ for each set X, and $(-)^\star$ an operation

$$\frac{f : X \to TY}{f^\star : TX \to TY}$$

s.t. the following laws hold: $\eta^\star = \mathrm{id}$, $f^\star \cdot \eta = f$, $(f^\star \cdot g)^\star = f^\star \cdot g^\star$

These laws are required to forbid "weird" computational behaviour

Exercise

Show that the set-constructor

$$X \mapsto \mathbb{N} \times X$$

can be equipped with a monadic structure

Show that the set-constructor

$$X \mapsto X + 1$$

can be equipped with a monadic structure