

# Sound Operational Game Semantics for generative algebraic effects and handlers

---

Hamza Jaafar, Guilhem Jaber

September 27, 2025

Nantes Université, LS2N, INRIA Gallinette

## Notions of Computation Determine Monads

## Algebraic Operations and Generic Effects

Gordon Plotkin and John Power \*

Division of Informatics, University of Edinburgh, King's Buildings,  
Edinburgh EH9 3JZ, Scotland

**Abstract.** Given a complete and cocomplete symmetric monoidal closed category  $V$  and a symmetric monoidal  $V$ -category  $C$  with cotensors and a strong  $V$ -monad  $T$  on  $C$ , we investigate axioms under which an  $ObC$ -indexed family of operations of the form  $\alpha_x : (Tx)^r \rightarrow (Tx)^m$  provides semantics for algebraic operations on the computational  $\lambda$ -calculus. We recall a definition for which we have elsewhere given adequacy results, and we show that an enrichment of it is equivalent to a range of other possible natural definitions of algebraic operation. In particular, we define  $\alpha_x$  to give a generic effect is equiv-

**Main idea:** Computational effects are *specified* by an *algebraic theory*.

- *Signature:*

$$\mathbf{choose} : \tau \times \tau \rightarrow \tau \qquad \mathbf{fail} : \mathbb{1} \rightarrow$$

- *Equations*

$$\mathbf{choose}(t, t) \equiv t \qquad (\text{Idem})$$

$$\mathbf{choose}(t, u) \equiv \mathbf{choose}(u, t) \qquad (\text{Sym})$$

$$\mathbf{choose}(\mathbf{choose}(t, u), e) \equiv \mathbf{choose}(e, \mathbf{choose}(t, u)) \qquad (\text{Asso})$$

## Constructors of effects, concretely

Every operation symbol **op** comes with an arity

$$\mathbf{op} : \tau \rightarrow \sigma$$

defining an algebraic operation

$$\overline{\mathbf{op}}(v, \kappa) \quad v : \tau \text{ and } \kappa : v \rightarrow \varphi$$

or equivalently<sup>1</sup>, a *generic operation*

$$\mathbf{op} \ v$$

---

<sup>1</sup>by algebraicity,  $E[\mathbf{op} \ v] = \overline{\mathbf{op}}(v, \lambda x.E[x])$

## Handler as destructors of effects

A generalization of exception handlers (constructs such as **try**  $\dots$  **catch** or **try**  $\dots$  **with**) that can capture the *delimited continuation*<sup>2</sup>.

$$\begin{aligned} h = & \{ \mathbf{ret} \ x \mapsto u \} && (\textit{return clause}) \\ & \{ \dots, \overline{\mathbf{op}}_i(x, k) \mapsto t_i, \dots \} && (\mathbf{op} \textit{ clauses}) \end{aligned}$$

---

<sup>2</sup>Benton and Kennedy, "Exceptional Syntax".

## Handler as destructors of effects

A generalization of exception handlers (constructs such as **try**  $\cdots$  **catch** or **try**  $\cdots$  **with**) that can capture the *delimited continuation*<sup>2</sup>.

$$\begin{aligned} \mathbf{h} = & \{ \mathbf{ret} \ x \mapsto u \} && (\text{return clause}) \\ & \{ \cdots, \overline{\mathbf{op}}_i(x, k) \mapsto \tau_i, \cdots \} && (\mathbf{op} \text{ clauses}) \end{aligned}$$

$$\mathbf{handle}(\mathbf{ret} \ v) \ \mathbf{with} \ \mathbf{h} \ \rightarrow_v \ u\{v/x\}$$

$$\mathbf{handle} \ \mathbf{E}[\mathbf{op} \ v] \ \mathbf{with} \ \mathbf{h} \ \rightarrow_v \ \tau_i\{v/x\} \{(\lambda y. \mathbf{handle} \ \mathbf{E}[y] \ \mathbf{with} \ \mathbf{h})/k\}$$

when  $\mathbf{E}$  does not handle  $\mathbf{op}$

<sup>2</sup>Benton and Kennedy, "Exceptional Syntax".

Effect interfaces are given by signatures. e.g. **I/O**:

$$\{\mathbf{write} : \tau \rightarrow \mathbb{1}, \mathbf{read} : \mathbb{1} \rightarrow \tau\}$$

*What if we want multiple states holding values of type  $\tau$ , multiple open file descriptors, etc., without forfeiting modularity?*

# The Eff<sup>3</sup> approach

⇒ First-class identifier  $\iota$  for each *instance* of an effect  $\mathbb{E}$ .

- Performing an effect:  $\iota \# \text{op}_{\mathbb{E}} \ v$
- Handling an effect:  $\mathfrak{h} = \{ \iota \# \text{op}_{\mathbb{E}} \ p \ \kappa \mapsto \mathfrak{t} \}$

⇒ Dynamic generation:

$$\text{let } x \Leftarrow \text{new } \mathbb{E} \text{ in handle } \mathfrak{t} \text{ with } \mathfrak{h} \quad \rightarrow_v \quad \text{handle } \mathfrak{t} \{ \iota / x \} \text{ with } \mathfrak{h} \{ \iota / x \}$$

fresh instance  $\iota$

---

<sup>3</sup>Bauer and Pretnar, “Programming with algebraic effects and handlers”.

# Programming Language

---

## Fine-grained<sup>4</sup> call-by-value + algebraic effects & handlers

values  $v, w := x \mid \lambda x : \tau. t \mid \iota$

terms  $t, u :=$  **ret**  $v \mid v \ v \mid$  **match**  $v$  **with**  $(P_i \rightarrow u_i)_{i \in I}$   
| **let**  $x \leftarrow t$  **in**  $u$   
| **new**  $\mathbb{E} \mid v \# \text{op } w \mid$  **handle**  $t$  **with**  $h$

handlers  $h := \{\text{ret } x \mapsto t\} \mid \{v \# \text{op } x \ k \mapsto t\} \uplus h$

---

<sup>4</sup>Levy, *Call-By-Push-Value: A Functional/Imperative Synthesis*.

## Characterizing Contextual Equivalence

---

# Contextual Equivalence

We consider a standard notion of contextual equivalence  $\simeq_{ctx}$  whereby only termination is *observed*.

## Definition (Observation)

$t$  is terminating  
*if and only if*  
 $\exists v$  s.t.  $t \Downarrow_{op} \mathbf{ret} \ v$

## Definition (Contextual equivalence)

$t \simeq_{ctx} u \iff \forall C. C[t] \Downarrow_{op} \text{ iff } C[u] \Downarrow_{op}$

## Two equivalent encodings of state

Consider the standard encoding of state

$$h_{state}(\iota) := \left\{ \begin{array}{l} \mathbf{ret} \ x \mapsto \lambda s. \mathbf{ret} \ x; \\ \iota \# \mathbf{get} \ \langle \rangle \ k \mapsto \lambda s. k \ s \ s; \\ \iota \# \mathbf{set} \ x \ k \mapsto \lambda s. k \ \langle \rangle \ s \end{array} \right\}$$

---

<sup>5</sup>Biernacki et al., “Handle with care: relational interpretation of algebraic effects and handlers”.

## Two equivalent encodings of state

Consider the standard encoding of state

$$\mathbf{h}_{state}(\iota) := \left\{ \begin{array}{l} \mathbf{ret} \ x \mapsto \lambda s. \mathbf{ret} \ x; \\ \iota\#\mathbf{get} \ \langle \rangle \ k \mapsto \lambda s. k \ s \ s; \\ \iota\#\mathbf{set} \ x \ k \mapsto \lambda s. k \ \langle \rangle \ s \end{array} \right\}$$

Another elegant encoding due to Biernacki et al.<sup>5</sup>

$$\mathbf{h}_{init}(\iota) := \{\iota\#\mathbf{get} \ \langle \rangle \ k \mapsto k \ 0\}$$

$$\mathbf{h}_{set}(\iota) := \{\iota\#\mathbf{set} \ x \ k \mapsto \mathbf{handle} \ k \ \langle \rangle \ \mathbf{with} \ \{\iota\#\mathbf{get} \ \langle \rangle \ y \mapsto y \ x\}\}$$

---

<sup>5</sup>Biernacki et al., “Handle with care: relational interpretation of algebraic effects and handlers”.

$$\begin{aligned} & (\mathbf{handle} \ \tau \ \mathbf{with} \ h_{state}(\iota)) \ 0 \\ & \quad \simeq_{ctx} \\ & \mathbf{handle} \ \tau \ \mathbf{with} \ h_{init} \cdot h_{set}(\iota) \end{aligned} \tag{1}$$

# Interplay of generativity and contextual equivalence

$$\begin{aligned} & (\text{handle } \tau \text{ with } h_{state}(\iota)) 0 \\ & \quad \simeq_{ctx} \\ & \text{handle } \tau \text{ with } h_{init} \cdot h_{set}(\iota) \end{aligned} \tag{1}$$

$$\begin{aligned} & \text{let } x \leftarrow \text{new } \mathbb{E} \text{ in} \\ & (\text{handle } \tau \text{ with } h_{state}(x)) 0 \\ & \quad \not\approx_{ctx} \\ & \text{handle } \tau \text{ with } h_{init} \cdot h_{set}(\iota) \end{aligned} \tag{2}$$

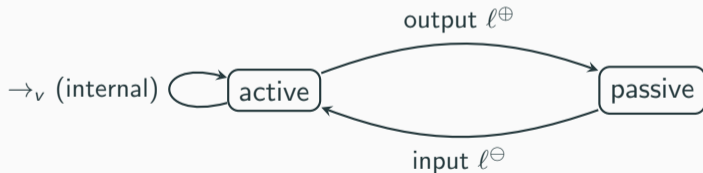
## Operational game semantics model

---

# Operational Game Semantics OGS

*Trace semantics* following the operational evaluation of a program (Proponent) and tracing its interaction with its environment (Opponent).

$\implies$  Interaction is modelled by a bi-partite transition system :



## Operational Game Semantics (cont.)

Normal Forms:

$$n ::= E[x \ v] \mid \mathbf{ret} \ v$$

- $\mathbf{ret} \ v$  calls for an answer.
- $E[x \ v]$  calls for a question.

$\implies$  Labels:

$$l ::= \langle \mathbf{ret} \ A \mid c \rangle \mid \langle x \ A \mid c \rangle$$

where  $c$  is an abstract continuation.

## Example 1

Let's consider the trace of

$$f(\lambda x.5)$$

representing the interaction with the environment given by the context represented by  $c$

**let**  $f \Leftarrow (\lambda g.g \ v; \mathbf{ret} \ \mathbf{tt})$  **in**  $[\ ]$

$$\langle f \ g \mid c \rangle^\oplus$$

## Example 1

Let's consider the trace of

$$f(\lambda x.5)$$

representing the interaction with the environment given by the context represented by  $c$

**let**  $f \Leftarrow (\lambda g.g \mathbf{v}; \mathbf{ret} \ \mathbf{tt})$  **in**  $[\ ]$

$$\langle f \ g \mid c \rangle^{\oplus} \langle g \ \mathbf{A} \mid d \rangle^{\ominus}$$

## Example 1

Let's consider the trace of

$$f(\lambda x.5)$$

representing the interaction with the environment given by the context represented by  $c$

**let**  $f \Leftarrow (\lambda g.g \mathbf{v}; \mathbf{ret} \ \mathbf{tt})$  **in**  $[\ ]$

$$\langle f \ g \mid c \rangle^{\oplus} \langle g \ \mathbf{A} \mid d \rangle^{\ominus} \langle \mathbf{ret} \ 5 \mid d \rangle^{\oplus}$$

## Example 1

Let's consider the trace of

$$f(\lambda x.5)$$

representing the interaction with the environment given by the context represented by  $c$

**let**  $f \Leftarrow (\lambda g.g \mathbf{v}; \mathbf{ret} \ \mathbf{tt})$  **in**  $[\ ]$

$$\langle f \ g \mid c \rangle^{\oplus} \langle g \ \mathbf{A} \mid d \rangle^{\ominus} \langle \mathbf{ret} \ 5 \mid d \rangle^{\oplus} \langle \mathbf{ret} \ \mathbf{tt} \mid c \rangle^{\ominus}$$

The denotation  $\llbracket t \rrbracket_{\text{ogs}}$  of a given program  $t$  is the set of all the traces it generates, *i.e.* all its execution runs against all possible environments.

### Definition (trace equivalence)

$$t \simeq_{tr} u \quad :\iff \quad \llbracket t \rrbracket_{\text{ogs}} = \llbracket u \rrbracket_{\text{ogs}}$$

Algebraic effects introduce new normal forms:

$$n ::= \dots \mid E[\iota\#\mathbf{op}\ v] \quad \text{when } \iota\#\mathbf{op} \notin \mathbf{hdl}(E)$$

$\implies$  It requires extending the interaction interface with new moves that account for observable effectful actions.

Algebraic effects introduce new normal forms:

$$n ::= \dots \mid E[\iota\#\mathbf{op} v] \quad \text{when } \iota\#\mathbf{op} \notin \mathbf{hdl}(E)$$

$\implies$  It requires extending the interaction interface with new moves that account for observable effectful actions.

But, what counts as *observable*?

## Accommodating the OGS model for effect name disclosure

When the program performs an effect

$\iota \# \text{op } v$

$\implies$  *What is the disclosure status of the instance  $\iota$ ?*

## Accommodating the *OGS* model for effect name disclosure

When the program performs an effect

$\iota\#\text{op } v$

$\implies$  *What is the disclosure status of the instance  $\iota$ ?*

- **Public:** Opponent could *potentially* handle the effect.
- **Private:** Opponent can only forward the effect to any enclosing Player's handling context.

## Accommodating the *OGS* model for effect name disclosure

$$\mathfrak{n} = \dots \mid \mathbf{E}[\iota\#\mathbf{op} \ v] \quad \text{when } \iota\#\mathbf{op} \notin \mathbf{hdl}(\mathbf{E})$$

⇒ New interactive moves capturing the delimited continuation  $\mathbf{E}$  and the effect:

- **private effect**      $\langle \kappa[e] \mid d \rangle$
- **public effect**      $\langle \kappa[\iota\#\mathbf{op} \ A] \mid c \rangle$

where  $\kappa$  is an abstract delimited continuation.

## Example 2

Now we look at how the term

**handle**  $f$  ( $\lambda x. \iota\#op \langle \rangle$ ) **with**  $\{\iota\#op \times k \mapsto k \ 5\}$

interacts with the same environment

**let**  $f \Leftarrow (\lambda g. g \ v; \mathbf{ret} \ \mathbf{tt})$  **in**  $[\ ]$

$\langle f \ g \mid c \rangle^\oplus$

## Example 2

Now we look at how the term

**handle**  $f$  ( $\lambda x. \iota\#op \langle \rangle$ ) **with**  $\{\iota\#op \times k \mapsto k\ 5\}$

interacts with the same environment

**let**  $f \Leftarrow (\lambda g. g\ v; \mathbf{ret}\ tt)$  **in**  $[\ ]$

$\langle f\ g \mid c \rangle^{\oplus} \langle g\ A \mid d \rangle^{\ominus}$

## Example 2

Now we look at how the term

**handle**  $f$  ( $\lambda x. \iota\#\mathbf{op} \langle \rangle$ ) **with**  $\{\iota\#\mathbf{op} \times k \mapsto k\ 5\}$

interacts with the same environment

**let**  $f \Leftarrow (\lambda g. g\ v; \mathbf{ret}\ tt)$  **in**  $[\ ]$

$\langle f\ g \mid c \rangle^{\oplus} \langle g\ A \mid d \rangle^{\ominus} \langle \kappa[e] \mid d \rangle^{\oplus}$

## Example 2

Now we look at how the term

**handle**  $f$  ( $\lambda x. \iota\#op \langle \rangle$ ) **with**  $\{\iota\#op \times k \mapsto k\ 5\}$

interacts with the same environment

**let**  $f \Leftarrow (\lambda g. g\ v; \mathbf{ret}\ tt)$  **in**  $[\ ]$

$$\langle f\ g \mid c \rangle^{\oplus} \langle g\ A \mid d \rangle^{\ominus} \langle \kappa[e] \mid d \rangle^{\oplus} \langle \kappa_d \cdot \kappa[e] \mid c \rangle^{\ominus}$$

## Example 2

Now we look at how the term

**handle**  $f$  ( $\lambda x. \iota\#op \langle \rangle$ ) **with**  $\{\iota\#op \times k \mapsto k\ 5\}$

interacts with the same environment

**let**  $f \Leftarrow (\lambda g. g\ v; \mathbf{ret}\ tt)$  **in**  $[\ ]$

$\langle f\ g \mid c \rangle^{\oplus} \langle g\ A \mid d \rangle^{\ominus} \langle \kappa[e] \mid d \rangle^{\oplus} \langle \kappa_d \cdot \kappa[e] \mid c \rangle^{\ominus} \langle \kappa_d[\mathbf{ret}\ 5] \mid c' \rangle^{\oplus}$

## Example 2

Now we look at how the term

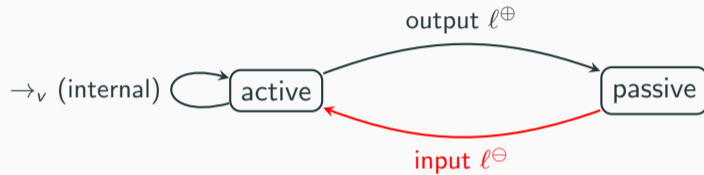
**handle**  $f$  ( $\lambda x. \iota\#op \langle \rangle$ ) **with**  $\{\iota\#op \times k \mapsto k\ 5\}$

interacts with the same environment

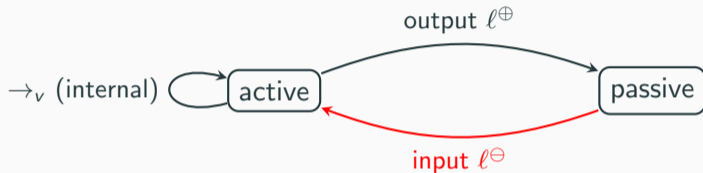
**let**  $f \Leftarrow (\lambda g. g\ v; \mathbf{ret}\ \mathbf{tt})$  **in**  $[\ ]$

$\langle f\ g \mid c \rangle^{\oplus} \langle g\ A \mid d \rangle^{\ominus} \langle \kappa[e] \mid d \rangle^{\oplus} \langle \kappa_d \cdot \kappa[e] \mid c \rangle^{\ominus} \langle \kappa_d[\mathbf{ret}\ 5] \mid c' \rangle^{\oplus} \langle \mathbf{ret}\ \mathbf{tt} \mid c' \rangle^{\ominus}$

## Constraining Opponent's behaviour

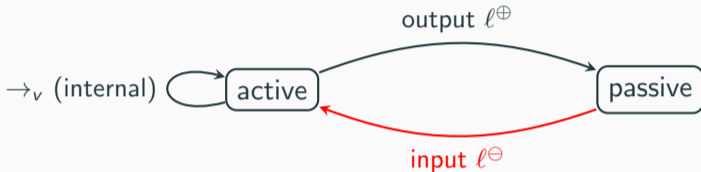


## Constraining Opponent's behaviour



**No constraints**  $\implies$  Opponent can simulate undelimited control (à la **call/cc**), and hence can discriminate more programs than any *real* context.

## Constraining Opponent's behaviour



**No constraints**  $\implies$  Opponent can simulate undelimited control (à la **call/cc**), and hence can discriminate more programs than any *real* context.

**Well-bracketed**  $\implies$  Opponent cannot simulate effect handlers, hence its discriminating powers are no match to *real* program contexts.

## Modelling delimited control flow

The capturing of a delimited continuation via a handler corresponds to the transition

$$\mathbf{handle} E_{n+1}[\kappa_n[E_n[\dots \kappa_0[E_0[\iota\#\mathbf{op} v]] \dots ]]] \mathbf{with} h \rightarrow_v \dots$$

where  $\iota\#\mathbf{op} \in \mathbf{hdl}(h)$

... hence, the *actual* resumable continuation is

$$\lambda x. (\mathbf{handle} E_{n+1}[\kappa_n[E_n[\dots \kappa_0[E_0[\mathbf{ret} x]] \dots ]]] \mathbf{with} h)$$

## Modelling delimited control flow

The capturing of a delimited continuation via a handler corresponds to the transition

$$\mathbf{handle} E_{n+1}[\kappa_n[E_n[\dots \kappa_0[E_0[\iota\#\mathbf{op} v]] \dots ]]] \mathbf{with} h \rightarrow_v \dots$$

where  $\iota\#\mathbf{op} \in \mathbf{hdl}(h)$

... hence, the *actual* resumable continuation is

$$\lambda x. (\mathbf{handle} E_{n+1}[\kappa_n[E_n[\dots \kappa_0[E_0[\mathbf{ret} x]] \dots ]]] \mathbf{with} h)$$

$\implies$  Stack discipline can be disrupted, but the fragments  $\kappa_i$  of the captured continuation can only be used in the order in which they have been disclosed.

## Definition 4 (Trace Equivalence)

$$t \simeq_{tr} u \iff \llbracket t \rrbracket_{ogs} = \llbracket u \rrbracket_{ogs}$$

## Theorem 5 (Soundness)

$$\simeq_{tr} \subseteq \simeq_{ctx}$$

- Contextual equivalence is more subtle in the presence of generativity of first-class effect instances.
- Extending the OGS model to account for effectful behaviour.
- Relaxing trace equivalence to coincide with the contextual one.

# QUESTIONS?

## Extra slides

---

- Impure behaviour given by operations on computations<sup>6</sup>  
(e.g chose for non-deterministic choice, raise for exceptions...)
- Impure behaviour is described by an equational theory on these operations
- Account for monadic effects whose behaviour is independent of the current evaluation context

$$\text{chose}(E[t], E[u]) \sim_{\text{op}} E[\text{chose}(t, u)]$$

- Easier to structure compared to combining monadic effects.
- Handlers arise as homomorphisms between models of such algebraic theories.

---

<sup>6</sup>Gordon Plotkin and John Power. “**Semantics for algebraic operations**”. In: *Electronic Notes in Theoretical Computer Science* 45 (2001), pp. 332–345.

$t$  is terminating  
*if and only if*  
 $\exists v$  s.t.  $t \Downarrow_{\text{op}} \mathbf{ret} \ v$

## Definition 6 (Contextual equivalence)

$$t \simeq_{\text{ctx}} u : \iff \forall C. C[t] \Downarrow_{\text{op}} \text{ iff } C[u] \Downarrow_{\text{op}}$$

## Disclosure and contextual equivalence

Consider the following variation of an example from<sup>7</sup>

$$f(\lambda x. 5)$$
$$\simeq_{ctx}$$

let  $y \leftarrow \text{new } \mathbb{E}$  in

handle

$$f(\lambda x. y\#op \langle \rangle)$$

with  $\{y\#op \ x \ \kappa \mapsto \kappa \ 5\}$

---

<sup>7</sup>Dariusz Biernacki et al. “**Handle with care: relational interpretation of algebraic effects and handlers**”. In: *Proceedings of the ACM on Programming Languages* 2.POPL (2017), pp. 1–30.

We need a coarser notion of trace equivalence in which

$$\begin{aligned}
 & \langle f \ g \mid c \rangle^{\oplus} \langle g \ A \mid d \rangle^{\ominus} \langle \mathbf{ret} \ d \mid 5 \rangle^{\oplus} \langle \mathbf{ret} \ c \mid \mathbf{tt} \rangle^{\ominus} \\
 & \qquad \qquad \qquad \simeq_{tr} \\
 & \langle f \ g \mid c \rangle^{\oplus} \langle g \ A \mid d \rangle^{\ominus} \langle \kappa[e] \mid d \rangle^{\oplus} \langle \mathcal{X} \cdot \kappa[e] \mid c \rangle^{\oplus} \langle k_d \ 5 \mid c' \rangle^{\oplus} \langle \mathbf{ret} \ c' \mid \mathbf{tt} \rangle^{\ominus}
 \end{aligned}$$

## Disclosure and contextual equivalence (cont.)

Now consider a variation of the previous example:

```
let  $y \leftarrow$  new  $\mathbb{E}$  in  $g\ y; f(\lambda x.5)$ 
```

$\not\sim_{ctx}$

```
let  $y =$ new  $\mathbb{E}$  in  
handle  
   $g\ y; f(\lambda x. y\#op\ \langle\rangle)$   
with  $\{y\#op\ x\ \kappa \mapsto \kappa\ 5\}$ 
```