Effectful Trace Semantics: Extended Abstract

Filippo Bonchi
Università di Pisa

Elena Di Lavore
Università di Pisa
Tallinn University of Technology

Mario Román
University of Oxford
Tallinn University of Technology

ABSTRACT

We introduce effectful streams, a coinductive semantic universe for effectful dataflow programming and traces. In monoidal categories with conditionals and ranges, we show that effectful streams particularize to families of morphisms satisfying a causality condition. Effectful streams allow us to develop notions of trace and bisimulation for effectful Mealy machines; we prove that bisimulation implies effectful trace equivalence.

Effectful traces. Traces are sequences that record the outputs of a transition system or a state machine along an execution. They constitute a successful and flexible formalism that can be adapted to the different flavours of transition systems (from *non-deterministic* to *stochastic*). Traces are also a semantic universe of interest for dataflow networks; in fact, they help extending Kahn's original model [Kah74] to a compositional semantics for the non-deterministic case [Jon89].

1 SETTING: EFFECTFUL CATEGORIES

The categorical setting we choose is that of effectful copy-discard categories: they allow us to distinguish between values, pure, and effectful morphisms.

Definition 1.1. An *effectful copy-discard category* is a triple of categories with two identity-on-objects functors, $\mathbf{V} \to \mathbf{P} \to \mathbf{C}$, where *(i)* $\mathbf{V}$ is a cartesian monoidal category; *(ii)* $\mathbf{P}$ is a monoidal category; and *(iii)* $\mathbf{C}$, is a premonoidal category.

The first identity-on-objects functor, $\mathbf{V} \to \mathbf{C}$, must preserve the monoidal structure strictly; the second identity-on-objects functor, $\mathbf{P} \to \mathbf{C}$, must preserve the premonoidal structure strictly.

Remark 1.2. Do-notation is the internal language of effectful copy-discard categories: it constructs the free such category over a signature.

$$\textsc{Return}\ \frac{\Gamma \vdash t_1 : X_1 \quad ... \quad \Gamma \vdash t_n : X_n}{\Gamma \Vdash \mathsf{return}(t_1, ..., t_n) : X_1, ..., X_n} \qquad \textsc{Variable}\ \frac{}{(x_i : X_i) \in \Gamma \vdash x_i : X_i}$$

$$\textsc{Value Generator}\ \frac{\Gamma \vdash t_1 : X_1 \quad ... \quad \Gamma \vdash t_n : X_n}{\Gamma \vdash f(t_1, ..., t_n) : Y}$$

$$\textsc{Pure Generator}\ \frac{y_1 : Y_1, ..., y_m : Y_m, \Gamma \Vdash p : Z_1, ..., Z_m \quad \Gamma \vdash t_1 : X_1, ..., \Gamma \vdash t_n : X_n}{\Gamma \Vdash g(t_1, ..., t_n) \to y_1, ..., y_m \fatsemi p : Z_1, ..., Z_m}$$

$$\textsc{Effectful Generator}\ \frac{y_1 : Y_1, ..., y_m : Y_m, \Gamma \Vdash p : Z_1, ..., Z_m \quad \Gamma \vdash t_1 : X_1, ..., \Gamma \vdash t_n : X_n}{\Gamma \Vdash h(t_1, ..., t_n) \rightsquigarrow y_1, ..., y_m \fatsemi p : Z_1, ..., Z_m}$$

For the last three rules, we have one instance of the rule, respectively, for each

- value, $f \in \mathcal{V}(X_1, ..., X_n; Y)$;
- pure morphism, $g \in \mathcal{P}(X_1, ..., X_n; Y_1, ..., Y_m)$;
- and effectful morphism $h \in \mathcal{E}(X_1, ..., X_n; Y_1, ..., Y_m)$.

2 EFFECTFUL STREAMS

We introduce a definition of stream in an effectful copy-discard category. An effectful stream consists of a process followed by an effectful stream: a "memory" object allows the proccess to communicate with the rest of the stream.

Definition 2.1. An *effectful stream*, $\boldsymbol{f} \colon \boldsymbol{A} \to \boldsymbol{B}$, over an effectful category $(\mathbf{P}, \mathbf{C})$ with inputs $\boldsymbol{A} = (A_0, A_1...)$ and outputs $\boldsymbol{B} = (B_0, B_1, ...)$ is a triple consisting of

- $M_f \in \mathbf{P}_{obj}$, the *memory*;
- $f^\circ \colon \boldsymbol{A}^\circ \rightsquigarrow M_f \otimes \boldsymbol{B}^\circ$, the *first action*, or *head*;
- $f^+ \colon M_f \cdot \boldsymbol{A}^+ \rightsquigarrow \boldsymbol{B}^+$, the *tail of the stream.*

Effectful streams are quotiented by dinaturality over the memory: the minimal equivalence $(\sim)$ such that $(M_f, f^\circ, f^+) \sim (M_g, g^\circ, g^+)$, for each pure morphism $r \colon M_g \to M_f$ in $\mathbf{P}$ such that $g^\circ \fatsemi r = f^\circ$ and $r \cdot f^+ \sim g^+$. Effectful streams from $\boldsymbol{A}$ to $\boldsymbol{B}$, quotiented by dinaturality form a set, $\mathsf{Stream}(\mathbf{P}, \mathbf{C})(\boldsymbol{A}; \boldsymbol{B})$.

Remark 2.2. Definition 2.1 can be recast in coalgebraic terms. The set $\mathsf{Stream}(\mathbf{P}, \mathbf{C})(\boldsymbol{A}; \boldsymbol{B})$ of effectful streams from $\boldsymbol{A}$ to $\boldsymbol{B}$ in the effectful category $(\mathbf{P}, \mathbf{C})$ is the final fixpoint of the functor $\phi \colon [(\mathbf{C}^\omega)^{\mathrm{o}} \times \mathbf{C}^\omega, \mathbf{Set}] \to [(\mathbf{C}^\omega)^{\mathrm{o}} \times \mathbf{C}^\omega, \mathbf{Set}]$ defined by

$$\phi(\mathrm{Q})(\boldsymbol{A}, \boldsymbol{B}) \coloneqq \int^{M \in \mathbf{P}} \hom_{\mathbf{C}}(\boldsymbol{A}^\circ; M \otimes \boldsymbol{B}^\circ) \times \mathrm{Q}(M \cdot \boldsymbol{A}^+; \boldsymbol{B}^+) \ .$$

2.1 Characterization as Causal Processes

Even if the definition seems technical, effectful streams still coincide with a notion of causal process in monoidal categories.

Definition 2.3. A *causal process*, $\boldsymbol{f} \colon \boldsymbol{X} \to \boldsymbol{Y}$, in a copy-discard category $(\mathbf{V}, \mathbf{P})$, is a family of morphisms

$$f_n \colon X_0 \otimes \ldots \otimes X_n \to Y_0 \otimes \ldots \otimes Y_n$$

that are *causal*: each $f_n \otimes \multimap$ is a marginal of f_{n+1}. Explicitly, there must exist morphisms $c_n \colon Y_0 \otimes ... \otimes Y_n \otimes X_0 \otimes ... \otimes X_{n+1} \to Y_{n+1}$, the *conditionals*, that satisfy the following equation[1].

$$f_{n+1}(\vec{x}_{n+1}) \ \to \ \vec{y}_{n+1}\} = \begin{matrix} f_n(\vec{x}_n) \ \to \ \vec{y}_n \\ c_n(\vec{y}_n, \vec{x}_{n+1}) \ \to \ y_{n+1} \end{matrix} \Big\}$$

Causal processes over a copy-discard category $(\mathbf{V}, \mathbf{P})$ with quasi-total conditionals [DLR23] form a copy-discard category. When the base category also has ranges [DLdFR22], causal processes are isomorphic to effectful streams.

[0] This is an extended abstract for a manuscript currently submitted for review. For the purpose of Open Access the Author has applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

[1] For convenience, we denote with $\vec{x}_n$ a list $(x_0, \ldots, x_n)$ of elements of $X_0 \otimes \cdots \otimes X_n$.

Theorem 2.4. *For a copy-discard category* $(\mathbf{V}, \mathbf{P})$ *with quasi-total conditionals and ranges, the category of effectful streams is monoidal and isomorphic to the monoidal category of causal processes,*

$$\mathsf{Stream}(\mathsf{Tot}(\mathbf{P}), \mathbf{P}) \cong \mathsf{Causal}(\mathbf{P}).$$

3 EXAMPLE: STREAM CIPHER

Stream cipher protocols encrypt messages of any given length. They are a repeated version of the *one-time pad protocol*. A first party (e.g. Alice) wants to send a private message, $\boldsymbol{m}$, to a second party (e.g. Bob), through a public channel; both apply the XOR operation against a pre-shared key $\boldsymbol{k}$: an attacker listening to the public channel will only receive the encrypted message, $\boldsymbol{m} \oplus \boldsymbol{k}$, which is perfectly uninformative.

$$\begin{array}{ll}
\mathtt{cipher(m)}^{\circ} = & \mathtt{cipher(m)}^{+\circ} = \\
\mathtt{init}() \rightsquigarrow () & \mathtt{rand}_a() \rightsquigarrow k_a \\
\mathtt{rand}_a() \rightsquigarrow k_a & \mathtt{rand}_b() \rightsquigarrow k_b \\
\mathtt{rand}_b() \rightsquigarrow k_b & \mathtt{return}(m \oplus k_a \oplus k_b, m \oplus k_a) \\
\mathtt{return}(m \oplus k_a \oplus k_b, m \oplus k_a) & \mathtt{cipher(m)}^{++} = \mathtt{cipher(m)}^{+}
\end{array}$$

Let us discuss security for the stream cipher protocol by giving it appropriate semantics.

Of course, it is impossible to prove that the stream cipher protocol is exactly equal to a secure channel: it can be easily seen that there exist no perfect random generators in the category of finitely-supported distributions, **Stoch**. Instead, we will prove that the protocol is "approximately equal" ($\approx$) to the secure channel, assuming a pseudorandom number generator tht is "approximately equal" to a perfect one.

Assumption 3.1 (Broadbent and Karvonen, [BK23, §7.4])**.** Let ($\approx$) be a congruence, preserved by composition and tensoring. An ($\approx$)*-pseudorandom number generator* over a finite alphabet is a deterministic morphism, $\mathtt{prng}\colon \mathsf{Seed} \to \mathsf{Char} \otimes \mathsf{Seed}$, that satisfies the following equation,

$$\left.\begin{array}{l} \mathtt{unif_s}() \to g \\ \mathtt{prng}(g) \to h, k \\ \mathtt{return}(h, k) \end{array}\right\} \approx \left.\begin{array}{l} \mathtt{unif_s}() \to g \\ \mathtt{unif_c}() \to k \\ \mathtt{return}(g, k) \end{array}\right\}$$

where $\mathtt{unif}_s$ and $\mathtt{unif}_c$ are the uniform distribution on seeds and characters.

We now interpret the generators of the stream cipher in terms of the following fragments of do-notation.

$$\begin{array}{lll}
[\![\mathtt{rand}_a]\!](g_a, g_b) = & [\![\mathtt{rand}_b]\!](g_a, g_b) = & [\![\mathtt{seed}]\!](g_a, g_b) = \\
\quad \mathtt{prng}(g_a) \to h_a, k & \quad \mathtt{prng}(g_b) \to h_b, k & \quad \mathtt{unif}() \to g \\
\quad \mathtt{return}(h_a, g_b, k) & \quad \mathtt{return}(g_a, h_b, k) & \quad \mathtt{return}(g, g)
\end{array}$$

Finally, we can state security for the stream cipher: it means that it is approximately equal to using secure channel that sends the message directly from Alice to Bob and outputs random noise to an external observer.

Definition 3.2. The secure channel is the following morphism in the syntactic category **Cipher**.

$$\begin{array}{l}
\mathtt{secure}^{\circ}(m) = \\
\quad \mathtt{unif}() \to n \\
\quad \mathtt{return}(m, n);
\end{array}$$

$$\mathtt{secure}^{+} = \mathtt{secure}$$

Theorem 3.3. *The interpretation of the stream cipher is approximately equal to the interpretation of the secure channel,*

$$[\![\mathtt{cipher}]\!] \approx [\![\mathtt{secure}]\!].$$

4 EFFECTFUL MEALY MACHINES

Finally, we employ effectful streams as semantic universe for effectful state machines: their trace is the effectful stream they generate.

Definition 4.1. An *effectful Mealy machine* in an effectful copy-discard category, $(\mathbf{V}, \mathbf{P}, \mathbf{C})$, taking inputs on $A \in \mathrm{Obj}(\mathbf{C})$ and producing outputs in $B \in \mathrm{Obj}(\mathbf{C})$, is a triple (U, i, f) consisting of a *state space* $U \in \mathrm{Obj}(\mathbf{C})$, an initial state $i\colon I \to U$ in $\mathbf{C}$, and a *transition morphism*, $f : U \otimes A \rightsquigarrow U \otimes B$ in $\mathbf{C}$.

Every effectful Mealy machine induces an effectful stream that represents its execution. An object A can be repeated to form a stream $[\![A]\!]$, defined by $[\![A]\!]^{\circ} = A$ and $[\![A]\!]^{+} = [\![A]\!]$. Analogously, a transition morphism, $f\colon U \otimes A \to U \otimes B$ with state space U, can be repeated to form an effectful stream $[\![f]\!]\colon U \cdot [\![A]\!] \to [\![B]\!]$ defined by $[\![f]\!]^{\circ} = f$ and $[\![f]\!]^{+} = [\![f]\!]$. When the transition morphism has an initial state, s_0, it defines a Mealy machine, and we can use it to construct an effectful stream that represents the execution trace of the Mealy machine. The operation $\cdot$ attaches the initial state i at the beginning of the execution of the machine f and gives its trace.

Definition 4.2. The *trace* $\mathrm{tr}(U, i, f)\colon [\![A]\!] \to [\![B]\!]$ of an effectful Mealy machine $(U, i, f)\colon A \to B$ is the effectful stream defined by $\mathrm{tr}(U, i, f) = i \cdot [\![f]\!]$. We say that two Mealy machines are *trace-equivalent* if their traces coincide.

Definition 4.3. A homomorphism of effectful Mealy machines with the same inputs and outputs, $(U, i, f) \Rightarrow (V, j, g)$, is a value morphism $\alpha\colon U \to V$ such that $i \fatsemi \alpha = j$ and moreover

$$\left.\begin{array}{l} f(i, x) \rightsquigarrow u_1, y \\ \mathtt{return}(\alpha(u_1), y) \end{array}\right\} = \left.\begin{array}{l} g(\alpha(i), x) \rightsquigarrow v_1, y \\ \mathtt{return}(v_1, y) \end{array}\right\}.$$

Theorem 4.4. *We say that two effectful Mealy machines are* bisimilar *if they are connected by homomorphisms. This coincides with the usual definition in Kleisli categories of monds. If two effectful Mealy machines are bisimilar, then they are trace equivalent.*

REFERENCES

[BK23] Anne Broadbent and Martti Karvonen. Categorical composable cryptography: extended version. *Log. Methods Comput. Sci.*, 19(4), 2023. URL: https://doi.org/10.46298/lmcs-19(4:30)2023, doi:10.46298/LMCS-19(4:30)2023.

[DLdFR22] Elena Di Lavore, Giovanni de Felice, and Mario Román. Monoidal streams for dataflow programming. In *Proceedings of the 37th Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '22, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3531130.3533365.

[DLR23] Elena Di Lavore and Mario Román. Evidential decision theory via partial markov categories. In *2023 38th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–14, 2023. doi:10.1109/LICS56636.2023.10175776.

[Jon89] Bengt Jonsson. A fully abstract trace model for dataflow networks. In *Proceedings of the 16th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 155–165, 1989.

[Kah74] Gilles Kahn. The semantics of a simple language for parallel programming. *Information processing*, 74(471-475):15–28, 1974.