Commutativity logic for probabilistic processes: complete or not?

Nathan Bowler (Hamburg) and Paul Blain Levy (Birmingham)

April 23, 2017

Prove radically different things equivalent.

Outline

1. Background
2. The language
3. Traces
4. The problem

Computational effects

A computational effect is something a program does

- lookup and update memory (state)
- make nondeterministic choices
- make probabilistic choices
- input an output
- ...

Moggi said: model an effect by a monad on **Set**

e.g. a monad of strategies for I/O.

Computational effects

A computational effect is something a program does

- lookup and update memory (state)
- make nondeterministic choices
- make probabilistic choices
- input an output
- . . .

Moggi said: model an effect by a monad on **Set**
e.g. a monad of strategies for I/O.

Plotkin and Power said: don't just give a monad.
Present it with operations and equations.

They did this for state.

Computational effects

A computational effect is something a program does

- lookup and update memory (state)
- make nondeterministic choices
- make probabilistic choices
- input an output
- ...

Moggi said: model an effect by a monad on **Set**
e.g. a monad of strategies for I/O.

Plotkin and Power said: don't just give a monad.
Present it with operations and equations.

They did this for state.

We do it for a monad combining nondeterminism and I/O.

Computational effects

A computational effect is something a program does

- lookup and update memory (state)
- make nondeterministic choices
- make probabilistic choices
- input an output
- ...

Moggi said: model an effect by a monad on **Set**
e.g. a monad of strategies for I/O.

Plotkin and Power said: don't just give a monad.
Present it with operations and equations.

They did this for state.

We do it for a monad combining nondeterminism and I/O.

And try to do it for a monad that combining probability and I/O.

An imperative language

$$
\begin{aligned}
M \quad ::= \quad & \mathsf{c} \mid M * M \\
& \text{Nondeterministic choice} \\
& \mid M \texttt{ or } M \\
& \mid \texttt{choose } (M_n)_{n \in \mathbb{N}} \\
& \text{Probabilistic choice} \\
& \mid p_0 M_0 + p_1 M_1 + \cdots + p_{R-1} M_{R-1} \;\; (\sum_{n<R} p_n = 1) \\
& \mid p_0 M_0 + p_1 M_1 + \cdots \;\; (\sum_{n \in \mathbb{N}} p_n = 1)
\end{aligned}
$$

- $M * N$ prints `Happy?` and pauses. If the user then inputs `Yes` it executes M, and if `No` it executes N.
- c prints `Goodbye` and pauses.

Signature

An I/O signature consists of

- a set K of outputs

$$\textit{Example} \quad K = \{\mathrm{bye}, \mathrm{happy}\}$$

Signature

An I/O signature consists of

- a set K of outputs

$$\textit{Example} \quad K = \{\mathrm{bye}, \mathrm{happy}\}$$

- for each operation k, a set $\mathrm{Ar}(k)$ of inputs, called the arity of k.

$$\begin{aligned} \textit{Example} \quad \mathrm{Ar}(\mathrm{happy}) &= \{\mathrm{yes}, \mathrm{no}\} \\ \mathrm{Ar}(\mathrm{bye}) &= \emptyset \end{aligned}$$

Signature

An I/O signature consists of

- a set K of outputs

$$\textit{Example} \quad K = \{\text{bye}, \text{happy}\}$$

- for each operation k, a set $\text{Ar}(k)$ of inputs, called the arity of k.

$$\begin{aligned} \textit{Example} \quad \text{Ar}(\text{happy}) &= \{\text{yes}, \text{no}\} \\ \text{Ar}(\text{bye}) &= \emptyset \end{aligned}$$

Each signature $S = (\text{Ar}(k))_{k \in K}$ gives rise to a language:

$$M ::= \ \texttt{input}_k(M_i)_{i \in \text{Ar}(k)} \mid \sum_{n<R} p_n M_n \mid \sum_{n \in \mathbb{N}} p_n M_n$$

Special kinds of signature

Finitary signature

All arities are finite.

Countablary signature

All arities are countable.

Sub-unary signature

Unary operations and constants (nullary operations).

Output only.

Operational semantics

Nondeterministic language

The set Comm of commands

forms a nondeterministic transition system

$$\zeta \ : \ \mathsf{Comm} \to \mathcal{P}^{+}(1 + \mathsf{Comm} \times \mathsf{Comm})$$

Operational semantics

Nondeterministic language

The set Comm of commands

forms a nondeterministic transition system

$$\zeta \ : \ \mathsf{Comm} \to \mathcal{P}^{+}(1 + \mathsf{Comm} \times \mathsf{Comm})$$

Probabilistic language

The set Comm of commands

forms a probabilistic transition system.

$$\zeta \ : \ \mathsf{Comm} \to \mathcal{D}(1 + \mathsf{Comm} \times \mathsf{Comm})$$

Traces

A play is a sequence $k_0, i_0, k_1, i_1, \ldots$, where for all n

- $k_n \in K$
- $i_n \in \mathrm{Ar}(k_n)$.

Example

```
Happy?
Yes
Happy?
Yes
Goodbye
```

An odd-length play is passive ending.

An even-length play is active-ending.

Trace sets

Any command in the nondeterministic language

more generally, any state in a nondeterministic transition system

has a trace set D.

that contains a passive-ending play $s = k_0, i_0, k_1, i_1, \ldots, k_n$

when that play is possible if the user supplies the stated inputs.

The trace set is prefix-closed:

$$sik \in D \Rightarrow s \in D$$

Trace distributions

Any command in the probabilistic language

more generally, any state in a probabilistic transition system

has a trace distribution μ

associating to each passive-ending play $s = k_0, i_0, k_1, i_1, \ldots, k_n$

the probability that it happens if the user supplies the stated inputs.

Calculated as a sum of products.

Not actually a probability distribution over traces, but it satisfies

$$\begin{aligned} 1 &= \sum_{k \in K} \mu(k) \\ \mu(sl) &= \sum_{k \in K} \mu(slik) \quad \text{for all } i \in \mathrm{Ar}(l) \end{aligned}$$

Example

The signature

A binary operation $*$ with arity $\{L, R\}$.
Constants a, b, c.

$$\frac{1}{2}(a * c) + \frac{1}{4}(b * (a * c)) + \frac{1}{8}(b * (b * (a * c))) + \frac{1}{16}(b * (b * (b * (a * c))) + \cdots$$

Example

The signature

A binary operation $*$ with arity $\{\mathsf{L}, \mathsf{R}\}$.
Constants $\mathsf{a}, \mathsf{b}, \mathsf{c}$.

$$\tfrac{1}{2}(\mathsf{a} * \mathsf{c}) + \tfrac{1}{4}(\mathsf{b} * (\mathsf{a} * \mathsf{c})) + \tfrac{1}{8}(\mathsf{b} * (\mathsf{b} * (\mathsf{a} * \mathsf{c}))) + \tfrac{1}{16}(\mathsf{b} * (\mathsf{b} * (\mathsf{b} * (\mathsf{a} * \mathsf{c}))) + \cdots$$

$$\begin{aligned}
(*\mathsf{R})^n * \quad &\mapsto \quad \frac{1}{2^n} \\
(*\mathsf{R})^n \mathsf{c} \quad &\mapsto \quad \frac{1}{2^n} \\
(*\mathsf{R})^n * \mathsf{La} \quad &\mapsto \quad \frac{1}{2^{n+1}} \\
(*\mathsf{R})^n * \mathsf{Lb} \quad &\mapsto \quad \frac{1}{2^{n+1}} \\
\text{everything else} \quad &\mapsto \quad 0
\end{aligned}$$

Definability

- When is a trace set definable by a command?
- When is a trace distribution definable by a command?

Definability: finite nondeterminism

A prefix-closed set of plays D is

- total when every active-ending $t \in D^+$ has at least one response
- finitely nondeterministic when every $t \in D^+$ has only finitely many responses; we then write D^∞ for the set of infinite plays whose prefixes are all in D
- König when it is finitely nondeterministic and D^∞ is empty.

Definability: finite nondeterminism

A prefix-closed set of plays D is

- total when every active-ending $t \in D^+$ has at least one response
- finitely nondeterministic when every $t \in D^+$ has only finitely many responses; we then write D^∞ for the set of infinite plays whose prefixes are all in D
- König when it is finitely nondeterministic and D^∞ is empty.

The trace set of any command is total and König.

Definability: finite nondeterminism

A prefix-closed set of plays D is

- total when every active-ending $t \in D^+$ has at least one response
- finitely nondeterministic when every $t \in D^+$ has only finitely many responses; we then write D^∞ for the set of infinite plays whose prefixes are all in D
- König when it is finitely nondeterministic and D^∞ is empty.

The trace set of any command is total and König.

More generally, the trace set of any well-founded finitely nondeterministic transition system.

Definability: finite nondeterminism

A prefix-closed set of plays D is

- total when every active-ending $t \in D^+$ has at least one response
- finitely nondeterministic when every $t \in D^+$ has only finitely many responses; we then write D^∞ for the set of infinite plays whose prefixes are all in D
- König when it is finitely nondeterministic and D^∞ is empty.

The trace set of any command is total and König.

More generally, the trace set of any well-founded finitely nondeterministic transition system.

Conversely, any total and König play process is the trace set of a command.

Definability: countable nondeterminism

A play process D is

- *countably nondeterministic* when every $t \in D^+$ has only countably many responses
- *well-foundedly total* when for all $t \in D^+$ there is a well-founded tree E such that $\{ts \mid s \in E\} \subseteq D$.

Definability: countable nondeterminism

A play process D is

- *countably nondeterministic* when every $t \in D^+$ has only countably many responses
- *well-foundedly total* when for all $t \in D^+$ there is a well-founded tree E such that $\{ts \mid s \in E\} \subseteq D$.

The trace set of any command has these properties.

More generally, the trace set of any well-founded countably nondeterministic transition system.

Definability: countable nondeterminism

A play process D is

- *countably nondeterministic* when every $t \in D^+$ has only countably many responses
- *well-foundedly total* when for all $t \in D^+$ there is a well-founded tree E such that $\{ts \mid s \in E\} \subseteq D$.

The trace set of any command has these properties.

More generally, the trace set of any well-founded countably nondeterministic transition system.

Conversely, any play process with these properties is the trace set of a command.

Definability: finite probabilistic choice

A trace distribution is König when its support

the set of plays with positive probability

is König.
The trace set of any command is total and König.

Definability: finite probabilistic choice

A trace distribution is König when its support

the set of plays with positive probability

is König.
The trace set of any command is total and König.

More generally, the trace set of any well-founded finitely probabilistic transition system.

Definability: finite probabilistic choice

A trace distribution is König when its support

the set of plays with positive probability

is König.
The trace set of any command is total and König.

More generally, the trace set of any well-founded finitely probabilistic transition system.

Conversely, any total and König play process is the trace set of a command.

Definability

When is a trace distribution definable from a command?

Definability

When is a trace distribution definable from a command?

When, regardless of how the user plays, the probability of infinite interaction is zero.

Definability

When is a trace distribution definable from a command?

When, regardless of how the user plays, the probability of infinite interaction is zero.

Victoriousness

A trace distribution is victorious when every counterstrategy almost surely fails against it.

The monad

A computation of type `bool` should represent a victorious trace distribution but there are two extra constants, returning `true` and `false`.

The monad

A computation of type `bool` should represent a victorious trace distribution but there are two extra constants, returning `true` and `false`.

A monad T on Set

TX is the set of victorious trace distributions over the signature S extended with X many constants.

Equations

Commands with the same trace distribution are trace equivalent.

We want an equational theory that represents trace equivalence.

Equations

Commands with the same trace distribution are trace equivalent.

We want an equational theory that represents trace equivalence.

It must be an equivalence relation and preserved by each term constructor, i.e. a congruence.

Equations

Commands with the same trace distribution are trace equivalent.

We want an equational theory that represents trace equivalence.

It must be an equivalence relation and preserved by each term constructor, i.e. a congruence.

It must include the laws of a convex space:

$$\begin{array}{rcll} \sum_{i<R} (\delta\hat{\imath})_i M_i & \equiv & M_{\hat{\imath}} & ((\delta\hat{\imath})_i \text{ is 1 if } i = \hat{\imath}, \text{ otherwise } 0) \\ \sum_{i<R} p_i \sum_{j<S} q_{i,j} M_j & \equiv & \sum_{j<S} (\sum_{i<R} p_i q_{i,j}) M_j & \end{array}$$

and their infinitary counterparts.

If we stop here, we have described bisimilarity.

Commutativity between probability and I/O

$$\sum_{n<R} p_n(M_n * M'_n) \;\;\equiv\;\; (\sum_{n<R} p_n M_n) * (\sum_{n<R} p_n M'_n)$$

With a general I/O operation:

$$\sum_{n<R} p_n\,\mathtt{input}_k(M_{n,i})_{i\in A_k} \;\;\equiv\;\; \mathtt{input}_k(\sum_{n<R} p_n M_{n,i})_{i\in A_k}$$

Commutativity between probability and I/O

$$\sum_{n<R} p_n(M_n * M'_n) \quad \equiv \quad (\sum_{n<R} p_n M_n) * (\sum_{n<R} p_n M'_n)$$

With a general I/O operation:

$$\sum_{n<R} p_n \, \mathtt{input}_k(M_{n,i})_{i \in A_k} \quad \equiv \quad \mathtt{input}_k(\sum_{n<R} p_n M_{n,i})_{i \in A_k}$$

Let $\equiv$ be the least congruence that includes the ω-convex laws and the commutativity law.

This is the tensor of the probability theory and the equationless I/O theory.

Soundness and completeness

Soundness: if $M \equiv M'$ then M and M' are trace equivalent.

So commutativity logic is sound, for every signature.

Soundness and completeness

Soundness: if $M \equiv M'$ then M and M' are trace equivalent.

So commutativity logic is sound, for every signature.

The converse (completeness) holds if M and M' are finitely probabilistic.

Does it hold for general M and M'?

Soundness and completeness

Soundness: if $M \equiv M'$ then M and M' are trace equivalent.

So commutativity logic is sound, for every signature.

The converse (completeness) holds if M and M' are finitely probabilistic.

Does it hold for general M and M'?

Conjecture (Bowler)

No, not even for the signature $\{*, \mathrm{c}\}$.

Conjecture (Levy)

Yes, for every signature.

Questions for audience

1. Is commutativity logic complete?
2. If not, what can be salvaged?