A unified approach to the semantics of effects

GaLoP 2018, Thessaloniki

John Gowers

April 15, 2018

Outline

Section 1

Some Examples

Nondeterminism and innocence

The game semantics of nondeterminism models nondeterministic programs by *nondeterministic strategies*, where P can have multiple replies to an O-position. But there is no satisfactory definition of an *innocent* nondeterministic strategy.

Nondeterminism and innocence

The game semantics of nondeterminism models nondeterministic programs by *nondeterministic strategies*, where P can have multiple replies to an O-position. But there is no satisfactory definition of an *innocent* nondeterministic strategy.

For example, consider the term

$$(\lambda f.f0) \text{ OR } (\lambda f.f1)$$

whose denotation is the union of the denotations of the same components. But this strategy is not innocent!

Nondeterminism and innocence

The game semantics of nondeterminism models nondeterministic programs by *nondeterministic strategies*, where P can have multiple replies to an O-position. But there is no satisfactory definition of an *innocent* nondeterministic strategy.

For example, consider the term

$$(\lambda f.f0) \text{ OR } (\lambda f.f1)$$

whose denotation is the union of the denotations of the same components. But this strategy is not innocent!

Tsukada and Ong: the problem is missing *branching time information*.

Is there another approach?

Harmer's thesis: a nondeterministic innocent strategy is any strategy of the form

$$1 \xrightarrow{\top_{\mathbb{B}}} \mathbb{B} \xrightarrow{\sigma} A$$

where σ is innocent.

Is there another approach?

Harmer's thesis: a nondeterministic innocent strategy is any strategy of the form

$$1 \xrightarrow{\top_{\mathbb{B}}} \mathbb{B} \xrightarrow{\sigma} A$$

where σ is innocent.

A more "synthetic", a priori definition of what it is to be an innocent strategy would be preferable.

– Harmer

Is there another approach?

Harmer's thesis: a nondeterministic innocent strategy is any strategy of the form

$$1 \xrightarrow{\top_{\mathbb{B}}} \mathbb{B} \xrightarrow{\sigma} A$$

where σ is innocent.

A more "synthetic", a priori definition of what it is to be an innocent strategy would be preferable.

– Harmer

Such an approach is ultimately unsatisfactory as it provides no new insights on nondeterministic stateless computation.

– Tsukada, Ong

Is there another approach?

Harmer's thesis: a nondeterministic innocent strategy is any strategy of the form

$$1 \xrightarrow{\top_{\mathbb{B}}} \mathbb{B} \xrightarrow{\sigma} A$$

#unsatisfactory

where σ is innocent.

A more "synthetic", a priori definition of what it is to be an innocent strategy would be preferable.

– Harmer

Such an approach is ultimately unsatisfactory as it provides no new insights on nondeterministic stateless computation.

– Tsukada, Ong

Is there another approach?

Harmer's thesis: a nondeterministic innocent strategy is any strategy of the form

$$1 \xrightarrow{\top_{\mathbb{B}}} \mathbb{B} \xrightarrow{\sigma} A$$

#unsatisfactory

where σ is innocent.

A more "synthetic", a priori definition of what it is to be an innocent strategy would be preferable.

– Harmer

Such an approach is ultimately unsatisfactory as it provides no new insights on nondeterministic stateless computation.

– Tsukada, Ong

Tsukada, Ong: We can solve the problem using a sheaf-based model that retains branching-time information.

Is there another approach?

Harmer's thesis: a nondeterministic innocent strategy is any strategy of the form

$$1 \xrightarrow{\top_{\mathbb{B}}} \mathbb{B} \xrightarrow{\sigma} A$$

#unsatisfactory

where σ is innocent.

A more "synthetic", a priori definition of what it is to be an innocent strategy would be preferable.

– Harmer

Such an approach is ultimately unsatisfactory as it provides no new insights on nondeterministic stateless computation.

– Tsukada, Ong

Tsukada, Ong: We can solve the problem using a sheaf-based model that retains branching-time information. #satisfactory

A purely formal treatment of effects

Idea: If we use the indirect approach, we can treat $\top_{\mathbb{B}}$ as a purely formal symbol and say that a nondeterministic innocent strategy for a game A is a deterministic innocent strategy for $\mathbb{B} \to A$.

A purely formal treatment of effects

Idea: If we use the indirect approach, we can treat $\top_{\mathbb{B}}$ as a purely formal symbol and say that a nondeterministic innocent strategy for a game A is a deterministic innocent strategy for $\mathbb{B} \to A$.

We don't need to mention nondeterministic strategies at all!

A purely formal treatment of effects

Idea: If we use the indirect approach, we can treat $\top_{\mathbb{B}}$ as a purely formal symbol and say that a nondeterministic innocent strategy for a game A is a deterministic innocent strategy for $\mathbb{B} \to A$.

We don't need to mention nondeterministic strategies at all!

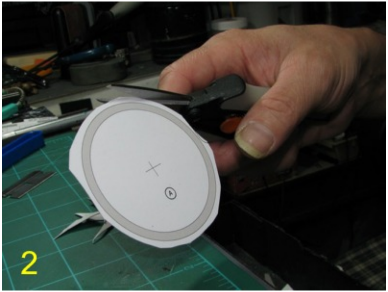
Slight drawback: this approach distinguishes strategies that are in fact identical. E.g., $[\![\lambda b.b]\!]$ and $[\![\lambda b.\text{not } b]\!] : \mathbb{B}$ both denote the term t OR f.

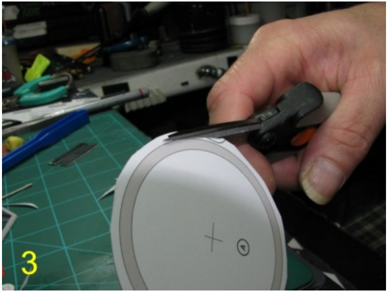
A purely formal treatment of effects

Idea: If we use the indirect approach, we can treat $\top_{\mathbb{B}}$ as a purely formal symbol and say that a nondeterministic innocent strategy for a game A is a deterministic innocent strategy for $\mathbb{B} \to A$.

We don't need to mention nondeterministic strategies at all!

Slight drawback: this approach distinguishes strategies that are in fact identical. E.g., $[\![\lambda b.b]\!]$ and $[\![\lambda b.\text{not } b]\!] : \mathbb{B}$ both denote the term t OR f.





The formal treatment and branching-time information

Clearly, the strategy $\sigma\colon \mathbb{B} \to A$ contains all the information about branching time that we might need if we were to reason about $\top_{\mathbb{B}}; \sigma$.

The formal treatment and branching-time information

Clearly, the strategy $\sigma \colon \mathbb{B} \to A$ contains all the information about branching time that we might need if we were to reason about $\top_{\mathbb{B}} ; \sigma$.

Since deterministic factorization automatically holds, the problem becomes one of working out the right equivalence relation to impose on strategies for $\mathbb{N} \to A$ (cutting out the circle).

The formal treatment and branching-time information

Clearly, the strategy $\sigma\colon \mathbb{B} \to A$ contains all the information about branching time that we might need if we were to reason about $\top_{\mathbb{B}}; \sigma$.

Since deterministic factorization automatically holds, the problem becomes one of working out the right equivalence relation to impose on strategies for $\mathbb{N} \to A$ (cutting out the circle).

We can use a semi-indirect approach informed by the operational semantics of the language (e.g., using linear decomposition: $\sigma, \tau\colon \mathbb{B} \to A$ are equivalent if and only if for all $\alpha\colon !\mathbb{B}$ there exists some $\beta\colon !\mathbb{B}$ such that $\alpha; \sigma = \beta; \tau$ – and vice versa).

The formal treatment and branching-time information

Clearly, the strategy $\sigma \colon \mathbb{B} \to A$ contains all the information about branching time that we might need if we were to reason about $\top_{\mathbb{B}}; \sigma$.

Since deterministic factorization automatically holds, the problem becomes one of working out the right equivalence relation to impose on strategies for $\mathbb{N} \to A$ (cutting out the circle).

We can use a semi-indirect approach informed by the operational semantics of the language (e.g., using linear decomposition: $\sigma, \tau \colon \mathbb{B} \to A$ are equivalent if and only if for all $\alpha \colon !\mathbb{B}$ there exists some $\beta \colon !\mathbb{B}$ such that $\alpha; \sigma = \beta; \tau$ – and vice versa).

Or we can use more ingenious techniques. For example, we can recover the Tsukada-Ong sheaf-theoretic approach by identifying σ with the following sheaf.

$$\sigma(s) = \{t \in \sigma \; : \; t|_A = s\}$$

Another example – bell games

We want to model the language $\mathrm{PCF}^{🔔+}$, obtained by enlarging PCF with command types and a constant $\mathsf{ding} : \mathtt{com}$ with the following operational semantics.

$$\frac{}{\mathsf{ding} \longrightarrow \mathsf{skip}}\ 🔔$$

(🔔 = ‘ring the bell every time you evaluate this rule’)

Another example – bell games

We want to model the language PCF$^{\spadesuit+}$, obtained by enlarging PCF with command types and a constant ding : com with the following operational semantics.

$$\frac{}{\text{ding} \longrightarrow \text{skip}}\ \spadesuit$$

($\spadesuit$ = 'ring the bell every time you evaluate this rule')

Game Semantics (Ghica's 'slot games'): insert an additional move $\spadesuit$ (belonging to neither player) into the play whenever we ring the bell (so $[\![\text{ding}]\!]$ is the strategy with maximal play $q\spadesuit a$).

The $\spadesuit$s are not hidden during composition:

Another example – bell games

We want to model the language PCF$^{🔔+}$, obtained by enlarging PCF with command types and a constant $\mathsf{ding} \colon \mathtt{com}$ with the following operational semantics.

$$\frac{}{\mathsf{ding} \longrightarrow \mathsf{skip}}\ 🔔$$

(🔔 = 'ring the bell every time you evaluate this rule')

Game Semantics (Ghica's 'slot games'): insert an additional move 🔔 (belonging to neither player) into the play whenever we ring the bell (so $[\![\mathsf{ding}]\!]$ is the strategy with maximal play $q🔔a$).

The 🔔s are not hidden during composition:

	com	$\rightarrow$	com	$\rightarrow$	com
					q
	q				
🔔					
	a				
			q		
🔔					
			a		
					a

$\rightsquigarrow$

q
🔔
🔔
a

The formal approach to bell games

Suppose instead that we attempt to model ding as a purely formal morphism $1 \to \mathbb{C}$. In our new model, a strategy for A will be a strategy for $\mathbb{C} \to A$ in the original model, where we interpret $\sigma : \mathbb{C} \to A$ as the strategy ding; $\sigma : A$.

The formal approach to bell games

Suppose instead that we attempt to model ding as a purely formal morphism $1 \to \mathbb{C}$. In our new model, a strategy for A will be a strategy for $\mathbb{C} \to A$ in the original model, where we interpret $\sigma : \mathbb{C} \to A$ as the strategy ding; σ : A.

For example, the term ding; ding : `com` is interpreted as the following composite.

$$\mathbb{C} \xrightarrow{\Delta} \mathbb{C} \times \mathbb{C} \xrightarrow{-\,;\,-} \mathbb{C}$$

The formal approach to bell games

Suppose instead that we attempt to model ding as a purely formal morphism $1 \to \mathbb{C}$. In our new model, a strategy for A will be a strategy for $\mathbb{C} \to A$ in the original model, where we interpret $\sigma : \mathbb{C} \to A$ as the strategy $\mathrm{ding}; \sigma : A$.

For example, the term $\mathrm{ding}; \mathrm{ding} : \texttt{com}$ is interpreted as the following composite.

$$\mathbb{C} \xrightarrow{\Delta} \mathbb{C} \times \mathbb{C} \xrightarrow{\ -\, ;\, -\ } \mathbb{C}$$

As a a strategy:

$$\begin{array}{ccc}
\mathbb{C} & \to & \mathbb{C} \\
 & & q \\
q & & \\
a & & \\
q & & \\
a & & \\
 & & a
\end{array}$$

The formal approach to bell games

Suppose instead that we attempt to model ding as a purely formal morphism $1 \to \mathbb{C}$. In our new model, a strategy for A will be a strategy for $\mathbb{C} \to A$ in the original model, where we interpret $\sigma : \mathbb{C} \to A$ as the strategy ding; $\sigma : A$.

For example, the term ding; ding : `com` is interpreted as the following composite.

$$\mathbb{C} \xrightarrow{\Delta} \mathbb{C} \times \mathbb{C} \xrightarrow{\;-\,;\,-\;} \mathbb{C}$$

As a a strategy:

$$\begin{array}{ccc}
\mathbb{C} & \to & \mathbb{C} \\
 & & q \\
q & & \\
a & & \\
q & & \\
a & & \\
 & & a
\end{array}$$

🔔 $= qa$ on the left.

A common theme

A common theme in both these examples is that formal morphisms expose moves that are normally hidden under composition.

A common theme

A common theme in both these examples is that formal morphisms expose moves that are normally hidden under composition.

For some computational effects, this is exactly what we want to do. For nondeterminism, the nondeterministic strategy approach hides too much information, and the formal approach reveals too much.

A common theme

A common theme in both these examples is that formal morphisms expose moves that are normally hidden under composition.

For some computational effects, this is exactly what we want to do. For nondeterminism, the nondeterministic strategy approach hides too much information, and the formal approach reveals too much.

Topics for exploration:

A common theme

A common theme in both these examples is that formal morphisms expose moves that are normally hidden under composition.

For some computational effects, this is exactly what we want to do. For nondeterminism, the nondeterministic strategy approach hides too much information, and the formal approach reveals too much.

Topics for exploration:

Can we add formal morphisms $A \to A$ for all A in order to make a 'revealed game semantics' that is still a Cartesian closed category?

A common theme

A common theme in both these examples is that formal morphisms expose moves that are normally hidden under composition.

For some computational effects, this is exactly what we want to do. For nondeterminism, the nondeterministic strategy approach hides too much information, and the formal approach reveals too much.

Topics for exploration:

Can we add formal morphisms $A \to A$ for all A in order to make a 'revealed game semantics' that is still a Cartesian closed category?

Are these two examples the same? (E.g., ding = 'unary nondeterminism')

A common theme

A common theme in both these examples is that formal morphisms expose moves that are normally hidden under composition.

For some computational effects, this is exactly what we want to do. For nondeterminism, the nondeterministic strategy approach hides too much information, and the formal approach reveals too much.

Topics for exploration:

Can we add formal morphisms $A \to A$ for all A in order to make a 'revealed game semantics' that is still a Cartesian closed category?

Are these two examples the same? (E.g., ding = 'unary nondeterminism')

Can we build a model of nondeterministic innocence based on bell games?

Section 2

Cones on monoidal functors

We talked about formally adding morphisms into a category. In this section, we'll look at a slightly more general construction.

We talked about formally adding morphisms into a category. In this section, we'll look at a slightly more general construction.

The goals of this generalization are twofold:

We talked about formally adding morphisms into a category. In this section, we'll look at a slightly more general construction.

The goals of this generalization are twofold:

- To allow us to add more than one morphism at a time.

We talked about formally adding morphisms into a category. In this section, we'll look at a slightly more general construction.

The goals of this generalization are twofold:

- To allow us to add more than one morphism at a time.
- To give a more systematic account of the equivalence relation on morphisms.

We talked about formally adding morphisms into a category. In this section, we'll look at a slightly more general construction.

The goals of this generalization are twofold:

- To allow us to add more than one morphism at a time.
- To give a more systematic account of the equivalence relation on morphisms.

Previously, we wanted to add a morphism $\phi\colon 1 \to A$ for some object A or, equivalently, a natural transformation from I to the 'constant A functor'.

We talked about formally adding morphisms into a category. In this section, we'll look at a slightly more general construction.

The goals of this generalization are twofold:

- To allow us to add more than one morphism at a time.
- To give a more systematic account of the equivalence relation on morphisms.

Previously, we wanted to add a morphism $\phi\colon 1 \to A$ for some object A or, equivalently, a natural transformation from I to the 'constant A functor'.

We replace I with an arbitrary symmetric monoidal category $\mathcal{C}'$ and require that ϕ be an oplax monoidal functor.

We talked about formally adding morphisms into a category. In this section, we'll look at a slightly more general construction.

The goals of this generalization are twofold:

- To allow us to add more than one morphism at a time.
- To give a more systematic account of the equivalence relation on morphisms.

Previously, we wanted to add a morphism $\phi\colon 1 \to A$ for some object A or, equivalently, a natural transformation from I to the 'constant A functor'.

We replace I with an arbitrary symmetric monoidal category $\mathcal{C}'$ and require that ϕ be an oplax monoidal functor.

Note: for every object A of a CCC, the 'constant A' functor is oplax monoidal with coherence given by the diagonal and projections.

The setup: we have an oplax monoidal functor $j\colon \mathcal{C}' \to \mathcal{C}$, where $\mathcal{C}, \mathcal{C}'$ are symmetric monoidal categories.

The setup: we have an oplax monoidal functor $j\colon \mathcal{C}' \to \mathcal{C}$, where $\mathcal{C}, \mathcal{C}'$ are symmetric monoidal categories.

We desire a symmetric monoidal category $\mathcal{D}$ that contains $\mathcal{C}$ plus additional morphisms corresponding to the image of j.

The setup: we have an oplax monoidal functor $j\colon \mathcal{C}' \to \mathcal{C}$, where $\mathcal{C}, \mathcal{C}'$ are symmetric monoidal categories.

We desire a symmetric monoidal category $\mathcal{D}$ that contains $\mathcal{C}$ plus additional morphisms corresponding to the image of j.

I.e., this (a *j-category*):

$$\begin{tikzcd}
\mathcal{C}' \arrow[r, "()"] \arrow[d, "j"'] \arrow[dr, phantom, "\Downarrow\phi"] & I \arrow[d, "I"] \\
\mathcal{C} \arrow[r, "J"'] & \mathcal{D}
\end{tikzcd},$$

where J is a lax monoidal functor and ϕ is a ‘generalized monoidal natural transformation’.

The setup: we have an oplax monoidal functor $j\colon \mathcal{C}' \to \mathcal{C}$, where $\mathcal{C}, \mathcal{C}'$ are symmetric monoidal categories.

We desire a symmetric monoidal category $\mathcal{D}$ that contains $\mathcal{C}$ plus additional morphisms corresponding to the image of j.

I.e., this (a *j-category*):

$$
\begin{tikzcd}
\mathcal{C}' \arrow[r, "()"] \arrow[d, "j"'] \arrow[dr, phantom, "\Downarrow\phi"] & I \arrow[d, "I"] \\
\mathcal{C} \arrow[r, "J"'] & \mathcal{D}
\end{tikzcd},
$$

where J is a lax monoidal functor and ϕ is a 'generalized monoidal natural transformation'.

The more objects there are in $\mathcal{C}'$, the more morphisms there are in $\mathcal{D}$. The more morphisms there are in $\mathcal{C}'$, the *fewer* morphisms there are in $\mathcal{D}$ (or: a finer equivalence relation on morphisms).

The Cone on j

We can define a morphism of j-categories $(\mathcal{D}', J', \phi') \to (\mathcal{D}, J, \phi)$ to be an oplax monoidal functor $f\colon \mathcal{D}' \to \mathcal{D}$ making the following diagram commute.

$$
\begin{tikzcd}
\mathcal{C}' \arrow[r, "()"] \arrow[d, "j"'] \arrow[dr, phantom, "\Downarrow\phi'"] \arrow[ddr, phantom, "\Downarrow\phi"'] & I \arrow[d, "I"] \\
\mathcal{C} \arrow[r, "J'"'] \arrow[d, "\mathrm{id}"'] \arrow[dr, phantom, "\Downarrow\mathrm{id}"] & \mathcal{D}' \arrow[d, "f"] \\
\mathcal{C} \arrow[r, "J"'] & \mathcal{D}
\end{tikzcd}
.
$$

The Cone on j

We can define a morphism of j-categories $(\mathcal{D}', J', \phi') \to (\mathcal{D}, J, \phi)$ to be an oplax monoidal functor $f \colon \mathcal{D}' \to \mathcal{D}$ making the following diagram commute.

$$
\begin{tikzcd}
\mathcal{C}' \arrow[r, "()"] \arrow[d, "j"'] & I \arrow[d, "I"] \\
\mathcal{C} \arrow[r, "J'"'] \arrow[d, "\mathrm{id}"'] \arrow[ur, phantom, "\Downarrow \phi'"] & \mathcal{D}' \arrow[d, "f"] \\
\mathcal{C} \arrow[r, "J"'] \arrow[ur, phantom, "\Downarrow \mathrm{id}"] \arrow[uur, phantom, "\Downarrow \phi"] & \mathcal{D}
\end{tikzcd}
.
$$

We get a category of j-categories. We can show that this category has an initial object C_j, called the *cone on j*.

The Cone on j

We can define a morphism of j-categories $(\mathcal{D}', J', \phi') \to (\mathcal{D}, J, \phi)$ to be an oplax monoidal functor $f \colon \mathcal{D}' \to \mathcal{D}$ making the following diagram commute.

$$
\begin{tikzcd}
\mathcal{C}' \arrow[r, "()"] \arrow[d, "j"'] \arrow[dr, phantom, "\Downarrow \phi'"] \arrow[dd, phantom, bend left, "\Downarrow \phi"] & I \arrow[d, "I"] \\
\mathcal{C} \arrow[r, "J'"'] \arrow[d, "\mathrm{id}"'] \arrow[dr, phantom, "\Downarrow \mathrm{id}"] & \mathcal{D}' \arrow[d, "f"] \\
\mathcal{C} \arrow[r, "J"] & \mathcal{D}
\end{tikzcd}
.
$$

We get a category of j-categories. We can show that this category has an initial object C_j, called the *cone on j*.

C_j is the category obtained from $\mathcal{C}$ by 'formally adding morphisms $\psi_X \colon I \multimap jX$'.

Construction of C_j

The construction of C_j is straightforward (and quite familiar in the Cartesian case).

Construction of C_j

The construction of C_j is straightforward (and quite familiar in the Cartesian case).

The objects of C_j are the objects of $\mathcal{C}$.

A morphism $A \to B$ in C_j is a pair (X, f), where X is an object of $\mathcal{C}'$ and $f : jX \otimes A \to B$ is a morphism in $\mathcal{C}'$.

Two morphisms (X, f) and (X', f') are considered to be equivalent if there is some morphism $h\colon X' \to X$ in $\mathcal{C}'$ making the following diagram commute.

$$\begin{tikzcd}
jX' \otimes A \arrow[r, "f'"] \arrow[d, "jh\otimes A"'] & B \\
jX \otimes A \arrow[ur, "f"'] &
\end{tikzcd}$$

Construction of C_j

The construction of C_j is straightforward (and quite familiar in the Cartesian case).

The objects of C_j are the objects of $\mathcal{C}$.

A morphism $A \to B$ in C_j is a pair (X, f), where X is an object of $\mathcal{C}'$ and $f \colon jX \otimes A \to B$ is a morphism in $\mathcal{C}'$.

Two morphisms (X, f) and (X', f') are considered to be equivalent if there is some morphism $h \colon X' \to X$ in $\mathcal{C}'$ making the following diagram commute.

$$\begin{tikzcd}
jX' \otimes A \arrow[r, "f'"] \arrow[d, "jh \otimes A"'] & B \\
jX \otimes A \arrow[ur, "f"'] &
\end{tikzcd}$$

Composition is given by 'the only thing you can write down'.

Constructing models from C_j

We said that we wanted to model effects by finding a particular j-category $\mathcal{D}$ for a suitable j that captures the effect in some way.

Constructing models from C_j

We said that we wanted to model effects by finding a particular j-category $\mathcal{D}$ for a suitable j that captures the effect in some way.

Even if we know nothing else about $\mathcal{D}$, we know that there is a functor $C_j \to \mathcal{D}$ that is a morphism of j-categories.

Constructing models from C_j

We said that we wanted to model effects by finding a particular j-category $\mathcal{D}$ for a suitable j that captures the effect in some way.

Even if we know nothing else about $\mathcal{D}$, we know that there is a functor $C_j \to \mathcal{D}$ that is a morphism of j-categories.

If $\mathcal{D}$ satisfies 'j-factorization', then this functor is full.

Constructing models from C_j

We said that we wanted to model effects by finding a particular j-category $\mathcal{D}$ for a suitable j that captures the effect in some way.

Even if we know nothing else about $\mathcal{D}$, we know that there is a functor $C_j \to \mathcal{D}$ that is a morphism of j-categories.

If $\mathcal{D}$ satisfies 'j-factorization', then this functor is full.

In such a case, we can recover $\mathcal{D}$ by imposing a suitable equivalence relation on the morphisms in C_j.

Constructing models from C_j

We said that we wanted to model effects by finding a particular j-category $\mathcal{D}$ for a suitable j that captures the effect in some way.

Even if we know nothing else about $\mathcal{D}$, we know that there is a functor $C_j \to \mathcal{D}$ that is a morphism of j-categories.

If $\mathcal{D}$ satisfies 'j-factorization', then this functor is full.

In such a case, we can recover $\mathcal{D}$ by imposing a suitable equivalence relation on the morphisms in C_j.

The more morphisms we can include in $\mathcal{C}'$, the closer C_j will be to our desired model.

Section 3

More examples

Unbounded nondeterminism

Let $\mathcal{G}$ be the category of Hyland-Ong games.

Unbounded nondeterminism

Let $\mathcal{G}$ be the category of Hyland-Ong games.

Now suppose that $\mathcal{D}$ is some extension of $\mathcal{G}$ that somehow models nondeterminism. For each object A of $\mathcal{D}$, there should be a strategy $\top_A \colon 1 \to A$ that includes every position in A.

Unbounded nondeterminism

Let $\mathcal{G}$ be the category of Hyland-Ong games.

Now suppose that $\mathcal{D}$ is some extension of $\mathcal{G}$ that somehow models nondeterminism. For each object A of $\mathcal{D}$, there should be a strategy $\top_A \colon 1 \to A$ that includes every position in A.

$\top_A$ is not a natural transformation $1 \to \mathcal{G}$: this would mean that $\top_A ; \sigma = \top_B$ for every $\sigma \colon A \to B$.

Unbounded nondeterminism

Let $\mathcal{G}$ be the category of Hyland-Ong games.

Now suppose that $\mathcal{D}$ is some extension of $\mathcal{G}$ that somehow models nondeterminism. For each object A of $\mathcal{D}$, there should be a strategy $\top_A \colon 1 \to A$ that includes every position in A.

$\top_A$ is not a natural transformation $1 \to \mathcal{G}$: this would mean that $\top_A ; \sigma = \top_B$ for every $\sigma \colon A \to B$.

However, if we require that σ be *surjective*; i.e., that every position in B is the restriction of some position occurring in σ, then $\top_A ; \sigma = \top_B$.

Unbounded nondeterminism

Let $\mathcal{G}$ be the category of Hyland-Ong games.

Now suppose that $\mathcal{D}$ is some extension of $\mathcal{G}$ that somehow models nondeterminism. For each object A of $\mathcal{D}$, there should be a strategy $\top_A \colon 1 \to A$ that includes every position in A.

$\top_A$ is not a natural transformation $1 \to \mathcal{G}$: this would mean that $\top_A; \sigma = \top_B$ for every $\sigma \colon A \to B$.

However, if we require that σ be *surjective*; i.e., that every position in B is the restriction of some position occurring in σ, then $\top_A; \sigma = \top_B$.

If j_s is the inclusion of the category $\mathcal{G}_s$ of games and *surjective strategies* into $\mathcal{G}$, then $\top_A$ is a natural transformation $1 \to j_s A$.

Unbounded nondeterminism

Let $\mathcal{G}$ be the category of Hyland-Ong games.

Now suppose that $\mathcal{D}$ is some extension of $\mathcal{G}$ that somehow models nondeterminism. For each object A of $\mathcal{D}$, there should be a strategy $\top_A \colon 1 \to A$ that includes every position in A.

$\top_A$ is not a natural transformation $1 \to \mathcal{G}$: this would mean that $\top_A; \sigma = \top_B$ for every $\sigma \colon A \to B$.

However, if we require that σ be *surjective*; i.e., that every position in B is the restriction of some position occurring in σ, then $\top_A; \sigma = \top_B$.

If j_s is the inclusion of the category $\mathcal{G}_s$ of games and *surjective strategies* into $\mathcal{G}$, then $\top_A$ is a natural transformation $1 \to j_s A$.

Therefore, $\mathcal{D}$ is some quotient of C_{j_s}.

Unbounded nondeterminism and must-testing

C_j models may-testing: in a model of unbounded nondeterminism with must-testing, we may have $\top_A; \sigma \neq \top_B$ even if σ is surjective. The reason is that σ might have a play ending with infinitely many moves in A.

Unbounded nondeterminism and must-testing

C_j models may-testing: in a model of unbounded nondeterminism with must-testing, we may have $\top_A ; \sigma \neq \top_B$ even if σ is surjective. The reason is that σ might have a play ending with infinitely many moves in A.

We say that a strategy $\sigma \colon A \to B$ is *winning* if it contains no such play. Then if j_{sw} is the inclusion of the category $\mathcal{G}_{sw}$ of games and *surjective winning* strategies, then any model of unbounded nondeterminism with must testing that satisfies deterministic factorization is a quotient of C_j.

Unbounded nondeterminism and must-testing

C_j models may-testing: in a model of unbounded nondeterminism with must-testing, we may have $\top_A ; \sigma \neq \top_B$ even if σ is surjective. The reason is that σ might have a play ending with infinitely many moves in A.

We say that a strategy $\sigma \colon A \to B$ is *winning* if it contains no such play. Then if j_{sw} is the inclusion of the category $\mathcal{G}_{sw}$ of games and *surjective winning* strategies, then any model of unbounded nondeterminism with must testing that satisfies deterministic factorization is a quotient of C_j.

Notice that $\mathcal{G}_{sw}$ has *fewer* morphisms than $\mathcal{G}_s$ and so the cone $C_{j_{sw}}$ has *more* morphisms than the cone C_{j_s}: we no longer identify all strategies that have the same convergent behaviours.

Finite nondeterminism, revisited

We modelled finite nondeterminism in the cone $C_{c_{\mathbb{B}}}$, where $c_{\mathbb{B}} \colon I \to \mathcal{G}$ is the constant $\mathbb{B}$ functor.

Finite nondeterminism, revisited

We modelled finite nondeterminism in the cone $C_{c_{\mathbb{B}}}$, where $c_{\mathbb{B}} \colon I \to \mathcal{G}$ is the constant $\mathbb{B}$ functor.

A problem with this model was that it did not distinguish enough morphisms.

Finite nondeterminism, revisited

We modelled finite nondeterminism in the cone $C_{c_{\mathbb{B}}}$, where $c_{\mathbb{B}}\colon I \to \mathcal{G}$ is the constant $\mathbb{B}$ functor.

A problem with this model was that it did not distinguish enough morphisms.

Let us replace I with the one-object category s_2 whose morphisms are the surjective functions $\texttt{bool} \to \texttt{bool}$. Then $c_{\mathbb{B}}$ extends to a functor $c'_{\mathbb{B}}\colon s_2 \to \mathcal{G}$ in an obvious way.

Finite nondeterminism, revisited

We modelled finite nondeterminism in the cone $C_{c_{\mathbb{B}}}$, where $c_{\mathbb{B}} \colon I \to \mathcal{G}$ is the constant $\mathbb{B}$ functor.

A problem with this model was that it did not distinguish enough morphisms.

Let us replace I with the one-object category s_2 whose morphisms are the surjective functions `bool` $\to$ `bool`. Then $c_{\mathbb{B}}$ extends to a functor $c'_{\mathbb{B}} \colon s_2 \to \mathcal{G}$ in an obvious way.

The cone on $c'_{\mathbb{B}}$ now no longer distinguishes the two terms $\lambda b.b$ and $\lambda b.\mathrm{not}\ b$.

Finite nondeterminism, revisited

We modelled finite nondeterminism in the cone $C_{c_{\mathbb{B}}}$, where $c_{\mathbb{B}}\colon I \to \mathcal{G}$ is the constant $\mathbb{B}$ functor.

A problem with this model was that it did not distinguish enough morphisms.

Let us replace I with the one-object category s_2 whose morphisms are the surjective functions `bool` $\to$ `bool`. Then $c_{\mathbb{B}}$ extends to a functor $c'_{\mathbb{B}}\colon s_2 \to \mathcal{G}$ in an obvious way.

The cone on $c'_{\mathbb{B}}$ now no longer distinguishes the two terms $\lambda b.b$ and $\lambda b.\text{not } b$.

Alternatively: there is a morphism of oplax monoidal functors from $c_{\mathbb{B}} \to j_{sw}$, which induces a functor $C_{c_{\mathbb{B}}} \to C_{j_{sw}}$. If we take the image of $C_{c_{\mathbb{B}}}$ inside $C_{j_{sw}}$ then we can use the finer equivalence relation with the simpler category.

Probabilistic game semantics

The last example was a category with one object and multiple morphisms.

Probabilistic game semantics

The last example was a category with one object and multiple morphisms.

Now we consider an example with multiple objects and very few morphisms.

Probabilistic game semantics

The last example was a category with one object and multiple morphisms.

Now we consider an example with multiple objects and very few morphisms.

We turn the monoid $([0,1], \times)$ into a (strict) monoidal category and then add additional morphisms $\neg_p \colon p \to 1 - p$.

Take the functor $c^p_{\mathbb{B}} \colon [0,1] \to \mathcal{G}$ that sends every element of $[0,1]$ to the boolean object $\mathbb{B}$ and sends each morphism $\neg_p$ to the morphism $\mathsf{not} \colon \mathbb{B} \to \mathbb{B}$.

Probabilistic game semantics

The last example was a category with one object and multiple morphisms.

Now we consider an example with multiple objects and very few morphisms.

We turn the monoid $([0,1], \times)$ into a (strict) monoidal category and then add additional morphisms $\neg_p \colon p \to 1 - p$.

Take the functor $c^p_{\mathbb{B}} \colon [0,1] \to \mathcal{G}$ that sends every element of $[0,1]$ to the boolean object $\mathbb{B}$ and sends each morphism $\neg_p$ to the morphism $\mathrm{not} \colon \mathbb{B} \to \mathbb{B}$.

Then the cone over $c^p_{\mathbb{B}}$ interprets probabilistic PCF.

Probabilistic game semantics

The last example was a category with one object and multiple morphisms.

Now we consider an example with multiple objects and very few morphisms.

We turn the monoid $([0,1], \times)$ into a (strict) monoidal category and then add additional morphisms $\neg_p \colon p \to 1 - p$.

Take the functor $c^p_{\mathbb{B}} \colon [0,1] \to \mathcal{G}$ that sends every element of $[0,1]$ to the boolean object $\mathbb{B}$ and sends each morphism $\neg_p$ to the morphism $\mathsf{not} \colon \mathbb{B} \to \mathbb{B}$.

Then the cone over $c^p_{\mathbb{B}}$ interprets probabilistic PCF.

The equivalence relation on morphisms in the cone automatically gives us equations such as the following.

$$\text{if } \psi_{2/3} \text{ then } (\text{if } \psi_{1/2} \text{ then } a \text{ else } b) \text{ else } c =$$
$$\text{if } \psi_{1/3} \text{ then } a \text{ else } (\text{if } \psi_{1/2} \text{ then } b \text{ else } c)$$

Game semantics of state

Let $\mathbf{Set}^{\rightleftarrows}$ be the category whose objects are sets and where the morphisms $A \to B$ are pairs of functions $f\colon A \to B, g\colon B \to A$ such that $f \circ g = \mathrm{id}_B$.

Then we get an oplax monoidal functor $\mathtt{Var}\colon \mathbf{Set}^{\rightleftarrows} \to \mathcal{G}$ sending a set X to the game $\mathtt{Var}[X] = \mathtt{com}^X \times \underline{X}$.

Game semantics of state

Let $\mathbf{Set}^{\rightleftarrows}$ be the category whose objects are sets and where the morphisms $A \to B$ are pairs of functions $f\colon A \to B, g\colon B \to A$ such that $f \circ g = \mathrm{id}_B$.

Then we get an oplax monoidal functor $\mathtt{Var}\colon \mathbf{Set}^{\rightleftarrows} \to \mathcal{G}$ sending a set X to the game $\mathtt{Var}[X] = \mathtt{com}^X \times \underline{X}$.

The function $\mathrm{cell}_X\colon 1 \to !\mathtt{Var}[X]$ that is used to encapsulate state in the Abramsky-McCusker model of Idealized Algol is then a natural transformation from 1 to $!\mathtt{Var}[X]$.

Game semantics of state

Let $\mathbf{Set}^{\rightleftarrows}$ be the category whose objects are sets and where the morphisms $A \to B$ are pairs of functions $f\colon A \to B, g\colon B \to A$ such that $f \circ g = \mathrm{id}_B$.

Then we get an oplax monoidal functor $\mathtt{Var}\colon \mathbf{Set}^{\rightleftarrows} \to \mathcal{G}$ sending a set X to the game $\mathtt{Var}[X] = \mathtt{com}^X \times \underline{X}$.

The function $\mathrm{cell}_X\colon 1 \to {!}\mathtt{Var}[X]$ that is used to encapsulate state in the Abramsky-McCusker model of Idealized Algol is then a natural transformation from 1 to ${!}\mathtt{Var}[X]$.

So we can model stateful strategies on a game A as pairs (X, σ) where X is a set and $\sigma\colon {!}\mathtt{Var}[X] \to A$ is an innocent strategy.

Game semantics of state

Let $\mathbf{Set}^{\rightleftarrows}$ be the category whose objects are sets and where the morphisms $A \to B$ are pairs of functions $f\colon A \to B, g\colon B \to A$ such that $f \circ g = \mathrm{id}_B$.

Then we get an oplax monoidal functor $\mathtt{Var}\colon \mathbf{Set}^{\rightleftarrows} \to \mathcal{G}$ sending a set X to the game $\mathtt{Var}[X] = \mathtt{com}^X \times \underline{X}$.

The function $\mathrm{cell}_X\colon 1 \to !\mathtt{Var}[X]$ that is used to encapsulate state in the Abramsky-McCusker model of Idealized Algol is then a natural transformation from 1 to $!\mathtt{Var}[X]$.

So we can model stateful strategies on a game A as pairs (X, σ) where X is a set and $\sigma\colon !\mathtt{Var}[X] \to A$ is an innocent strategy.

So what? We can already characterize these as visible strategies.

Call-by-value state

Murawski + Tzevelekos: There is no implicit characterization of the stateful strategies of call-by-value Idealized Algol.

Call-by-value state

Murawski + Tzevelekos: There is no implicit characterization of the stateful strategies of call-by-value Idealized Algol.

Contrast

$\text{RML} = \text{PCF}^+ + \text{ref} : \texttt{Var}$

with

$\text{IA}_{cbv} = \text{PCF}^+ + \texttt{new}_X : ((\texttt{Var} \to X) \to X)$.

Call-by-value state

Murawski + Tzevelekos: There is no implicit characterization of the stateful strategies of call-by-value Idealized Algol.

Contrast

RML = PCF^{+} + ref: $\mathtt{Var}$

with

IA$_{cbv}$ = PCF^{+} + $\mathtt{new}_X : ((\mathtt{Var} \to X) \to X)$.

Cone models make Idealized Algol's *block structure* explicit:

$$[\mathrm{RML}]\ \mathtt{Var} \to A \qquad\qquad [\mathrm{IA}_{cbv}]\ ((\mathtt{Var} \to X) \to X) \to A$$

Call-by-value state

Murawski + Tzevelekos: There is no implicit characterization of the stateful strategies of call-by-value Idealized Algol.

Contrast

RML $=$ PCF$^+$ $+$ ref: $\mathtt{Var}$

with

$\mathsf{IA}_{cbv} = \mathsf{PCF}^+ + \mathtt{new}_X : ((\mathtt{Var} \to X) \to X)$.

Cone models make Idealized Algol's *block structure* explicit:

$$[\mathsf{RML}]\ \mathtt{Var} \to A \qquad\qquad [\mathsf{IA}_{cbv}]\ ((\mathtt{Var} \to X) \to X) \to A$$

M+T: S-strategies for modelling IA_{cbv} – make state and block-structure explicit, keeping just enough information to define a notion of *innocence*.

Call-by-value state

Murawski + Tzevelekos: There is no implicit characterization of the stateful strategies of call-by-value Idealized Algol.

Contrast

$\mathsf{RML} = \mathsf{PCF}^{+} + \mathsf{ref} \colon \mathtt{Var}$

with

$\mathsf{IA}_{cbv} = \mathsf{PCF}^{+} + \mathtt{new}_X \colon ((\mathtt{Var} \to X) \to X)$.

Cone models make Idealized Algol's *block structure* explicit:

$$[\mathsf{RML}]\ \mathtt{Var} \to A \qquad\qquad [\mathsf{IA}_{cbv}]\ ((\mathtt{Var} \to X) \to X) \to A$$

M+T: S-strategies for modelling IA_{cbv} – make state and block-structure explicit, keeping just enough information to define a notion of *innocence*.

Then discard the explicit state to get the semantics of IA_{cbv}.

Questions?