Sequoidal Categories and Transfinite Games: A Coalgebraic Approach to Stateful Objects in Game Semantics

University of Bath, 2017

John Gowers and James Laird

June 5, 2017

Semantics of Stateful Programs

In traditional domain-theoretic semantics, a function without side effects is represented as a map

$$X \to Y$$

Semantics of Stateful Programs

In traditional domain-theoretic semantics, a function without side effects is represented as a map

$$X \to Y$$

while a function with side effects is represented as a map

$$S \times X \to S \times Y$$

Semantics of Stateful Programs

In traditional domain-theoretic semantics, a function without side effects is represented as a map

$$X \to Y$$

while a function with side effects is represented as a map

$$S \times X \to S \times Y$$

or equivalently an algebra for the monad

$$[S, S \times \_\ ]$$

Semantics of Stateful Programs

In traditional domain-theoretic semantics, a function without side effects is represented as a map

$$X \to Y$$

while a function with side effects is represented as a map

$$S \times X \to S \times Y$$

or equivalently an algebra for the monad

$$[S, S \times \_\ ]$$

We can then represent programs with a fixed store S as morphisms in the Kleisli category for this monad.

Semantics of Stateful Programs

In traditional domain-theoretic semantics, a function without side effects is represented as a map

$$X \to Y$$

while a function with side effects is represented as a map

$$S \times X \to S \times Y$$

or equivalently an algebra for the monad

$$[S, S \times \_\,]$$

We can then represent programs with a fixed store S as morphisms in the Kleisli category for this monad. This is an example of *explicit state*.

Game Semantics and Implicit State

By contrast, the state in Game Semantics is *implicit*.

Game Semantics and Implicit State

By contrast, the state in Game Semantics is *implicit*.

Consider, for example, the game where player O, at any turn, can either say b (button press) or c (count).

Player P must reply to b with the move $\checkmark$ and can reply to c by saying any natural number.

Game Semantics and Implicit State

By contrast, the state in Game Semantics is *implicit*.

Consider, for example, the game where player O, at any turn, can either say b (button press) or c (count).

Player P must reply to b with the move $\checkmark$ and can reply to c by saying any natural number.

There is a strategy for this game corresponding to a simple counter.

Game Semantics and Implicit State

By contrast, the state in Game Semantics is *implicit*.

Consider, for example, the game where player O, at any turn, can either say b (button press) or c (count).

Player P must reply to b with the move $\checkmark$ and can reply to c by saying any natural number.

There is a strategy for this game corresponding to a simple counter.

An example play in that strategy is:

$$c\ 0\ b\ \checkmark\ b\ \checkmark\ b\ \checkmark\ c\ 3\ c\ 3\ b\ \checkmark \cdots$$

Game Semantics and Implicit State

By contrast, the state in Game Semantics is *implicit*.

Consider, for example, the game where player O, at any turn, can either say b (button press) or c (count).

Player P must reply to b with the move $\checkmark$ and can reply to c by saying any natural number.

There is a strategy for this game corresponding to a simple counter.

An example play in that strategy is:

$$c\ 0\ b \checkmark\ b \checkmark\ b \checkmark\ c\ 3\ c\ 3\ b \checkmark \cdots$$

This play has implicit state:

Moves	c	0	b	$\checkmark$	b	$\checkmark$	b	$\checkmark$	c	3	c	3	b	$\checkmark$
State	0	0	1	1	2	2	3	3	3	3	3	3	3	3

Case Study: The Abramsky-McCusker model of Idealized Algol

Abramsky and McCusker's model of Idealized Algol (1998) uses a stateful strategy cell, representing the behaviour of a storage cell.

Case Study: The Abramsky-McCusker model of Idealized Algol

Abramsky and McCusker's model of Idealized Algol (1998) uses a stateful strategy cell, representing the behaviour of a storage cell.

They use a combinatorial definition to define cell directly.

Case Study: The Abramsky-McCusker model of Idealized Algol

Abramsky and McCusker's model of Idealized Algol (1998) uses a stateful strategy cell, representing the behaviour of a storage cell.

They use a combinatorial definition to define cell directly.

Because state is implicit, it is not always easy to keep track of it.

Case Study: The Abramsky-McCusker model of Idealized Algol

Abramsky and McCusker's model of Idealized Algol (1998) uses a stateful strategy cell, representing the behaviour of a storage cell.

They use a combinatorial definition to define cell directly.

Because state is implicit, it is not always easy to keep track of it.

Defining strategies combinatorially doesn't tell us what properties of the category of games make it suitable for modelling stateful programs.

Case Study: The Abramsky-McCusker model of Idealized Algol

Abramsky and McCusker's model of Idealized Algol (1998) uses a stateful strategy cell, representing the behaviour of a storage cell.

They use a combinatorial definition to define cell directly.

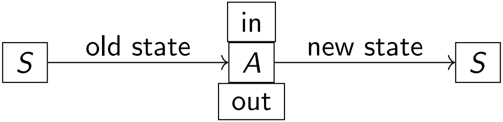
Because state is implicit, it is not always easy to keep track of it.

Defining strategies combinatorially doesn't tell us what properties of the category of games make it suitable for modelling stateful programs.

We want to find a way to *encapsulate* an explicit state in a systematic category-theoretic way.

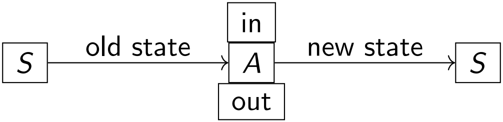
Final Coalgebras as State Transformers

We can think of a state transformer as a box



Final Coalgebras as State Transformers

We can think of a state transformer as a box



Then *encapsulation* means plugging an unbounded number of boxes together: