

# Guards, Structure and Power

Dan Marsden<sup>1</sup>

March 26, 2020

---

<sup>1</sup>Joint work with Samson Abramsky, Tom Paine and Nihil Shah 

# Outline

- ▶ Background on Games
- ▶ Introduction to the Structure and Power set up
- ▶ Detailed examination of modal logic related comonads
- ▶ Guarded logic and appropriate comonads

# Model Comparison Games

- ▶ Pebbling games - model equivalence with  $k$ -variable FOL.
- ▶ Ehrenfeucht-Fraïssé games - model equivalence with quantifier depth  $k$  FOL.
- ▶ **Bisimulation games** - “Behavioural equivalence” for  $k$ -steps

# Basic Bisimilarity

## Bisimulations and Bisimilarity

Given two non-deterministic transition systems, *Left* and *Right*, a *bisimulation between Left and Right* is a binary relation  $B$  such that if  $B(l, r)$ :

- ▶ If  $l \rightarrow l'$  then there exists  $r'$  such that  $r \rightarrow r'$  and  $B(l', r')$ .
- ▶ If  $r \rightarrow r'$  then there exists  $l'$  such that  $l \rightarrow l'$  and  $B(l', r')$ .

If two states are related by a bisimulation, we say that they are *bisimilar*.

# Basic Bisimilarity

## Bisimulations and Bisimilarity

Given two non-deterministic transition systems, Left and Right, a *bisimulation between Left and Right* is a binary relation  $B$  such that if  $B(l, r)$ :

- ▶ If  $l \rightarrow l'$  then there exists  $r'$  such that  $r \rightarrow r'$  and  $B(l', r')$ .
- ▶ If  $r \rightarrow r'$  then there exists  $l'$  such that  $l \rightarrow l'$  and  $B(l', r')$ .

If two states are related by a bisimulation, we say that they are *bisimilar*.

## Interactive Perspective

We can phrase this as a 2-player game between:

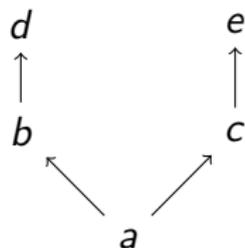
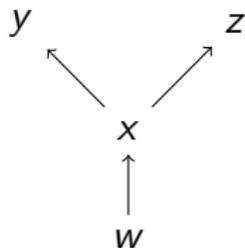
- ▶ Spoiler - choosing moves to show that two states are not bisimilar
- ▶ Duplicator - choosing responses maintaining that the states are bisimilar

# Bisimulation Games

## A Win for Duplicator

### Example (A Play of the Bisimulation Game)

In the structures below,  $w$  and  $a$  are bisimilar.

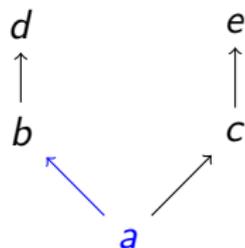
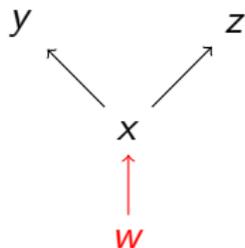


# Bisimulation Games

## A Win for Duplicator

### Example (A Play of the Bisimulation Game)

In the structures below,  $w$  and  $a$  are bisimilar.

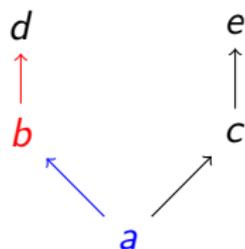
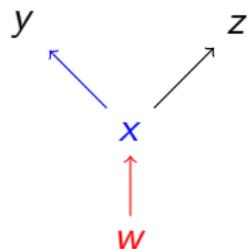


# Bisimulation Games

## A Win for Duplicator

### Example (A Play of the Bisimulation Game)

In the structures below,  $w$  and  $a$  are bisimilar.

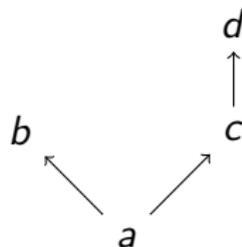


# Bisimulation Games

A Win for Spoiler

## Example (Another Play of the Bisimulation Game)

In the structures below,  $w$  and  $a$  are *not* bisimilar:

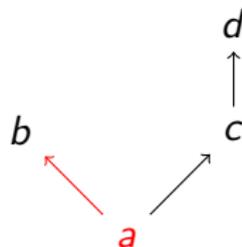


# Bisimulation Games

A Win for Spoiler

## Example (Another Play of the Bisimulation Game)

In the structures below,  $w$  and  $a$  are *not* bisimilar:

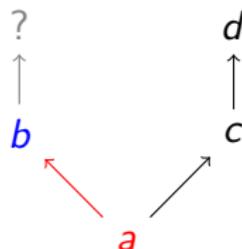


# Bisimulation Games

## A Win for Spoiler

### Example (Another Play of the Bisimulation Game)

In the structures below,  $w$  and  $a$  are *not* bisimilar:



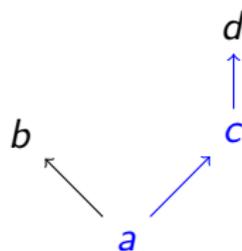
And spoiler has won. But notice it took 2 rounds to force the win.

# A Simulation Game

## A Restricted Game

We now consider games where players are restricted to one side.

Example (Spoiler on the left)



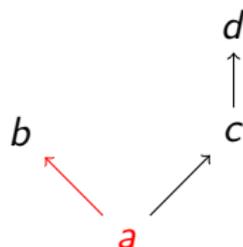
If Duplicator has a winning strategy, as in this case, we say *Right simulates Left*.

# A Simulation Game

## A Restricted Game

We now consider games where players are restricted to one side.

Example (Spoiler on the right 1)

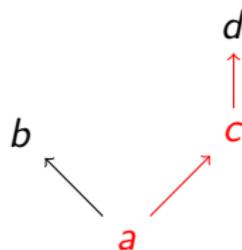


# A Simulation Game

## A Restricted Game

We now consider games where players are restricted to one side.

Example (Spoiler on the right 2)



So we also have Left simulates Right. But they are *not* bisimilar.

# Relational Structures

## Relational Structures

Basic definitions:

- ▶ A *relational signature*  $\Sigma$  is a set of relation symbols, each with an associated arity.
- ▶ A *relational structure* over  $\Sigma$  is a set  $A$  equipped with a relation  $R^\sigma \subseteq A^n$  for each relation symbol  $\sigma \in \Sigma$  with arity  $n$ .
- ▶ A *homomorphism of relational structures* of type  $h : A \rightarrow B$  is a function between the underlying sets such that:

$$R^\sigma(a_1, \dots, a_n) \Rightarrow R^\sigma(ha_1, \dots, ha_n)$$

# Making Life Easier

## Informal Question

If there is no homomorphism of type:

$$A \rightarrow B$$

How can we make it *easier* to construct one?

# Making Life Easier

## Informal Question

If there is no homomorphism of type:

$$A \rightarrow B$$

How can we make it *easier* to construct one?

## Imprecise Answer

We make the codomain “bigger”:

$$A \rightarrow B$$

# Making Life Harder

## Dual Informal Question

If there is a homomorphism of type:

$$A \rightarrow B$$

How can we make it *harder* to construct one?

## Dual Imprecise Answer

$$A \rightarrow B$$

# Measuring How Hard Life Is?

$$A \rightarrow B$$

- ▶ Bigger on the right, life gets easier, as we have more *resources*.
- ▶ Bigger on the left, life gets harder, as we have more *coresources*.

# Keisler-Shelah Isomorphism Theorem

## An Analogous Result

### Elementary Equivalence as Isomorphism

Given relational structures  $A$  and  $B$  we can find “bigger” structures such that we have an *isomorphism*:

$$A \cong B$$

if and only if  $A$  and  $B$  are elementary equivalent.

# Categorical Framework

## The Plan

For a given notion of game, we wish to introduce a comonad  $D$  such that homomorphisms:

$$D(A) \rightarrow B$$

correspond to winning strategies for duplicator in the *existential version* of the game.

# Categorical Framework

## The Plan

For a given notion of game, we wish to introduce a comonad  $D$  such that homomorphisms:

$$D(A) \rightarrow B$$

correspond to winning strategies for duplicator in the *existential version* of the game. In fact, these comonads will be graded, with the grading quantifying the amount of coresources that can be catered for.

$$D^k(A) \rightarrow B$$

# Categorical Framework

## Concretely for Bisimulation

We construct a comonad  $D^k$  such that homomorphisms:

$$D^k(A) \rightarrow B$$

correspond to a winning strategy for the  $k$ -round simulation game showing  $B$  can simulate  $A$ .

# Recap: Whats a comonad?

## Comonads

A *comonad* on a category  $\mathcal{C}$  consists of:

- ▶ An endofunctor  $D : \mathcal{C} \rightarrow \mathcal{C}$ .
- ▶ A counit natural transformation  $\epsilon : D \Rightarrow 1$ .
- ▶ A comultiplication natural transformation  $\delta : D \Rightarrow D \circ D$ .

Satisfying obvious coherence conditions. More succinctly, a comonad on  $\mathcal{C}$  is a comonoid in the endofunctor monoidal category  $([\mathcal{C}, \mathcal{C}], \circ, 1)$ .

# An Instructive Example

## Example (Non-empty lists)

There is a comonad on the category of sets and functions with:

- ▶  $D(X)$  is the set of non-empty finite lists of elements from  $X$ .
- ▶  $\epsilon$  is the tail function, e.g.  $\epsilon[x, y, z] = z$ .
- ▶  $\delta$  is the prefix function, e.g.  $\delta[x, y, z] = [[x], [x, y], [x, y, z]]$ .

## A Baby Comonad for Bisimulation

We will be looking at bisimilarity between a particular pair of elements  $(a_0, b_0)$ , so we use pointed structures  $(A, a_0)$ .

- ▶ We would like elements of  $D(X)$  to encode spoiler's moves so far, starting from  $a_0$ .

## A Baby Comonad for Bisimulation

We will be looking at bisimilarity between a particular pair of elements  $(a_0, b_0)$ , so we use pointed structures  $(A, a_0)$ .

- ▶ We would like elements of  $D(X)$  to encode spoilers moves so far, starting from  $a_0$ .
- ▶ A natural choice would be sequences  $[a_0, a_1, \dots, a_n]$  where there is a transition  $a_i \rightarrow a_{i+1}$ . It will be convenient to write these:

$$[a_0 \rightarrow a_1 \rightarrow \dots \rightarrow a_n]$$

## A Baby Comonad for Bisimulation

We will be looking at bisimilarity between a particular pair of elements  $(a_0, b_0)$ , so we use pointed structures  $(A, a_0)$ .

- ▶ We would like elements of  $D(X)$  to encode spoilers moves so far, starting from  $a_0$ .
- ▶ A natural choice would be sequences  $[a_0, a_1, \dots, a_n]$  where there is a transition  $a_i \rightarrow a_{i+1}$ . It will be convenient to write these:

$$[a_0 \rightarrow a_1 \rightarrow \dots \rightarrow a_n]$$

- ▶ To lift this to a transition system, we add transitions:

$$[a_0 \rightarrow \dots \rightarrow a_n] \rightarrow [a_0 \rightarrow \dots \rightarrow a_n \rightarrow a_{n+1}]$$

## A Baby Comonad for Bisimulation

We will be looking at bisimilarity between a particular pair of elements  $(a_0, b_0)$ , so we use pointed structures  $(A, a_0)$ .

- ▶ We would like elements of  $D(X)$  to encode spoilers moves so far, starting from  $a_0$ .
- ▶ A natural choice would be sequences  $[a_0, a_1, \dots, a_n]$  where there is a transition  $a_i \rightarrow a_{i+1}$ . It will be convenient to write these:

$$[a_0 \rightarrow a_1 \rightarrow \dots \rightarrow a_n]$$

- ▶ To lift this to a transition system, we add transitions:

$$[a_0 \rightarrow \dots \rightarrow a_n] \rightarrow [a_0 \rightarrow \dots \rightarrow a_n \rightarrow a_{n+1}]$$

- ▶ Finally, we make  $[a_0]$  the point of the new structure.

## A Baby Comonad for Bisimulation

We will be looking at bisimilarity between a particular pair of elements  $(a_0, b_0)$ , so we use pointed structures  $(A, a_0)$ .

- ▶ We would like elements of  $D(X)$  to encode spoilers moves so far, starting from  $a_0$ .
- ▶ A natural choice would be sequences  $[a_0, a_1, \dots, a_n]$  where there is a transition  $a_i \rightarrow a_{i+1}$ . It will be convenient to write these:

$$[a_0 \rightarrow a_1 \rightarrow \dots \rightarrow a_n]$$

- ▶ To lift this to a transition system, we add transitions:

$$[a_0 \rightarrow \dots \rightarrow a_n] \rightarrow [a_0 \rightarrow \dots \rightarrow a_n \rightarrow a_{n+1}]$$

- ▶ Finally, we make  $[a_0]$  the point of the new structure.
- ▶ We restrict to sequences of length  $k$  to yield a comonad for  $k$ -step bisimilarity.

# Grown-up Bisimulation

Our notion of bisimilarity was as simple as possible. Two natural extensions:

- ▶ Allow for multiple different transition relations  $\alpha, \beta, \dots$
- ▶ Allow unary predicates on states,  $P, Q, \dots$

## Labelled Transition System Bisimilarity

Given two non-deterministic labelled transition systems, Left and Right, a *bisimulation between Left and Right* is a binary relation  $B$  such that if  $B(l, r)$ :

- ▶ For all unary predicates  $P(l)$  if and only if  $P(r)$ .
- ▶ If  $l \xrightarrow{\alpha} l'$  then there exists  $r'$  such that  $r \xrightarrow{\alpha} r'$  and  $B(l', r')$ .
- ▶ If  $r \xrightarrow{\alpha} r'$  then there exists  $l'$  such that  $l \xrightarrow{\alpha} l'$  and  $B(l', r')$ .

If two states are related by a bisimulation, we say that they are *bisimilar*.

# A Grown-Up Bisimulation Comonad

We adjust our baby comonad as follows:

- ▶ We now define  $D(X)$  to be sequences of the form:

$$[a_0 \xrightarrow{\alpha} a_1 \dots a_{n-1} \xrightarrow{\gamma} a_n]$$

- ▶ We generate the transition relations on our new structure as follows:

$$[a_0 \dots a_n] \xrightarrow{\alpha} [a_0 \dots a_n \xrightarrow{\alpha} a_{n+1}]$$

- ▶ Predicates are defined on the new structure by:

$$P([a_0 \dots a_n]) \Leftrightarrow P(a_n)$$

# Modal Logic

## Syntax

$$\varphi = p \mid \neg\varphi \mid \varphi \wedge \psi \mid \diamond_{\alpha}\varphi \mid \perp$$

## Intuitive Reading

- ▶  $p$  - proposition  $P$  holds in the current state.
- ▶  $\diamond_{\alpha}p$  - we can make an  $\alpha$  transition to a state in which proposition  $P$  holds.
- ▶ Logical connectives have their usual reading, for example  $\varphi \wedge \psi$  -  $\varphi$  and  $\psi$  hold in the current state.

# Semantics

## Translation

Given a modal formula, we can construct a unary formula  $\llbracket \varphi \rrbracket(x)$  in FOL as follow:

$$\llbracket \diamond_{\alpha} \varphi \rrbracket(x) := \exists y. R_{\alpha}(x, y) \wedge \llbracket \varphi \rrbracket(y)$$

$$\llbracket p \rrbracket(x) := P(x)$$

$$\llbracket \neg \varphi \rrbracket(x) := \neg \llbracket \varphi \rrbracket(x)$$

$$\llbracket \varphi \wedge \psi \rrbracket(x) := \llbracket \varphi \rrbracket(x) \wedge \llbracket \psi \rrbracket(x)$$

$$\llbracket \perp \rrbracket(x) := x \neq x$$

Call formulae equivalent to those in the image of this translation the *modal fragment of FOL*.

# Tying a Knot

## ML and Bisimulation

Modal logic is the bisimulation invariant fragment of FOL:

$$\text{FO}/\sim \equiv \text{ML}$$

# Tying a Knot

## ML and Bisimulation

Modal logic is the bisimulation invariant fragment of FOL:

$$\text{FO}/\sim \equiv \text{ML}$$

## But why do we care?

- ▶ Modal logic is a bit weak - we cannot make natural sounding statements about transition systems in ML.
- ▶ Modal logic has good computational and model theoretic properties - decidable, finite model property, tree model property.

# Tying a Knot

## ML and Bisimulation

Modal logic is the bisimulation invariant fragment of FOL:

$$\text{FO}/\sim \equiv \text{ML}$$

## But why do we care?

- ▶ Modal logic is a bit weak - we cannot make natural sounding statements about transition systems in ML.
- ▶ Modal logic is has good computational and model theoretic properties - decidable, finite model property, tree model property.
- ▶ Modal logic is remarkably well behaved when extended with useful features.

# Why is ML Nice?

## The Straw Man

The ML translation lives within the 2-variable fragment of FOL logic. Is this the source of the good properties?

$$\begin{aligned}\llbracket \diamond_{\alpha} \diamond_{\beta} p \rrbracket(x) &= \exists y. R_{\alpha}(x, y) \wedge (\exists z. R_{\beta}(y, z) \wedge P(z)) \\ &= \exists y. R_{\alpha}(x, y) \wedge (\exists x. R_{\beta}(y, x) \wedge P(x))\end{aligned}$$

# Why is ML Nice?

## The Straw Man

The ML translation lives within the 2-variable fragment of FOL logic. Is this the source of the good properties?

$$\begin{aligned}\llbracket \diamond_{\alpha} \diamond_{\beta} p \rrbracket(x) &= \exists y. R_{\alpha}(x, y) \wedge (\exists z. R_{\beta}(y, z) \wedge P(z)) \\ &= \exists y. R_{\alpha}(x, y) \wedge (\exists x. R_{\beta}(y, x) \wedge P(x))\end{aligned}$$

Did anybody really believe this?

# Extensions with Fixed Points

- ▶ We can extend modal logic (variants) with fixed point operators. This greatly improves the expressive power.
- ▶ Generally this doesn't affect the notion of bisimilarity, but instead leads to expressive completeness results in stronger logics, for example:

$$\text{MSO}/\sim \equiv L_{\mu}$$

- ▶ From a model comparison point of view, these extensions “come for free”.

# Going Backwards

What if we add backwards modalities?

$$\llbracket \diamond_{\alpha}^{-} \varphi \rrbracket (x) = \exists y. R_{\alpha}(y, x) \wedge \llbracket \varphi \rrbracket (y)$$

# Going Backwards

What if we add backwards modalities?

$$\llbracket \diamond_{\alpha}^{-} \varphi \rrbracket (x) = \exists y. R_{\alpha}(y, x) \wedge \llbracket \varphi \rrbracket (y)$$

We need to adjust our notion of bisimilarity, by adding two new clauses, for  $B(l, r)$

- ▶ For all unary predicates  $P(l)$  if and only if  $P(r)$ .
- ▶ If  $l \xrightarrow{\alpha} l'$  then there exists  $r'$  such that  $r \xrightarrow{\alpha} r'$  and  $B(l', r')$ .
- ▶ If  $r \xrightarrow{\alpha} r'$  then there exists  $l'$  such that  $l \xrightarrow{\alpha} l'$  and  $B(l', r')$ .
- ▶ If  $l' \xrightarrow{\alpha} l$  then there exists  $r'$  such that  $r' \xrightarrow{\alpha} r$  and  $B(l', r')$ .
- ▶ If  $r' \xrightarrow{\alpha} r$  then there exists  $l'$  such that  $l' \xrightarrow{\alpha} l$  and  $B(l', r')$ .

# A Comonad for Two-Way Bisimulation

We can adjust our previous comonad for ML as follows:

- ▶ We now consider sequences with forwards and backwards edges, for example:

$$[a_0 \xrightarrow{\alpha} a_1 \xleftarrow{\beta} a_2]$$

Respecting transition relations appropriately.

- ▶ We extend the edge relations in the resulting structure, now with two rules:

$$\begin{aligned} [a_0 \dots a_n] &\xrightarrow{\alpha} [a_0 \dots a_n \xrightarrow{\alpha} a_{n+1}] \\ [a_0 \dots a_n \xleftarrow{\alpha} a_{n+1}] &\xrightarrow{\alpha} [a_0 \dots a_n] \end{aligned}$$

- ▶ The remaining structure is analogous to before.

# Jumping About

What if we add a global modality?

$$\llbracket \exists \varphi \rrbracket (x) = \exists y. \llbracket \varphi \rrbracket (y)$$

# Jumping About

What if we add a global modality?

$$\llbracket \exists \varphi \rrbracket (x) = \exists y. \llbracket \varphi \rrbracket (y)$$

We need to adjust our notion of bisimilarity:

- ▶ For all unary predicates  $P(l)$  if and only if  $P(r)$ .
- ▶ If  $l \xrightarrow{\alpha} l'$  then there exists  $r'$  such that  $r \xrightarrow{\alpha} r'$  and  $B(l', r')$ .
- ▶ If  $r \xrightarrow{\alpha} r'$  then there exists  $l'$  such that  $l \xrightarrow{\alpha} l'$  and  $B(l', r')$ .
- ▶ For all  $a'$  there exists  $b'$  such that  $B(a, b')$
- ▶ For all  $b'$  there exists  $a'$  such that  $B(a', b)$

# A Comonad Incorporating Global Modalities

We can further adjust our comonad as follows:

- ▶ We add a third edge type  $\exists \rightarrow$ , so we now have sequences of the form:

$$[a_0 \xrightarrow{\alpha} a_1 \xleftarrow{\beta} a_2 \xrightarrow{\exists} a_3]$$

Where  $\exists$ -edges may appear between any two states.

- ▶ We don't add a new edges in the resulting structure, and the remaining components remain as before.

# A Comonad Incorporating Global Modalities

We can further adjust our comonad as follows:

- ▶ We add a third edge type  $\exists \rightarrow$ , so we now have sequences of the form:

$$[a_0 \xrightarrow{\alpha} a_1 \xleftarrow{\beta} a_2 \xrightarrow{\exists} a_3]$$

Where  $\exists$ -edges may appear between any two states.

- ▶ We don't add a new edges in the resulting structure, and the remaining components remain as before.
- ▶ But, we could have instead have allowed sequences to start anywhere, not just at  $a_0$ . Take home message - *there will in general be non-isomorphic comonads encoding the same game.*

## So what have we added?

The various modalities appear in FOL as:

- ▶ Ordinary ML modality:

$$\exists y. R_\alpha(x, y) \wedge \varphi(y)$$

- ▶ Backwards ML modality:

$$\exists y. R_\alpha(y, x) \wedge \varphi(x)$$

- ▶ Global modality:

$$\exists y. \varphi(y)$$

$$\exists y. (y = y) \wedge \varphi(y)$$

## So what have we added?

The various modalities appear in FOL as:

- ▶ Ordinary ML modality:

$$\exists y. R_\alpha(x, y) \wedge \varphi(y)$$

- ▶ Backwards ML modality:

$$\exists y. R_\alpha(y, x) \wedge \varphi(x)$$

- ▶ Global modality:

$$\exists y. \varphi(y)$$

$$\exists y. (y = y) \wedge \varphi(y)$$

- ▶ We could also consider polyadic modalities:

$$\llbracket \diamond_\pi(\varphi, \psi) \rrbracket(x) = \exists y, z. R_\pi(x, y, z) \wedge \llbracket \varphi \rrbracket(y) \wedge \llbracket \psi \rrbracket(z)$$

(although bisimilarity and the comonad get uglier)

# Re-inventing (Atom) Guarded Logic

The previous use of quantifiers were all of the form:

$$\exists \bar{y}. \alpha(\bar{x}, \bar{y}) \wedge \varphi(\bar{y})$$

Widely generalizing what we saw on the previous two slides, we restrict to *any* use of quantifiers of the form:

$$\exists \bar{y}. \alpha(\bar{x}, \bar{y}) \wedge \varphi(\bar{x}, \bar{y}) \quad \forall \bar{y}. \alpha(\bar{x}, \bar{y}) \Rightarrow \varphi(\bar{x}, \bar{y})$$

Here:

- ▶ We use vector quantifiers as things cannot be done iteratively in general.
- ▶  $\alpha$  is an atom, referred to as the *guard*, an  $\bar{x}, \bar{y}$  *must* appear in  $\alpha$ . The variable appearing in  $\bar{x}, \bar{y}$  is called a *guarded set*.
- ▶  $\varphi$  is a formula in which only variables in  $\bar{x}, \bar{y}$  *may* appear.
- ▶ Note that we are certainly not restricted to two variables!

# Guarded Bisimulation

Following the pattern that has emerged, we need yet another notion of bisimulation.

## Guarded Bisimulation

We consider a non-empty set  $I$  of partial isomorphisms rather than a binary relation  $B$ .

- ▶ For every guarded set  $X' \subseteq A$  there exists  $f' \in I$  with domain  $X'$  such that  $f$  and  $f'$  agree on  $X \cap X'$ .
- ▶ For every guarded set  $Y' \subseteq B$  there exists  $f' \in I$  with range  $Y'$  such that  $f^{-1}$  and  $f'^{-1}$  agree on  $Y \cap Y'$ .

# GF Comonad, Take One

## Back and Forth Condition

For every guarded set  $X' \subseteq A$  there exists  $f' \in I$  with domain  $X'$  such that  $f$  and  $f'$  agree on  $X \cap X'$ .

- ▶ So we're interested in sequences of guarded sets  $[S_1, S_2, \dots, S_n]$ .
- ▶ We restrict to sequences  $S_i \cap S_{i+1} \neq \emptyset$ .
- ▶ We need to say where each element of these sets should go, we instead we consider pairs of the form:

$$([S_1, \dots, S_n], a)$$

with the  $S_i$  overlapping, and  $a \in S_n$ .

# GF Comonad, Take One

## Back and Forth Condition

For every guarded set  $X' \subseteq A$  there exists  $f' \in I$  with domain  $X'$  such that  $f$  and  $f'$  agree on  $X \cap X'$ .

- ▶ We need to force the “agree on overlaps condition”, so we quotient:

$$([S_1, \dots, S_n], a) \sim ([S_1, \dots, S_n, S_{n+1}], a)$$

- ▶ We add relations based on the second components of the pairs.
- ▶  $\epsilon$  extracts the second component, and  $\delta$  is a bit icky!
- ▶ This all works out after detailed checking, and yields a legitimate comonad on relational structures.

# GF Comonad, Take One

## Discussion

### Choices

- ▶ We chose to enforce pairwise overlap in our sequences  $[S_1, \dots, S_n]$ . This is not essential, just less “flabby”.

# GF Comonad, Take One

## Discussion

### Choices

- ▶ We chose to enforce pairwise overlap in our sequences  $[S_1, \dots, S_n]$ . This is not essential, just less “flabby”.
- ▶ More importantly, the quotient is icky, and seems slightly morally wrong from a comonadic point of view.

# GF Comonad, Take Two

Cleaning up a bit

- ▶ Consider two pairs:

$$([S_1, \dots, S_n], a) \quad \text{and} \quad ([S_1, \dots, S_n, S_{n+1}], a)$$

We don't really need the second pair, so we can just throw it away.

# GF Comonad, Take Two

## Cleaning up a bit

- ▶ Consider two pairs:

$$([S_1, \dots, S_n], a) \quad \text{and} \quad ([S_1, \dots, S_n, S_{n+1}], a)$$

We don't really need the second pair, so we can just throw it away.

- ▶ More generally, we call a pair:

$$([S_1, \dots, S_n, S_{n+1}], a)$$

*canonical* if  $a$  appears in  $S_{n+1}$ , but not in  $S_n$ . We restrict our attention to canonical pairs.

# GF Comonad, Take Two

## Cleaning up a bit

- ▶ Consider two pairs:

$$([S_1, \dots, S_n], a) \quad \text{and} \quad ([S_1, \dots, S_n, S_{n+1}], a)$$

We don't really need the second pair, so we can just throw it away.

- ▶ More generally, we call a pair:

$$([S_1, \dots, S_n, S_{n+1}], a)$$

*canonical* if  $a$  appears in  $S_{n+1}$ , but not in  $S_n$ . We restrict our attention to canonical pairs.

- ▶ During our constructions, non-canonical pairs naturally arise. We can always *canonicalize* by “working backwards” to a canonical pair.
- ▶ This again yields a legitimate comonad after detailed checking, circumventing the aesthetically distracting quotient.

# GF Comonad, Take Two

## Discussion

By working with canonical pairs:

- ▶ We simplify studying homomorphisms:

$$D(X) \rightarrow Y$$

as  $D(X)$  avoids the need for a quotient.

- ▶ Some of the structure needs to be carefully canonicalized in places, so depending on your preferences, some of the comonadic structure may seem slightly more complicated.

# Conclusions

- ▶ So far we have candidate comonads for the guarded fragment, and various intermediate logics extending ordinary ML. These should generalize smoothly to more general guards, as far as clique guarded logics.
- ▶ There is also Unary Negation Logic (UNFO), and the very general Guarded Negation Logic (GNFO) - comonads for these are work in progress.
- ▶ The aim then is to study computational and model theoretic aspects of these logics, from the semanticists point of view.
- ▶ It would be nice to be able to present these comonads in a cleaner way. For monads equational presentations are incredibly useful, ambition to have analogous tools for the dual situation.