



## Outline

### Basics

Induction and coinduction  
Algebras and coalgebras

Bisimilarity and finality  
Language example

Coalgebras and quantum computing  
Introducing QBits  
Combining qubits  
Monads for computations  
Walks illustrating computation types  
Quantum walks, coalgebraically  
Reversibility of computation

## Introduction to Coalgebra

Bart Jacobs

Institute for Computing and Information Sciences – Digital Security  
Radboud University Nijmegen

EWSCS 2011: 16th Estonian Winter School in Computer Science  
28 feb. – 4 march, Estonia

4th March 2011

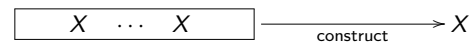


## What are coalgebras, intuitively

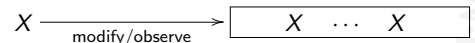
- Mathematical models for **state-based** computation
  - your computer has a complicated internal state
  - you as user can only modify it to some extent
  - also, you can observe only so much
- Basically only two kinds of operations:
  - move to a next state, somehow (deterministically, non-deterministically, probabilistically, ...)
  - make an observation ("measurement") about the current state
  - These operations can be combined ("observation has side-effect")
- Describing such state-based systems and their dynamics requires a new kind of mathematics.
  - related to dynamical systems (and evolution via differential equations)

## (Co)algebras, pictorially

**Algebras** with "carrier"  $X$  are maps **into**  $X$ , of the form:



**Coalgebras** with "state space"  $X$  are maps **out of**  $X$ , of the form:



Think of a fields & methods in an object-oriented class

- Formally, the boxes are **functors**, usually **Sets**  $\rightarrow$  **Sets**, ...
- ... but at this stage you may think of them as **types**  $\sigma(X)$  containing a single type variable  $X$



## Intermezzo: type constructors for "boxes"

- Identity  $X$ , and constants  $A$ 
  - including empty set  $\emptyset$ , often written as  $0$
  - singleton set  $1 = \{*\}$
- **products**  $X \times Y = \{\langle x, y \rangle \mid x \in X, y \in Y\}$ , with **projections**:

$$X \xleftarrow{\pi_1} X \times Y \xrightarrow{\pi_2} Y \quad \begin{array}{c} Z \xrightarrow{\langle f, g \rangle} X \times Y \\ \hline \begin{array}{cc} Z \xrightarrow{f} X & Z \xrightarrow{g} Y \end{array} \end{array}$$

- infinite version usually written as  $\prod_{i \in I} X_i$

- **coproducts**  $X + Y = \{\langle x, 0 \rangle \mid x \in X\} \cup \{\langle y, 1 \rangle \mid y \in Y\}$ , with **coprojections**

$$X \xrightarrow{\kappa_1} X + Y \xleftarrow{\kappa_2} Y \quad \begin{array}{c} X + Y \xrightarrow{\langle f, g \rangle} Z \\ \hline \begin{array}{cc} X \xrightarrow{f} Z & Y \xrightarrow{g} Z \end{array} \end{array}$$

- infinite version written as  $\prod_{i \in I} X_i$

## Algebra examples $F(X) \rightarrow X$

- **Natural numbers**

$$1 + \mathbb{N} \xrightarrow{[\text{zero}, \text{succ}]} \mathbb{N} \quad \text{i.e.} \quad \begin{cases} \text{zero: } 1 \rightarrow \mathbb{N} \\ \text{succ: } \mathbb{N} \rightarrow \mathbb{N} \end{cases}$$

The functor/type here is  $F(X) = 1 + X$ .

- **Groups**

$$1 + (G \times G) + G \xrightarrow{[0, +, -]} G \quad \text{i.e.} \quad \begin{cases} 0: 1 \rightarrow G \\ +: G \times G \rightarrow G \\ -: G \rightarrow G \end{cases}$$

- Clearly this works for many **algebraic structures**: for each signature  $\Sigma$  there is a functor:

$$X \mapsto \prod_{f \in \Sigma} X^{\text{arity}(f)} \quad (\text{disjoint union over all operations})$$

whose algebras are precisely the  $\Sigma$ -algebras.

**Note: this captures algebraic operations, not equations**



## Coalgebra examples $X \rightarrow F(X)$

- Streams (infinite lists)  $A^{\mathbb{N}}$  of elements of a set  $A$ :

$$A^{\mathbb{N}} \xrightarrow{\langle \text{head}, \text{tail} \rangle} A \times A^{\mathbb{N}}$$

for the functor  $F(X) = A \times X$ , where:

$$\begin{cases} \text{head}(\alpha) = \alpha(0) \\ \text{tail}(\alpha) = \alpha' = \lambda n \in \mathbb{N}. \alpha(n+1) \end{cases}$$

- Both finite and infinite lists  $A^{\infty} = A^* + A^{\mathbb{N}}$

$$A^{\infty} \xrightarrow{\text{possible head}} 1 + (A \times A^{\infty})$$

for the functor  $F(X) = 1 + A \times X$ .

**Note:** observation with side-effect



## Markov chains

For a set  $X$ , define the set of discrete distributions on  $X$

$$\mathcal{D}(X) = \{\varphi: X \rightarrow [0, 1] \mid \text{support}(\varphi) \text{ is finite, and } \sum_x \varphi(x) = 1\}$$

Such  $\varphi \in \mathcal{D}(X)$  is a formal convex combination:

$$r_1 x_1 + \dots + r_n x_n \quad \text{where} \quad \begin{cases} \text{support}(\varphi) = \{x_1, \dots, x_n\} \\ r_i = \varphi(x_i) > 0 \\ r_1 + \dots + r_n = 1 \end{cases}$$

Coalgebras  $X \rightarrow \mathcal{D}(X)$  are Markov chains, giving probabilistic transitions:

$$x \xrightarrow{r_i} x_i \quad \text{with} \quad \sum_i r_i = 1.$$

They can also be used in labeled form, as

$$X \rightarrow \mathcal{D}(X)^A \quad \text{or as} \quad X \rightarrow \mathcal{D}(A \times X)$$



## Functors, part I

- Before we can proceed we need to know a little bit more about functors, for the time being only on sets
- In functional programming "functoriality" is often described by "map" properties, such as `list_map`
- A functor  $F$  maps sets  $X$  to sets  $F(X)$ , and also functions to functions, in an orderly manner:
  - if  $g: X \rightarrow Y$ , then  $F(g): F(X) \rightarrow F(Y)$
  - $F$  preserves identity maps:  $F(X \xrightarrow{\text{id}} X) = F(X) \xrightarrow{\text{id}} F(X)$
  - $F$  preserves composition:  $F(h \circ g) = F(h) \circ F(g)$ , in:
 
$$F(X \xrightarrow{g} Y \xrightarrow{h} Z) = (F(X) \xrightarrow{F(g)} F(Y) \xrightarrow{F(h)} F(Z))$$
- Often, when the action on functions is "obvious", it is omitted from the description of a functor



## Automata-theoretic coalgebra examples

- A deterministic automaton with input actions  $A$  is a coalgebra:

$$X \xrightarrow{\langle \text{step}, \text{final?} \rangle} X^A \times 2$$

where  $2 = \{0, 1\}$ .

- A non-deterministic automaton, or transition system, with input actions  $A$  is a coalgebra:

$$X \xrightarrow{\text{succs}} \mathcal{P}(X)^A$$

- Other functors (actually "monads") than powerset  $\mathcal{P}$  may be used: eg. partial automata via lift, or probabilistic automata via distribution.



## Coalgebra essentials & Plan

- Generic framework for describing state-based systems ✓
- Coinductive definition and reasoning principles Lect. II, III
- Bisimilarity, as indistinguishability of states Lect. III, IV
- Modal logic, also generically
- Coherence between operational & denotational semantics
- General trace semantics
- Current own interest: coalgebra & quantum computing Lect. V
  - Quantum language has strong coalgebraic flavour: states, transitions, observations, indistinguishability, ...
  - Can this be made mathematically precise?



## Functor example I

- For a set  $X$  write  $\mathcal{P}(X)$  for the powerset: the set of all subsets of  $X$
- This  $\mathcal{P}(-)$  forms a functor
- For a function  $g: X \rightarrow Y$  there is a suitable function  $\mathcal{P}(X) \rightarrow \mathcal{P}(Y)$ 
  - this is new function is written  $\mathcal{P}(g): \mathcal{P}(X) \rightarrow \mathcal{P}(Y)$
  - it is given by image:

$$\begin{aligned} \mathcal{P}(g)(U \subseteq X) &= \{g(x) \mid x \in U\} \subseteq Y \\ &= \{y \in Y \mid \exists x \in U. y = g(x)\} \\ &= g(U) \quad \text{as it is sometimes written} \\ &= g[U] \quad \text{also possible} \\ &= \coprod_g(U) \quad \text{for categorical logicians} \end{aligned}$$





## Functor example I

We check explicitly:

$$\mathcal{P}(\text{id})(U) = \{\text{id}(x) \mid x \in U\} = U$$

Hence  $\mathcal{P}(\text{id}) = \text{id}$ . Similarly,  $\mathcal{P}$  preserves composition:

$$\begin{aligned} (\mathcal{P}(h) \circ \mathcal{P}(g))(U) &= \mathcal{P}(h)(\mathcal{P}(g)(U)) \\ &= \mathcal{P}(h)(\{y \in Y \mid \exists x \in U. y = g(x)\}) \\ &= \{z \in Z \mid \exists y \in Y. \exists x \in U. y = g(x) \text{ and } z = h(y)\} \\ &= \{z \in Z \mid \exists x \in U. z = h(g(x))\} \\ &= \mathcal{P}(h \circ g)(U) \end{aligned}$$

## Functor example II

- For a set  $X$  write  $X^*$  for the associated set of finite lists:

$$X^* = \{\langle x_1, x_2, \dots, x_n \rangle \mid x_i \in X\}$$

- The mapping  $X \mapsto X^*$  is a **functor**
- For  $g: X \rightarrow Y$  we have  $g^*: X^* \rightarrow Y^*$  given by:

$$g^*(\langle x_1, x_2, \dots, x_n \rangle) = \langle g(x_1), g(x_2), \dots, g(x_n) \rangle$$

- It is not hard to check the functoriality properties:

$$\text{id}^* = \text{id} \quad \text{and} \quad (h \circ g)^* = h^* \circ g^*$$



## Functor examples II

- Infinite lists:**  $X \mapsto X^{\mathbb{N}}$  with:  $g^{\mathbb{N}}: X^{\mathbb{N}} \rightarrow Y^{\mathbb{N}}$ , given by:

$$g^{\mathbb{N}}(\langle x_0, x_1, x_2, \dots \rangle) = \langle g(x_0), g(x_1), g(x_2), \dots \rangle$$

- Product with a constant**  $X \mapsto X \times A$ , with:

$$g \times A: X \times A \rightarrow Y \times A \quad \text{given by} \quad (g \times A)(x, a) = (g(x), a)$$

- Coproduct with a constant**  $X \mapsto X + A$ , with

$$(g+A): X+A \rightarrow Y+A \quad \text{given by} \quad \begin{cases} (g+A)(\kappa_1 x) = \kappa_1 g(x) \\ (g+A)(\kappa_2 a) = \kappa_2 a \end{cases}$$

### Definition

Let  $F$  be a functor

- An **algebra** for  $F$  is a map  $F(X) \rightarrow X$
- A **coalgebra** for  $F$  is a map  $X \rightarrow F(X)$

### Definition

- An **algebra homomorphism** from  $F(X) \xrightarrow{a} X$  to  $F(Y) \xrightarrow{b} Y$  is a map  $f: X \rightarrow Y$  for which the rectangle on the left commutes.

$$\begin{array}{ccc} F(X) & \xrightarrow{F(f)} & F(Y) \\ a \downarrow & f & \downarrow b \\ X & \xrightarrow{f} & Y \end{array} \quad \begin{array}{ccc} F(X) & \xrightarrow{F(g)} & F(Y) \\ c \uparrow & g & \uparrow d \\ X & \xrightarrow{g} & Y \end{array}$$

- A **coalgebra homomorphism** from  $X \xrightarrow{c} F(X)$  to  $Y \xrightarrow{d} F(Y)$  is a map  $g: X \rightarrow Y$  for which the diagram on the right commutes.



## Algebra homomorphism example

- Consider the signature functor  $F(X) = 1 + (X \times X)$  for **monoids**
- The **natural numbers**  $\mathbb{N}$  form an algebra, via:

$$1 + (\mathbb{N} \times \mathbb{N}) \xrightarrow{[0,+]} \mathbb{N}$$

- Also **lists**  $A^*$  for an algebra via empty list  $\langle \rangle$  and list concatenation ; in:

$$1 + (A^* \times A^*) \xrightarrow{[\langle \rangle, ;]} A^*$$

- There is an **algebra homomorphism**:

$$\begin{array}{ccc} 1 + (\mathbb{N} \times \mathbb{N}) & \xrightarrow{\text{id} + (f \times f)} & 1 + (A^* \times A^*) \\ [0,+] \downarrow & f & \downarrow [\langle \rangle, ;] \\ \mathbb{N} & \xrightarrow{f} & A^* \end{array}$$

where  $f(n) = \underbrace{\langle a, a, \dots, a \rangle}_{n \text{ times}}$ , for some  $a \in A$ .

For  $A = 1 = \{*\}$   
we get  $\mathbb{N} \cong 1^*$

## Initiality and finality

### Definition

- An algebra  $F(A) \xrightarrow{\alpha} A$  is **initial** if for each algebra  $F(X) \xrightarrow{b} X$  there is a unique algebra map:

$$\begin{array}{ccc} F(A) & \xrightarrow{F(!_b)} & F(X) \\ \alpha \downarrow & & \downarrow b \\ A & \xrightarrow{!_b} & X \end{array}$$

- A coalgebra  $Z \xrightarrow{\zeta} F(Z)$  is **final** if for each coalgebra  $X \xrightarrow{c} F(X)$  there is a unique coalgebra map:

$$\begin{array}{ccc} F(X) & \xrightarrow{F(!_c)} & F(Z) \\ c \uparrow & & \uparrow \zeta \\ X & \xrightarrow{!_c} & Z \end{array}$$





## Fixed point results

### Fact

- An initial algebra is an isomorphism  $F(A) \xrightarrow{\cong} A$
- a final coalgebra is an isomorphism  $Z \xrightarrow{\cong} F(Z)$

**Consequence:** By Cantor's theorem, the powerset functor  $\mathcal{P}$  has neither an initial algebra nor a final coalgebra.

### Unique existence

- **existence** is used for definition by (co)induction
- **uniqueness** is used for reasoning by (co)induction



## Induction example, II

We wish to define the **length** function  $\text{len}: A^* \rightarrow \mathbb{N}$  formally, by initiality.

This requires an algebra map  $??$  on  $\mathbb{N}$  in:

$$1 + (A \times A^*) \xrightarrow{\text{id} + (\text{id} \times \text{len})} 1 + (A \times \mathbb{N})$$

$$\begin{array}{ccc} \downarrow [(\cdot, \cdot)] & & \downarrow [??] \\ A^* & \xrightarrow{\text{len}} & \mathbb{N} \end{array}$$

This  $??$  must be of the form  $[p, q]$ , with  $p: 1 \rightarrow \mathbb{N}$  and  $q: A \times \mathbb{N} \rightarrow \mathbb{N}$  satisfy:

$$\text{len}(\langle \rangle) = p \quad \text{and} \quad \text{len}(a \cdot \sigma) = q(a, n)$$

Hence it is clear how to choose  $p, q$ , namely as:

$$p = 0 \quad \text{and} \quad q(a, n) = n + 1 \quad \text{ie. as: } q = \lambda(a, n). n + 1$$



## Induction example, III (cntd)

We sketch how to do **tree traversal**  $\text{BinTree}(A) \rightarrow A^*$ , in two ways ("inorder" and "preorder")

$$1 + (\text{BinTree}(A) \times A \times \text{BinTree}(A)) \xrightarrow{\cong} 1 + (A^* \times A \times A^*)$$

$$\begin{array}{ccc} \cong \downarrow & & \downarrow [(\cdot, \cdot, \cdot)] \\ \text{BinTree}(A) & \xrightarrow{\quad} & A^* \end{array}$$

This map  $??: A^* \times A \times A^* \rightarrow A^*$  is:

$$\begin{cases} \text{for } \text{inorder} \text{ traversal} & (\sigma, a, \tau) \mapsto \sigma; a \cdot \tau \\ \text{for } \text{preorder} \text{ traversal} & (\sigma, a, \tau) \mapsto a \cdot \sigma; \tau \end{cases}$$



## Induction example, I

### Lemma

Fix a set  $A$  and consider the functor  $F(X) = 1 + (A \times X)$ .  
The initial algebra of  $F$  is given by finite lists  $A^*$  with algebra structure given by empty list  $(\text{nil}) \langle \rangle$  and prefix  $\cdot$  in:

$$1 + (A \times A^*) \xrightarrow[\cong]{[(\cdot, \cdot)]} A^*$$

**Proof:** The dashed map  $f$  in:

$$1 + (A \times A^*) \xrightarrow{\text{id} + (\text{id} \times f)} 1 + (A \times X)$$

$$\begin{array}{ccc} \downarrow [(\cdot, \cdot)] & & \downarrow [p, q] \\ A^* & \xrightarrow{f} & X \end{array}$$

is given by:  $f(\langle \rangle) = p$  and  $f(a \cdot \sigma) = q(a, f(\sigma))$ .  
It is a homomorphism by construction.



## Induction example, III

- Consider the functor  $F(X) = 1 + (X \times A \times X)$ , for a fixed set  $A$
- its operations give **binary,  $A$ -labeled trees**
  - what if we had taken  $F(X) = A + (X \times A \times X)$
  - or:  $F(X) = B + (X \times A \times X)$
  - or:  $F(X) = B + (X \times X \times X)$

- We write its **initial algebra** as:

$$1 + (\text{BinTree}(A) \times A \times \text{BinTree}(A)) \xrightarrow[\cong]{[\text{nil}, \text{node}]} \text{BinTree}(A)$$

- We do not really need to know what is inside the set  $\text{BinTree}(A)$ ; its **universal properties** suffice to be able to use it
  - a form of mathematical behaviourism



## Summary, so far

- Definitions by induction illustrated via initiality
- Initiality formulation is "generic": it works for every functor / signature
- This genericity is convenient when formalising induction in (functional) programming languages and theorem provers
- Examples of reasoning by induction (via uniqueness) are missing so far, but will be given next, for coinduction





## Coinduction example I

### Lemma

The *final coalgebra* of the functor  $F(X) = A \times X$  is the set  $A^{\mathbb{N}}$  of infinite sequences  $\langle a_0, a_1, \dots \rangle$  of elements  $a_i \in A$ , with coalgebra structure:

$$A^{\mathbb{N}} \xrightarrow[\cong]{\langle \text{hd}, \text{tl} \rangle} A \times A^{\mathbb{N}} \quad \begin{cases} \text{hd}(\sigma) = \sigma(0) \\ \text{tl}(\sigma) = \langle \sigma(1), \sigma(2), \dots \rangle \end{cases}$$

**Proof:** The required unique dashed map  $f$  in:

$$\begin{array}{ccc} A \times X & \xrightarrow{\text{id} \times f} & A \times A^{\mathbb{N}} \\ \langle p, q \rangle \uparrow & & \uparrow \langle \text{hd}, \text{tl} \rangle \\ X & \xrightarrow{f} & A^{\mathbb{N}} \end{array}$$

is given by  $f(x) = \langle p(x), p(q(x)), p(q^2(x)), p(q^3(x)), \dots \rangle$ .

## Coinduction example II

- We wish to define  $\text{even}: A^{\mathbb{N}} \rightarrow A^{\mathbb{N}}$
- This involves a coalgebra  $??$  in:

$$\begin{array}{ccc} A \times A^{\mathbb{N}} & \xrightarrow{\text{id} \times \text{even}} & A \times A^{\mathbb{N}} \\ ?? \uparrow & & \cong \uparrow \langle \text{hd}, \text{tl} \rangle \\ A^{\mathbb{N}} & \xrightarrow{\text{even}} & A^{\mathbb{N}} \end{array}$$

- If we write  $?? = \langle p, q \rangle$ , then commutation says:
 
$$\begin{cases} p(\sigma) = \text{hd}(\text{even}(\sigma)) = \sigma(0) = \text{hd}(\sigma) \\ \text{even}(q(\sigma)) = \text{tl}(\text{even}(\sigma)) = \langle \sigma(2), \sigma(4), \dots \rangle \end{cases}$$
- Thus we use:  $?? = \langle \text{hd}, \text{tl} \circ \text{tl} \rangle$

**Definition by coinduction: from single-step to "and-so-forth"**

## Coinduction example III

- We also wish  $\text{odd}: A^{\mathbb{N}} \rightarrow A^{\mathbb{N}}$
- Which coalgebra do we need now on the left?

$$\begin{array}{ccc} A \times A^{\mathbb{N}} & \xrightarrow{\text{id} \times \text{odd}} & A \times A^{\mathbb{N}} \\ \langle \text{hd}, \text{tl}, \text{tl} \rangle \uparrow & & \cong \uparrow \langle \text{hd}, \text{tl} \rangle \\ A^{\mathbb{N}} & \xrightarrow{\text{odd}} & A^{\mathbb{N}} \end{array}$$

- How to prove  $\text{odd}(\sigma) = \text{even}(\text{tl}(\sigma))$ ?
- Use **uniqueness**, and show that  $f(\sigma) = \text{even}(\text{tl}(\sigma))$  is also a coalgebra homomorphism in the above diagram!

$$\begin{aligned} \text{hd}(f(\sigma)) &= \text{hd}(\text{even}(\text{tl}(\sigma))) & \text{tl}(f(\sigma)) &= \text{tl}(\text{even}(\text{tl}(\sigma))) \\ &= \text{hd}(\text{tl}(\sigma)) & &= \text{even}(\text{tl}(\text{tl}(\sigma))) \\ & & &= f(\text{tl}(\sigma)) \end{aligned}$$

## Coinduction example IV

- The next aim is  $\text{merge}: A^{\mathbb{N}} \times A^{\mathbb{N}} \rightarrow A^{\mathbb{N}}$
- We thus have to provide a coalgebra  $c$  in a diagram:

$$\begin{array}{ccc} A \times (A^{\mathbb{N}} \times A^{\mathbb{N}}) & \xrightarrow{\text{id} \times \text{merge}} & A \times A^{\mathbb{N}} \\ c \uparrow & & \cong \uparrow \langle \text{hd}, \text{tl} \rangle \\ A^{\mathbb{N}} \times A^{\mathbb{N}} & \xrightarrow{\text{merge}} & A^{\mathbb{N}} \end{array}$$

- This coalgebra  $c: A^{\mathbb{N}} \times A^{\mathbb{N}} \rightarrow A \times (A^{\mathbb{N}} \times A^{\mathbb{N}})$  is:
 
$$c(\sigma, \tau) = \langle \text{hd}(\sigma), (\tau, \text{tl}(\sigma)) \rangle$$
- Then, for instance:
  - $\text{hd}(\text{merge}(\sigma, \tau)) = \text{hd}(\sigma)$
  - $\text{hd}(\text{tl}(\text{merge}(\sigma, \tau))) = \text{hd}(\text{merge}(\tau, \text{tl}(\sigma))) = \text{hd}(\text{merge}(\tau, \text{tl}(\sigma))) = \text{hd}(\tau)$
  - $\text{hd}(\text{tl}^2(\text{merge}(\sigma, \tau))) = \dots = \text{hd}(\text{tl}(\sigma))$  etc.

## Coinduction example V

- Now we wish to prove  $\text{merge}(\text{even}(\sigma), \text{odd}(\sigma)) = \sigma$
- Obviously, the identity  $\text{id}: A^{\mathbb{N}} \rightarrow A^{\mathbb{N}}$  is the unique coalgebra homomorphism  $g$  in:

$$\begin{array}{ccc} A \times A^{\mathbb{N}} & \xrightarrow{\text{id} \times g} & A \times A^{\mathbb{N}} \\ \langle \text{hd}, \text{tl} \rangle \uparrow & & \cong \uparrow \langle \text{hd}, \text{tl} \rangle \\ A^{\mathbb{N}} & \xrightarrow{g} & A^{\mathbb{N}} \end{array}$$

- Hence it suffices to prove that the map
 
$$f(\sigma) = \text{merge}(\text{even}(\sigma), \text{odd}(\sigma))$$
 is also such a coalgebra homomorphism.

## Coinduction example VI

Thus we calculate:

$$\begin{aligned} \text{hd}(f(\sigma)) &= \text{hd}(\text{merge}(\text{even}(\sigma), \text{odd}(\sigma))) \\ &= \text{hd}(\text{even}(\sigma)) \\ &= \text{hd}(\sigma) \end{aligned}$$

$$\begin{aligned} \text{tl}(f(\sigma)) &= \text{tl}(\text{merge}(\text{even}(\sigma), \text{odd}(\sigma))) \\ &= \text{merge}(\text{odd}(\sigma), \text{tl}(\text{even}(\sigma))) \\ &= \text{merge}(\text{even}(\text{tl}(\sigma)), \text{even}(\text{tl}(\text{tl}(\sigma)))) \\ &= \text{merge}(\text{even}(\text{tl}(\sigma)), \text{odd}(\text{tl}(\sigma))) \\ &= f(\text{tl}(\sigma)) \end{aligned}$$

**Conclusion:**  $\text{id} = f = \lambda \sigma. \text{merge}(\text{even}(\sigma), \text{odd}(\sigma))$



Summary on induction and coinduction

- Induction and coinduction, formulated via initiality and finality are each other's duals
- Initiality & finality mechanism is:
  - generic / uniform
  - powerful
  - abstract (can be formulated in other categories than sets)
- Initiality & finality can also be used in **binary** form; this is the next topic



Bisimilarity

- Formalisation of the intuitive idea: **indistinguishability of states**, via the available operations
- Several definitions possible, but they all coincide for "reasonable" functors
- Important relation with final coalgebras, and coinductive proof techniques
- Also relevant for **Hennessey-Milner** result in coalgebraic modal logic: states are bisimilar if and only if they satisfy the same formulas.



Bisimilarity (as behavioural equivalence)

Definition (cospan style)

Assume two coalgebras  $X \xrightarrow{c} F(X)$  and  $Y \xrightarrow{d} F(Y)$  of the same functor  $F$ .

Two states  $x \in X$  and  $y \in Y$  will be called **bisimilar**, written as  $x \leftrightarrow y$ , if there is a third coalgebra  $W \xrightarrow{e} F(W)$  with two homomorphisms:

$$(X \xrightarrow{c} F(X)) \xrightarrow{f} (W \xrightarrow{e} F(W)) \xleftarrow{g} (Y \xrightarrow{d} F(Y))$$

satisfying  $f(x) = g(y)$  in  $W$ .

The definition is often used where coalgebras  $c$  and  $d$  are the same. This gives bisimilarity for two states of the **same** coalgebra.



Bisimilarity alternative II

Lemma (Bisimilarity as greatest bisimulation)

Assume the functor  $F$  is "reasonable" (preserves weak pullbacks)  
States  $x \in X$  and  $y \in Y$  are bisimilar iff they are contained in a **bisimulation**  $R \subseteq X \times Y$

- such a relation  $R$  is **closed under the coalgebras**
- formally,  $\langle r_1, r_2 \rangle: R \hookrightarrow X \times Y$  carries a coalgebra itself in:

$$\begin{array}{ccccc} F(X) & \xleftarrow{F(r_1)} & F(R) & \xrightarrow{F(r_2)} & F(Y) \\ c \uparrow & & \uparrow & & \uparrow d \\ X & \xleftarrow{r_1} & R & \xrightarrow{r_2} & Y \end{array}$$



Bisimilarity alternative I

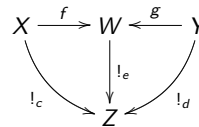
Lemma

Assume  $F$  as a **final coalgebra**  $Z \xrightarrow{\zeta} F(Z)$ .

Assume  $X \xrightarrow{c} F(X)$ ,  $Y \xrightarrow{d} F(Y)$ , and  $x \in X$  and  $y \in Y$  as before.

$$\begin{aligned} x \leftrightarrow y &\iff !_c(x) = !_d(y) \text{ in } Z \\ &\iff x, y \text{ are mapped to the same element} \\ &\quad \text{of the final coalgebra} \end{aligned}$$

**Proof:** ( $\Leftarrow$ ) obvious; ( $\Rightarrow$ ) Assume  $f(x) = g(y)$  in  $W \xrightarrow{e} F(W)$  in:



By finality:

$$!_e \circ f = !_c \quad !_e \circ g = !_d$$

Hence:

$$!_c(x) = !_e(f(x)) = !_e(g(y)) = !_d(y)$$



Final coalgebras

- Not all functors have final coalgebras (recall powerset  $\mathcal{P}$ )
- but under suitable **size conditions** they do exist (boundedness, accessibility)
- Given such restrictions, there are several constructions:
  - as limits of chains
  - via modal formulas
  - as quotients of known final coalgebras
- Here we do not care very much about how they are built, but focus on its finality and how to use it



## Final automaton example I

- Fix two sets:  $A$  for inputs,  $B$  for outputs
- We consider coalgebras, or **deterministic automata**:

$$X \xrightarrow{\langle \delta, \epsilon \rangle} X^A \times B$$

- For each state  $x \in X$ , it gives:
  - a **successor state**  $\delta(x)(a) \in X$ , for each input  $a \in A$
  - an **observation**  $\epsilon(x) \in B$
- For a finite sequence  $\sigma = \langle a_1, \dots, a_n \rangle \in A^*$  we put:

$$\delta^*(x)(\sigma) = \delta(\dots \delta(x)(a_1) \dots)(a_n)$$

This gives multiple-step transitions.



## Final automaton example III

For an arbitrary coalgebra  $X \xrightarrow{\langle \delta, \epsilon \rangle} X^A \times B$  we have:

$$\begin{array}{ccc} X^A \times B & \xrightarrow{\text{beh}^A \times \text{id}} & (B^{A^*})^A \times B \\ \langle \delta, \epsilon \rangle \uparrow & & \cong \uparrow \zeta = \langle \zeta_1, \zeta_2 \rangle \\ X & \xrightarrow{\text{beh}} & B^{A^*} \end{array}$$

where:  $\text{beh}(x)(\sigma) = \epsilon(\delta^*(x)(\sigma))$ .

This makes the diagram commute:

$$\begin{aligned} \zeta_2(\text{beh}(x)) &= \text{beh}(x)(\langle \rangle) \\ &= \epsilon(\delta^*(x)(\langle \rangle)) = \epsilon(x) \\ \zeta_2(\text{beh}(x))(a)(\sigma) &= \text{beh}(x)(a \cdot \sigma) \\ &= \epsilon(\delta^*(x)(a \cdot \sigma)) \\ &= \epsilon(\delta^*(\delta(x)(a))(\sigma)) \\ &= \text{beh}(\delta(x)(a))(\sigma) = \text{beh}^A(\delta(x))(a)(\sigma). \end{aligned}$$



## Final automaton example V

1 **Special case I**:  $A = 1$ , so functor is  $F(X) = X^1 \times B \cong X \times B$

- $A^* = 1^* \cong \mathbb{N}$
- final coalgebra consists of **infinite sequences / streams**  
 $B^{A^*} \cong B^{\mathbb{N}}$
- structure map is **tail** and **head**  $B^{\mathbb{N}} \xrightarrow{\cong} B^{\mathbb{N}} \times B$

2 **Special case I**:  $B = 2 = \{0, 1\}$ , so functor is  $F(X) = X^A \times 2$

- final coalgebra is  $2^{A^*} = \mathcal{P}(A^*)$
- this is the set of **languages** over alphabet  $A$
- behaviour map **beh**:  $X \rightarrow \mathcal{P}(A^*)$  maps a state to the **accepted language** in that state
- bisimilar states thus accept the same language



## Final automaton example II

### Theorem (Arbib & Manes, Reichel)

The final coalgebra of the automaton functor  $F(X) = X^A \times B$  is the set of functions  $B^{A^*}$  (from input sequences to observations) with:

$$B^{A^*} \xrightarrow[\cong]{\zeta = \langle \zeta_1, \zeta_2 \rangle} (B^{A^*})^A \times B$$

where for a function/state  $\varphi \in B^{A^*}$

$$\begin{cases} \zeta_1(\varphi)(a) = \lambda \sigma \in A^*. \varphi(a \cdot \sigma) \\ \zeta_2(\varphi) = \varphi(\langle \rangle) \end{cases}$$



## Final automaton example IV

### Automaton bisimulations

For a (single) automaton  $X \xrightarrow{\langle \delta, \epsilon \rangle} X^A \times B$  a **bisimulation** is a relation  $R \subseteq X \times X$  which is closed under the operations:

- $R(x, x') \implies \epsilon(x) = \epsilon(x')$
- $R(x, x') \implies R(\delta(x)(a), \delta(x')(a))$ , for all inputs  $a \in A$ .

Thus, states within  $R$  cannot be distinguished.



## Final language coalgebra (special case II)

Write  $\mathcal{L}(A) = \mathcal{P}(A^*) = 2^{A^*}$  for the final coalgebra of languages over alphabet  $A$

$$\mathcal{L}(A) \xrightarrow[\cong]{\langle \delta, \epsilon \rangle} \mathcal{L}(A)^A \times 2$$

where, for  $L \in \mathcal{L}(A)$ ,

$$\begin{aligned} \delta(L)(a) &= L_a, \quad \text{the Brzozowski derivative of } L \\ &= \{\sigma \in A^* \mid a \cdot \sigma \in L\} \end{aligned}$$

$$\epsilon(L) = 1 \iff \langle \rangle \in L$$

Two languages  $L, K \in \mathcal{L}(A)$  are **bisimilar** if there is a bisimulation  $R \subseteq \mathcal{L}(A) \times \mathcal{L}(A)$ ; this  $R$  satisfies

$$R(L, K) \implies \begin{cases} \langle \rangle \in L \text{ iff } \langle \rangle \in K \\ R(L_a, K_a), \text{ for each } a \in A \end{cases}$$





## Regular languages example I

Recall that the subset  $\mathcal{R}(A) \subseteq \mathcal{L}(A)$  of **regular languages** is the smallest one with:

- $0 = \emptyset \in \mathcal{R}(A)$
- $1 = \{\langle \rangle\} \in \mathcal{R}(A)$
- $\{a\} \in \mathcal{R}(A)$ , for each  $a \in A$  — often simply:  $a \in \mathcal{R}(A)$
- $L + K = L \cup K \in \mathcal{R}(A)$ , for  $L, K \in \mathcal{R}(A)$
- $LK = \{\sigma; \tau \mid \sigma \in L, \tau \in K\} \in \mathcal{R}(A)$ , for  $L, K \in \mathcal{R}(A)$
- $L^* = \bigcup_{n \in \mathbb{N}} L^n \in \mathcal{R}(A)$ , for  $L \in \mathcal{R}(A)$

**Examples:**  $a^*$ ,  $a^*b^*$ ,  $(ab)^*$ ,  $(1+a)^*$  etc.



## Equality of regular languages via coinduction

- Now that regular language  $\mathcal{R}(A)$  carry a coalgebra, we know what it means that languages  $L, K \in \mathcal{R}(A)$  are bisimilar
- namely: they are **equal, when mapped to the final coalgebra**
- But the finality map  $\mathcal{R}(A) \rightarrow \mathcal{L}(A)$  is inclusion, in:

$$\begin{array}{ccc} \mathcal{R}(A)^A \times 2 & \dashrightarrow & \mathcal{L}(A)^A \times 2 \\ \uparrow & & \uparrow \cong \\ \mathcal{R}(A) & \dashrightarrow & \mathcal{L}(A) = \mathcal{P}(A^*) \end{array}$$

- Thus, for regular languages  $L, K \in \mathcal{R}(A)$ ,  
 $L \Leftrightarrow R$  iff  $L = K$
- a **new proof method** for equality of regular languages
  - the traditional one is cumbersome: it involves either algebraic reasoning, or turning expressions into machines, and then minimalising them



## Equality of regular languages, example II

- Aim:  $(a+b)^* = (a^*b)^*a^*$ , over alphabet  $A = \{a, b\}$
- Take as relation  $R = \{(a+b)^*, (a^*b)^*a^*\}$
- Clearly, for  $(L, K) \in R$ ,  $\langle \rangle \in L \Leftrightarrow \text{true} \Leftrightarrow \langle \rangle \in K$
- $R$  is also closed under derivatives. Eg. for  $(-)_b$ :

$$\begin{aligned} ((a+b)^*)_b &= (a+b)_b(a+b)^* = (a_b + b_b)(a+b)^* \\ &= (0+1)(a+b)^* = (a+b)^* \end{aligned}$$

$$\begin{aligned} ((a^*b)^*a^*)_b &= ((a^*b)^*)_b a^* + (a^*)_b = (a^*b)_b (a^*b)^* a^* + a_b a^* \\ &= ((a^*)_b b + b_b) (a^*b)^* a^* + 0 a^* \\ &= (0+1)(a^*b)^* a^* + 0 = (a^*b)^* a^*. \end{aligned}$$

- Similarly for the derivative  $(-)_a$ . Hence  $R$  is a **bisimulation**, and the goal is proven.



## Regular languages form a coalgebra

Regular languages are **closed under the Brzozowski derivative**, via:

$$\begin{aligned} 0_a &= 0 & \langle \rangle &\notin 0 \\ 1_a &= 0 & \langle \rangle &\in 1 \\ b_a &= \begin{cases} 1 & \text{if } b = a \\ 0 & \text{otherwise} \end{cases} & \langle \rangle &\notin b \\ (K+L)_a &= K_a + L_a & \langle \rangle &\in K+L \text{ iff} \\ & & & \langle \rangle \in K \text{ or } \langle \rangle \in L \\ (KL)_a &= \begin{cases} K_a L + L_a & \text{if } \langle \rangle \in K \\ K_a L & \text{otherwise} \end{cases} & \langle \rangle &\in KL \text{ iff} \\ & & & \langle \rangle \in K \text{ and } \langle \rangle \in L \\ (K^*)_a &= K_a K^* & \langle \rangle &\in K^*. \end{aligned}$$

This yields a coalgebra / automaton

$$\mathcal{R}(A) \xrightarrow{(\delta, \epsilon)} \mathcal{R}(A)^A \times 2$$



## Equality of regular languages, example I

- Assume we wish to prove  $(1+a)^* = a^*$ , over alphabet  $A = \{a, b\}$
- We need  $((1+a)^*, a^*) \in R$  for a bisimulation  $R \subseteq \mathcal{R}(A) \times \mathcal{R}(A)$
- Take** the subset relation consisting of 2 pairs only:  
 $R = \{((1+a)^*, a^*)\} \cup \{(0, 0)\}$

- Assume  $(L, K) \in R$ ;
  - clearly,  $\langle \rangle \in L$  iff  $\langle \rangle \in K$
  - $R$  is also closed under derivatives

$$\begin{aligned} \text{Obviously, for } L = 0 = K, \text{ so take } L = (1+a)^*, K = a^*, \\ ((1+a)^*)_a &= (1+a)_a(1+a)^* = (1_a + a_a)(1+a)^* \\ &= (0+1)(1+a)^* = (1+a)^* \end{aligned}$$

$$(a^*)_a = a_a a^* = 1 a^* = a^*$$

$$((1+a)^*)_b = 0 = (a^*)_b \quad \checkmark$$



## Equality of regular languages, example III

- Non-trivial example:

$$\begin{aligned} a^* + a^*b(a+ba^*b)^*ba^* \\ = \{\sigma \in A^* \mid \sigma \text{ contains an even number of } b\text{'s}\} \end{aligned}$$

- The proof is non-trivial, but essentially straightforward via bisimulations
- This technique is now incorporated in a tool (CIRC, based on Maude)
- Many more examples of such equality proof via bisimulations in the work of **Jan Rutten**



## Prerequisites for quantum computation

- Do you need to be a quantum physicist to understand quantum computation?
- **NO!**  
*One can be a masterful practioner of computer science without having the foggiest notion of what a transistor is, not to mention how it works*  
(David Mermin, Quantum Computer Science. An Introduction.)
- Here, as usual, we only deal with the abstract, mathematical model for quantum computation, not with its possible future realisation — which is work for physicists.



## What is quantum computing about?

- Letting nature do matrix multiplications for us
- Computations are divided over multiple parallel worlds ("quantum parallelism")
- A new physical basis for computation
- Much focus so far on algorithms and complexity
- More recently also on semantics and quantum languages (Abramsky, Coeke, Panangaden, Selinger, ...)
- Actual, physical realisation of quantum computer still embryonic (in the order of 10 qubits)
- Quantum key distribution more developed (but also under attack, see *quantum hackers*)



## Quantum computation is powerful

- More efficient in certain tasks, via a new form of parallelism
  - Notably factorisation of numbers (Shor); threatening for RSA;
- New levels of security, e.g. with key exchange via entanglement



## What is needed?

### Main prerequisites for quantum computation

- Linear algebra (over complex numbers  $\mathbb{C}$ )
- Hilbert spaces (esp. finite dimensional ones)
- Tensors and spectral decomposition

### Prerequisites for this talk

- Basic linear algebra
- Basic category theory



## Quantum mechanics/computation is strange

### The four strangest phenomena

- Superposition: linear combinations of basic states
  - Entanglement of states: explained via tensors  $\otimes$
  - Computation is reversible: explained via unitary maps (via daggers  $(-)^{\dagger}$  ie. adjoints)
  - Measurement
    - side-effect: the state changes to the result of the measurement
    - entangled objects are **both** changed when **only one** is measured
- Mathematical explanation via diagonalisation of operators (using eigenvectors & eigenvalues)



## Attractions

- New area inbetween computer science, physics and logic
  - John Baez: category theory forms *Rosetta Stone*
- Great potential for both theory and applications
  - Chance to get theory in place before (programming) languages emerge
  - Early in CS: languages came before theory
- Issues are both familiar and new
  - See later discussion of 'walks'



## Dirac notation for vectors

- Following Dirac, physicists use the notation  $|x\rangle$  to denote an arbitrary vector. This is convenient with inner/outer products.
- Binary numbers inside  $|-\rangle$  are often used for base vectors.
- For example,
  - In  $\mathbb{C}^2$  one may read:

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

- And in  $\mathbb{C}^4$ ,

$$|00\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad |01\rangle = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \quad |10\rangle = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \quad |11\rangle = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

- Etc.



## Quantum bits and classical bits

### Quantum bit (or qubit)

A Qubit is a unit vector in  $\mathbb{C}^2$ , written as:

$$\alpha|0\rangle + \beta|1\rangle \quad \text{with } \alpha, \beta \in \mathbb{C} \text{ satisfying } |\alpha|^2 + |\beta|^2 = 1.$$

Thus, a qubit is a **superposition** of  $|0\rangle$  and  $|1\rangle$ .

The  $\alpha, \beta$  are called probability amplitudes.

### Classical bit (or cbit)

A cbit is either 0 or 1.

More formally, it is a unit vector over Booleans:

$$\alpha|0\rangle + \beta|1\rangle \quad \text{with } \alpha, \beta \in \mathbb{B} \text{ satisfying } \alpha \text{ XOR } \beta = 1.$$



## Measurement, informally

Given a qubit in state  $\alpha|0\rangle + \beta|1\rangle$  one can **measure** it, and:

- obtain either  $|0\rangle$  or  $|1\rangle$  as successor state (side effect)
  - the probability of getting  $|0\rangle$  is  $|\alpha|^2 \in [0, 1]$
  - the probability of getting  $|1\rangle$  is  $|\beta|^2 \in [0, 1]$
- (Recall:  $|\alpha|^2 + |\beta|^2 = 1$ .)

**Note** The state  $(\alpha, \beta)$  itself remains hidden

Measurement can be done with respect to any basis.



## Two products

Given vector spaces  $V, W$  we need:

- The **biproduct**  $V \times W$ , with elements  $(v, w)$
- The **tensor product**  $V \otimes W$ , with elements  $v \otimes w$

Tensor distributes over biproducts:

$$W \otimes (V \times U) \cong (W \otimes V) \times (W \otimes U).$$

In the finite dimensional case:

$$\dim(V \times W) = \dim(V) + \dim(W) \quad \dim(V \otimes W) = \dim(V) \cdot \dim(W)$$



## Biproducts

The biproduct  $V \times W$  of vector/Hilbert spaces is both a categorical **product** and **coproduct** (sum), with projections:

$$\begin{array}{ccccc} V & \xleftarrow{\pi_1} & V \times W & \xrightarrow{\pi_2} & W \\ v & \longleftarrow & (v, w) & \longrightarrow & w \end{array}$$

and coprojections (injections):

$$\begin{array}{ccccc} V & \xrightarrow{\kappa_2} & V \times W & \xleftarrow{\kappa_1} & W \\ v & \longrightarrow & (v, 0) & & \\ & & (0, w) & \longleftarrow & w \end{array}$$

The singleton space  $\{0\}$  is unit element for  $\times$ , ie.  $\{0\} \times V \cong V$ .



## Tensor products

The tensor product  $V \otimes W$  contains linear combinations of pairs  $v \otimes w$ . The mapping  $\otimes: V \times W \rightarrow V \otimes W$  is "bilinear" (and universal).

If  $\begin{cases} V \text{ has basis } |a_1\rangle, \dots, |a_n\rangle \\ W \text{ has basis } |b_1\rangle, \dots, |b_m\rangle, \end{cases}$

then  $V \otimes W$  has a basis given by:

$$|a_i b_j\rangle = |a_i\rangle \otimes |b_j\rangle \quad \text{for } i \leq n, j \leq m.$$

The scalars  $\mathbb{C}$  are unit element for  $\otimes$ , ie.  $\mathbb{C} \otimes V \cong V$ .

$\otimes$  is used for **parallel composition**, possibly with entanglement.





## Tensor example

- A **single qubit** lives in  $\mathbb{C}^2 = \mathbb{C} \times \mathbb{C}$ , with basis  $|0\rangle, |1\rangle$
- Two qubits in parallel** live in  $\mathbb{C}^2 \otimes \mathbb{C}^2$ , for which:

$$\begin{aligned} \mathbb{C}^2 \otimes \mathbb{C}^2 &= (\mathbb{C} \times \mathbb{C}) \otimes (\mathbb{C} \times \mathbb{C}) \\ &\cong (\mathbb{C} \otimes \mathbb{C}) \times (\mathbb{C} \otimes \mathbb{C}) \times (\mathbb{C} \otimes \mathbb{C}) \times (\mathbb{C} \otimes \mathbb{C}) \\ &\cong \mathbb{C} \times \mathbb{C} \times \mathbb{C} \times \mathbb{C} \\ &= \mathbb{C}^4 \end{aligned}$$

with basis  $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ , where  $|00\rangle = |0\rangle \otimes |0\rangle$  etc.

- $n$  qubits** live in  $\mathbb{C}^2 \otimes \dots \otimes \mathbb{C}^2 \cong \mathbb{C}^{2^n}$ , with base vectors  $|b_{n-1} \dots b_0\rangle$  for  $b_i \in \{0, 1\}$ .



## Two basic results

### LEMMA (separability/disentanglement condition)

An arbitrary element

$$\gamma_{00}|00\rangle + \gamma_{01}|01\rangle + \gamma_{10}|10\rangle + \gamma_{11}|11\rangle \in \mathbb{C}^2 \otimes \mathbb{C}^2$$

with  $|\gamma_{00}|^2 + |\gamma_{01}|^2 + |\gamma_{10}|^2 + |\gamma_{11}|^2 = 1$  is of non-entangled form:

$$(\alpha_0|0\rangle + \alpha_1|1\rangle) \otimes (\beta_0|0\rangle + \beta_1|1\rangle)$$

if and only if

$$\gamma_{00}\gamma_{11} = \gamma_{01}\gamma_{10}.$$

### LEMMA Classical bits can always be separated:

(Since classically, only one of the  $\gamma_{ij}$  is 1; the others are 0)



## Coalgebras of monads

- In this context we use coalgebras  $X \rightarrow T(X)$  where  $T$  is a **monad**: a special functor with *unit* and *multiplication*
- As a result, coalgebras can be composed
  - More formally: the set of coalgebras  $(TX)^X$  forms a monoid
  - Even more formally: a monoid in the category of  $T$ -algebras
- Main monad examples:
  - Powerset  $\mathcal{P}$
  - Distribution  $\mathcal{D}$
  - Multiset  $\mathcal{M}$



## Putting qubits together

Two qubits  $\alpha_0|0\rangle + \alpha_1|1\rangle$  and  $\beta_0|0\rangle + \beta_1|1\rangle$  can be put in parallel, in  $\mathbb{C}^2 \otimes \mathbb{C}^2$ , as:

$$\begin{aligned} &(\alpha_0|0\rangle + \alpha_1|1\rangle) \otimes (\beta_0|0\rangle + \beta_1|1\rangle) \\ &= \alpha_0\beta_0|00\rangle + \alpha_0\beta_1|01\rangle + \alpha_1\beta_0|10\rangle + \alpha_1\beta_1|11\rangle. \end{aligned}$$

However, an **arbitrary** element in  $\mathbb{C}^2 \otimes \mathbb{C}^2$  is of the form:

$$\gamma_{00}|00\rangle + \gamma_{01}|01\rangle + \gamma_{10}|10\rangle + \gamma_{11}|11\rangle$$

It captures two **entangled qubits**, since in general such separate  $\alpha$ 's and  $\beta$ 's cannot be recovered from  $\gamma$ 's.



## Recap on coalgebras

- Mathematical models for **state-based** computation:
  - state space, say  $X$
  - transition map  $X \rightarrow F(X)$
  - $F$  is a functor that fixes the **type of computation**
- A **deterministic automaton** involves a pair of maps:

$$X \xrightarrow{\text{(step, final?)}} X^A \times 2$$

where  $2 = \{0, 1\}$ .

- A **non-deterministic automaton**, or **transition system**, with input actions  $A$  is a coalgebra:

$$X \xrightarrow{\text{succs}} \mathcal{P}(X)^A$$

- By replacing powerset  $\mathcal{P}$  by distribution functor  $\mathcal{D}$  one obtains **probabilistic** automata (aka. Markov chain)



## Distribution monad $\mathcal{D}$

For a set  $X$ , define

$$\mathcal{D}(X) = \{\varphi: X \rightarrow [0, 1] \mid \text{support}(\varphi) \text{ is finite, and } \sum_x \varphi(x) = 1\}$$

Such  $\varphi \in \mathcal{D}(X)$  is a formal convex combination:

$$r_1x_1 + \dots + r_nx_n \quad \text{where} \quad \begin{cases} \text{support}(\varphi) = \{x_1, \dots, x_n\} \\ r_i = \varphi(x_i) > 0 \\ r_1 + \dots + r_n = 1 \end{cases}$$

Coalgebras  $X \rightarrow \mathcal{D}(X)$  are **Markov chains**, giving probabilistic transitions:

$$x \xrightarrow{r_i} x_i \quad \text{with} \quad \sum_i r_i = 1.$$



Multiset monad  $\mathcal{M}$

The multiset monad  $\mathcal{M}$  over complex numbers is similar to, but simpler than, the distribution monad  $\mathcal{D}$ . For a set  $X$ , now define

$$\mathcal{M}(X) = \{ \varphi: X \rightarrow \mathbb{C} \mid \text{support}(\varphi) \text{ is finite} \}$$

Such  $\varphi \in \mathcal{M}(X)$  is a formal linear combination:

$$z_1x_1 + \dots + z_nx_n \quad \text{where} \quad \begin{cases} \text{support}(\varphi) = \{x_1, \dots, x_n\} \\ z_i = \varphi(x_i) \in \mathbb{C} \end{cases}$$

Such formal linear combinations form the **free vector space** on  $X$ .

Coalgebras  $X \rightarrow \mathcal{M}(X)$  are like **weighted automata**, adding weights/resources in  $\mathbb{C}$  as labels to transitions.



Starting to walk (for the classically educated)



Tales from the Ministry of Coalgebraic Walks



Walk the walk

Consider a **line of integer points**  $\dots, -2, -1, 0, 1, 2, \dots \in \mathbb{Z}$ . Different styles of walks, say of a drunkard, on this line will be described next:

- non-deterministic
- probabilistic
- quantum

All three computational styles can & will be represented coalgebraically.

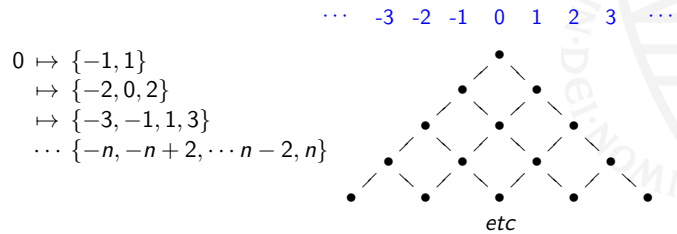


Non-deterministic walks: definition

Coalgebraic representation, of possible left-or-right stepping:

$$\begin{aligned} \mathbb{Z} &\xrightarrow{s} \mathcal{P}(\mathbb{Z}) \\ k &\longmapsto \{k-1, k+1\} \end{aligned}$$

Iteration, starting in  $0 \in \mathbb{Z}$ , yields:



Non-deterministic walks: iteration

Formally, iteration is done via the **Kleisli extension** endomap:

$$\begin{aligned} \mathcal{P}(\mathbb{Z}) &\xrightarrow{s^\#} \mathcal{P}(\mathbb{Z}) \\ U &\longmapsto \bigcup_{k \in U} \{k-1, k+1\} \end{aligned}$$

The subset of successors of  $0 \in \mathbb{Z}$ , after  $n$  steps, is obtained as the  $n$ -th iterate:

$$(s^\#)^n(\{0\}) = \{-n, -n+2, \dots, n-2, n\}.$$

**Aside:** categorically, this can be described directly as iteration in the Kleisli category of the powerset monad  $\mathcal{P}$ .

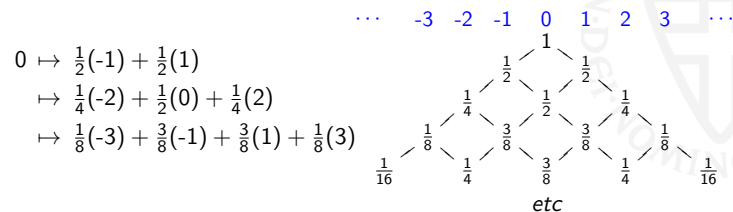


Probabilistic walks: definition

Probabilistic left-or-right stepping, each with chance  $\frac{1}{2}$  is expressed via a formal convex sum / distribution, as:

$$\begin{aligned} \mathbb{Z} &\xrightarrow{d} \mathcal{D}(\mathbb{Z}) \\ k &\longmapsto \frac{1}{2}(k-1) + \frac{1}{2}(k+1) \end{aligned}$$

Iteration, starting in  $0 \in \mathbb{Z}$ , now yields:





Probabilistic walks: the general formula

This tree of probabilities involves **Pascal's triangle**.

Starting in  $k \in \mathbb{Z}$ , after  $n$  iterations one obtains the formal convex sum:

$$\frac{\binom{n}{0}}{2^n}(k-n) + \frac{\binom{n}{1}}{2^n}(k-n+2) + \frac{\binom{n}{2}}{2^n}(k-n+4) + \dots + \frac{\binom{n}{n-1}}{2^n}(k+n-2) + \frac{\binom{n}{n}}{2^n}(k+n)$$

Using the sum formula for binomial coefficients:

$$\binom{n}{0} + \binom{n}{1} + \binom{n}{2} + \dots + \binom{n}{n-1} + \binom{n}{n} = 2^n.$$

Quantum walks

Situation / plan

- Quantum walks are standardly described as endomap  $S \rightarrow S$  (see eg. work of Julia Kempe)
- Here it will be shown that there is also an (equivalent) coalgebraic / monadic description.

Quantum walks: endomap definition II

The relevant endomap is:

$$\begin{aligned} \mathbb{C}^2 \otimes \mathcal{M}(\mathbb{Z}) &\xrightarrow{q} \mathbb{C}^2 \otimes \mathcal{M}(\mathbb{Z}) \\ |\uparrow\rangle \otimes |k\rangle &\longmapsto \frac{1}{\sqrt{2}}|\uparrow\rangle \otimes |k-1\rangle + \frac{1}{\sqrt{2}}|\downarrow\rangle \otimes |k+1\rangle \\ |\downarrow\rangle \otimes |k\rangle &\longmapsto \frac{1}{\sqrt{2}}|\uparrow\rangle \otimes |k-1\rangle - \frac{1}{\sqrt{2}}|\downarrow\rangle \otimes |k+1\rangle \end{aligned}$$

Implicitly, the **Hadamard** operator  $H: \mathbb{C}^2 \rightarrow \mathbb{C}^2$  is used:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$



Probabilistic walks: iteration

Again there is a **Kleisli extension** endomap  $d^\#: \mathcal{D}(\mathbb{Z}) \rightarrow \mathcal{D}(\mathbb{Z})$

$$\begin{aligned} (r_1 k_1 + \dots + r_n k_n) \\ \mapsto \left( \frac{1}{2} r_1 (k_1 - 1) + \frac{1}{2} r_1 (k_1 + 1) + \dots + \frac{1}{2} r_n (k_n - 1) + \frac{1}{2} r_n (k_n + 1) \right) \end{aligned}$$

The subset of successors of  $0 \in \mathbb{Z}$ , after  $n$  steps, is obtained as the  $n$ -th iterate:

$$(d^\#)^n(10) = \frac{\binom{n}{0}}{2^n}(-n) + \frac{\binom{n}{1}}{2^n}(-n+2) + \dots + \frac{\binom{n}{n-1}}{2^n}(n-2) + \frac{\binom{n}{n}}{2^n}(n)$$

It is iteration in the Kleisli category of the distribution monad  $\mathcal{D}$ .

Quantum walks: endomap definition I

Two vector spaces

- $\mathcal{M}(\mathbb{Z})$ , the free vector space on  $\mathbb{Z}$  (over  $\mathbb{C}$ ), with base vectors written as:  $|k\rangle \in \mathcal{M}(\mathbb{Z})$ , for  $k \in \mathbb{Z}$
- $\mathbb{C}^2$ , the one qubit space, with base vectors  $|\uparrow\rangle$  and  $|\downarrow\rangle$  (think of the direction of the walk)

The state space is  $\mathbb{C}^2 \otimes \mathcal{M}(\mathbb{Z})$ , with basis  $\left\{ \begin{aligned} &|\uparrow\rangle \otimes |k\rangle \\ &|\downarrow\rangle \otimes |k\rangle \end{aligned} \right\}$

Quantum walks: iteration and probabilities I

Obtaining probabilities in quantum walks

- We are interested in the probability of reaching  $|k\rangle$  after  $n$  steps. Starting from  $0$  — more precisely, from  $|\uparrow\rangle \otimes |0\rangle$ .
- We do so by  $n$ -times iterating the endomap  $q$ , and then measuring in the basis  $|k\rangle$ .
- The sum of norms  $|\alpha_i|^2$  of amplitudes  $\alpha_i$  for  $|k\rangle$  then gives the probability.



Quantum walks: iteration and probabilities II

$$|\uparrow\rangle \otimes |0\rangle \mapsto \frac{1}{\sqrt{2}}|\uparrow\rangle \otimes |-1\rangle + \frac{1}{\sqrt{2}}|\downarrow\rangle \otimes |1\rangle$$

$$\text{probabilities} \begin{cases} |-1\rangle & |\frac{1}{\sqrt{2}}|^2 = \frac{1}{2} \\ |1\rangle & |\frac{1}{\sqrt{2}}|^2 = \frac{1}{2} \end{cases}$$

$$\mapsto \frac{1}{2}|\uparrow\rangle \otimes |-2\rangle + \frac{1}{2}|\downarrow\rangle \otimes |0\rangle + \frac{1}{2}|\uparrow\rangle \otimes |0\rangle - \frac{1}{2}|\downarrow\rangle \otimes |2\rangle$$

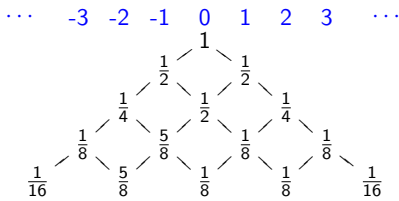
$$\text{probabilities} \begin{cases} |-2\rangle & |\frac{1}{2}|^2 = \frac{1}{4} \\ |0\rangle & |\frac{1}{2}|^2 + |\frac{1}{2}|^2 = \frac{1}{2} \\ |2\rangle & |-\frac{1}{2}|^2 = \frac{1}{4} \end{cases}$$

$\mapsto \dots$  (next page)



Quantum walks: iteration and probabilities III

The resulting tree of quantum walk probabilities starts as:



The matrix involved — Hadamard's  $H$  in this case — determines the drifting, and thus how the tree is traversed. This may yield optimisations in data processing.



Quantum endomorphism as coalgebra

**THEOREM** Linear maps (in  $\mathbf{Vect}_{\mathbb{C}}$ )

$$\mathbb{C}^2 \otimes \mathcal{M}(\mathbb{Z}) \longrightarrow \mathbb{C}^2 \otimes \mathcal{M}(\mathbb{Z})$$

correspond bijectively to functions (in  $\mathbf{Sets}$ )

$$\mathbb{Z} \longrightarrow \mathcal{M}(\mathbb{Z} + \mathbb{Z})^2$$

that is, to coalgebras with state space  $\mathbb{Z}$  of the functor  $\mathcal{M}(2 \cdot -)^2$ .

**Proof**

$$\mathbb{C}^2 \otimes \mathcal{M}(\mathbb{Z}) \longrightarrow \mathbb{C}^2 \otimes \mathcal{M}(\mathbb{Z}) \quad \text{linear}$$

$$\mathcal{M}(\mathbb{Z} + \mathbb{Z}) \longrightarrow \mathcal{M}(\mathbb{Z} + \mathbb{Z}) \quad \text{linear}$$

$$\mathbb{Z} + \mathbb{Z} \longrightarrow \mathcal{M}(\mathbb{Z} + \mathbb{Z})$$

$$\mathbb{Z} \longrightarrow \mathcal{M}(\mathbb{Z} + \mathbb{Z})^2$$



Quantum endomorphism as coalgebra



Quantum walks: iteration and probabilities II

$$\begin{aligned} \dots &\mapsto \frac{1}{2\sqrt{2}}|\uparrow\rangle \otimes |-3\rangle + \frac{1}{2\sqrt{2}}|\downarrow\rangle \otimes |-1\rangle + \frac{1}{2\sqrt{2}}|\uparrow\rangle \otimes |-1\rangle \\ &\quad - \frac{1}{2\sqrt{2}}|\downarrow\rangle \otimes |1\rangle + \frac{1}{2\sqrt{2}}|\uparrow\rangle \otimes |-1\rangle + \frac{1}{2\sqrt{2}}|\downarrow\rangle \otimes |1\rangle \\ &\quad - \frac{1}{2\sqrt{2}}|\uparrow\rangle \otimes |1\rangle + \frac{1}{2\sqrt{2}}|\downarrow\rangle \otimes |3\rangle \\ &= \frac{1}{2\sqrt{2}}|\uparrow\rangle \otimes |-3\rangle + \frac{1}{\sqrt{2}}|\uparrow\rangle \otimes |-1\rangle + \frac{1}{2\sqrt{2}}|\downarrow\rangle \otimes |-1\rangle \\ &\quad - \frac{1}{2\sqrt{2}}|\uparrow\rangle \otimes |1\rangle + \frac{1}{2\sqrt{2}}|\downarrow\rangle \otimes |3\rangle \end{aligned}$$

$$\text{probabilities} \begin{cases} |-3\rangle & |\frac{1}{2\sqrt{2}}|^2 = \frac{1}{8} \\ |-1\rangle & |\frac{1}{\sqrt{2}}|^2 + |\frac{1}{2\sqrt{2}}|^2 = \frac{5}{8} \\ |1\rangle & |-\frac{1}{2\sqrt{2}}|^2 = \frac{1}{8} \\ |3\rangle & |\frac{1}{2\sqrt{2}}|^2 = \frac{1}{8} \end{cases}$$

There is "drift to the left" due to **interference**.



Basic isomorphism

**LEMMA**  $\mathbb{C}^2 \otimes \mathcal{M}(X) \cong \mathcal{M}(X + X)$  where  $+$  is disjoint union

**Proof** By the following chain of isomorphisms:

$$\begin{aligned} \mathbb{C}^2 \otimes \mathcal{M}(X) &= (\mathbb{C} \oplus \mathbb{C}) \otimes \mathcal{M}(X) \\ &\cong \mathbb{C} \otimes \mathcal{M}(X) \oplus \mathbb{C} \otimes \mathcal{M}(X) \\ &\quad \text{by distributivity} \\ &\cong \mathcal{M}(X) \oplus \mathcal{M}(X) \\ &\quad \text{since } \mathbb{C} \text{ is tensor unit} \\ &\cong \mathcal{M}(X + X) \end{aligned}$$

$\oplus$  is also coproduct of spaces, and  $\mathcal{M}$  is a free functor.



Qubit action is monadic!

In the background, there is a new **monad transformer** result.

**LEMMA** If  $T$  is a monad, then so is  $X \mapsto T(n \cdot X)^n$ , for each  $n \in \mathbb{N}$

For quantum walks we use  $T = \mathcal{M}$  and  $n = 2$ .



Qubit action is monadic!





## Three categories for reversible computation

## Bifinite relations

- Reversible computation will be described in three categories:

**BifRel** sets with **bifinite relations**,  
for non-deterministic computation

**dBisRel** sets with discrete **bistochastic relations**,  
for probabilistic computation

**BifMRel** sets with **bifinite multirelations**,  
for quantum computation

- Each of the three walks will be described there
- Only the quantum walks turn out to be reversible

- The category **BifRel** is defined as follows

- object are sets  $X$ , and
- maps  $X \rightarrow Y$  are relations  $r: X \times Y \rightarrow 2 = \{0, 1\}$  such that:
  - for each  $x \in X$  the set  $\{y \in Y \mid r(x, y) \neq 0\}$  is finite;
  - also, for each  $y \in Y$  the set  $\{x \in X \mid r(x, y) \neq 0\}$  is finite.

- Thus: such  $r$  factors both as:

$$X \longrightarrow \mathcal{P}_{fin}(Y) \quad \text{and as} \quad Y \longrightarrow \mathcal{P}_{fin}(X)$$

- For a map  $r: X \rightarrow Y$  there is  $r^\dagger: Y \rightarrow X$  in the reverse direction, via argument swapping:  $r^\dagger(y, x) = r(x, y)$ .
- This makes **BifRel** a **dagger category**.



## Non-deterministic walks in BifRel

- The non-deterministic walks  $\mathbb{Z} \rightarrow \mathcal{P}_{fin}(\mathbb{Z})$  form an endomap:

$$\mathbb{Z} \xrightarrow{s} \mathbb{Z} \quad \text{in BifRel}$$

- Namely,  $s(n, m) = 1$  iff  $m = n - 1$  or  $m = n + 1$
- Its reverse is:  $s^\dagger(m, n) = 1 \iff s(n, m) = 1$ 
  - $\iff m = n - 1$  or  $m = n + 1$
  - $\iff n = m - 1$  or  $n = m + 1$
  - $\iff s(n, m) = 1$

- But  $s$  is **not unitary**: ie.  $s^\dagger \neq s^{-1}$

- For instance:

$$(s \circ s^\dagger)(n, n') = 1 \iff n' = n - 2 \text{ or } n' = n \text{ or } n' = n + 2$$

(The identity  $\mathbb{Z} \rightarrow \mathbb{Z}$  in **BifRel** is  $\text{id}(n, n') = 1$  iff  $n = n'$ )

## Discrete bistochastic relations

- The category **dBisRel** is defined as follows

- object are sets  $X$ , and
- maps  $X \rightarrow Y$  are functions  $r: X \times Y \rightarrow [0, 1]$  such that:
  - for each  $x$  there are finitely many  $y$  with  $r(x, y) \neq 0$  and  $\sum_y r(x, y) = 1$
  - also, for each  $y \in Y$  the set  $\{x \in X \mid r(x, y) \neq 0\}$  is finite and  $\sum_x r(x, y) = 1$

- we get discrete probability distributions in two directions:

$$X \longrightarrow \mathcal{D}(Y) \quad \text{and as} \quad Y \longrightarrow \mathcal{D}(X)$$

- For a map  $r: X \rightarrow Y$  in **dBisRel** there is again  $r^\dagger: Y \rightarrow X$  via argument swapping:  $r^\dagger(y, x) = r(x, y)$ .
- This makes also **dBisRel** a **dagger category**.



## Probabilistic walks in dBisRel

- The probabilistic walks  $\mathbb{Z} \rightarrow \mathcal{D}(\mathbb{Z})$  form an endomap:

$$\mathbb{Z} \xrightarrow{d} \mathbb{Z} \quad \text{in dBisRel}$$

- Namely,  $d(n, m) = \begin{cases} \frac{1}{2} & \text{if } m = n - 1 \text{ or } m = n + 1 \\ 0 & \text{otherwise.} \end{cases}$

- Its reverse is:  $d^\dagger(m, n) = d(n, m)$

- Also  $d$  is **not unitary**: ie.  $s^\dagger \neq s^{-1}$

- For instance:

$$(d \circ d^\dagger)(n, n') = \begin{cases} \frac{1}{4} & \text{if } n' = n - 2 \text{ or } n' = n + 2 \\ \frac{1}{2} & \text{if } n' = n \\ 0 & \text{otherwise.} \end{cases}$$

(The identity  $\mathbb{Z} \rightarrow \mathbb{Z}$  in **dBisRel** is  $\text{id}(n, n') = 1$  iff  $n = n'$ )

## Bifinite multirelations

- The category **BifMRel** is defined as follows

- object are sets  $X$ , and
- maps  $X \rightarrow Y$  are relations  $r: X \times Y \rightarrow \mathbb{C}$  such that:
  - for each  $x \in X$  the set  $\{y \in Y \mid r(x, y) \neq 0\}$  is finite;
  - also, for each  $y \in Y$  the set  $\{x \in X \mid r(x, y) \neq 0\}$  is finite.

- Thus: such  $r$  factors both as:

$$X \longrightarrow \mathcal{M}(Y) \quad \text{and as} \quad Y \longrightarrow \mathcal{M}(X)$$

- For a map  $r: X \rightarrow Y$  there is  $r^\dagger: Y \rightarrow X$  again via:  $r^\dagger(y, x) = r(x, y)$ .
- This makes **BifMRel** a **dagger category**.





Probabilistic walks in dBisRel

- The quantum walks  $\mathbb{Z} \rightarrow \mathcal{M}(\mathbb{Z} + \mathbb{Z})^2$  form an endomap:

$$\mathbb{Z} + \mathbb{Z} \xrightarrow{q} \mathbb{Z} + \mathbb{Z} \quad \text{in BifMRel}$$

- Namely,

$$\begin{cases} q(\kappa_1 n, \kappa_1(n-1)) = \frac{1}{\sqrt{2}} \\ q(\kappa_1 n, \kappa_2(n+1)) = \frac{1}{\sqrt{2}} \\ q(\kappa_2 n, \kappa_1(n-1)) = \frac{1}{\sqrt{2}} \\ q(\kappa_2 n, \kappa_2(n+1)) = -\frac{1}{\sqrt{2}} \end{cases}$$

Recall Hadamard

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

- Its reverse is:  $q^\dagger(u, v) = q(v, u)$
- This time:  $q$  is **unitary**: ie.  $q^\dagger \neq q^{-1}$



Future work

Better understand the three categories:

$$\text{BifRel} \longrightarrow \text{dBisRel} \longrightarrow \text{BifMRel}$$

and see what structure (categorical and logical) is typical for which kind of computing.



One of the four verifications

$$\begin{aligned} (q \circ q^\dagger)(\kappa_1 m, \kappa_1 n) &= \sum_x q^\dagger(\kappa_1 m, x) \cdot q(x, \kappa_1 n) \\ &= q^\dagger(\kappa_1 m, \kappa_1(m+1)) \cdot q(\kappa_1(m+1), \kappa_1 n) \\ &\quad + q^\dagger(\kappa_1 m, \kappa_2(m+1)) \cdot q(\kappa_2(m+1), \kappa_1 n) \\ &= \begin{cases} \frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{2}} & \text{if } n = m \\ 0 & \text{otherwise} \end{cases} \\ &= \begin{cases} 1 & \text{if } \kappa_1 n = \kappa_1 m \\ 0 & \text{otherwise} \end{cases} \\ &= \text{id}(\kappa_1 m, \kappa_1 n). \end{aligned}$$



Thanks for your attention; hopefully you feel inspired

