

Quantum Programs as Kleisli Maps

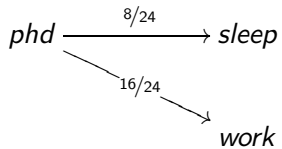
Abraham Westerbaan

Radboud University Nijmegen

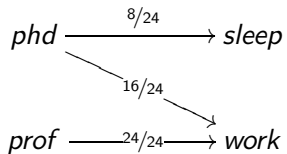
2016-06-09

<https://bram.westerbaan.name/kleisli.pdf>

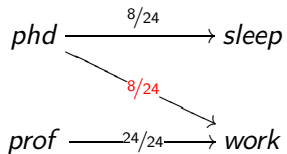
Kleisli Maps



Kleisli Maps

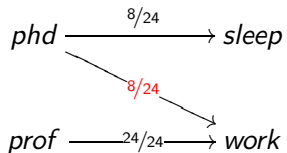


Kleisli Maps



Kleisli Maps

Substochastic map:



Kleisli Maps

Substochastic map:

$$phd \longrightarrow \frac{8}{24} |sleep\rangle + \frac{8}{24} |work\rangle$$

$$prof \longrightarrow |work\rangle$$

Kleisli Maps

$$\frac{X \xrightarrow{\text{substochastic}} Y}{X \xrightarrow{\text{("sharp") map}} \mathcal{S}(Y)}$$

Here $\mathcal{S}(Y)$ is the set of subdistributions on Y .

Kleisli Maps

$$\frac{X \xrightarrow{\text{substochastic}} Y}{X \xrightarrow{\text{("sharp") map}} \mathcal{S}(Y)}$$

Here $\mathcal{S}(Y)$ is the set of subdistributions on Y .

In fact, the category of sets and substochastic maps is isomorphic to the Kleisli category of the subdistribution monad \mathcal{S} .

Kleisli Maps

$$\begin{array}{c} X \xrightarrow{\text{substochastic}} Y \\ \hline X \xrightarrow{\text{("sharp") map}} \mathcal{S}(Y) \end{array}$$

Here $\mathcal{S}(Y)$ is the set of subdistributions on Y .

In fact, the category of sets and substochastic maps is isomorphic to the Kleisli category of the subdistribution monad \mathcal{S} .

So substochastic maps are “Kleisli maps”.

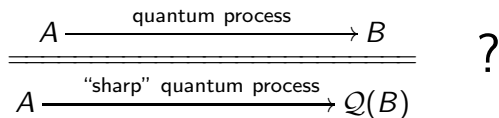
What about quantum processes?

What about quantum processes?

$$\begin{array}{c} A \xrightarrow{\text{quantum process}} B \\ \hline A \xrightarrow{\text{"sharp" quantum process}} Q(B) \end{array}$$

?

What about quantum processes?



It depends ...

What about quantum processes?

$$\begin{array}{c} A \xrightarrow{\text{quantum process}} B \\ \hline A \xrightarrow{\text{"sharp" quantum process}} Q(B) \end{array} \quad ?$$

It depends (on the exact definitions),

What about quantum processes?

$$\frac{A \xrightarrow{\text{quantum process}} B}{A \xrightarrow{\text{"sharp" quantum process}} Q(B)} \quad ?$$

It depends (on the exact definitions), but I would say:

NO! when A , B , and $Q(B)$ are to have **finite** dimension,

What about quantum processes?

$$\begin{array}{c} A \xrightarrow{\text{quantum process}} B \\ \hline A \xrightarrow{\text{"sharp" quantum process}} Q(B) \end{array} \quad ?$$

It depends (on the exact definitions), but I would say:

NO! when A , B , and $Q(B)$ are to have **finite** dimension,

YES! when A , B , and $Q(B)$ may have **infinite** dimension.

“Exact” definitions

“Exact” definitions

1. A **finite-dim. quantum datatype** is a ‘ring’ of matrices:

$$\mathcal{M}_2 \oplus \mathcal{M}_3 \oplus \mathbb{C}^2 \quad .$$

“Exact” definitions

1. A **finite-dim. quantum datatype** is a ‘ring’ of matrices:

$$\mathcal{M}_2 \oplus \mathcal{M}_3 \oplus \mathbb{C}^2 \subseteq \mathcal{M}_7 \quad .$$

“Exact” definitions

1. A **finite-dim. quantum datatype** is a ‘ring’ of matrices:

$$\mathcal{M}_2 \oplus \mathcal{M}_3 \oplus \mathbb{C}^2 \subseteq \mathcal{M}_7 \equiv \mathcal{B}(\mathbb{C}^7).$$

“Exact” definitions

1. A **finite-dim. quantum datatype** is a ‘ring’ of matrices:

$$\mathcal{M}_2 \oplus \mathcal{M}_3 \oplus \mathbb{C}^2 \subseteq \mathcal{M}_7 \equiv \mathcal{B}(\mathbb{C}^7).$$

2. A **quantum datatype** is a **von Neumann algebra**, that is, a ‘ring’ of bounded operators on some Hilbert space \mathcal{H} ,

$$\mathcal{A} \subseteq \mathcal{B}(\mathcal{H}),$$

“Exact” definitions

1. A **finite-dim. quantum datatype** is a ‘ring’ of matrices:

$$\mathcal{M}_2 \oplus \mathcal{M}_3 \oplus \mathbb{C}^2 \subseteq \mathcal{M}_7 \equiv \mathcal{B}(\mathbb{C}^7).$$

2. A **quantum datatype** is a **von Neumann algebra**, that is, a ‘ring’ of bounded operators on some Hilbert space \mathcal{H} ,

$$\mathcal{A} \subseteq \mathcal{B}(\mathcal{H}),$$

which is closed in a suitable topology on $\mathcal{B}(\mathcal{H})$.

“Exact” definitions

1. A **finite-dim. quantum datatype** is a ‘ring’ of matrices:

$$\mathcal{M}_2 \oplus \mathcal{M}_3 \oplus \mathbb{C}^2 \subseteq \mathcal{M}_7 \equiv \mathcal{B}(\mathbb{C}^7).$$

2. A **quantum datatype** is a **von Neumann algebra**, that is, a ‘ring’ of bounded operators on some Hilbert space \mathcal{H} ,

$$\mathcal{A} \subseteq \mathcal{B}(\mathcal{H}),$$

which is closed in a suitable topology on $\mathcal{B}(\mathcal{H})$.

3. $\mathbf{vN}_{\text{CPsU}}$ is the category of von Neumann algebras and normal, completely **p**ositive, **s**ub**u**nital linear maps.

“Exact” definitions

1. A **finite-dim. quantum datatype** is a ‘ring’ of matrices:

$$\mathcal{M}_2 \oplus \mathcal{M}_3 \oplus \mathbb{C}^2 \subseteq \mathcal{M}_7 \equiv \mathcal{B}(\mathbb{C}^7).$$

2. A **quantum datatype** is a **von Neumann algebra**, that is, a ‘ring’ of bounded operators on some Hilbert space \mathcal{H} ,

$$\mathcal{A} \subseteq \mathcal{B}(\mathcal{H}),$$

which is closed in a suitable topology on $\mathcal{B}(\mathcal{H})$.

3. $\mathbf{vN}_{\text{CPsU}}$ is the category of von Neumann algebras and normal, completely **positive**, **subunital** linear maps.
($\mathcal{A} \xrightarrow{\text{CPsU}} \mathcal{B}$ is a **quantum process** from \mathcal{B} to \mathcal{A} .)

“Exact” definitions

1. A **finite-dim. quantum datatype** is a ‘ring’ of matrices:

$$\mathcal{M}_2 \oplus \mathcal{M}_3 \oplus \mathbb{C}^2 \subseteq \mathcal{M}_7 \equiv \mathcal{B}(\mathbb{C}^7).$$

2. A **quantum datatype** is a **von Neumann algebra**, that is, a ‘ring’ of bounded operators on some Hilbert space \mathcal{H} ,

$$\mathcal{A} \subseteq \mathcal{B}(\mathcal{H}),$$

which is closed in a suitable topology on $\mathcal{B}(\mathcal{H})$.

3. $\mathbf{vN}_{\text{CPsU}}$ is the category of von Neumann algebras and normal, completely **positive**, **subunital** linear maps.
($\mathcal{A} \xrightarrow{\text{CPsU}} \mathcal{B}$ is a **quantum process** from \mathcal{B} to \mathcal{A} .)
4. $\mathbf{vN} \subseteq \mathbf{vN}_{\text{CPsU}}$ is the (wide) subcategory of multiplicative unital maps — the “sharp” quantum processes.

Main Theorem

$\mathbf{vN}_{\text{CPsU}}^{\text{op}}$ is isomorphic to the Kleisli category of a monad on \mathbf{vN}^{op} .

Main Theorem

$\mathbf{vN}_{\text{CPsU}}^{\text{op}}$ is isomorphic to the Kleisli category of a monad on \mathbf{vN}^{op} .

Proof sketch.

Main Theorem

$\mathbf{vN}_{\text{CPsU}}^{\text{op}}$ is isomorphic to the Kleisli category of a monad on \mathbf{vN}^{op} .

Proof sketch.

Since \mathbf{vN} and $\mathbf{vN}_{\text{CPsU}}$ have the same objects, we need only prove that $\mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$ has a left adjoint.

Main Theorem

$\mathbf{vN}_{\text{CPsU}}^{\text{op}}$ is isomorphic to the Kleisli category of a monad on \mathbf{vN}^{op} .

Proof sketch.

Since \mathbf{vN} and $\mathbf{vN}_{\text{CPsU}}$ have the same objects, we need only prove that $\mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$ has a left adjoint.

By the Adjoint Functor Theorem it suffices to show that

1. \mathbf{vN} has all limits, and $\mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$ preserves them, and
2. $\mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$ satisfies the solution set condition.

Main Theorem

$\mathbf{vN}_{\text{CPsU}}^{\text{op}}$ is isomorphic to the Kleisli category of a monad on \mathbf{vN}^{op} .

Proof sketch.

Since \mathbf{vN} and $\mathbf{vN}_{\text{CPsU}}$ have the same objects, we need only prove that $\mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$ has a left adjoint.

By the Adjoint Functor Theorem it suffices to show that

1. \mathbf{vN} has all limits, and $\mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$ preserves them, and
2. $\mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$ satisfies the solution set condition.

Point 1 follows without without tricks.

Main Theorem

$\mathbf{vN}_{\text{CPsU}}^{\text{op}}$ is isomorphic to the Kleisli category of a monad on \mathbf{vN}^{op} .

Proof sketch.

Since \mathbf{vN} and $\mathbf{vN}_{\text{CPsU}}$ have the same objects, we need only prove that $\mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$ has a left adjoint.

By the Adjoint Functor Theorem it suffices to show that

1. \mathbf{vN} has all limits, and $\mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$ preserves them, and
2. $\mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$ satisfies the solution set condition.

Point 1 follows without without tricks.

Point 2 follows from this: if \mathcal{B} is a von Neumann subalgebra generated by a subset X of a von Neumann algebra, then

$$\#\mathcal{B} \leq 2^{2^{\#\mathbb{C} \cdot \#X}}.$$



Ok, . . . but what is this left adjoint $\mathcal{F}: \mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$, concretely?

Ok, . . . but what is this left adjoint $\mathcal{F}: \mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$, concretely?

1. $\mathcal{F}(\{0\}) = \{0\}$, because $\{0\}$ is final.

Ok, . . . but what is this left adjoint $\mathcal{F}: \mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$, concretely?

1. $\mathcal{F}(\{0\}) = \{0\}$, because $\{0\}$ is final.
2. $\mathcal{F}(\mathbb{C}) = C[0, 1]$

Ok, . . . but what is this left adjoint $\mathcal{F}: \mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$, concretely?

1. $\mathcal{F}(\{0\}) = \{0\}$, because $\{0\}$ is final.
2. $\mathcal{F}(\mathbb{C}) = C[0, 1]**$

Ok, . . . but what is this left adjoint $\mathcal{F}: \mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$, concretely?

1. $\mathcal{F}(\{0\}) = \{0\}$, because $\{0\}$ is final.
2. $\mathcal{F}(\mathbb{C}) = C[0, 1]**$

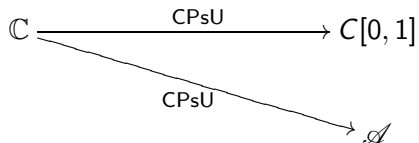
Ok, . . . but what is this left adjoint $\mathcal{F}: \mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$, concretely?

1. $\mathcal{F}(\{0\}) = \{0\}$, because $\{0\}$ is final.
2. $\mathcal{F}(\mathbb{C}) = C[0, 1]**$

$$\mathbb{C} \xrightarrow{\text{CPsU}} C[0, 1]$$

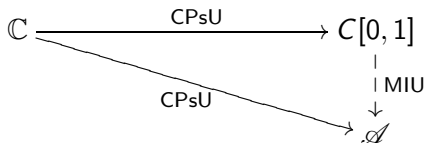
Ok, . . . but what is this left adjoint $\mathcal{F}: \mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$, concretely?

1. $\mathcal{F}(\{0\}) = \{0\}$, because $\{0\}$ is final.
2. $\mathcal{F}(\mathbb{C}) = C[0, 1]**$



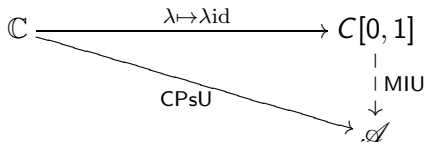
Ok, . . . but what is this left adjoint $\mathcal{F}: \mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$, concretely?

1. $\mathcal{F}(\{0\}) = \{0\}$, because $\{0\}$ is final.
2. $\mathcal{F}(\mathbb{C}) = C[0, 1]**$



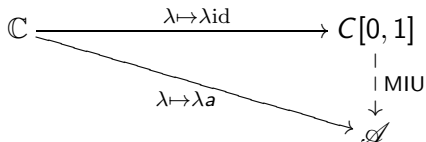
Ok, . . . but what is this left adjoint $\mathcal{F}: \mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$, concretely?

1. $\mathcal{F}(\{0\}) = \{0\}$, because $\{0\}$ is final.
2. $\mathcal{F}(\mathbb{C}) = C[0, 1]**$



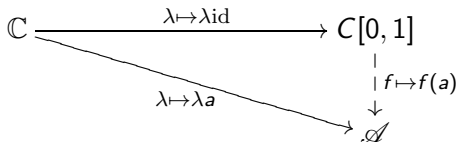
Ok, . . . but what is this left adjoint $\mathcal{F}: \mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$, concretely?

1. $\mathcal{F}(\{0\}) = \{0\}$, because $\{0\}$ is final.
2. $\mathcal{F}(\mathbb{C}) = C[0, 1]**$



Ok, . . . but what is this left adjoint $\mathcal{F}: \mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$, concretely?

1. $\mathcal{F}(\{0\}) = \{0\}$, because $\{0\}$ is final.
2. $\mathcal{F}(\mathbb{C}) = C[0, 1]**$



Ok, . . . but what is this left adjoint $\mathcal{F}: \mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$, concretely?

1. $\mathcal{F}(\{0\}) = \{0\}$, because $\{0\}$ is final.
2. $\mathcal{F}(\mathbb{C}) = C[0, 1]**$

$$\begin{array}{ccc} \mathbb{C} & \xrightarrow{\lambda \mapsto \widehat{\lambda \text{id}}} & C[0, 1]** \\ & \searrow \lambda \mapsto \lambda a & \downarrow f \mapsto f(a) \\ & & \mathcal{A} \end{array}$$

Ok, . . . but what is this left adjoint $\mathcal{F}: \mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$, concretely?

1. $\mathcal{F}(\{0\}) = \{0\}$, because $\{0\}$ is final.
2. $\mathcal{F}(\mathbb{C}) = C[0, 1]**$

$$\begin{array}{ccc} \mathbb{C} & \xrightarrow{\lambda \mapsto \widehat{\lambda \text{id}}} & C[0, 1]** \\ & \searrow \lambda \mapsto \lambda a & \downarrow f \mapsto f(a) \\ & & \mathcal{A} \end{array}$$

3. $\mathcal{F}(\mathbb{C}^2) = ??$

Ok, . . . but what is this left adjoint $\mathcal{F}: \mathbf{vN} \rightarrow \mathbf{vN}_{\text{CPsU}}$, concretely?

1. $\mathcal{F}(\{0\}) = \{0\}$, because $\{0\}$ is final.
2. $\mathcal{F}(\mathbb{C}) = C[0, 1]**$

$$\begin{array}{ccc} \mathbb{C} & \xrightarrow{\lambda \mapsto \widehat{\lambda \text{id}}} & C[0, 1]** \\ & \searrow \lambda \mapsto \lambda a & \downarrow f \mapsto f(a) \\ & & \mathcal{A} \end{array}$$

3. $\mathcal{F}(\mathbb{C}^2) = ??$

At least $\mathcal{F}(\mathbb{C}^2)$ is not commutative (because $f(1, 0)$ and $f(0, 1)$ might not commute for a CPsU-map $f: \mathbb{C}^2 \rightarrow \mathcal{A}$).

An Application

An Application

A model of the Quantum Lambda Calculus:

$$\begin{array}{ccccc} \mathbf{Set} & \begin{array}{c} \xrightarrow{\ell^\infty} \\ \perp \\ \xleftarrow{\mathbf{vN}(\cdot, \mathbb{C})} \end{array} & \mathbf{vN}^{\text{op}} & \begin{array}{c} \xrightarrow{\mathcal{J}} \\ \perp \\ \xleftarrow{\mathcal{F}} \end{array} & \mathbf{vN}_{\text{CPsU}}^{\text{op}} \end{array}$$

An Application

A model of the Quantum Lambda Calculus:

$$\mathbf{Set} \begin{array}{c} \xrightarrow{\ell^\infty} \\ \perp \\ \xleftarrow{\mathbf{vN}(\cdot, \mathbb{C})} \end{array} \mathbf{vN}^{\text{op}} \subset \begin{array}{c} \xrightarrow{\mathcal{J}} \\ \perp \\ \xleftarrow{\mathcal{F}} \end{array} \mathbf{vN}_{\text{CPsU}}^{\text{op}}$$

This is the subject of the next talk!

An Application

A model of the Quantum Lambda Calculus:

$$\mathbf{Set} \begin{array}{c} \xrightarrow{\ell^\infty} \\ \perp \\ \xleftarrow{\mathbf{vN}(\cdot, \mathbb{C})} \end{array} \mathbf{vN}^{\text{op}} \begin{array}{c} \xrightarrow{\mathcal{J}} \\ \perp \\ \xleftarrow{\mathcal{F}} \end{array} \mathbf{vN}_{\text{CPsU}}^{\text{op}}$$

This is the subject of the next talk!

Teaser:

$$\llbracket !A \rrbracket = \ell^\infty(\mathbf{vN}(\llbracket A \rrbracket, \mathbb{C})) \quad \llbracket A \multimap B \rrbracket = (\mathcal{F}\mathcal{J}\llbracket B \rrbracket)^{\ast\llbracket A \rrbracket}$$

where $(-)^{\ast\llbracket A \rrbracket}$ is the Kornell's free exponential.

An Application

A model of the Quantum Lambda Calculus:

$$\mathbf{Set} \begin{array}{c} \xrightarrow{\ell^\infty} \\ \perp \\ \xleftarrow{\mathbf{vN}(\cdot, \mathbb{C})} \end{array} \mathbf{vN}^{\text{op}} \begin{array}{c} \xrightarrow{\mathcal{J}} \\ \perp \\ \xleftarrow{\mathcal{F}} \end{array} \mathbf{vN}_{\text{CPsU}}^{\text{op}}$$

This is the subject of the next talk!

Teaser:

$$\llbracket !A \rrbracket = \ell^\infty(\mathbf{vN}(\llbracket A \rrbracket, \mathbb{C})) \quad \llbracket A \multimap B \rrbracket = (\mathcal{F}\mathcal{J}\llbracket B \rrbracket)^{*[\![A]\!]}$$

where $(-)^{*[\![A]\!]}$ is the Kornell's free exponential.

Questions?