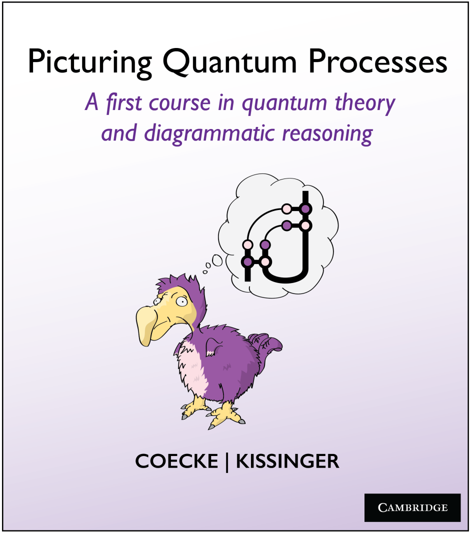
Process Theories and Graphical Language

Aleks Kissinger

Institute for Computing and Information Sciences
Radboud University Nijmegen

12th July 2016


Picturing Quantum Processes
A first course in quantum theory
and diagrammatic reasoning
COECKE | KISSINGER
CAMBRIDGE

Picturing Quantum Processes

Picturing Quantum Processes

When two systems [...] enter into temporary physical interaction due to known forces between them, [...] then they can no longer be described in the same way as before, viz. by endowing each of them with a representative of its own. **I would not call that one but rather the characteristic trait of quantum mechanics**, the one that enforces its entire departure from classical lines of thought.

— *Erwin Schrödinger, 1935.*

Picturing Quantum Processes

> When two systems [...] enter into temporary physical interaction due to known forces between them, [...] then they can no longer be described in the same way as before, viz. by endowing each of them with a representative of its own. **I would not call that one but rather the characteristic trait of quantum mechanics**, the one that enforces its entire departure from classical lines of thought.
>
> — *Erwin Schrödinger, 1935.*

In quantum theory, *interaction* of systems is everything. **Diagrams** are the language of interaction.

Picturing Quantum Processes

Q: How much of quantum theory can be understood just using diagrams and diagram transformation?

Picturing Quantum Processes

Q: How much of quantum theory can be understood just using diagrams and diagram transformation?

A: Pretty much everything!

Outline

Outline

Process theories and diagrams

Quantum processes

Classical and quantum interaction

Applications: a Hollywood-style trailer

Processes

- A process is anything with zero or more *inputs* and zero or more *outputs*

Processes

- A process is anything with zero or more *inputs* and zero or more *outputs*
- For example, this function:

$$f(x, y) = x^2 + y$$

Processes

- A process is anything with zero or more *inputs* and zero or more *outputs*
- For example, this function:

$$f(x, y) = x^2 + y$$

...is a process when takes two real numbers as input, and produces a real number as output.

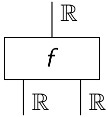
Processes

- A process is anything with zero or more *inputs* and zero or more *outputs*
- For example, this function:

$$f(x, y) = x^2 + y$$

...is a process when takes two real numbers as input, and produces a real number as output.

- We could also write it like this:



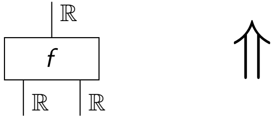
Processes

- A process is anything with zero or more *inputs* and zero or more *outputs*
- For example, this function:

$$f(x, y) = x^2 + y$$

...is a process when takes two real numbers as input, and produces a real number as output.

- We could also write it like this:



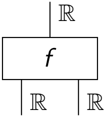
Processes

- A process is anything with zero or more *inputs* and zero or more *outputs*
- For example, this function:

$$f(x, y) = x^2 + y$$

...is a process when takes two real numbers as input, and produces a real number as output.

- We could also write it like this:



$\Uparrow$

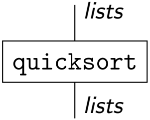
- The labels on wires are called system-types or just types

More processes

- Similarly, computer programs are processes

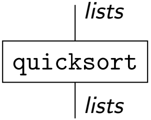
More processes

- Similarly, computer programs are processes
- For example, a program that sorts lists might look like this:

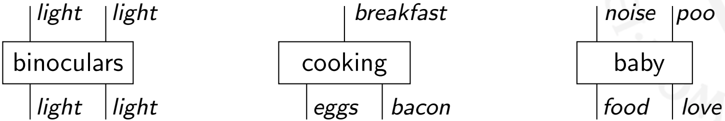


More processes

- Similarly, computer programs are processes
- For example, a program that sorts lists might look like this:

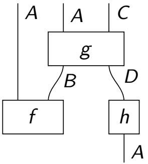


- These are also perfectly good processes:



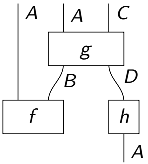
Diagrams

- We can combine simple processes to make more complicted ones, described by diagrams:

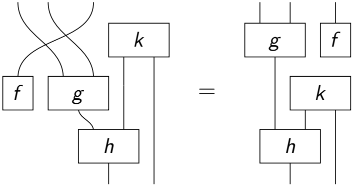


Diagrams

- We can combine simple processes to make more complicted ones, described by diagrams:



- The golden rule: only connectivity matters!

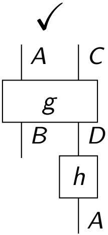


Types

- Connections are only allowed where the types match

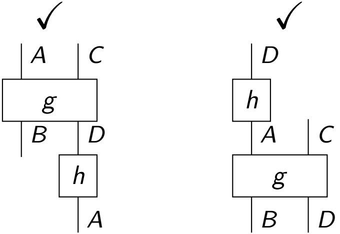
Types

- Connections are only allowed where the types match, e.g.:



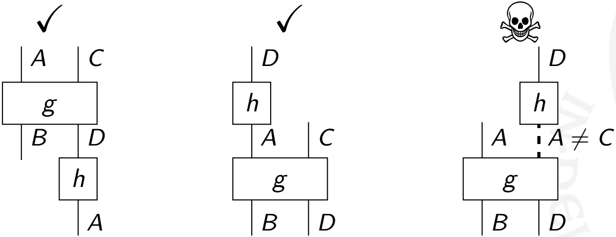
Types

- Connections are only allowed where the types match, e.g.:



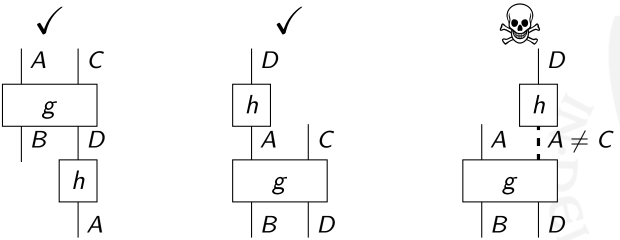
Types

- Connections are only allowed where the types match, e.g.:



Types

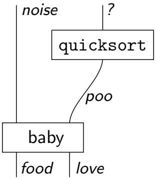
- Connections are only allowed where the types match, e.g.:



- Types tell us when it makes sense to plug processes together

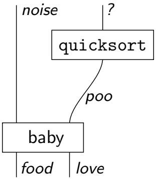
Types and Process Theories

- Ill-typed diagrams are undefined:



Types and Process Theories

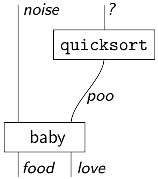
- Ill-typed diagrams are undefined:



- In fact, these processes don't ever make sense to plug together

Types and Process Theories

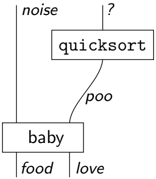
- Ill-typed diagrams are undefined:



- In fact, these processes don't <u>ever</u> make sense to plug together
- A family of processes which <u>do</u> make sense together is called a process theory

Types and Process Theories

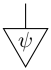
- Ill-typed diagrams are undefined:



- In fact, these processes don't ever make sense to plug together
- A family of processes which do make sense together is called a process theory, e.g.
  - **functions**
  - **linear maps**
  - **optical devices**
  - **proofs**, ...

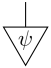
Special processes: states and effects

- Processes with no inputs are called states:



Special processes: states and effects

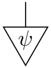
- Processes with no inputs are called states:



Interpret as: preparing a system in a particular configuration, where we don't care what came before.

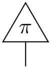
Special processes: states and effects

- Processes with no inputs are called states:



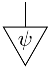
Interpret as: preparing a system in a particular configuration, where we don't care what came before.

- Processes with no outputs are called effects:



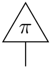
Special processes: states and effects

- Processes with no inputs are called states:



Interpret as: preparing a system in a particular configuration, where we don't care what came before.

- Processes with no outputs are called effects:



Interpret as: testing for a property π, where we don't care what happens after.

Numbers

- A number is a process with no inputs or outputs, written as:

or just: λ

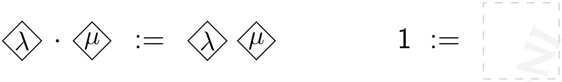
Radboud University Nijmegen

Numbers

- A number is a process with no inputs or outputs, written as:

or just: λ

- Numbers always form a commutative monoid:

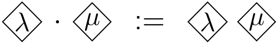


Numbers

- A number is a process with no inputs or outputs, written as:

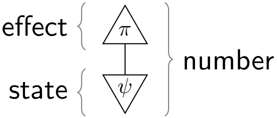
or just: λ

- Numbers always form a commutative monoid:



$1 :=$

- **Interpret as:** what happens when a state meets an effect



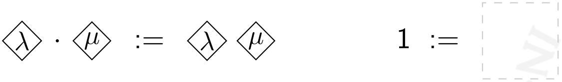
Radbound University Nijmegen

Numbers

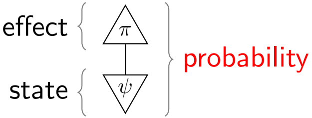
- A number is a process with no inputs or outputs, written as:

or just: λ

- Numbers always form a commutative monoid:



- **Interpret as:** what happens when a state meets an effect, e.g.

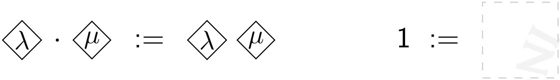


Numbers

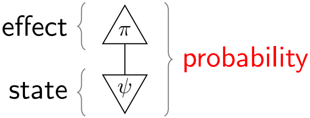
- A number is a process with no inputs or outputs, written as:

or just: λ

- Numbers always form a commutative monoid:



- **Interpret as:** what happens when a state meets an effect, e.g.



This is called the (generalised) Born rule

Process theories in general

Q: What kinds of behaviour can we study using just diagrams, and nothing else?

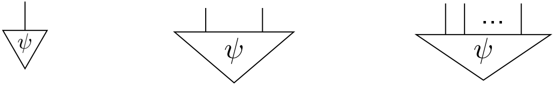
Process theories in general

Q: What kinds of behaviour can we study using just diagrams, and nothing else?

A: (Non-)separability

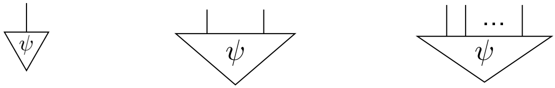
Separable states

- States can be on a single system, two systems, or many systems:

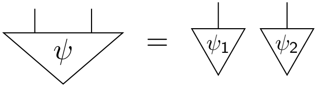


Separable states

- States can be on a single system, two systems, or many systems:

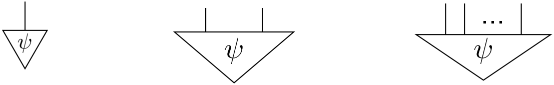


- A state ψ on two systems is $\otimes$-*separable* if there exist ψ_1, ψ_2 such that:

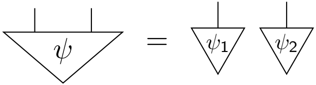


Separable states

- States can be on a single system, two systems, or many systems:



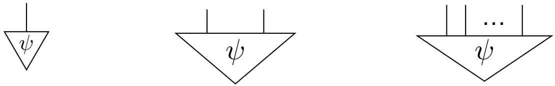
- A state ψ on two systems is $\otimes$-*separable* if there exist ψ_1, ψ_2 such that:



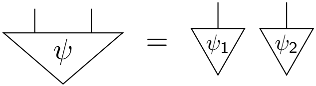
- **Intuitively:** the properties of the system on the left are *independent* from those on the right

Separable states

- States can be on a single system, two systems, or many systems:



- A state ψ on two systems is $\otimes$-*separable* if there exist ψ_1, ψ_2 such that:



- **Intuitively:** the properties of the system on the left are *independent* from those on the right
- In classical (deterministic) world, we expect *all states* to $\otimes$-separate

Characterising non-separability

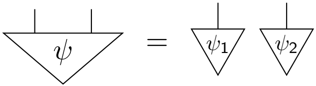
- ...which is why non-separable states are way more interesting!

Characterising non-separability

- ...which is why non-separable states are way more interesting!
- But, how do we know we've found one?

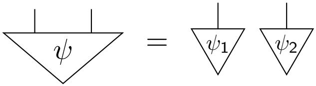
Characterising non-separability

- ...which is why non-separable states are way more interesting!
- But, how do we know we've found one?
- i.e. that there do not exist states ψ_1, ψ_2 such that:



Characterising non-separability

- ...which is why non-separable states are way more interesting!
- But, how do we know we've found one?
- i.e. that there do not exist states ψ_1, ψ_2 such that:



- **Problem:** Showing that something doesn't exist can be hard.

Characterising non-separability

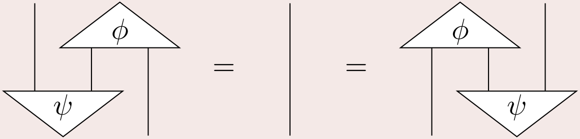
Solution: Replace a negative property with a (stronger) postive one:

Characterising non-separability

Solution: Replace a negative property with a (stronger) postive one:

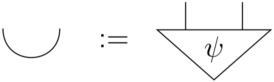
Definition

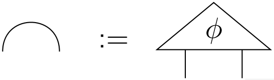
A state ψ is called *cup-state* if there exists an effect ϕ, called a *cap-effect*, such that:



Cup-states

- By introducing some clever notation:

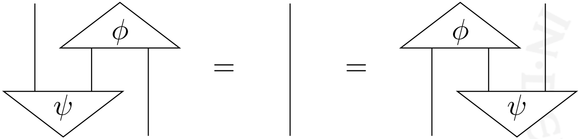




Cup-states

- By introducing some clever notation:

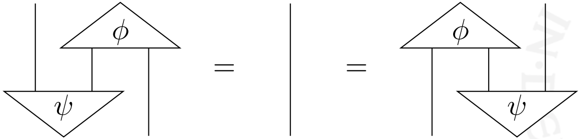
- Then these equations:



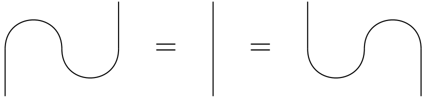
Cup-states

- By introducing some clever notation:

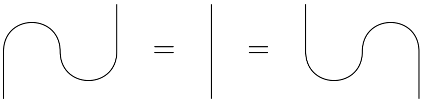
- Then these equations:



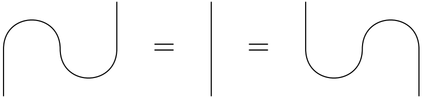
- ...look like this:



Yank the wire!



Yank the wire!



A no-go theorem for separability

Theorem

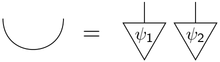
If a process theory (i) has cup-states for every type and (ii) every state separates, then it is trivial.

A no-go theorem for separability

Theorem

If a process theory (i) has cup-states for every type and (ii) every state separates, then it is trivial.

Proof. Suppose a cup-state separates:



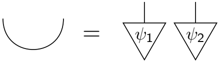
Then for any f:

A no-go theorem for separability

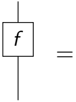
Theorem

If a process theory (i) has cup-states for every type and (ii) every state separates, then it is trivial.

Proof. Suppose a cup-state separates:



Then for any f:



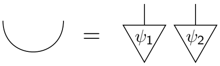
□

A no-go theorem for separability

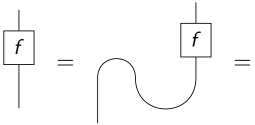
Theorem

If a process theory (i) has cup-states for every type and (ii) every state separates, then it is trivial.

Proof. Suppose a cup-state separates:



Then for any f:



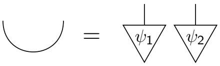
□

A no-go theorem for separability

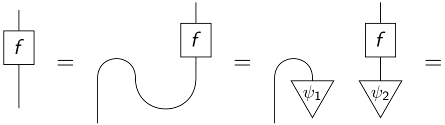
Theorem

If a process theory (i) has cup-states for every type and (ii) every state separates, then it is trivial.

Proof. Suppose a cup-state separates:



Then for any f:



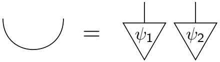
□

A no-go theorem for separability

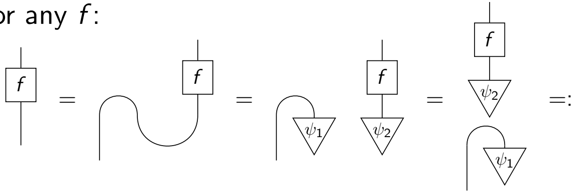
Theorem

If a process theory (i) has cup-states for every type and (ii) every state separates, then it is trivial.

Proof. Suppose a cup-state separates:



Then for any f:



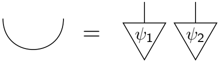
□

A no-go theorem for separability

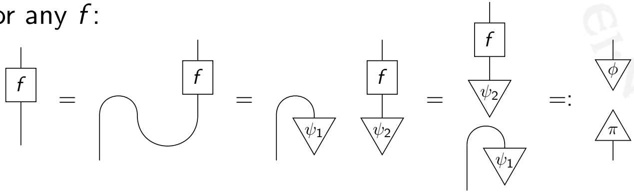
Theorem

If a process theory (i) has cup-states for every type and (ii) every state separates, then it is trivial.

Proof. Suppose a cup-state separates:

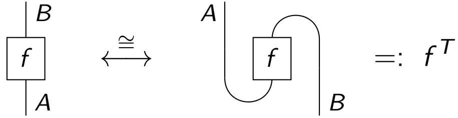


Then for any f:

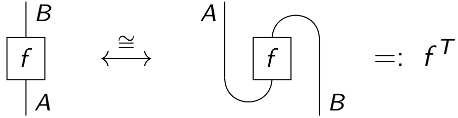


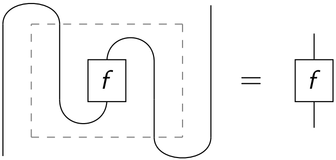
□

Transpose



Transpose



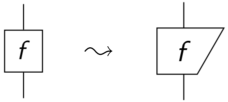


i.e.

$$(f^T)^T = f$$

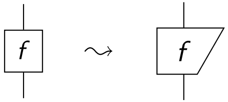
Tranpose = rotation

A bit of a deformation:

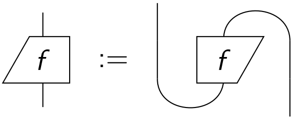


Tranpose = rotation

A bit of a deformation:

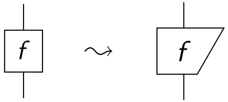


allows some clever notation:

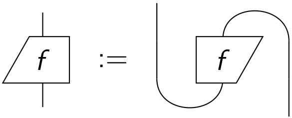


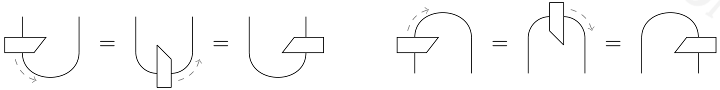
Tranpose = rotation

A bit of a deformation:

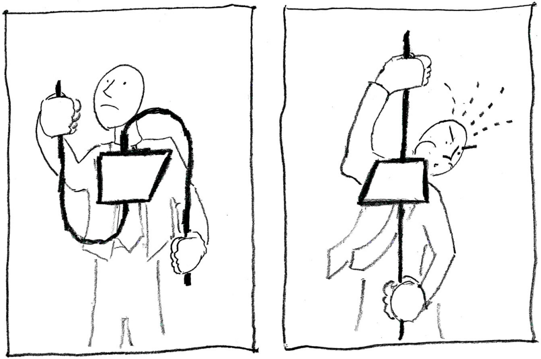


allows some clever notation:



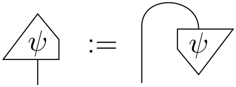


Transpose = rotation



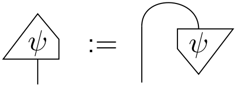
Tranpose = rotation

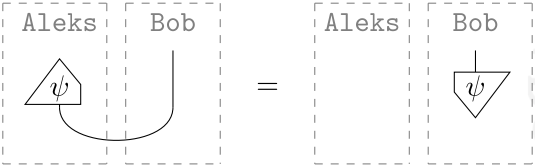
Specialised to states:



Tranpose = rotation

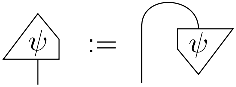
Specialised to states:

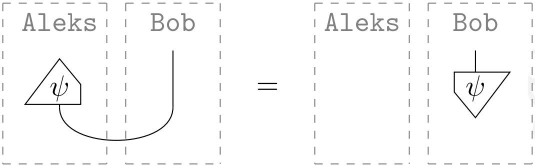


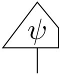


Tranpose = rotation

Specialised to states:



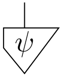


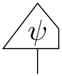
as soon as Aleks obtains 

Bob's system will be in state 

State/effect correspondence

states of system A $\cong$ *effects for correlated system B*

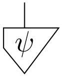


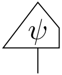


transpose

State/effect correspondence

states of system A $\cong$ *effects for correlated system B*

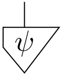




transpose

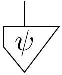
But what about...

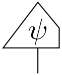
states of system A $\cong$ *effects for system A*



State/effect correspondence

states of system A $\cong$ *effects for correlated system B*

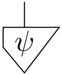


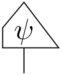


transpose

But what about...

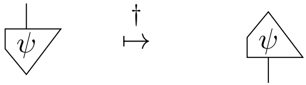
states of system A $\cong$ *effects for system A*





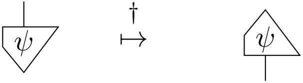
adjoint

Adjoints



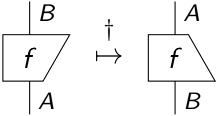
state ψ **testing for** ψ

Adjoints

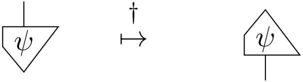


state ψ **testing for** ψ

Extends from states/effects to all processes:

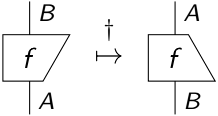


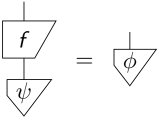
Adjoints



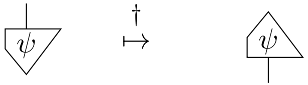
state ψ **testing for** ψ

Extends from states/effects to all processes:



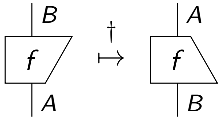


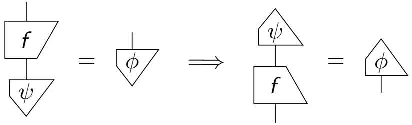
Adjoints



state ψ **testing for** ψ

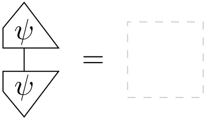
Extends from states/effects to all processes:





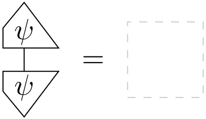
Normalised states and isometries

- Adjoints increase expressiveness, for instance can say when ψ is *normalised*:

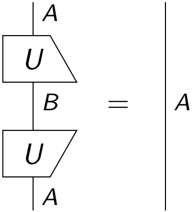


Normalised states and isometries

- Adjoints increase expressiveness, for instance can say when ψ is *normalised*:

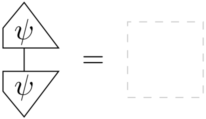


- U is an *isometry*:

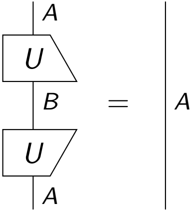


Normalised states and isometries

- Adjoints increase expressiveness, for instance can say when ψ is *normalised*:



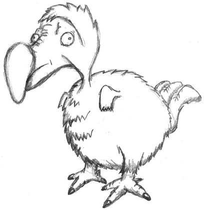
- U is an *isometry*:



- ...and unitary, self-adjoint, positive, etc.

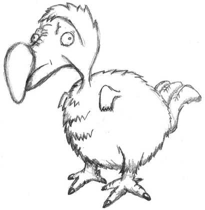
Conjugates

If we:

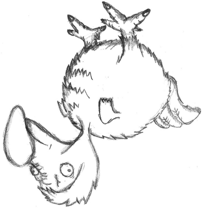


Conjugates

If we:

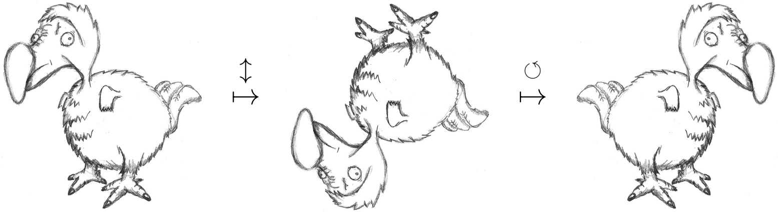


$\updownarrow$
$\mapsto$



Conjugates

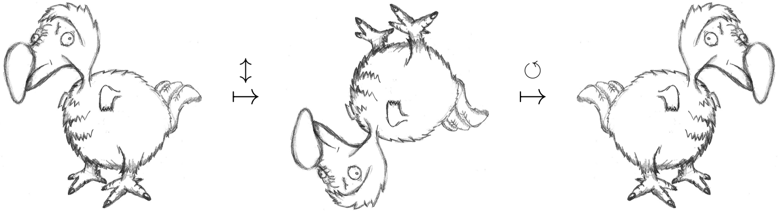
If we:



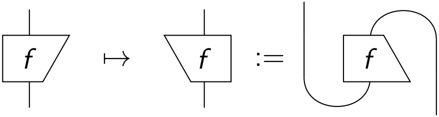
...we get horizontal reflection.

Conjugates

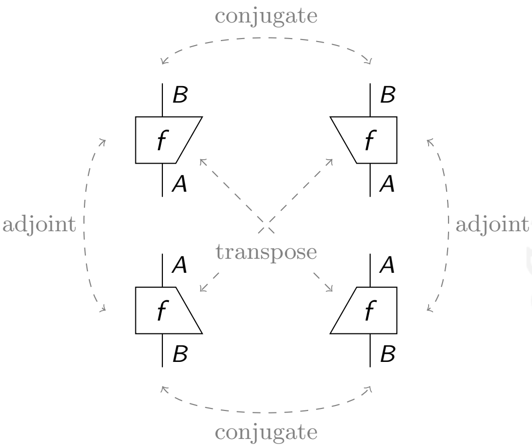
If we:



...we get horizontal reflection. The *conjugate*:



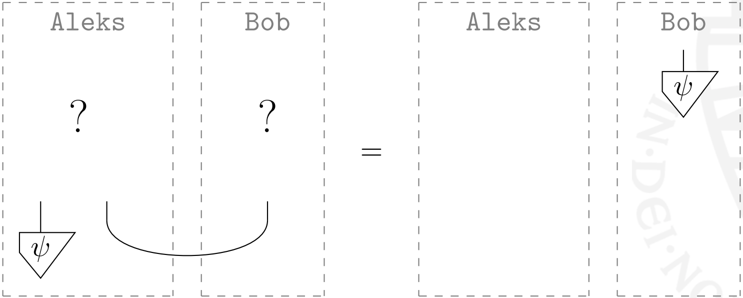
4 kinds of box



Quantum teleportation: take 1

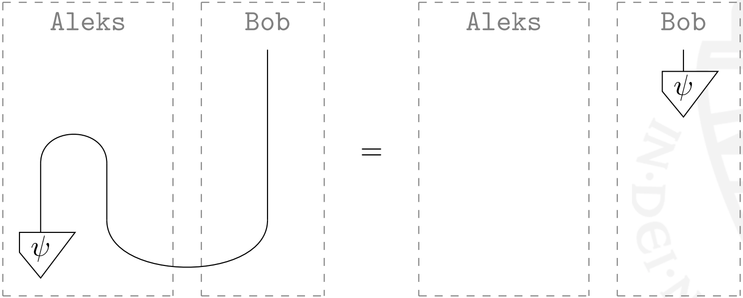
Quantum teleportation: take 1

Can we fill in '?' to get this?



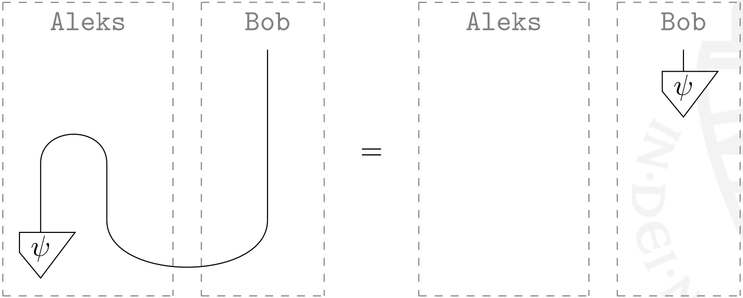
Quantum teleportation: take 1

Here's a simple solution:



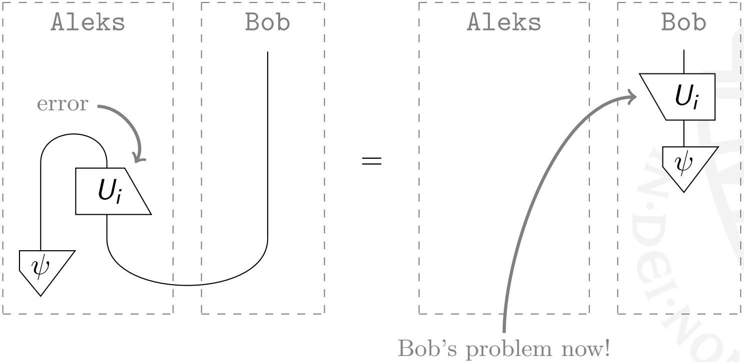
Quantum teleportation: take 1

Here's a simple solution:



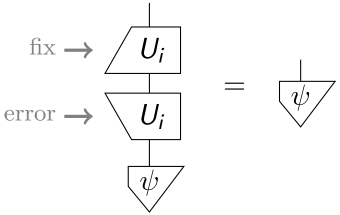
Problem: 'cap' can't be performed deterministically

Quantum teleportation: take 1

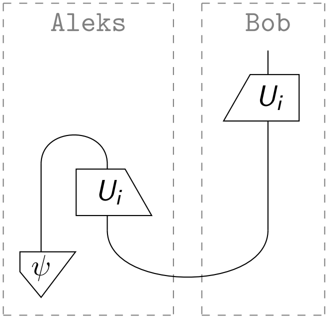


Quantum teleportation: take 1

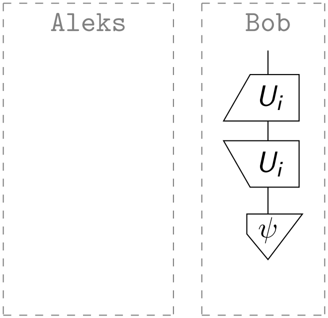
Solution: Bob fixes the error.



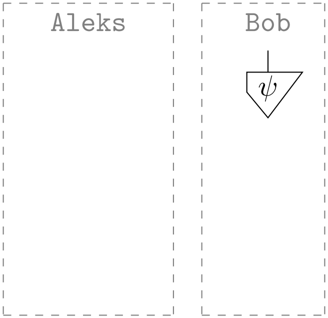
Quantum teleportation: take 1



Quantum teleportation: take 1



Quantum teleportation: take 1



Outline

Process theories and diagrams

Quantum processes

Classical and quantum interaction

Applications: a Hollywood-style trailer

Hilbert space

The starting point for quantum theory is the process theory of **linear maps**

Hilbert space

The starting point for quantum theory is the process theory of **linear maps**, which has:

1. **systems:** Hilbert spaces
2. **processes:** complex linear maps

Hilbert space

The starting point for quantum theory is the process theory of **linear maps**, which has:

1. **systems:** Hilbert spaces
2. **processes:** complex linear maps

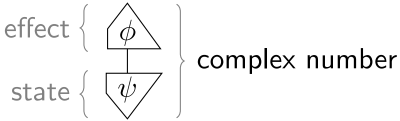
...in particular, numbers are *complex numbers*.

Hilbert space

Looking at the ‘Born rule’ for **linear maps**, we have a problem:

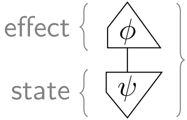
Hilbert space

Looking at the ‘Born rule’ for **linear maps**, we have a problem:



Hilbert space

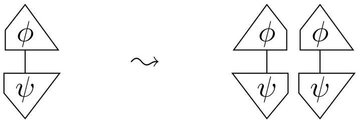
Looking at the 'Born rule' for **linear maps**, we have a problem:



complex number $\neq$ probability!

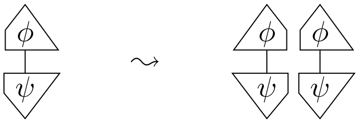
Doubling

Solution: multiply by the conjugate:

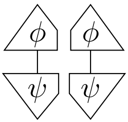


Doubling

Solution: multiply by the conjugate:

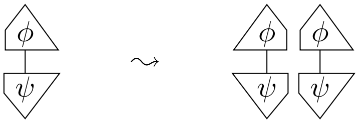


Then, for normalised ψ, ϕ:

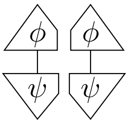
$0 \leq$  ≤ 1

Doubling

Solution: multiply by the conjugate:



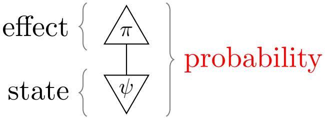
Then, for normalised ψ, ϕ:

$0 \leq$  ≤ 1

(i.e. the 'usual' Born rule: $\overline{\langle\phi|\psi\rangle}\langle\phi|\psi\rangle = |\langle\phi|\psi\rangle|^2$)

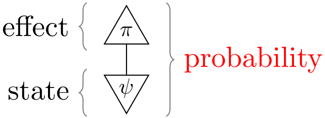
Doubling

New problem: We lost this:



Doubling

New problem: We lost this:



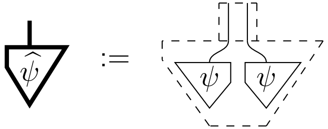
...which was the basis of our interpretation for states, effects, and numbers.

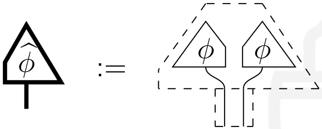
Doubling

Solution: Make a new process theory with doubling 'baked in':

Doubling

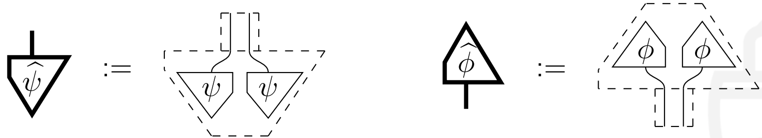
Solution: Make a new process theory with doubling 'baked in':



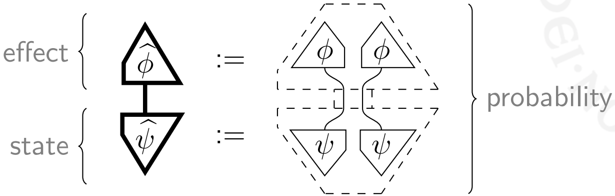


Doubling

Solution: Make a new process theory with doubling 'baked in':

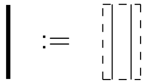


Then:



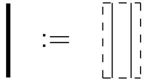
Doubling

The new process theory has doubled systems $\widehat{H} := H \otimes H$:

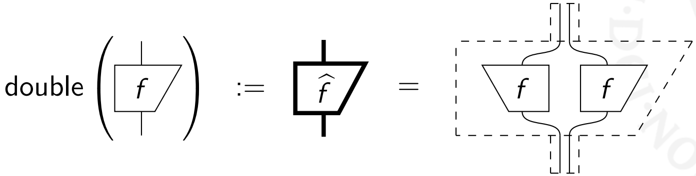


Doubling

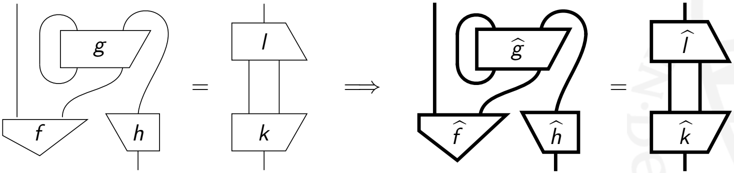
The new process theory has doubled systems $\widehat{H} := H \otimes H$:



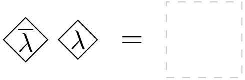
and processes:



Doubling preserves diagrams

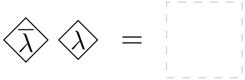


...but kills global phases



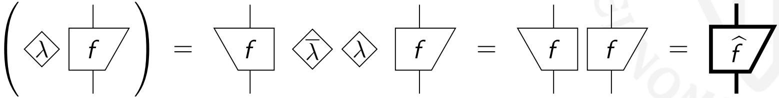
(i.e. $\lambda = e^{i\alpha}$)

...but kills global phases



(i.e. $\lambda = e^{i\alpha}$)

$$\Longrightarrow$$

double 

Discarding

Doubling also lets us do something we couldn't do before:

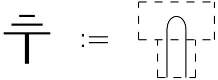
Discarding

Doubling also lets us do something we couldn't do before: throw stuff away!

Discarding

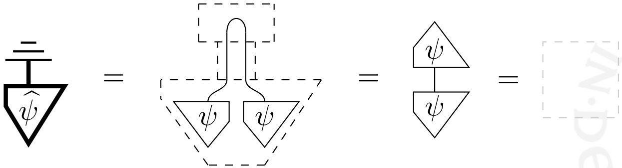
Doubling also lets us do something we couldn't do before: throw stuff away!

How? Like this:



Discarding

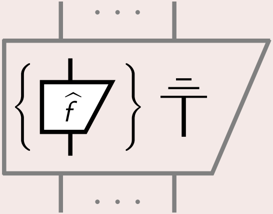
For normalised ψ, the two copies annihilate:



Quantum maps

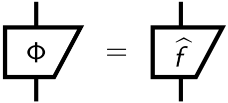
Definition

The process theory of **quantum maps** has as types (doubled) Hilbert spaces $\widehat{H}$ and as processes:



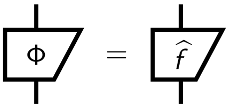
Two characterisations of ‘pure’

No discarding involved, i.e. for some f:



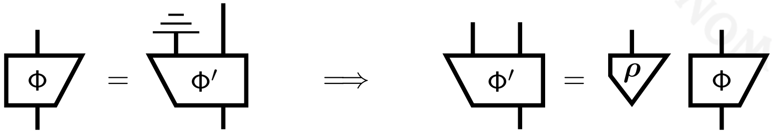
Two characterisations of 'pure'

No discarding involved, i.e. for some f:



$$\Longleftrightarrow$$

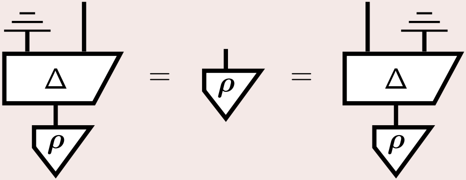
Any *extension* is trivial:



Consequence: no-broadcasting

Theorem (No universal broadcasting)

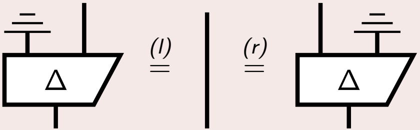
There exists no quantum map Δ *where:*



Consequence: no-broadcasting

Theorem (No universal broadcasting)

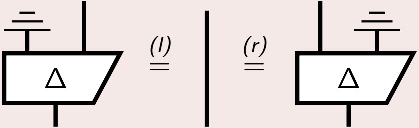
There exists no quantum map Δ where:



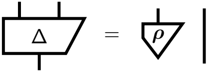
Consequence: no-broadcasting

Theorem (No universal broadcasting)

There exists no quantum map Δ where:



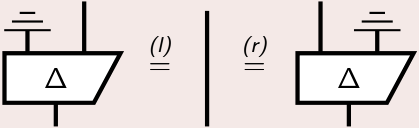
Proof. From (l):



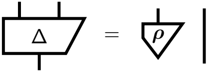
Consequence: no-broadcasting

Theorem (No universal broadcasting)

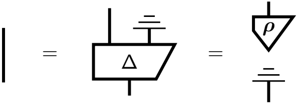
There exists no quantum map Δ *where:*



Proof. From (l):



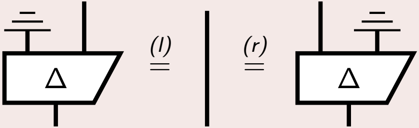
From (r):



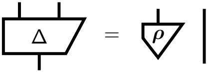
Consequence: no-broadcasting

Theorem (No universal broadcasting)

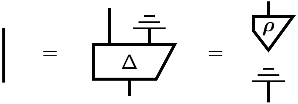
There exists no quantum map Δ where:



Proof. From (l):



From (r):

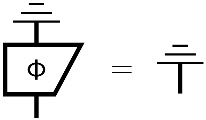


⇒ contradiction. □

Causality

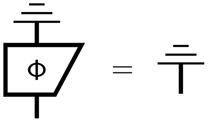
Causality

A quantum map is called *causal* if:



Causality

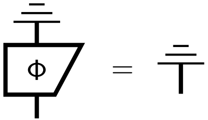
A quantum map is called *causal* if:



If we discard the output of a process,
it doesn't matter which process happened.

Causality

A quantum map is called *causal* if:

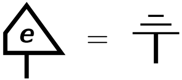


If we discard the output of a process,
it doesn't matter which process happened.

causal $\iff$ *deterministically physically realisable*

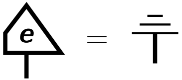
Consequence: no cap effect ☹

Consequence: there is a unique causal effect, discarding:

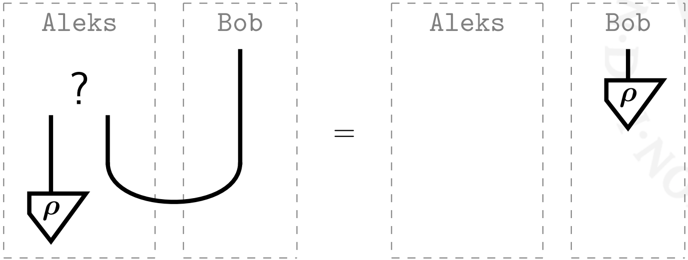


Consequence: no cap effect ☹

Consequence: there is a unique causal effect, discarding:

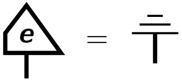


Hence 'deterministic quantum teleportation' must fail:

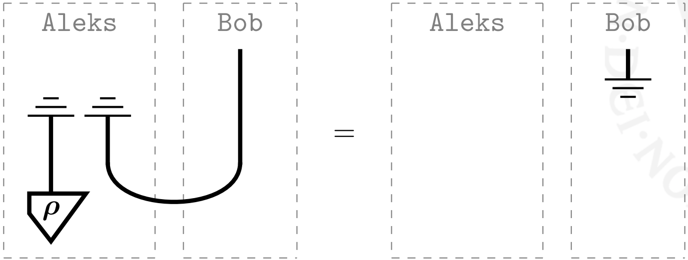


Consequence: no cap effect ☹

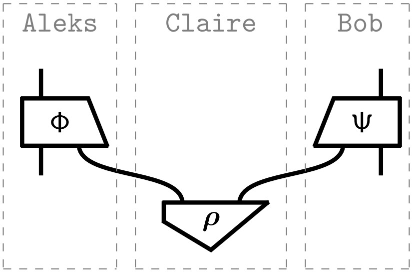
Consequence: there is a unique causal effect, discarding:



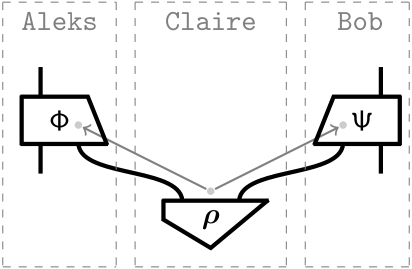
Hence 'deterministic quantum teleportation' must fail:



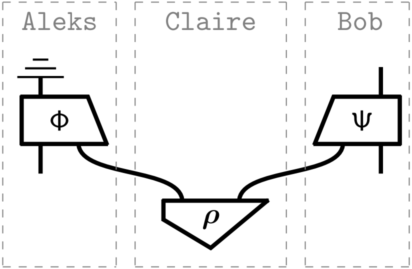
Consequence: no signalling ☺



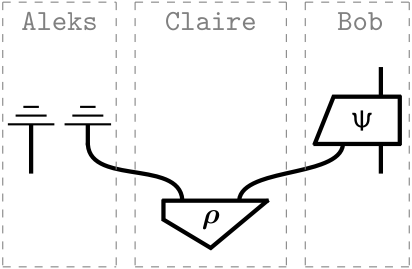
Consequence: no signalling ☺



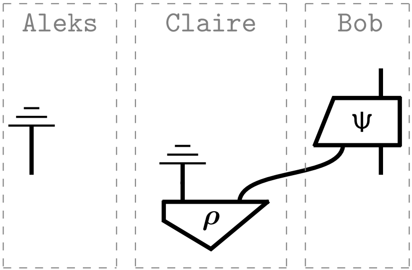
Consequence: no signalling ☺



Consequence: no signalling ☺



Consequence: no signalling ☺



Outline

Process theories and diagrams

Quantum processes

Classical and quantum interaction

Applications: a Hollywood-style trailer

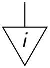
Double vs. single wires

$$\left(\text{quantum} \quad := \quad \Bigg| \right)$$

Double vs. single wires

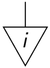
$$\left(\text{quantum} \quad := \quad \rule{3pt}{2em} \right) \quad \neq \quad \left(\text{classical} \quad := \quad \rule{0.5pt}{2em} \right)$$

Classical values

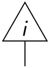


:= ‘providing classical value i’

Classical values

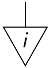


:= 'providing classical value i'

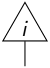


:= 'testing for classical value i'

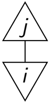
Classical values



:= 'providing classical value i'

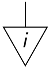


:= 'testing for classical value i'

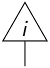


$$= \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

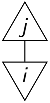
Classical values



:= ‘providing classical value i’



:= ‘testing for classical value i’



$$= \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

($\Rightarrow$ ONB)

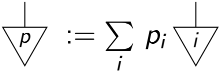
Classical states

General state of a classical system:

$$\bigtriangledown_{p} \;:=\; \sum_i p_i \, \bigtriangledown_{i} \qquad \leftarrow \quad \text{probability distributions}$$

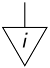
Classical states

General state of a classical system:



← probability distributions

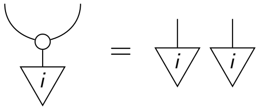
Hence:



← point distributions

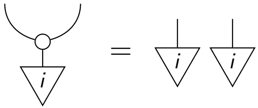
Copy and delete

Unlike quantum states, classical values can be *copied*:

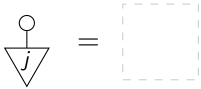


Copy and delete

Unlike quantum states, classical values can be *copied*:

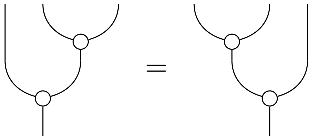


and *deleted*:



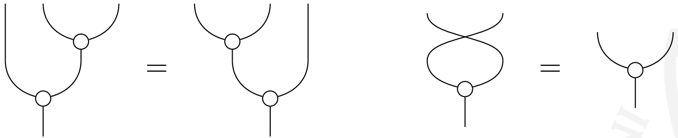
Copy and delete

These satisfy some equations you would expect:



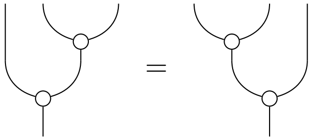
Copy and delete

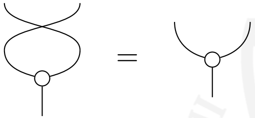
These satisfy some equations you would expect:



Copy and delete

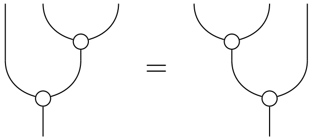
These satisfy some equations you would expect:

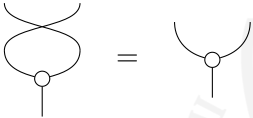


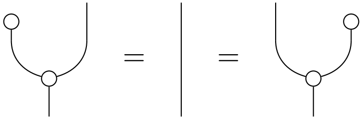


Copy and delete

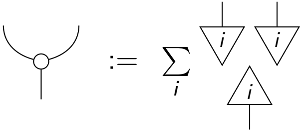
These satisfy some equations you would expect:

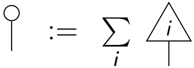




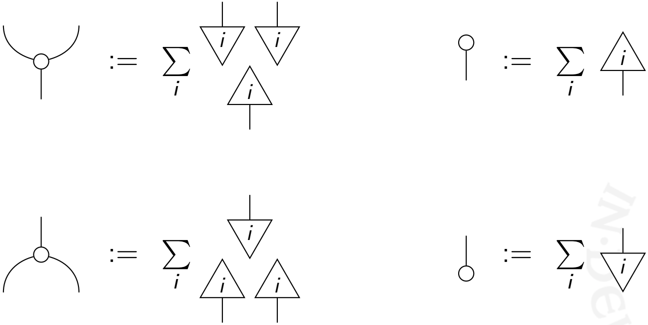


Other classical maps

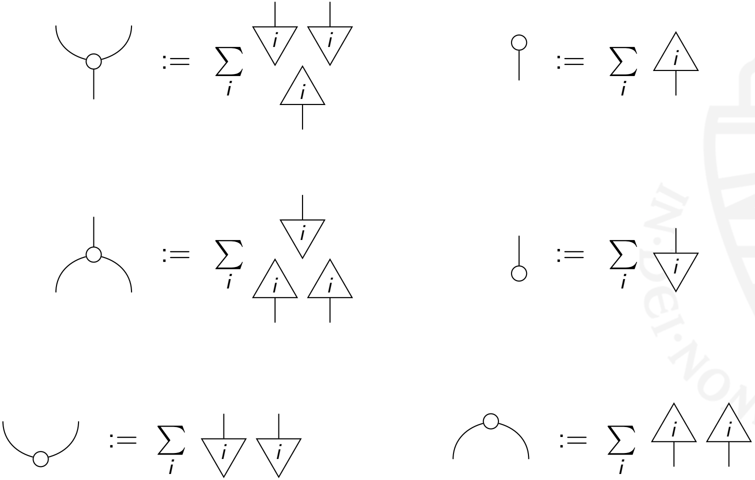




Other classical maps

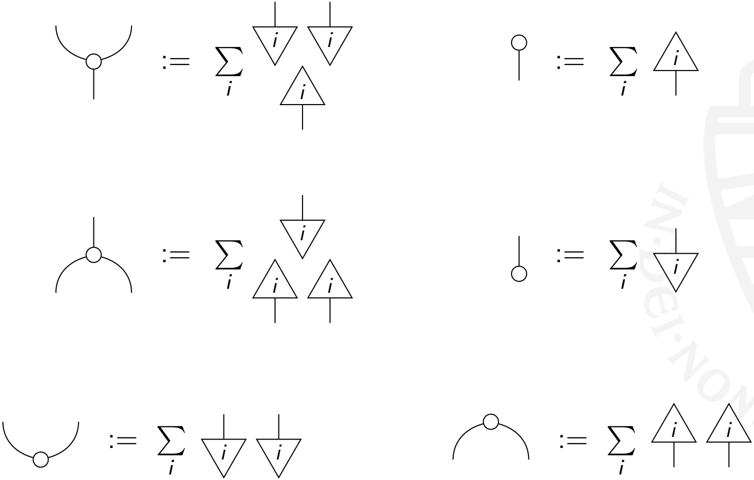


Other classical maps

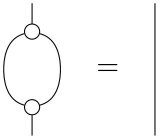


Radboud University Nijmegen

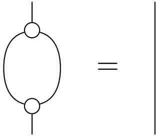
Other classical maps

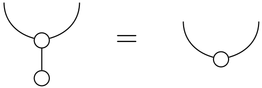


...satisfying lots of equations

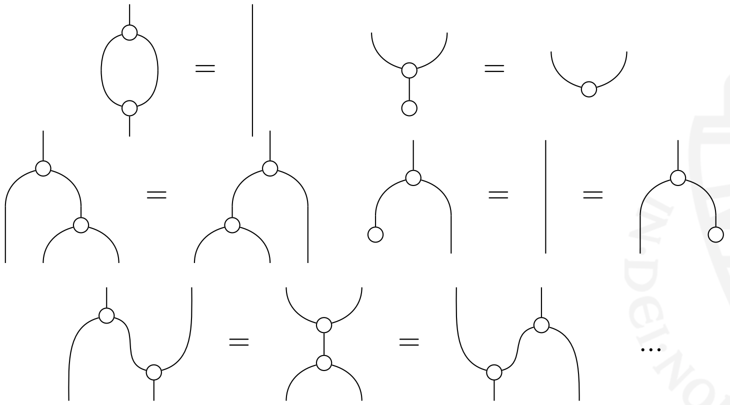


...satisfying lots of equations

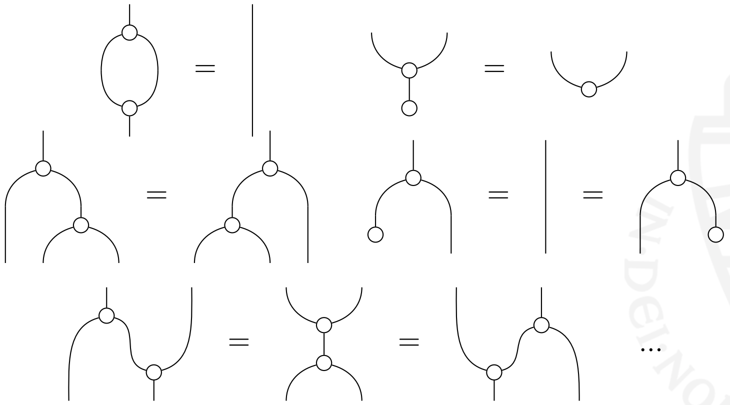




...satisfying lots of equations



...satisfying lots of equations

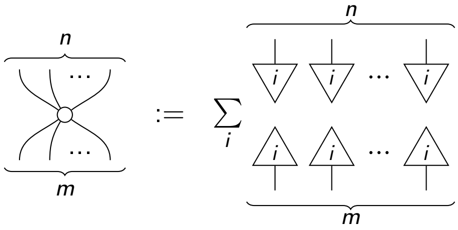


When does it end???

Spiders

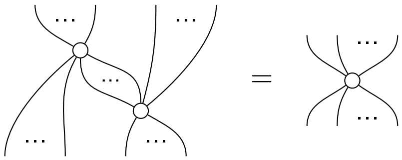
Spiders

All of these are special cases of *spiders*:



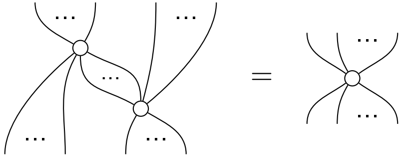
Spiders

The only equation you need to remember is this one:



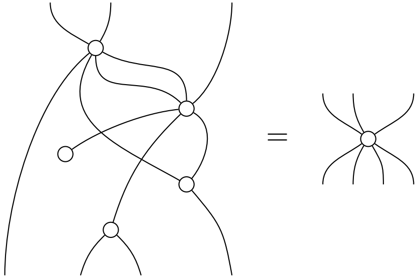
Spiders

The only equation you need to remember is this one:

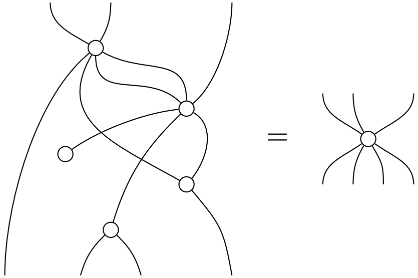


When spiders meet, they fuse together.

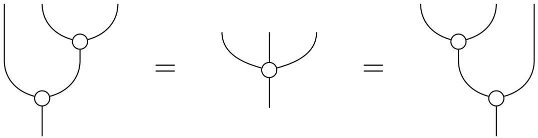
Spider reasoning



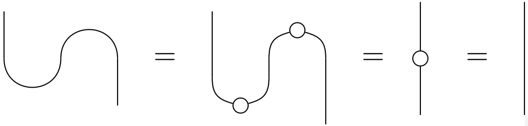
Spider reasoning



For example:



Spider reasoning ⇒ string diagram reasoning



How do we recognise spiders?

Suppose we have something that ‘behaves like’ a spider:

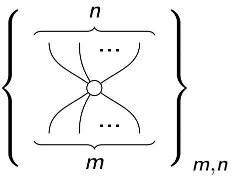
How do we recognise spiders?

Suppose we have something that 'behaves like' a spider:

Do we know it is one?

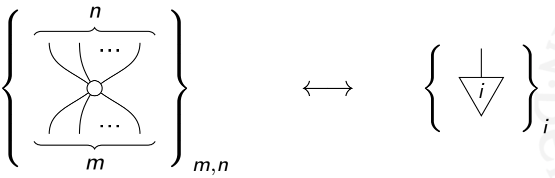
Spiders = 'diagrammatic ONBs'

Yes!



Spiders = 'diagrammatic ONBs'

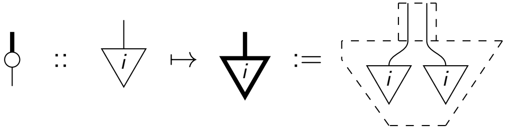
Yes!



Classical and quantum interaction

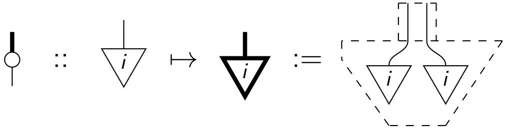
Classical and quantum interaction

Classical values can be encoded as quantum states, via doubling:



Classical and quantum interaction

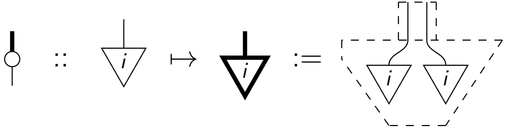
Classical values can be encoded as quantum states, via doubling:



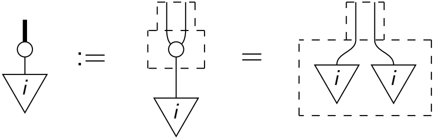
This is our first classical-quantum map, *encode*.

Classical and quantum interaction

Classical values can be encoded as quantum states, via doubling:



This is our first classical-quantum map, *encode*. It's a copy-spider in disguise:

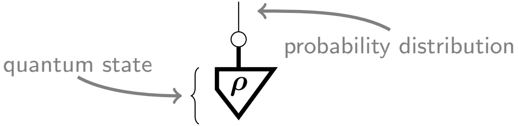


Measuring quantum states

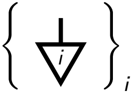
The adjoint of *encode* is *measure*:

Measuring quantum states

The adjoint of *encode* is *measure*:

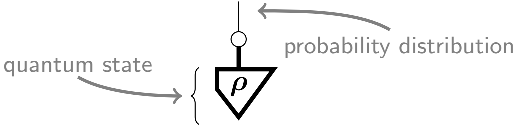


This represents measuring w.r.t.

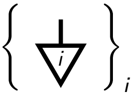


Measuring quantum states

The adjoint of *encode* is *measure*:



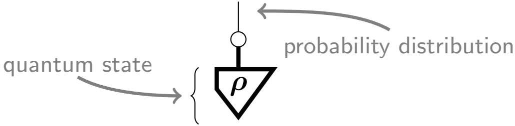
This represents measuring w.r.t.



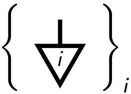
...where probabilities come from the Born rule:

Measuring quantum states

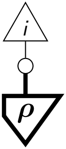
The adjoint of *encode* is *measure*:



This represents measuring w.r.t.

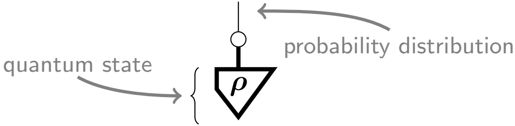


...where probabilities come from the Born rule:

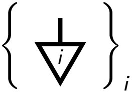
$P(i|\rho) :=$ 

Measuring quantum states

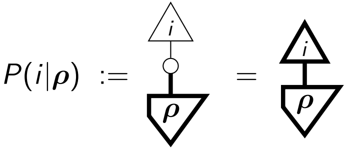
The adjoint of *encode* is *measure*:



This represents measuring w.r.t.



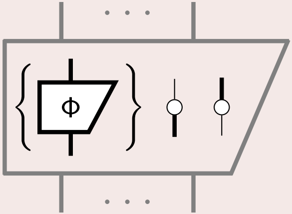
...where probabilities come from the Born rule:



Classical-quantum maps

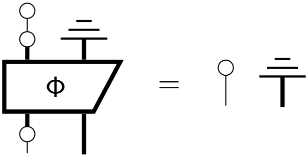
Definition

The process theory of **cq-maps** has as processes diagrams of quantum maps and encode/decode:



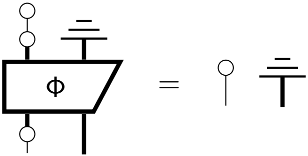
Quantum processes

Causality generalises to cq-maps:



Quantum processes

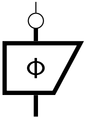
Causality generalises to cq-maps:



quantum processes := causal **cq-maps**

Special case: quantum measurements

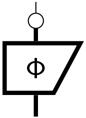
A *measurement* is any **quantum process** from a quantum system to a classical one:



$\xleftrightarrow{\cong}$ POVMs

Special case: quantum measurements

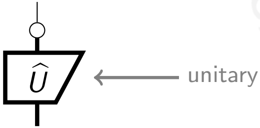
A *measurement* is any **quantum process** from a quantum system to a classical one:



$\overset{\cong}{\longleftrightarrow}$ POVMs

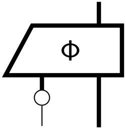
Special case:

ONB-measurement $:=$



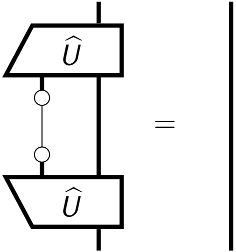
Special case: controlled-operations

A **quantum process** with a classical input is a *controlled operation*:



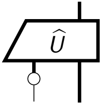
Special case: controlled-operations

A *controlled isometry* furthermore satisfies:



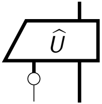
Special case: controlled-operations

Suppose we can use a single $\widehat{U}$ to build a *controlled isometry*:

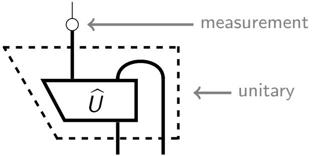


Special case: controlled-operations

Suppose we can use a single $\widehat{U}$ to build a *controlled isometry*:

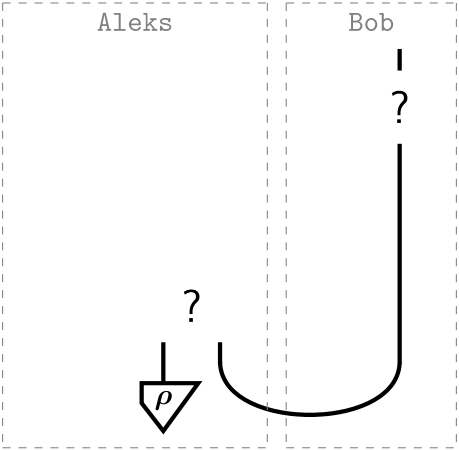


...and an ONB measurement:



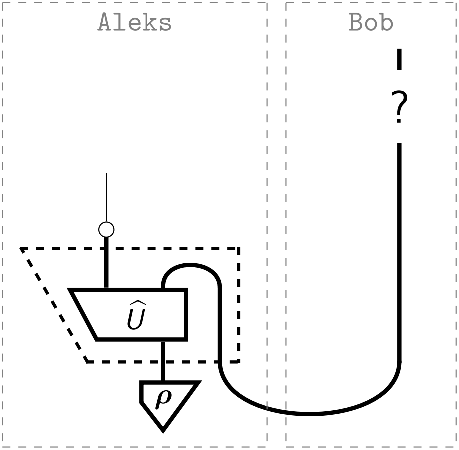
Quantum teleportation: take 2

...then teleportation is a snap!



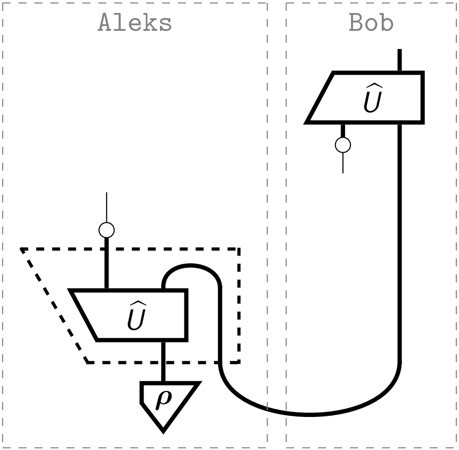
Quantum teleportation: take 2

...then teleportation is a snap!



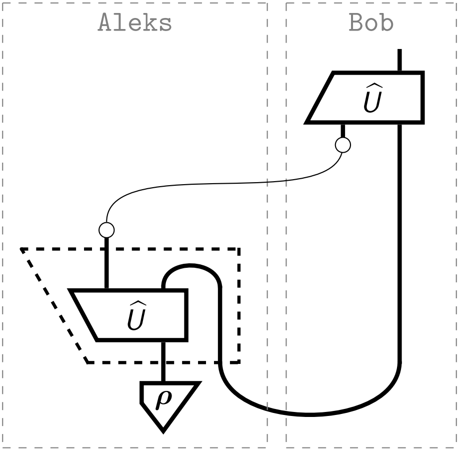
Quantum teleportation: take 2

...then teleportation is a snap!



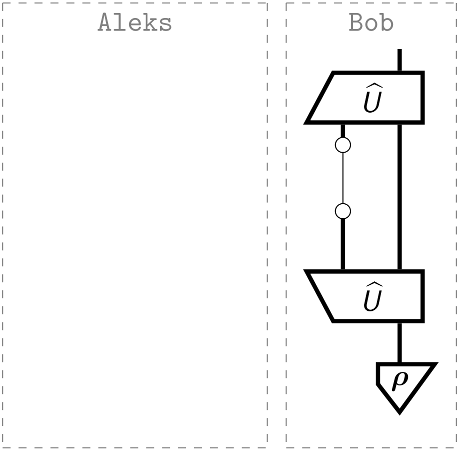
Quantum teleportation: take 2

...then teleportation is a snap!



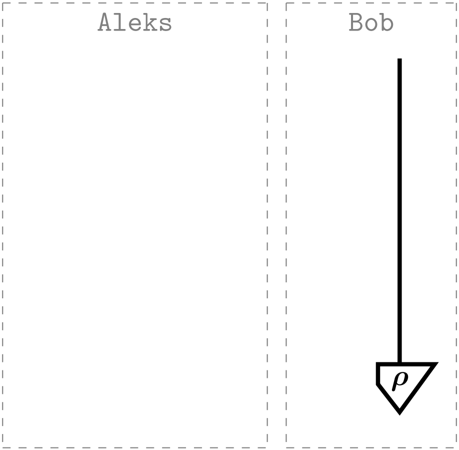
Quantum teleportation: take 2

...then teleportation is a snap!

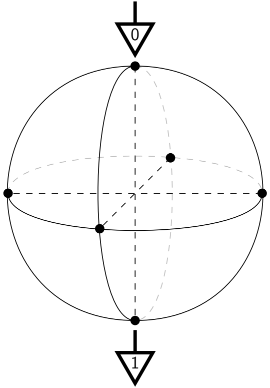


Quantum teleportation: take 2

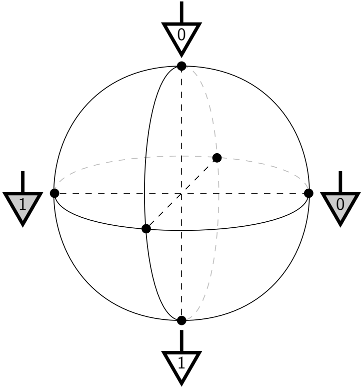
...then teleportation is a snap!



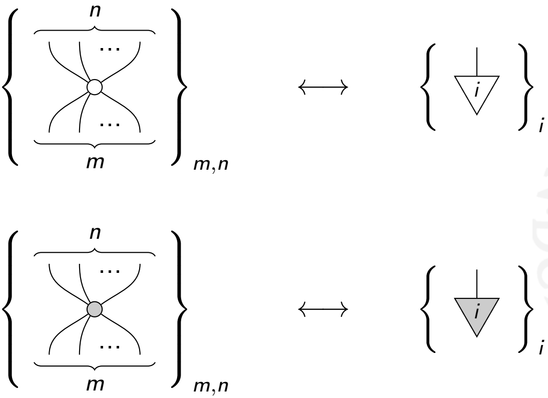
Complementary bases



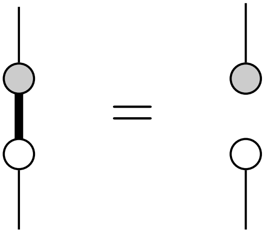
Complementary bases



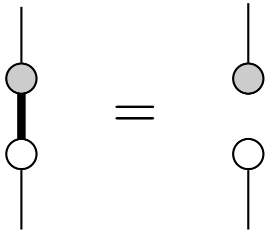
Complementary bases



Complementarity



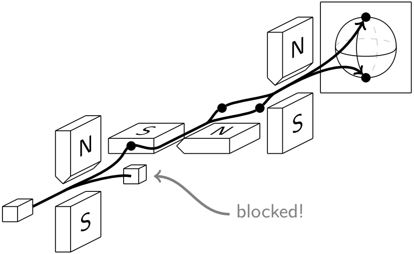
Complementarity

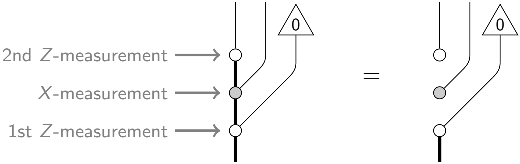


Interpretation:

(encode in ○) THEN (measure in ●) = (no data flow)

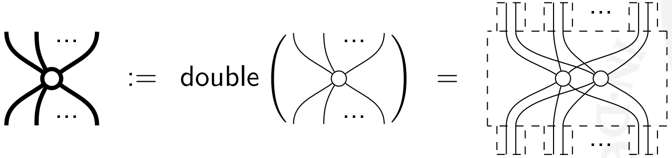
Consequence: Stern-Gerlach





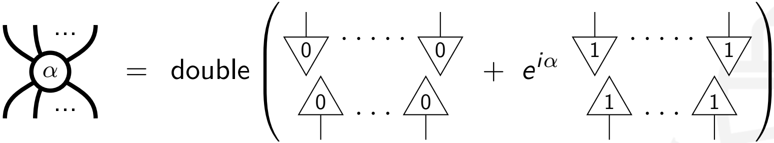
Quantum computation

Doubling a classical spider gives a *quantum spider*:



Universality

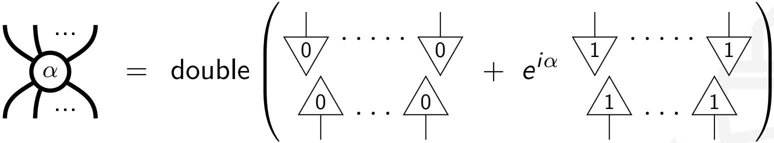
By decorating quantum spiders with *phases*:



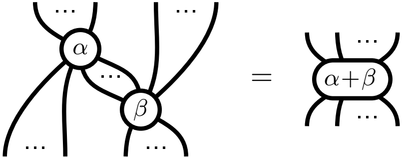
Radboud University Nijmegen

Universality

By decorating quantum spiders with *phases*:



we have:



and spider-diagrams become **universal for quantum computation**!

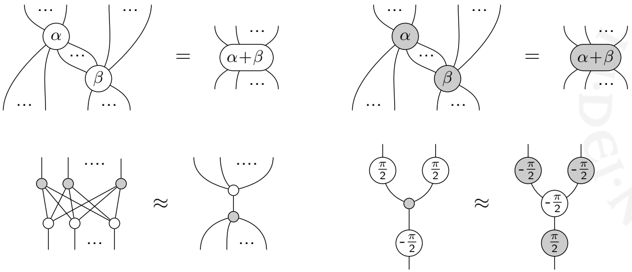
Soundness and completeness

Restricting the phase group to $\mathbb{Z}_4 \cong \{0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}\} \subset U(1)$ gives *stabiliser QT*.

Soundness and completeness

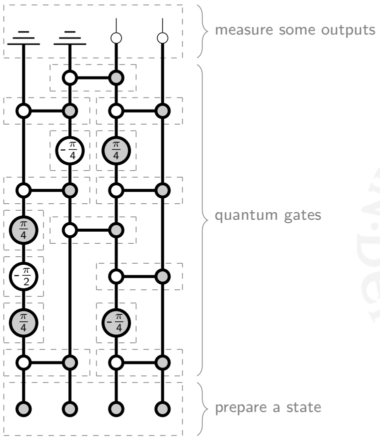
Restricting the phase group to $\mathbb{Z}_4 \cong \{0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}\} \subset U(1)$ gives *stabiliser QT*.

Sound and complete presentation via the **ZX-calculus**:

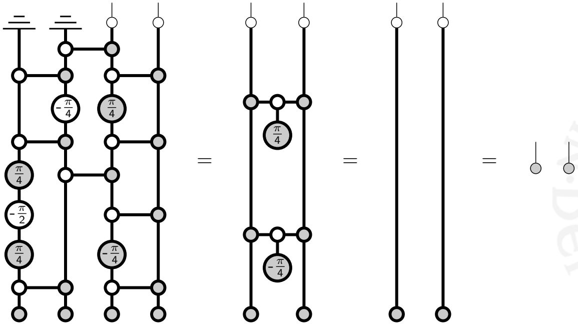


Outline

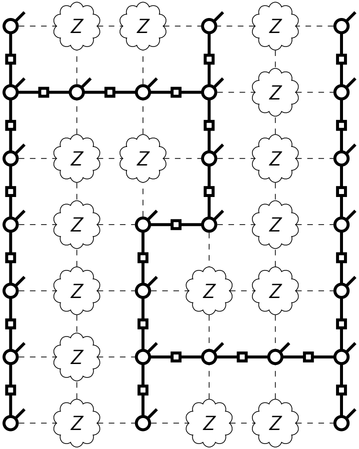
Quantum circuits and rewriting



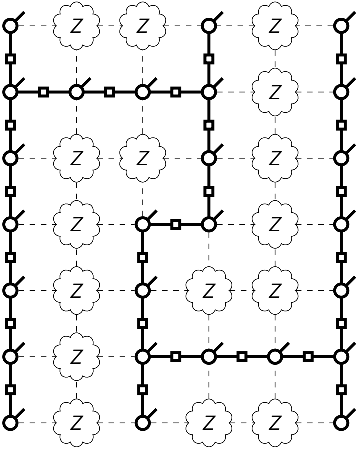
Quantum circuits and rewriting

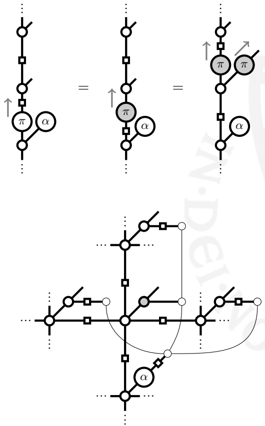


Measurement-based quantum computing



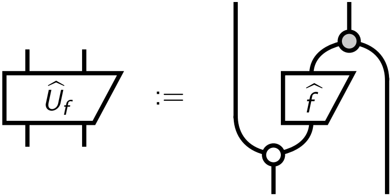
Measurement-based quantum computing





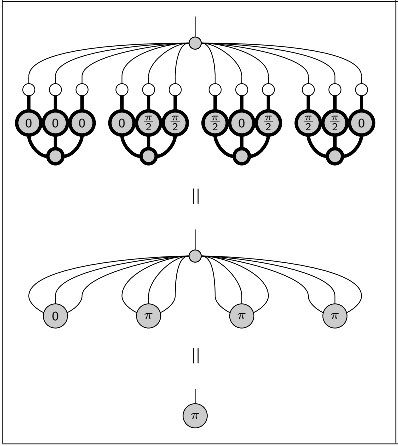
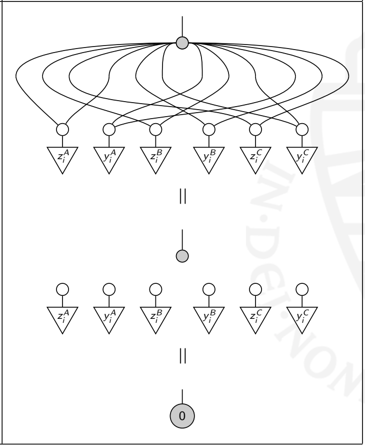
Quantum algorithms

Spiders can be used to build *quantum oracles*:



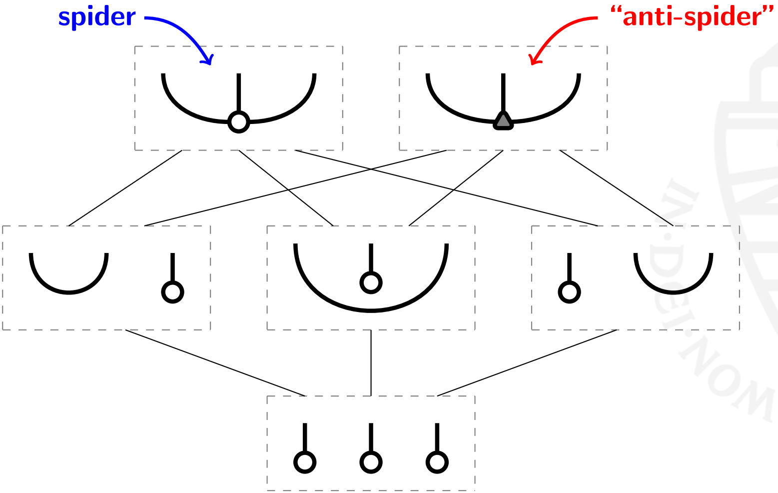
⇒ simple derivations of **Deutsch-Jozsa**, **quantum seach**, and **hidden subgroup** algorithms.

GHZ/Mermin non-locality

quantum theory	any local theory
	

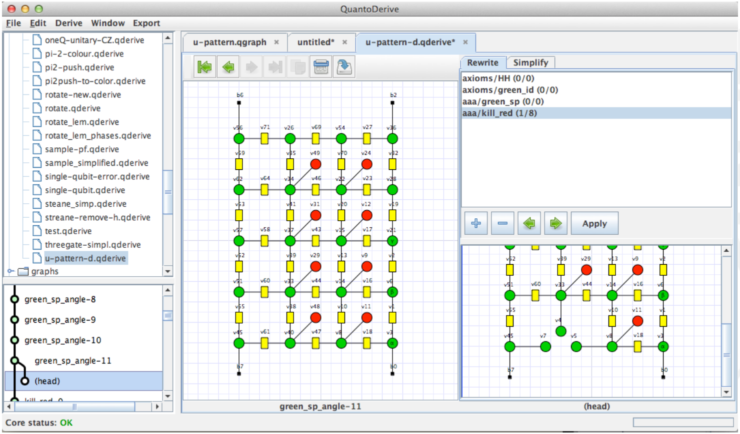
Multipartite entanglement

SLOCC-classification of 3 qubits:



Automation

Quantomatic:



Thanks! Joint work with Bob Coecke (book):

...and many more!

Abramsky, Backens, Duncan, Edwards, Gogioso, Hadzihasanovic, Heunen, Lal, Merry, Pavlovic, Perdrix, Quick, Selinger, Vicary, Zamdzhiev, ...

http://quantomatic.github.io