The Logic of Quantum Mechanics - take II

Bob Coecke – Oxford Univ. Computing Lab. – Quantum Group

INTRODUCTION
[von Neumann 1932] Formalized quantum mechanics in “Mathematische Grundlagen der Quantenmechanik”
[von Neumann 1932] Formalized quantum mechanics in “Mathematische Grundlagen der Quantenmechanik”

[von Neumann to Birkhoff 1935] “I would like to make a confession which may seem immoral: I do not believe absolutely in Hilbert space no more.” (sic)
[von Neumann 1932] Formalized quantum mechanics in “Mathematische Grundlagen der Quantenmechanik”

[von Neumann to Birkhoff 1935] “I would like to make a confession which may seem immoral: I do not believe absolutely in Hilbert space no more.” (sic)

— genesis —

[von Neumann 1932] Formalized quantum mechanics in “Mathematische Grundlagen der Quantenmechanik”

[von Neumann to Birkhoff 1935] “I would like to make a confession which may seem immoral: I do not believe absolutely in Hilbert space no more.” (sic)


[1936 – 2000] many followed them, ...
— genesis —

[von Neumann 1932] Formalized quantum mechanics in “Mathematische Grundlagen der Quantenmechanik”

[von Neumann to Birkhoff 1935] “I would like to make a confession which may seem immoral: I do not believe absolutely in Hilbert space no more.” (sic)


[1936 – 2000] many followed them, ... and FAILED.
the mathematics of it
Hilbert space stuff: continuum, field structure of complex numbers, vector space over it, inner-product, etc.
— the mathematics of it —

Hilber space stuff: continuum, field structure of complex numbers, vector space over it, inner-product, etc.

WHY?
— the mathematics of it —

**Hilber space stuff**: continuum, field structure of complex numbers, vector space over it, inner-product, etc.

**WHY?**

**von Neumann**: only used *it* since *it* was ‘available’.
— the mathematics of it —

**Hilber space stuff**: continuum, field structure of complex numbers, vector space over it, inner-product, etc.

**WHY?**

**von Neumann**: only used *it* since *it* was ‘available’.

**Model theory**: one can do almost anything with *it*.  
— the mathematics of it —

Hilber space stuff: continuum, field structure of complex numbers, vector space over it, inner-product, etc.

WHY?

von Neumann: only used it since it was ‘available’.

Model theory: one can do almost anything with it.

⇒ it comes with little conceptual content ⇐
— the mathematics of it —

Hilber space stuff: continuum, field structure of complex numbers, vector space over it, inner-product, etc.

WHY?

von Neumann: only used it since it was ‘available’.

Model theory: one can do almost anything with it.

⇒ it comes with little conceptual content ⇐

SO WHAT ELSE SHOULD ONE USE THEN?
— the physics of it —
von Neumann crafted Birkhoff-von Neumann Quantum ‘Logic’ to capture the concept of superposition.
von Neumann crafted Birkhoff-von Neumann Quantum ‘Logic’ to capture the concept of superposition.

Schrödinger (1935): the stuff which is the true soul of quantum theory is how quantum systems compose.
von Neumann crafted Birkhoff-von Neumann Quantum ‘Logic’ to capture the concept of superposition.

Schrödinger (1935): the stuff which is the true soul of quantum theory is how quantum systems compose.

Quantum Computer Scientists: Schrödinger is right!
von Neumann crafted Birkhoff-von Neumann Quantum ‘Logic’ to capture the concept of superposition.

Schrödinger (1935): the stuff which is the true soul of quantum theory is how quantum systems compose.

Quantum Computer Scientists: Schrödinger is right!

75 years later Birkhoff-von Neumann quantum ‘logicians’ still fail to elegantly capture composition of quantum systems, ... nor much else neither, ...
the game plan
Task 0. Solve:
\[
\frac{\text{tensor product structure}}{\text{the other stuff}} = ???
\]
Task 0. Solve:

\[
\frac{\text{tensor product structure}}{\text{the other stuff}} = ???
\]

i.e. \textit{axiomatize “⊗” without reference to spaces.}
Task 0. Solve: \[
\frac{\text{tensor product structure}}{\text{the other stuff}} = ???
\]
i.e. \textit{axiomatize “⊗” without reference to spaces.}

Task 1. Investigate which assumptions (i.e. which structure) on \(⊗\) is needed to deduce \textit{physical phenomena}?
— the game plan —

Task 0. Solve:

\[
\begin{array}{c}
\text{tensor product structure} \\
\text{the other stuff}
\end{array} = ???
\]

i.e. **axiomatize “⊗” without reference to spaces.**

Task 1. Investigate which assumptions (i.e. which structure) on ⊗ is needed to deduce **physical phenomena**?

Task 2. Investigate whether such an “interaction structure” appear elsewhere in **“our classical reality”**
Outcome 1a:
Outcome 1a: “Sheer ratio of results to assumptions”

Outcome 1a: “Sheer ratio of results to assumptions” confirms that we are probing something very essential.

Outcome 1a: “Sheer ratio of results to assumptions” confirms that we are probing something very essential.

Outcome 1b: Exposing this structure has already helped to solve open problems elsewhere. (e.g. 2× ICALP’10)

Outcome 1a: “Sheer ratio of results to assumptions” confirms that we are probing something very essential.

Outcome 1b: Exposing this structure has already helped to solve open problems elsewhere. (e.g. 2× ICALP’10)

Outcome 1c: Framework is a simple intuitive (but rigorous) diagrammatic language,

Outcome 1a: “Sheer ratio of results to assumptions” confirms that we are probing something very essential.

Outcome 1b: Exposing this structure has already helped to solve open problems elsewhere. (e.g. 2× ICALP’10)

Outcome 1c: Framework is a simple intuitive (but rigorous) diagrammatic language, meanwhile adopted by others e.g. Lucien Hardy in arXiv:1005.5164:

“... we join the quantum picturalism revolution [1]”

Outcome 2a:
Outcome 2a:

**Behaviors of matter** *(Abramsky-C; LiCS’04, quant-ph/0402130)*:

Meaning in language *(Clark-C-Sadrzadeh; Linguistic Analysis, arXiv:1003.4394)*:

Knowledge updating *(C-Spekkens; Synthese, arXiv:1102.2368)*:
the logic of it
WHAT IS “LOGIC”?
WHAT IS “LOGIC”? 

Definitely not Birkhoff-von Neumann quantum logic, which, by definition, is ‘logic without deduction’,
Definitely not Birkhoff-von Neumann quantum logic, which, by definition, is ‘logic without deduction’, nor is it some metaphysical jargon (cf. Putnam, Omnés, ...) aiming to solve the measurement problem.
WHAT IS “LOGIC”?  
Pragmatic option 1: Logic is structure in language.
— the logic of it —

WHAT IS “LOGIC”?

Pragmatic option 1: Logic is structure in language.

“Alice and Bob ate everything or nothing, then got sick.”

connectives ($\land, \lor$): and, or
negation ($\neg$): not (cf. nothing = not something)
entailment ($\Rightarrow$): then
quantifiers ($\forall, \exists$): every(thing), some(thing)
constants ($a, b$): thing
variable ($x$): Alice, Bob
predicates ($P(x), R(x, y)$): eating, getting sick
truth valuation (0, 1): true, false

($\forall z : Eat(a, z) \land Eat(b, z)) \land \neg(\exists z : Eat(a, z) \land Eat(b, z)) \Rightarrow Sick(a), Sick(b)$
WHAT IS “LOGIC”? 

Pragmatic option 1: Logic is structure in language.

Pragmatic option 2: Logic lets machines reason.
WHAT IS “LOGIC”?

Pragmatic option 1: Logic is structure in language.

Pragmatic option 2: Logic lets machines reason.

Cf. the soft incarnation of AI in robotics, automated theorem proving, automated theory exploration, ...
WHAT IS “LOGIC”?  

Pragmatic option 1: Logic is structure in language. 

Pragmatic option 2: Logic lets machines reason. 

Our framework appeals to both senses of logic,
WHAT IS “LOGIC”?

Pragmatic option 1: Logic is structure in language.

Pragmatic option 2: Logic lets machines reason.

Our framework appeals to both senses of logic, and moreover induces important new applications:

From truth to meaning in natural language processing: [NewScientist](http://www.newscientist.com) (December 2010)
WHAT IS “LOGIC”?

Pragmatic option 1: Logic is structure in language.

Pragmatic option 2: Logic lets machines reason.

Our framework appeals to both senses of logic, and moreover induces important new applications:

From truth to meaning in natural language processing:

— NewScientist (December 2010)

Automated theorem generation for graphical theories:

— http://sites.google.com/site/quantomatic/

the data of tensor logic

Box := \[ f \]

\[ \begin{array}{l}
\text{output wire(s)} \\
\text{input wire(s)} \\
\end{array} \]
the data of tensor logic

Box := \( f \)

Interpretation: wire := system ; box := process
— the data of tensor logic —

Box := $f$

**Interpretation:** wire := system ; box := process

one system: \( n \) subsystems: no system:

1

\( n \)

0
— the connectives of tensor logic —
the connectives of tensor logic

sequential or causal or connected composition:

\[ g \circ f \equiv \]

[Diagram of sequential or causal or connected composition]
the connectives of tensor logic

sequential or causal or connected composition:

$$g \circ f \equiv \begin{array}{c} \hline g \end{array} \begin{array}{c} \hline f \end{array}$$

parallel or acausal or disconnected composition:

$$f \otimes g \equiv \begin{array}{c} \hline f \end{array} \begin{array}{c} \hline g \end{array}$$
merely a new notation?
merely a new notation?

\[(g \circ f) \otimes (k \circ h) = (g \otimes k) \circ (f \otimes h)\]
merely a new notation?

\[(g \circ f) \otimes (k \circ h)\]
merely a new notation?

\[(g \otimes k) \circ (f \otimes h)\]
merely a new notation?

\[(g \circ f) \otimes (k \circ h) = (g \otimes k) \circ (f \otimes h)\]
merely a new notation?

\[(g \circ f) \otimes (k \circ h) = (g \otimes k) \circ (f \otimes h)\]

peel potato and then fry it, while, clean carrot and then boil it = peel potato while clean carrot, and then, fry potato while boil carrot
topological connectedness as a paradigm

classified as connected

classified as disconnected
topological connectedness as a paradigm

This is a digital paradigm:
This is a digital paradigm:

\[ 0 \sim \text{no information-flow} \quad 1 \sim \text{information-flow} \]
topological connectedness as a paradigm

dot =
ground =

mixedness
decoherence
eigenstate

complementarity

GHZ-state vs W-state
A MINIMAL LANGUAGE
FOR QUANTUM REASONING


--- graphical notation for states and effects ---
— graphical notation for states and effects —

\[ \psi : \text{I} \to \text{A} \]
\[ \pi : \text{A} \to \text{I} \]
\[ \pi \circ \psi : \text{I} \to \text{I} \]
graphical notation for states and effects
graphical notation for states and effects
graphical notation for states and effects

$\psi: I \rightarrow A$

$\pi: A \rightarrow I$

$\pi \circ \psi: I \rightarrow I$
graphical notation for states and effects
graphical notation for states and effects
graphical notation for states and effects
— graphical notation for states and effects —

ψ: I → A
π: A → I
π ◦ ψ: I → I
graphical notation for states and effects

\[ \psi : I \rightarrow A \]

\[ \pi : A \rightarrow I \]

\[ \pi \circ \psi : I \rightarrow I \]
--- adjoint ---

\[ f : A \rightarrow B \]
adjoint

\[ f^\dagger : B \rightarrow A \]
asserting (pure) entanglement

\[ \frac{\text{quantum}}{\text{classical}} = \frac{\text{\[ diagram \]}}{\text{\[ diagram \]}} = \]
--- asserting (pure) entanglement ---

\[
\frac{\text{quantum}}{\text{classical}} = \frac{\text{\hspace{1cm}}}{\text{\hspace{1cm}}} = \frac{\text{\hspace{1cm}}}{\text{\hspace{1cm}}} = \text{\hspace{1cm}}
\]

\[\Rightarrow \text{introduce ‘parallel connection between systems’:} \quad \bigcup \]
quantum-like

\[ \mu = \nu \]
quantum-like
sliding
classical data flow?
classical data flow?
⇒ quantum teleportation
symbolically: dagger compact categories
Thm. [Kelly-Laplaza ’80; Selinger ’05] An equational statement between expressions in dagger compact categorical language holds if and only if it is derivable in the graphical notation via homotopy.
Thm. [Kelly-Laplaza ’80; Selinger ’05] An equational statement between expressions in dagger compact categorical language holds if and only if it is derivable in the graphical notation via homotopy.

there are many models, i.e. concrete dagger compact categories, e.g. linear maps, relations, ...
Thm. [Kelly-Laplaza ’80; Selinger ’05] An equational statement between expressions in dagger compact categorical language holds if and only if it is derivable in the graphical notation via homotopy.

Thm. [Selinger ’08] An equational statement between expressions in dagger compact categorical language holds if and only if it is derivable in the dagger compact category of finite dimensional Hilbert spaces, linear maps, tensor product and adjoints.
In words: *Any equation involving:*

- states, operations, effects
- unitarity, adjoints (e.g. self-adjoint), projections
- Bell-states/effects, transpose, conjugation
- inner-product, trace, Hilbert-Schmidt norm
- positivity, completely positive maps, ...

*holds in quantum theory if and only if it can be derived in the graphical language via homotopy.*
MEANING IN NATURAL LANGUAGE

Consider meanings of **words** (for Google these are vectors):
How do we/machines compute meaning of sentences?
How do we/machines compute meaning of sentences?
Information flow within a verb:
Information flow within a verb:

\[ \text{object} \rightarrow \text{verb} \rightarrow \text{subject} \]

Again we have:

\[ \text{object} \neq \text{verb} \neq \text{subject} \]
\[ I \eta^l \to A \otimes A^l \quad A^l \otimes A \overset{\epsilon^l}{\to} I \quad I \eta^r \to A^r \otimes A \quad A \otimes A^r \overset{\epsilon^r}{\to} I \]

\[ \begin{array}{c}
A \quad A^l \\
A^l \quad A
\end{array} \quad \begin{array}{c}
A^r \quad A \\
A \quad A^r
\end{array} \quad \begin{array}{c}
A \quad A^l \\
A^l \quad A
\end{array} \quad \begin{array}{c}
A^r \quad A \\
A \quad A^r
\end{array} \]
For noun type “n”, verb type is “−1n · s · n−1”, so:

\[ n \cdot -1 n \cdot s \cdot n^{-1} \cdot n = s \]
For noun type “n”, verb type is “\(-1n \cdot s \cdot n^{-1}\)”, so:

\[ n \cdot -1n \cdot s \cdot n^{-1} \cdot n = s \]

Diagrammatic typing:
For noun type “n” and verb type “$-1n \cdot s \cdot n^{-1}$”:

$$n \cdot -1n \cdot s \cdot n^{-1} \cdot n = s$$

Diagrammatic meaning:
Alice does not like Bob.
Alice ⊗ does ⊗ not ⊗ like ⊗ Bob

meaning vectors of words

grammar
Alice $\otimes$ does $\otimes$ not $\otimes$ like $\otimes$ Bob

grammar

meaning vectors of words
Alice $\otimes$ does $\otimes$ not $\otimes$ like $\otimes$ Bob

meaning vectors of words
\[ \overrightarrow{\text{Alice}} \otimes \overrightarrow{\text{does}} \otimes \overrightarrow{\text{not}} \otimes \overrightarrow{\text{like}} \otimes \overrightarrow{\text{Bob}} \]

meaning vectors of words

grammer

\[ = \begin{array}{c}
\overrightarrow{\text{Alice}} \\
\text{not} \\
\overrightarrow{\text{like}} \\
\overrightarrow{\text{Bob}} \\
\overrightarrow{\text{like}}
\end{array} \]
— experiment: word disambiguation —

E.g. what is “saw” in: “Alice saw Bob with a saw”.

<table>
<thead>
<tr>
<th>Model</th>
<th>High</th>
<th>Low</th>
<th>(\rho)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>0.47</td>
<td>0.44</td>
<td>0.16</td>
</tr>
<tr>
<td>Add</td>
<td>0.90</td>
<td>0.90</td>
<td>0.05</td>
</tr>
<tr>
<td>Multiply</td>
<td>0.67</td>
<td>0.59</td>
<td>0.17</td>
</tr>
<tr>
<td>Categorical</td>
<td>0.73</td>
<td>0.72</td>
<td>0.21</td>
</tr>
</tbody>
</table>

Alice hates Bob
meaning vectors of words
grammar

measurements

states
--- analogy: “non-local” info-flows ---

English (& French):

Hindi:

Persian:

Arabic (and Hebrew):

**analogy: quantizing grammar!**

**Topological quantum field theory:**

\[ F : n\text{Cob} \rightarrow \text{FVect}_\mathbb{C} \mapsto V \]

**Grammatical quantum field theory:**

\[ F : \text{Pregroup} \rightarrow \text{FVect}_{\mathbb{R}^+} \mapsto V \]

Louis Crane was the first one to notice this analogy.
THE EXTENDED LANGUAGE:
BASES AND COMPLEMENTARITY


commutative Frobenius algebras
A commutative monoid is a set $X$ with a binary map

$$- \bullet - : X \times X \to X$$

which is commutative, associative and unital i.e

$$(a \bullet b) \bullet c = a \bullet (b \bullet c) \quad a \bullet b = b \bullet a \quad a \bullet 1 = a$$
commutative Frobenius algebras

A commutative monoid is a set $X$ with a binary map

$$\mu(-, -) : X \times X \to X$$

which is commutative, associative and unital i.e

$$\mu(\mu(a, b), c) = \mu(a, \mu(b, c)) \quad \mu(a, b) = \mu(b, a) \quad \mu(a, 1) = a$$
A commutative monoid is a set $X$ with a binary map

$$\mu : X \times X \to X$$

which is commutative, associative and unital i.e

$$\mu \circ (\mu \times 1_X) = \mu \circ (1_X \times \mu) \quad \mu = \mu \circ \sigma \quad \mu \circ (1_X \times e) = 1_X$$

with:

$$\sigma : X \times X \to X \times X :: (a, b) \mapsto (b, a)$$

$$e : \{\ast\} \to X :: \ast \mapsto 1$$
A **commutative monoid** is a set $X$ with a binary map

$$\mu : X \times X \rightarrow X$$

which is commutative, associative and unital i.e

$$\mu \circ (\mu \times 1_X) = \mu \circ (1_X \times \mu) \quad \mu = \mu \circ \sigma \quad \mu \circ (1_X \times e) = 1_X$$

A **cocomutative comonoid** is a set $X$ with a binary map

$$\delta : X \rightarrow X \times X$$

which is cocommutative, coassociative and counital i.e

$$(\delta \times 1_X) \circ \delta = (1_X \times \delta) \circ \delta \quad \delta = \sigma \circ \delta \quad (1_X \times e') \circ \delta = 1_X$$
<table>
<thead>
<tr>
<th>commutative monoid</th>
<th>cocommutative comonoid</th>
</tr>
</thead>
</table>
| $\oplus_{\mathbb{Z}_2}$ $\def\range{\left\{\right.} \begin{array}{ll}
(0, 0), (1, 1) & \mapsto 0 \\
(0, 1), (1, 0) & \mapsto 1 
\end{array}$ | $\Delta$ $\def\range{\left\{\right.} \begin{array}{ll}
0 & \mapsto (0, 0) \\
1 & \mapsto (1, 1) 
\end{array}$ |
| $\times_{\mathbb{B}}$ $\def\range{\left\{\right.} \begin{array}{ll}
(0, 0), (0, 1), (1, 0) & \mapsto 0 \\
(1, 1) & \mapsto 1 
\end{array}$ | |
|                   | .                         |
**commutative Frobenius algebras**

A commutative monoid is vect. space $V$ & lin. map

$$\mu : V \otimes V \rightarrow V$$

which is commutative, associative and unital i.e

$$\mu \circ (\mu \otimes 1_V) = \mu \circ (1_V \otimes \mu) \quad \mu = \mu \circ \sigma \quad \mu \circ (1_V \otimes e) = 1_V$$

A cocommutative comonoid is vect. space $V$ & lin. map

$$\delta : V \rightarrow V \otimes V$$

which is cocommutative, coassociative and counital i.e

$$(\delta \otimes 1_V) \circ \delta = (1_V \otimes \delta) \circ \delta \quad \delta = \sigma \circ \delta \quad (1_V \otimes e') \circ \delta = 1_V$$
<table>
<thead>
<tr>
<th>commutative monoid</th>
<th>cocommutative comonoid</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\hat{+}_{\mathbb{Z}_2} :: \begin{cases}</td>
<td>00\rangle,</td>
</tr>
<tr>
<td>$\hat{\times}_B :: \cdots$</td>
<td>$\hat{\times}^\dagger_B :: \cdots$</td>
</tr>
</tbody>
</table>

| $\delta^\dagger_Z :: \begin{cases} |00\rangle &\mapsto |0\rangle \\ |11\rangle &\mapsto |1\rangle \end{cases}$ | $\delta_Z :: \begin{cases} |0\rangle &\mapsto |00\rangle \\ |1\rangle &\mapsto |11\rangle \end{cases}$ |
| $\delta^\dagger_X :: \begin{cases} |++\rangle &\mapsto |++\rangle \\ |--\rangle &\mapsto |--\rangle \end{cases}$ | $\delta_X :: \begin{cases} |+\rangle &\mapsto |++\rangle \\ |--\rangle &\mapsto |--\rangle \end{cases}$ |
| $\delta^\dagger_Y :: \cdots$ | $\delta_Y :: \cdots$ |
A commutative monoid is:

\[
\begin{array}{c}
\begin{array}{c}
\begin{array}{c}
\begin{array}{c}
\text{such that}
\end{array}
\end{array}
\end{array}
\end{array}
\]

\[
\begin{array}{c}
\begin{array}{c}
\begin{array}{c}
\begin{array}{c}
\end{array}
\end{array}
\end{array}
\end{array}
\]

A cocomutative comonoid is:

\[
\begin{array}{c}
\begin{array}{c}
\begin{array}{c}
\begin{array}{c}
\text{such that}
\end{array}
\end{array}
\end{array}
\end{array}
\]

\[
\begin{array}{c}
\begin{array}{c}
\begin{array}{c}
\begin{array}{c}
\end{array}
\end{array}
\end{array}
\end{array}
\]

---

*commutative Frobenius algebras* ---
— commutative Frobenius algebras —

A commutative Frobenius algebra is a commutative monoid and a commutative comonoid:

which moreover satisfy:
<table>
<thead>
<tr>
<th>commutative monoid</th>
<th>cocommutative comonoid</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \hat{+} \mathbb{Z}_2 :: \begin{cases}</td>
<td>00\rangle \mapsto</td>
</tr>
<tr>
<td>( \hat{\delta}^\dagger \mathbb{Z} :: \begin{cases}</td>
<td>+\rangle \mapsto</td>
</tr>
<tr>
<td>( \hat{\delta}_Y :: \cdots )</td>
<td>( \hat{\delta}_Y :: \cdots )</td>
</tr>
</tbody>
</table>
The notion of an (orthonormal) basis can be captured purely in the language of tensor logic:

**Thm.** *Dagger special* commutative Frobenius algebras:

\[
\begin{align*}
\begin{tikzpicture}[baseline=-.5ex,thick,xscale=1,yscale=1] 
\draw (0,0) -- (0,.5) -- (1,.5) -- (1,0); 
\end{tikzpicture} &= \left( \begin{tikzpicture}[baseline=-.5ex,thick,xscale=1,yscale=.5] 
\draw (0,0) -- (0,.5) -- (1,.5) -- (1,0); 
\end{tikzpicture} \right)^\dagger \\
\begin{tikzpicture}[baseline=-.5ex,thick,xscale=1,yscale=1] 
\draw (0,0) -- (0,.5) -- (1,.5) -- (1,0); 
\end{tikzpicture} &= \left( \begin{tikzpicture}[baseline=-.5ex,thick,xscale=1,yscale=1] 
\draw (0,0) -- (0,.5) -- (1,.5) -- (1,0); 
\end{tikzpicture} \right)^\dagger \\
\begin{tikzpicture}[baseline=-.5ex,thick,xscale=1,yscale=1] 
\draw (0,0) -- (0,.5) -- (1,.5) -- (1,0); 
\end{tikzpicture} &= \begin{tikzpicture}[baseline=-.5ex,thick,xscale=1,yscale=1] 
\draw (0,0) -- (0,.5) -- (1,.5) -- (1,0); 
\end{tikzpicture}
\end{align*}
\]

are the same thing as *orthonormal bases*; the basis vectors are the vectors copied by the comultiplication.

---

CFA normal form depends on \#inputs, \#outputs, \#loops:
such that, for $k > 0$: 

\[
\begin{align*}
\{ m+n'-k \} & \quad \{ m+m'-k \} \\
\{ n+n'-k \} & \quad \{ m+n-k \} = \{ m+m-k \} \\
\end{align*}
\]
\[ '\text{(co-)}\text{mult.}' = \left\{ \begin{array}{ccc} m \\ n \end{array} \right\} \]

such that, for \( k > 0 \):

\[ m + m' - k \]
\[ n + n' - k \]
such that, for $k > 0$:
spiders

\[
\text{‘cups/caps’} = \left\{ \begin{array}{c}
m \\
n \end{array} \right\}
\]

such that, for \( k > 0 \):

\[
\begin{align*}
m + m' &- k \\ n + n' &- k
\end{align*}
\]
complementary bases
Thm.

--- complementary bases ---

**quantum gates**

**Z-spin:**

\[ \delta_Z : |i\rangle \leftrightarrow |ii\rangle \]

**X-spin:**

\[ \delta_X : |\pm\rangle \leftrightarrow |\pm\pm\rangle \]
i.e.

\[(\delta^\dagger_Z \otimes 1) \circ (1 \otimes \delta_X) = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0
\end{pmatrix} = CNOT\]
quantum gates

\[
\begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 \\
\end{pmatrix}
\circ
\begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 \\
\end{pmatrix}
= ?
\]
quantum gates
quantum gates
quantum gates
quantum gates
classical communication and control
classical communication and control
classical communication and control
classical communication and control
classical communication and control
classical communication and control
Translating measurement based quantum computations to circuits without additional resources:

Example 18. The ubiquitous CNOT operation can be computed by the pattern
\[ P = X_{34} Z_{24} Z_{21} M_{03} M_{02} E_{13} E_{23} E_{34} N_{34} \ldots \]

Definition 21. Given a geometry \( \Gamma = ((V, E), I, O) \) we can define a diagram \( D_{\Gamma} = ((V_D, E_D), I_D, O_D) \) as follows:

Spekkens’ toy qubit qubit theory and stabilizer quantum theory within one framework.

(Non-)locality can be reduced to the ‘phase group’:

\[
\frac{\text{Spekkens’ qubit QM}}{\text{stabilizer qubit QM}} = \frac{\mathbb{Z}_2 \times \mathbb{Z}_2}{\mathbb{Z}_4} = \frac{\text{local}}{\text{non-local}}
\]

---

— automated theory exploration —

Overview

Recent graph-based formalisms of quantum computation provide an abstract and symbolic way to represent and simulate computations. However, manual manipulation of such graphs is slow and error-prone. This project employs a formalism, based on compact closed categories, that supports mechanised reasoning about such graphs. This gives a compositional account of graph rewriting that preserves the underlying categorical semantics.

Using this representation, we are developing a generic system with a fixed logical kernel that supports reasoning about models of compact closed category. A salient feature of the system is that it provides a formal and declarative account of derived results that can include ellipsis-style notation. The main application is to develop a graph-based language for reasoning about quantum computation: Quantomatic.

The System

We have an implementation built using libraries from IsaPlanner. This builds on PolyML, Java, and Graphviz/dot. To build and run Quantomatic, you need to be able to run the command: ant, java, javac, poly, and dot.

For Mac, you can install Graphviz/dot from the Graphviz download site. Once installed, test it by making sure that the dot command can be executed in a shell (that it is in the path). On Ubuntu/Debian Linux you quickly and easily install graphviz by typing:

```
sudo apt-get install graphviz
```

Assuming you have java, a recent java compiler, and graphviz installed, you can install everything else with the following shell commands:

```
### Make the Quanto directory, for all things quanto, and go there
mkdir Quanto; cd Quanto
### download and build PolyML
svn co https://polyml.eva.sourceforge.net/svnroot/polyml/fixes-5.3 polyml
cd polyml/polyml; ./configure --prefix=(cd ..; sed); make; make install; cd ..
### download and build Quantomatic
svn co https://isaplanner.eva.sourceforge.net/svnroot/isaplanner/trunk/isapl-lib isapl
svn co https://isaplanner.eva.sourceforge.net/svnroot/isaplanner/trunk/quantomatic quantomatic
cd quantomatic/glu; ant; cd ..
```
THE EXTENDED LANGUAGE II: 
GHZ vs W ENTANGLEMENT


Defn. Special CFA (SCFA):
**Defn.** Special CFA (SCFA):

\[
\begin{array}{c}
\text{\includegraphics[width=0.3\textwidth]{diagram.png}}
\end{array}
\]

**Lem.** Loop invertible $\Rightarrow$ special up to phase.
**Defn. Special CFA (SCFA):**

![Diagram of SCFA]

**Lem.** Loop invertible $\Rightarrow$ special up to to phase.

**Defn. Anti-special CFA (ACFA):**

![Diagram of ACFA]
**Defn.** Special CFA (SCFA):

\[ \text{Diagram of SCFA} \]

**Lem.** Loop invertible ⇒ special up to phase.

**Defn.** Anti-special CFA (ACFA):

\[ \text{Diagram of ACFA} \]

**Lem.** [Herrmann] Loop disconnected ⇒ antispecial.
\[ C(F)As \leftrightarrow \text{tripartite states} \]

A comultiplication & unit yield a tripartite state:
Theorem. If a CFA on $\mathbb{C}^2$ is special, that is,

$$\begin{array}{c}
\circ \quad = \\
\end{array}$$

it induces a GHZ-class Frobenius state, and vice versa.
**Theorem.** If a CFA on $\mathbb{C}^2$ is **special**, that is,

![Special Diagram](image)

it induces a **GHZ-class Frobenius state**, and vice versa.

**Theorem.** If a CFA on $\mathbb{C}^2$ is **anti-special**, that is,

![Anti-Special Diagram](image)

it induces a **W-class Frobenius state**, and vice versa.
GHZ algebra

\[ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \]

\[ \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 \end{pmatrix} \]

\[ \begin{pmatrix} 1 & 0 & 0 & 1 \end{pmatrix} = \langle 00 \rangle + \langle 11 \rangle \]
W algebra

\[
\begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
1 \\
0
\end{pmatrix}
\begin{pmatrix}
1 \\
0 \\
1 \\
0
\end{pmatrix}
\begin{pmatrix}
0 & 1 \\
0 & 1
\end{pmatrix}
\begin{pmatrix}
0 & 1 \\
1 & 0
\end{pmatrix}
\begin{pmatrix}
0 & 1 & 1 & 0
\end{pmatrix}
= \langle 01 \rangle + \langle 10 \rangle
**GHZ “spiders”**

**Data:**

\[
\begin{align*}
\{ & m \\
\{ & n \\
\} & | \quad n, m \in \mathbb{N}
\end{align*}
\]

**Rules:**

\[
\begin{align*}
m + m' - k & (\geq 1) \\
n + n' - k & (\geq 1) \\
\end{align*}
\]

\[
\begin{align*}
m + m' - k & (\geq 1) \\
n + n' - k & (\geq 1) \\
\end{align*}
\]
W “exploding spiders”

Data:

\[
\begin{cases}
m \\
n
\end{cases}
\text{, } \begin{array}{c}
, \\
\end{array}
| n, m \in \mathbb{N}
\]

Rules:

\[
\begin{array}{c}
m + m' - 1 \\
n + n' - 1
\end{array}
= \begin{array}{c}
m + m' - 1 \\
n + n' - 1
\end{array}
\]
**ACFA “exploding spiders”**

**Data:**

\[
\begin{cases}
  m \\
  n
\end{cases}, \quad n, m \in \mathbb{N}
\]

**Rules:**

\[
m + m' - k (\geq 2) \quad = \quad n + n' - k (\geq 2)
\]
GHZ-W interaction
**GHZ-W interaction**

**Informatics**
- Information flow vs no information flow

**Topology**
- Connected vs disconnected

**Algebra**
- \( g \circ f = g \) vs \( f \otimes g = f \)

**Entanglement**
- GHZ vs W
**GHZ-W interaction**

**Informatics**
- information flow vs no information flow

**Topology**
- connected vs disconnected

**Calculus**
- $\times$ vs $+$

**Entanglement**
- GHZ vs W

**Algebra**
- $g \circ f$ vs $f \otimes g$
\[
\begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 \\
\end{pmatrix}
\begin{pmatrix}
\begin{pmatrix}
1 \\
z \\
\end{pmatrix} \otimes \begin{pmatrix}
1 \\
z' \\
\end{pmatrix}
\end{pmatrix}
= 
\begin{pmatrix}
1 \\
z \times z' \\
\end{pmatrix}
\]

GHZ-mult.
\[
\begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 \\
\end{pmatrix}
\begin{pmatrix}
(1) \\
(z) \\
\end{pmatrix}
\otimes
\begin{pmatrix}
1 \\
(1' ) \\
\end{pmatrix}
= \begin{pmatrix}
1 \\
(z \times z') \\
\end{pmatrix}
\]

GHZ-mult.

\[
\begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 \\
\end{pmatrix}
\begin{pmatrix}
(1) \\
(z) \\
\end{pmatrix}
\otimes
\begin{pmatrix}
1 \\
(1' ) \\
\end{pmatrix}
= \begin{pmatrix}
1 \\
(z + z') \\
\end{pmatrix}
\]

W-mult.
\[ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \left( \begin{pmatrix} 1 \\ z \end{pmatrix} \otimes \begin{pmatrix} 1 \\ z' \end{pmatrix} \right) = \begin{pmatrix} 1 \\ z \times z' \end{pmatrix} \]

**GHZ-mult.**

\[ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix} \left( \begin{pmatrix} 1 \\ z \end{pmatrix} \otimes \begin{pmatrix} 1 \\ z' \end{pmatrix} \right) = \begin{pmatrix} 1 \\ z + z' \end{pmatrix} \]

**W-mult.**

Encoding states \( \begin{pmatrix} 1 \\ z \end{pmatrix} \) as \( z \) \( \Rightarrow \) **GHZ** = \( \times \), **W** = \( + \)
\[
\left( \begin{array}{cccc}
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 \\
\end{array} \right) \left( \left( \begin{array}{c}
1 \\
z \\
\end{array} \right) \otimes \left( \begin{array}{c}
1 \\
z' \\
\end{array} \right) \right) = \left( \begin{array}{c}
1 \\
z \times z' \\
\end{array} \right)
\]

\text{GHZ-mult.}

\[
\left( \begin{array}{cccc}
1 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 \\
\end{array} \right) \left( \left( \begin{array}{c}
1 \\
z \\
\end{array} \right) \otimes \left( \begin{array}{c}
1 \\
z' \\
\end{array} \right) \right) = \left( \begin{array}{c}
1 \\
z + z' \\
\end{array} \right)
\]

\text{W-mult.}

Encoding states \( \left( \begin{array}{c}
1 \\
z \\
\end{array} \right) \) as \( z \Rightarrow \text{GHZ} = \times, \text{W} = + \)

\( \Rightarrow \text{GHZ and W interact via distributivity} \leftrightarrow \)
challenge

What can we do with stuff of this kind:

Given that:

- GHZ/W-calculus encodes rational arithmetic
- We have automation support
Posters on this line of work:

- **Aleks Kissinger**: *Synthesising physical theories.*
- **Raymond Lal**: *Causal structure in categorical quantum mechanics.*
- **Johan Paulsson**: *Towards a diagrammatic representation of classical data.*

Also from the Oxford gang:

- **Shane Mansfield**: *Hardy’s non-locality paradox and other possibilistic non-locality conditions.*

Want to learn this kind of category theory? Try: