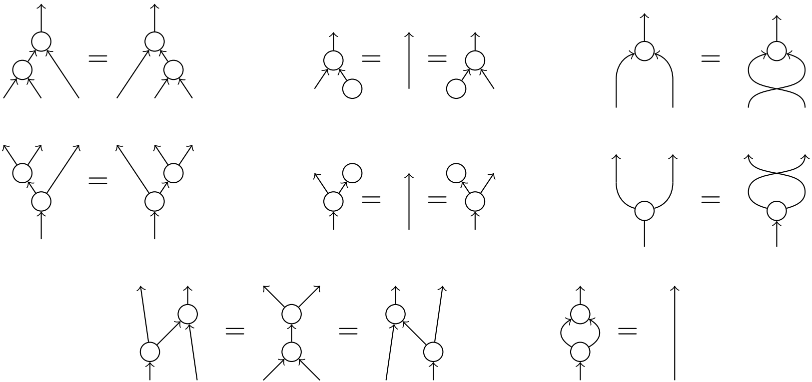
Simplification by Rotation for Frobenius/Hopf algebras

Aleks Kissinger

September 9, 2017

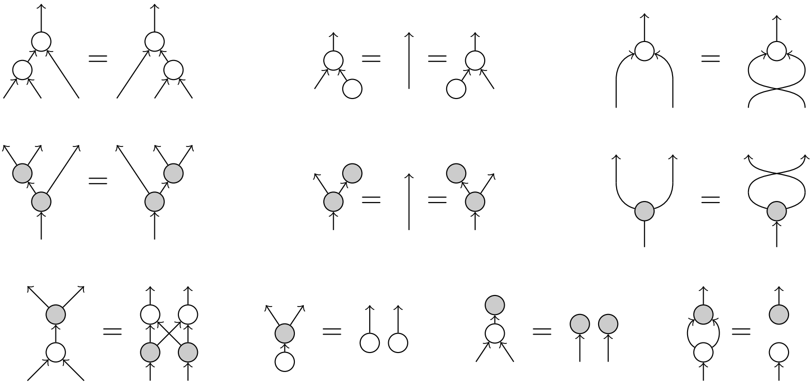
The goal

Simplification for special commutative Frobenius algebras:



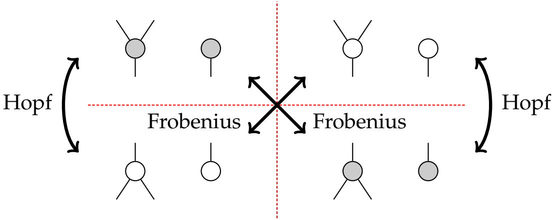
The goal

Simplification for commutative Hopf algebras:



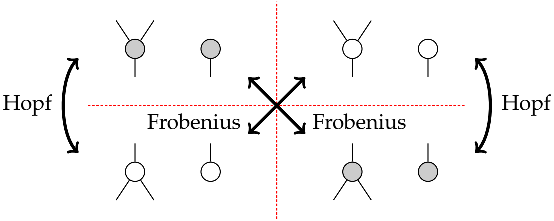
The goal

Simplification for the system $\mathbb{IB}$:



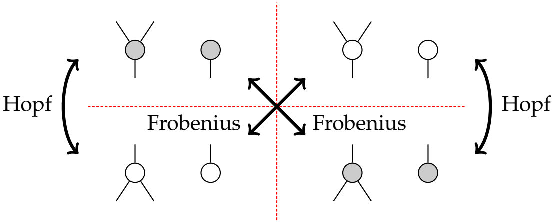
The goal

Simplification for the system $\mathbb{IB}$:



The goal

Simplification for the system $\mathbb{IB}$:



(a.k.a. the *phase-free fragment of the ZX-calculus*)

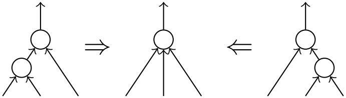
The (first) problem

- (Biased) AC rules are not terminating:

The (first) problem

- (Biased) AC rules are not terminating:

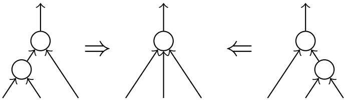
- **Solution:** use *unbiased* simplifications:



The (first) problem

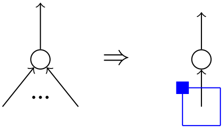
- (Biased) AC rules are not terminating:

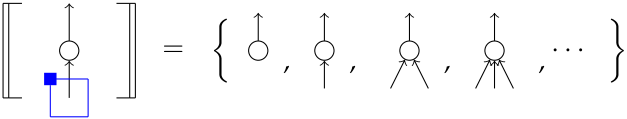
- **Solution:** use *unbiased* simplifications:



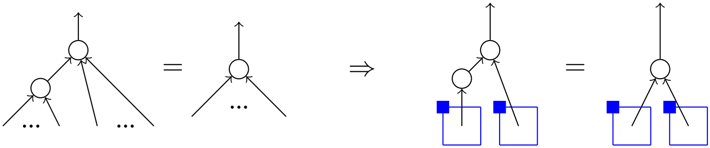
- $\implies$ need infinitely many rules, or *rule schemas*

!-boxes: simple diagram schemas

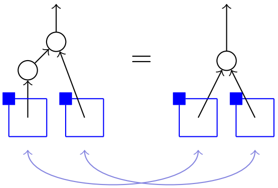




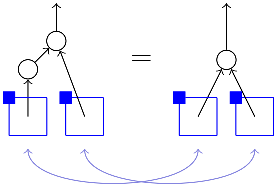
!-boxes: simple diagram rule schemas

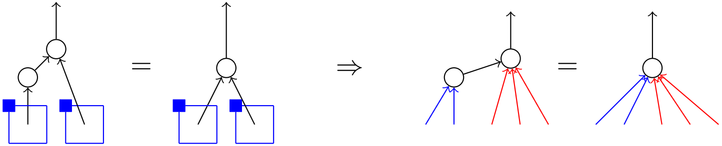


!-boxes

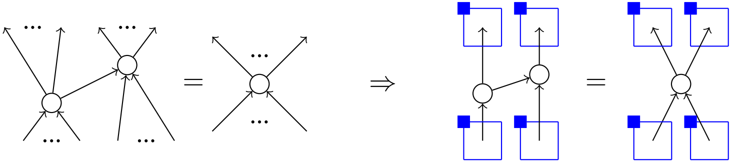


!-boxes

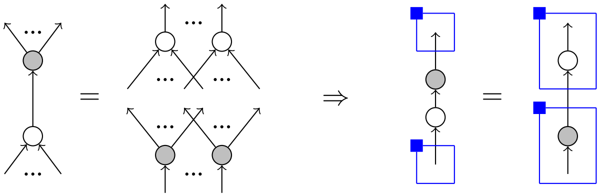




Unbiased Frobenius algebras



Unbiased bialgebras



To quanto!

Interacting bialgebras are linear relations

$$\mathbb{IB} \cong \mathrm{LinRel}_{\mathbb{Z}_2}$$

Interacting bialgebras are linear relations

$$\mathbb{IB} \cong \mathrm{LinRel}_{\mathbb{Z}_2}$$

- $\mathrm{LinRel}_{\mathbb{Z}_2}$ has:
    - **objects:** $\mathbb{N}$
    - **morphisms:** $R : m \to n$ is a subspace $R \subseteq \mathbb{Z}_2^m \times \mathbb{Z}_2^n$
    - **tensor** is $\oplus$, **composition** is relation-style

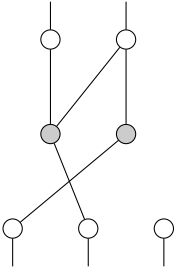
Interacting bialgebras are linear relations

$$\mathbb{IB} \cong \mathrm{LinRel}_{\mathbb{Z}_2}$$

- $\mathrm{LinRel}_{\mathbb{Z}_2}$ has:
  - **objects:** $\mathbb{N}$
  - **morphisms:** $R : m \to n$ is a subspace $R \subseteq \mathbb{Z}_2^m \times \mathbb{Z}_2^n$
  - **tensor** is $\oplus$, **composition** is relation-style
- Pseudo-normal forms can be interpreted as:
  - **white spiders** := place-holders
  - **grey spiders** := vectors spanning the subspace

Lets see how this works...

- Subspaces can be represented as:

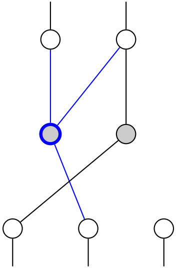


$$\leftrightarrow \quad \left\langle \begin{pmatrix} 0 \\ 1 \\ 0 \\ \hline 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 0 \\ \hline 0 \\ 1 \end{pmatrix} \right\rangle$$

- The 1′s indicate where edges appear for each vector.

Lets see how this works...

- Subspaces can be represented as:

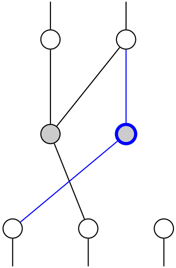


$$\leftrightarrow \quad \left\langle \begin{pmatrix} 0 \\ 1 \\ 0 \\ \hline 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 0 \\ \hline 0 \\ 1 \end{pmatrix} \right\rangle$$

- The 1′s indicate where edges appear for each vector.

Lets see how this works...

- Subspaces can be represented as:

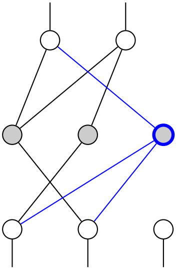


$$\leftrightarrow \quad \left\langle \begin{pmatrix} 0 \\ 1 \\ 0 \\ \hline 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 0 \\ \hline 0 \\ 1 \end{pmatrix} \right\rangle$$

- The 1′s indicate where edges appear for each vector.

Lets see how this works...

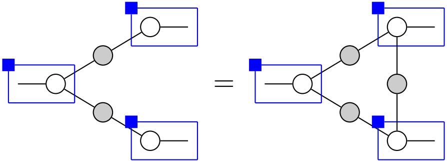
- Not unique! We can always add or remove a vector that is the sum of two other spanning vectors and get the same space:



$$\leftrightarrow \quad \left\langle \begin{pmatrix} 0 \\ 1 \\ 0 \\ \hline 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 0 \\ \hline 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \\ 0 \\ \hline 1 \\ 0 \end{pmatrix} \right\rangle$$

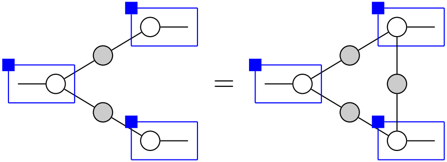
Addition is a !-box rule

- 'Addition' operation can be written as a !-box rule:

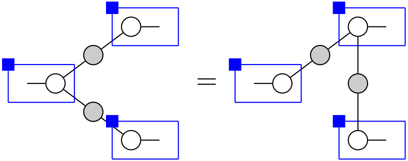


Addition is a !-box rule

- 'Addition' operation can be written as a !-box rule:

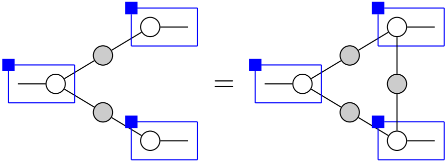


- We can also apply this forward then backward to get a 'rotation' rule:

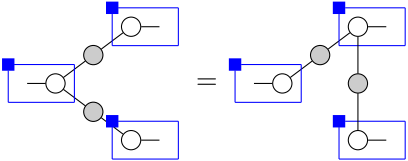


Addition is a !-box rule

- 'Addition' operation can be written as a !-box rule:



- We can also apply this forward then backward to get a 'rotation' rule:



- Note this rule decreases the arity of the white dot on the left by 1.

Thanks!

- Joint work with Lucas Dixon, Alex Merry, Ross Duncan, Vladimir Zamdzhiev, David Quick, Hector Miller-Bakewell and others
- See: `quantomatic.github.io`