

Tensors, \ast -graphs, and non-commutative quantum structures

Aleks Kissinger David Quick

QPL
June 2014

Table of Contents

Introduction

Tensor notation

Definitions

Induction

Summary

QuantoDerive

File Edit Derive Window

- TestDerive
 - axioms
 - derivations
 - graphs
 - rotate_lhs.qgraph
 - sample.qgraph
 - test.qgraph
 - threegates.qgraph
 - simprocs
 - theorems

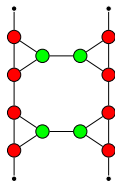
rotate_lhs.qgraph × sample.qgraph × test.qgraph × threegates.qgraph ×

Vertex Type: `X` Edge Type: `string` directed

Core status: OK

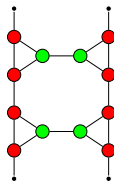
Defining Nodes

Given the graph:

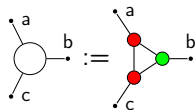


Defining Nodes

Given the graph:

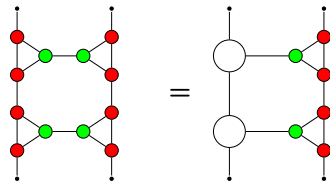


Could we define:

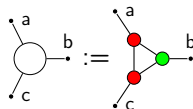


Defining Nodes

Given the graph:

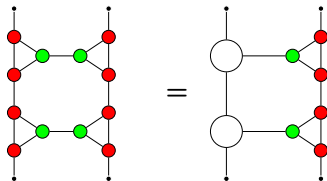


Could we define:

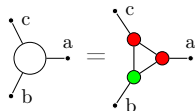
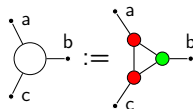


Defining Nodes

Given the graph:

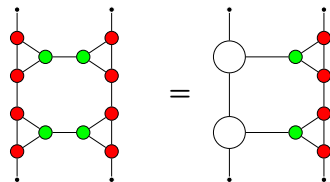


Could we define:

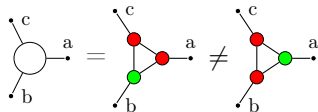
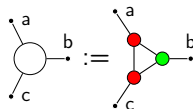


Defining Nodes

Given the graph:

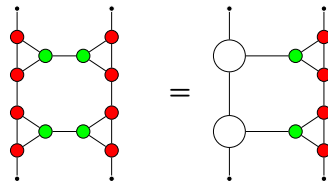


Could we define:

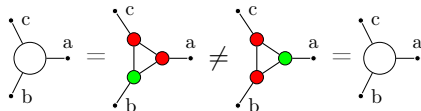
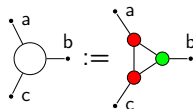


Defining Nodes

Given the graph:

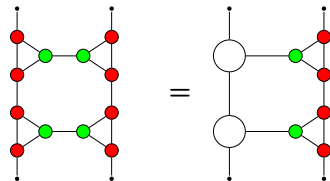


Could we define:

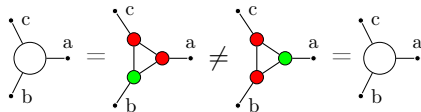
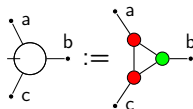


Defining Nodes

Given the graph:

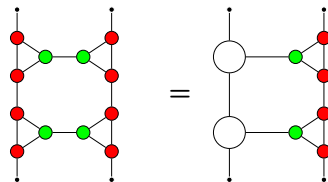


Could we define:

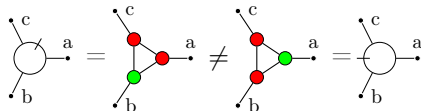
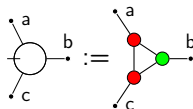


Defining Nodes

Given the graph:

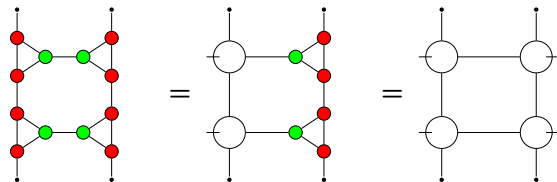


Could we define:

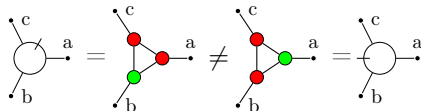
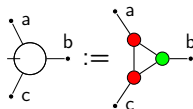


Defining Nodes

Given the graph:

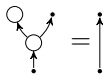
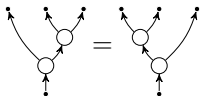
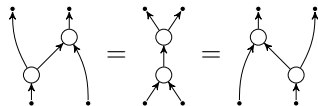
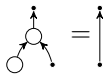
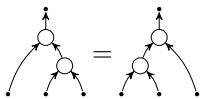


Could we define:



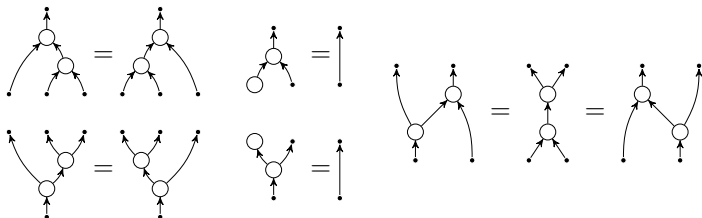
Recursively Defining Nodes

Given $\Sigma = \left\{ \begin{array}{c} \uparrow \\ \circ \\ \swarrow \quad \searrow \\ \bullet \quad \bullet \end{array} \right\}$, $\left\{ \begin{array}{c} \uparrow \\ \circ \end{array} \right\}$, $\left\{ \begin{array}{c} \uparrow \\ \swarrow \quad \searrow \\ \bullet \quad \bullet \end{array} \right\}$, $\left\{ \begin{array}{c} \uparrow \\ \circ \\ \uparrow \end{array} \right\}$ } satisfying:

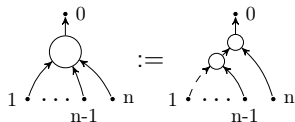


Recursively Defining Nodes


Given $\Sigma = \left\{ \begin{array}{c} \uparrow \\ \circ \\ \swarrow \uparrow \searrow \\ \circ \\ \swarrow \uparrow \searrow \\ \circ \\ \uparrow \end{array} \right\}$ satisfying:





We may want to define nodes of the form:




Recursively Defining Nodes


0 inputs: 


Recursively Defining Nodes

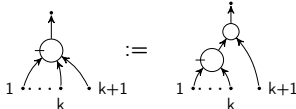
0 inputs:  := 

Given k inputs: 


Recursively Defining Nodes


0 inputs: 

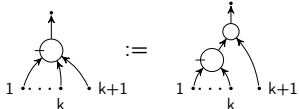
Given k inputs: 

define $k+1$ inputs: 

Recursively Defining Nodes

0 inputs: 

Given k inputs: 

define k+1 inputs: 

Recursive definitions suggest proof by induction. To do this we need to be more formal with variable arity nodes.

!-Boxes

Use !-boxes for multiple copies of a section from a graph.



!-Boxes

Use !-boxes for multiple copies of a section from a graph.



Then we have operations allowing deletion of !-boxes and creation of a new instance of the contents:

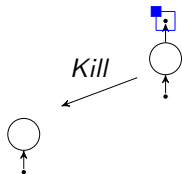


!-Boxes

Use !-boxes for multiple copies of a section from a graph.



Then we have operations allowing deletion of !-boxes and creation of a new instance of the contents:

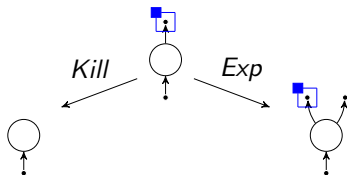


!-Boxes

Use !-boxes for multiple copies of a section from a graph.



Then we have operations allowing deletion of !-boxes and creation of a new instance of the contents:

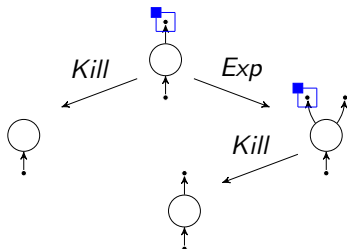


!-Boxes

Use !-boxes for multiple copies of a section from a graph.



Then we have operations allowing deletion of !-boxes and creation of a new instance of the contents:

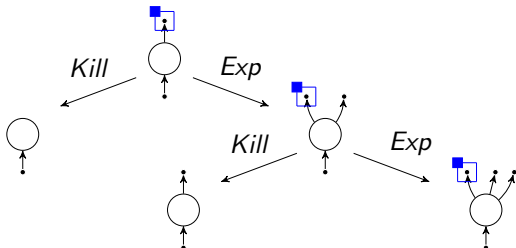


!-Boxes

Use !-boxes for multiple copies of a section from a graph.



Then we have operations allowing deletion of !-boxes and creation of a new instance of the contents:

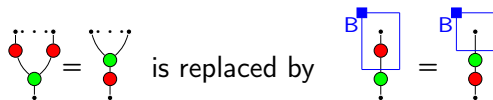


!-Box Equations

!-Boxes can be used in equations:

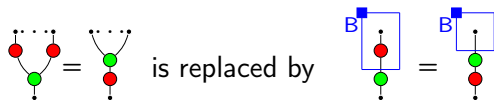
!-Box Equations

!-Boxes can be used in equations:

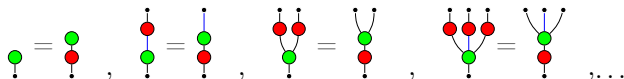


!-Box Equations

!-Boxes can be used in equations:

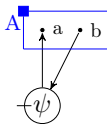


Which represents concrete equations like:



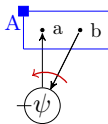
!-Box Expansion

This !-box (labelled A) has many different possible expansions:



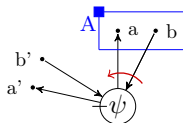
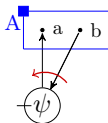
!-Box Expansion

This !-box (labelled A) has many different possible expansions:



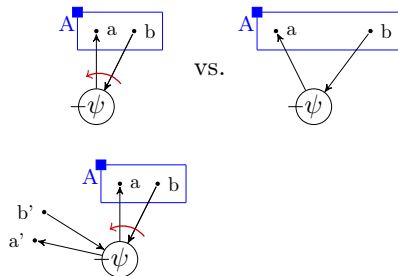
!-Box Expansion

This !-box (labelled A) has many different possible expansions:



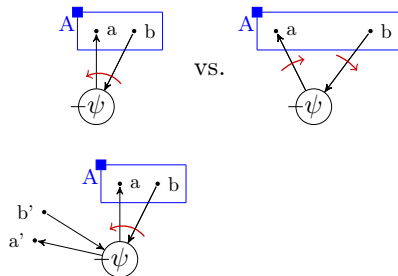
!-Box Expansion

This !-box (labelled A) has many different possible expansions:



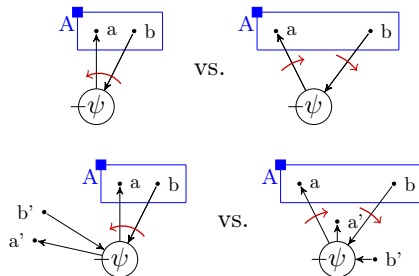
!-Box Expansion

This !-box (labelled A) has many different possible expansions:



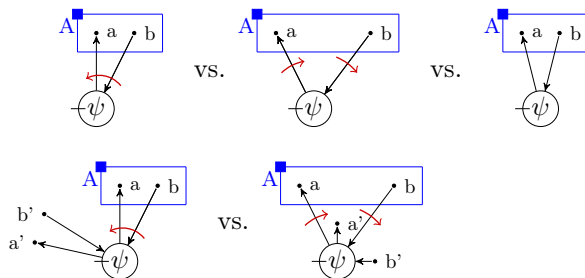
!-Box Expansion

This !-box (labelled A) has many different possible expansions:



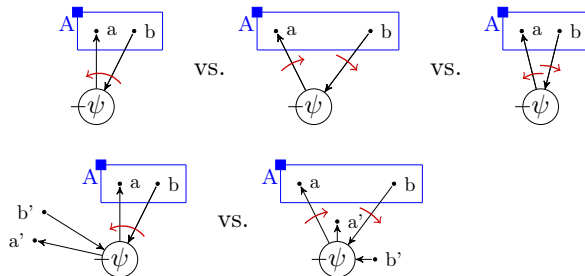
!-Box Expansion

This !-box (labelled A) has many different possible expansions:



!-Box Expansion

This !-box (labelled A) has many different possible expansions:



!-Box Expansion

This !-box (labelled A) has many different possible expansions:

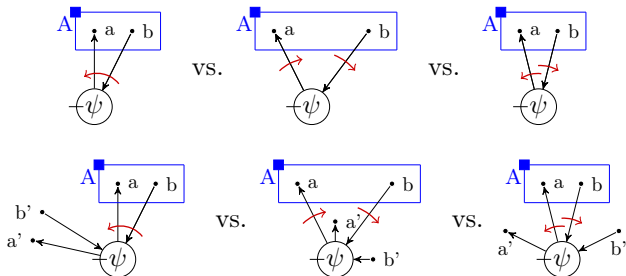


Table of Contents

Introduction

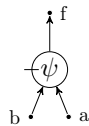
Tensor notation

Definitions

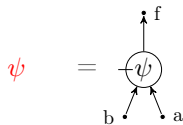
Induction

Summary

Building a Tensor



Building a Tensor



Building a Tensor

$$\psi_{fab} = \text{---} \circlearrowleft \psi \begin{matrix} \uparrow f \\ \swarrow b \searrow a \end{matrix}$$

The diagram shows the tensor ψ_{fab} represented as a circle containing the symbol ψ . Three arrows originate from the circle: one pointing upwards to the index f , one pointing down and to the left to the index b , and one pointing down and to the right to the index a . The index f is positioned above the circle, b is to the bottom-left, and a is to the bottom-right.

Building a Tensor

$$\psi_{f\check{a}\check{b}} = \text{---} \circlearrowleft \psi \begin{matrix} \uparrow f \\ \swarrow b \\ \searrow a \end{matrix}$$

The diagram shows a central circle containing the Greek letter ψ . Three arrows originate from the circle: one pointing upwards to the label f , one pointing down and to the left to the label b , and one pointing down and to the right to the label a . To the left of the circle is an equals sign, and further left is the text $\psi_{f\check{a}\check{b}}$, where the f is black and the \check{a} and \check{b} are red.

Building a Tensor

$$\psi_{\hat{f}\hat{a}\hat{b}} = \begin{array}{c} \bullet \\ \uparrow \\ \textcircled{\psi} \\ \swarrow \quad \searrow \\ \bullet \quad \bullet \\ \text{b} \quad \text{a} \end{array}$$

$$\phi_{\hat{a}\hat{b}\hat{c}\hat{d}\hat{e}} = \begin{array}{c} \bullet \quad \bullet \\ \uparrow \quad \uparrow \\ \textcircled{\phi} \\ \leftarrow \quad \rightarrow \\ \bullet \quad \bullet \\ \text{e} \quad \text{d} \end{array}$$

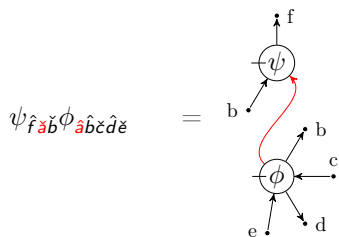
Building a Tensor

$$\psi_{\hat{f}\check{a}\check{b}}\phi_{\hat{a}\hat{b}\check{c}\hat{d}\check{e}}$$

=

The diagram shows two circular nodes, ψ and ϕ , connected by two internal lines. The ψ node has an outgoing line labeled f and two incoming lines labeled a and b . The ϕ node has an outgoing line labeled d and three incoming lines labeled a , b , and c . The lines labeled a and b connect the two nodes, with the a line on the left and the b line on the right.

Building a Tensor



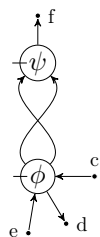
Building a Tensor

$$\psi_{\hat{f}\check{a}\check{b}}\phi_{\hat{a}\hat{b}\check{c}\hat{d}\check{e}} =$$

Building a Tensor

$$\psi_{\hat{f}\check{a}\check{b}}\phi_{\hat{a}\hat{b}\check{c}\hat{d}\check{e}}$$

=



The diagram shows two circular nodes, the top one labeled ψ and the bottom one labeled ϕ . The ψ node has an outgoing arrow labeled f at the top and an incoming arrow from the left. The ϕ node has an incoming arrow labeled c from the right, an outgoing arrow labeled d at the bottom right, and an incoming arrow labeled e at the bottom left. Two curved arrows connect the nodes: one from the left of ψ to the left of ϕ , and another from the right of ϕ to the right of ψ , forming a figure-eight shape.

Building a Tensor

$$\psi_{\hat{f}\hat{x}\hat{b}}\phi_{\hat{a}\hat{b}\hat{c}\hat{d}\hat{e}} = \begin{array}{c} \text{f} \\ \bullet \\ \uparrow \\ \textcircled{\psi} \\ \leftarrow \quad \rightarrow \\ \text{e} \quad \uparrow \quad \downarrow \quad \text{d} \\ \textcircled{\phi} \\ \leftarrow \quad \rightarrow \\ \text{c} \end{array} = \psi_{\hat{f}\hat{x}\hat{y}}\phi_{\hat{x}\hat{y}\hat{z}\hat{d}\hat{e}}$$

Building a Tensor

$$\psi_{\hat{f}\check{a}\check{b}}\phi_{\hat{a}\hat{b}\check{c}\hat{d}\check{e}}$$

=

The diagram shows two circular nodes, ψ and ϕ , connected by two lines that cross each other. The top node ψ has an upward-pointing arrow labeled f . The bottom node ϕ has an arrow pointing left labeled c , an arrow pointing down-right labeled d , and an arrow pointing up-left labeled e . A blue square box labeled B encloses the ψ node and the two crossing lines. The ϕ node and its three outgoing arrows are outside the box.

Building a Tensor

$$[\psi_{\hat{f}\hat{a}\hat{b}}]^B \phi_{\hat{a}\hat{b}\hat{c}\hat{d}\hat{e}} =$$

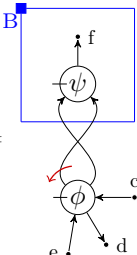
Building a Tensor

$$[\psi_{\hat{f}\hat{a}\hat{b}}]^B \phi_{\hat{a}\hat{b}\hat{c}\hat{d}\hat{e}}$$

=

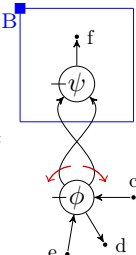
The diagram illustrates the contraction of two tensors. The upper tensor, labeled ψ , has an output index f pointing upwards and two input indices pointing downwards. The lower tensor, labeled ϕ , has an input index e pointing upwards, an output index d pointing downwards, and two other indices c and b pointing horizontally to the left and right, respectively. A red checkmark is placed to the left of the ϕ node. Two curved lines connect the two downward-pointing input indices of the ψ node to the upward-pointing input index e of the ϕ node, forming a figure-eight shape. A blue square box labeled B is drawn around the ψ node and its two input indices.

Building a Tensor

$$[\psi_{\hat{f}\hat{a}\hat{b}}]^B \phi_{\langle\hat{a}\rangle^B\hat{b}\hat{c}\hat{d}\hat{e}} =$$


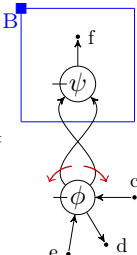
The diagram illustrates the contraction of two tensors. The upper tensor, labeled ψ , has three legs: an incoming leg from the left, an outgoing leg labeled f pointing upwards, and an incoming leg from the bottom. The lower tensor, labeled ϕ , has four legs: an incoming leg from the left, an outgoing leg labeled c pointing to the right, an outgoing leg labeled d pointing downwards, and an incoming leg labeled e pointing upwards. A blue square box encloses the top part of the diagram, with a blue square symbol and the letter B to its left. A red checkmark is placed to the left of the contraction point between the two tensors.

Building a Tensor

$$[\psi_{\hat{f}\hat{a}\hat{b}}]^B \phi_{\langle\hat{a}\rangle^B\hat{b}\hat{c}\hat{d}\hat{e}} =$$


The diagram illustrates the contraction of two tensors. The upper tensor, labeled ψ , has three legs: an output leg f pointing upwards, and two input legs a and b pointing downwards. The lower tensor, labeled ϕ , has five legs: an input leg c pointing left, an output leg d pointing down-right, and two other legs e and a pointing upwards. The two downward-pointing legs of ψ and the two upward-pointing legs of ϕ are connected by two crossing lines, forming a figure-eight shape. A blue square box encloses the upper tensor ψ and the top portion of the crossing lines. A blue letter B with a small blue square above it is positioned to the left of the box, indicating the index of the contraction. Red curved arrows are drawn above the crossing lines, pointing from the right towards the left, indicating the direction of the contraction.

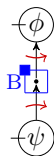
Building a Tensor

$$[\psi \hat{f} \check{a} \check{b}]^B \phi_{\langle \hat{a} \rangle^B [\hat{b}]^B \check{c} \hat{d} \check{e}} =$$


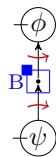
Building a Tensor

$$[\psi \hat{f} \check{a} \check{b}]^B \phi_{\langle \hat{a} \rangle^B [\hat{b}]^B \check{c} \hat{d} \check{e}} =$$

An Example

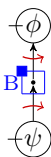


An Example

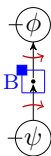
$$\psi_{[a]B} \phi_{[a]B} =$$


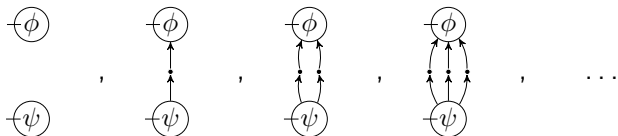
The diagram shows two circular nodes, one labeled ϕ at the top and one labeled ψ at the bottom. A vertical line connects the two nodes. A blue square labeled B is positioned on this line. A red arrow points from the bottom of the B square to the top of the ϕ node. Another red arrow points from the bottom of the B square to the top of the ψ node. This represents the contraction of the two tensors over the index B .

An Example

$$\psi_{[a]B} \phi_{[a]B} [1]^B =$$


An Example

$$\psi_{[a]B} \phi_{[a]B} [1]^B =$$


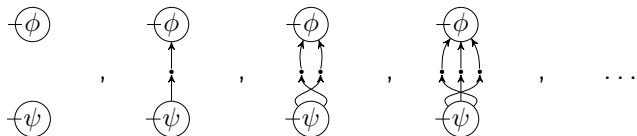


Twisted Example

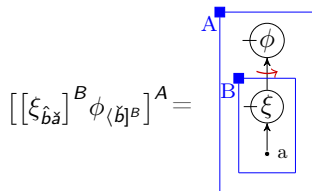
$$\psi_{[\hat{a}]^B} \phi_{[\hat{a}]^B} [1]^B =$$

Twisted Example

$$\psi_{[\tilde{a}]^B} \phi_{[\tilde{a}]^B} [1]^B = \text{Diagram}$$

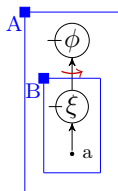


Nesting Example (Inner)

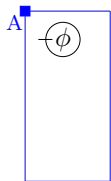


Nesting Example (Inner)

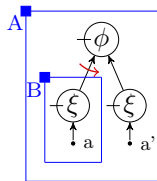
$$[[\xi_{b\check{a}}]^B \phi_{\langle b \rangle^B}]^A =$$



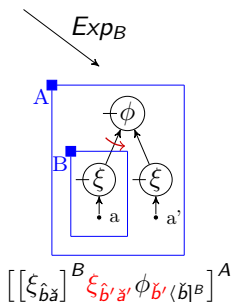
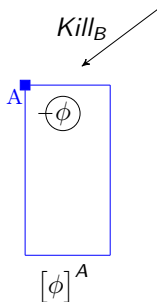
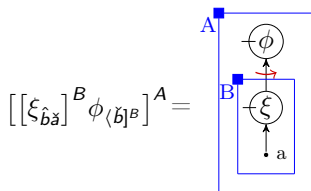
$Kill_B$



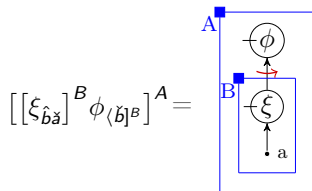
Exp_B



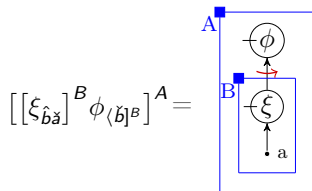
Nesting Example (Inner)



Nesting Example (Outer)

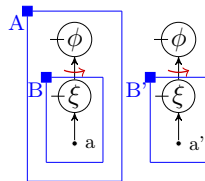


Nesting Example (Outer)

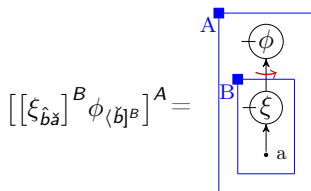


$Kill_A$

Exp_A



Nesting Example (Outer)



$Kill_A$

Exp_A

1

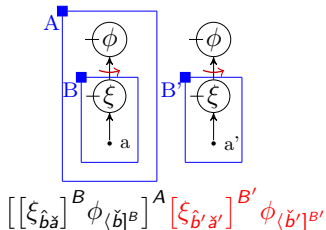


Table of Contents

Introduction

Tensor notation

Definitions

Induction

Summary

Edgeterms

Definition

Fix disjoint, infinite sets \mathcal{E} and \mathcal{B} of edge names and !-box names, respectively. The set of *edgeterms* \mathcal{T}_e is defined recursively as follows:

- $\epsilon \in \mathcal{T}_e$ (i.e empty)
- $\check{a}, \hat{a} \in \mathcal{T}_e$ $a \in \mathcal{E}$
- $[e]^A, \langle e \rangle^A \in \mathcal{T}_e$ $e \in \mathcal{T}_e, A \in \mathcal{B}$
- $ef \in \mathcal{T}_e$ $e, f \in \mathcal{T}_e$

Edgeterms

Definition

Fix disjoint, infinite sets \mathcal{E} and \mathcal{B} of edge names and !-box names, respectively. The set of *edgeterms* \mathcal{T}_e is defined recursively as follows:

- $\epsilon \in \mathcal{T}_e$ (i.e empty)
- $\check{a}, \hat{a} \in \mathcal{T}_e$ $a \in \mathcal{E}$
- $[e]^A, \langle e \rangle^A \in \mathcal{T}_e$ $e \in \mathcal{T}_e, A \in \mathcal{B}$
- $ef \in \mathcal{T}_e$ $e, f \in \mathcal{T}_e$

Two edgeterms are equivalent if one can be transformed into the other by:

$$\epsilon e \equiv e \equiv e\epsilon \quad e(fg) \equiv (ef)g \quad [e]^A \equiv \epsilon \equiv \langle e \rangle^A$$

!-Tensors

Definition

The set of all !-tensor expressions \mathcal{T}_Σ for a signature Σ is defined recursively as:

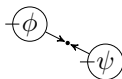
- $1 \in \mathcal{T}_\Sigma$ (empty tensor)
- $1_{\hat{a}\check{b}} \in \mathcal{T}_\Sigma$ $a, b \in \mathcal{E}$
- $\phi_e \in \mathcal{T}_\Sigma$ $e \in \mathcal{T}_e, \phi \in \Sigma$
- $[G]^A \in \mathcal{T}_\Sigma$ $G \in \mathcal{T}_\Sigma, A \in \mathcal{B}$
- $GH \in \mathcal{T}_\Sigma$ $G, H \in \mathcal{T}_\Sigma$

Satisfying conditions F1-2, C1-3

Conditions: F1-F2

F1: No directed edge name can appear more than once as these

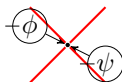
can not be plugged together: $\phi_{\hat{a}}\psi_{\hat{a}}$ =



Conditions: F1-F2

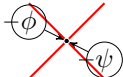
F1: No directed edge name can appear more than once as these

can not be plugged together: $\phi_{\hat{a}}\psi_{\hat{a}} =$

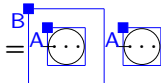


Conditions: F1-F2

F1: No directed edge name can appear more than once as these

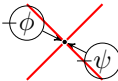
can not be plugged together: $\phi_{\hat{a}}\psi_{\hat{a}} =$ 

F2: No box can appear more than once to prevent overlap (which

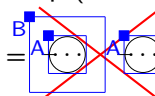
we do not allow in this formalism): $[[\dots]^A]^B [\dots]^A =$ 

Conditions: F1-F2

F1: No directed edge name can appear more than once as these

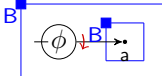
can not be plugged together: $\phi_{\hat{a}}\psi_{\hat{a}} =$ 

F2: No box can appear more than once to prevent overlap (which

we do not allow in this formalism): $[[\dots]^A]^B [\dots]^A =$ 

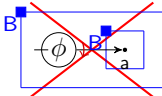
Conditions: C1-C3

C1: An edge entering a !-box can't be on a node already in that

!-Box: $[\phi_{[\hat{a}]^B}]^B =$ 

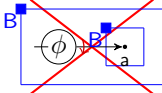
Conditions: C1-C3

C1: An edge entering a !-box can't be on a node already in that

!-Box: $[\phi_{[\hat{a}]^B}]^B =$ 

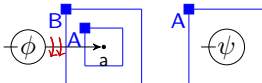
Conditions: C1-C3

C1: An edge entering a !-box can't be on a node already in that

!-Box: $[\phi_{[\hat{a}]^B}]^B =$ 

The diagram shows a blue square box labeled '!' in the top-left corner. Inside the box is a circle containing the symbol ϕ . An arrow points from the circle to a smaller blue square box labeled '!' in its top-left corner, which contains the symbol a . A red 'X' is drawn over the entire diagram, indicating that this configuration is invalid.

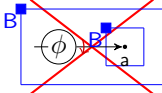
C2: Nested !-Boxes around an edge must be nested in the same way

in the rest of the tensor: $\phi_{[[\hat{a}]^A]^B} [\psi]^A =$ 

The diagram shows two blue square boxes. The left box is labeled '!' in the top-left corner and contains a circle with ϕ . An arrow points from this circle to a smaller blue square box labeled '!' in its top-left corner, which contains a circle with ψ and the symbol a . A red arrow points to the edge between the two boxes. The right box is labeled '!' in the top-left corner and contains a circle with ψ . The labels 'A' and 'B' are placed above the boxes, with 'A' above the inner box and 'B' above the outer box.

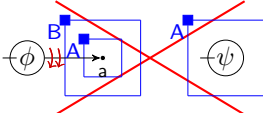
Conditions: C1-C3

C1: An edge entering a !-box can't be on a node already in that

!-Box: $[\phi_{[\hat{a}]^B}]^B =$ 

The diagram shows a blue square box labeled '!' in the top-left corner. Inside the box is a circle containing the Greek letter phi (φ). An arrow points from the circle to a node labeled 'a'. A smaller blue square box labeled '!' is nested inside the larger one, also containing the node 'a'. A red 'X' is drawn over the entire diagram, indicating that this configuration is invalid because the edge entering the inner !-box is on a node already present in the outer !-box.

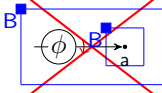
C2: Nested !-Boxes around an edge must be nested in the same way

in the rest of the tensor: $\phi_{[[\hat{a}]^A]^B} [\psi]^A =$ 

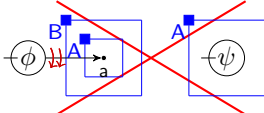
The diagram shows two blue square boxes labeled '!' in the top-left corners. The left box contains a circle with phi (φ) and an arrow pointing to a node 'a'. The right box contains a circle with psi (ψ). A red 'X' is drawn over the entire diagram, indicating that this configuration is invalid because the boxes are not nested in the same way around the edge between the two nodes.

Conditions: C1-C3

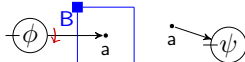
C1: An edge entering a !-box can't be on a node already in that

!-Box: $[\phi_{[\hat{a}]^B}]^B =$ 

C2: Nested !-Boxes around an edge must be nested in the same way

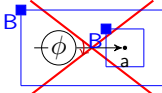
in the rest of the tensor: $\phi_{[[\hat{a}]^A]^B} [\psi]^A =$ 

C3: Bound edges must be in compatible !-boxes:

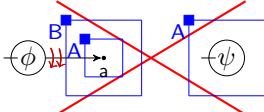
$\phi_{[\hat{a}]^B} \psi_{\check{a}}$ = 

Conditions: C1-C3

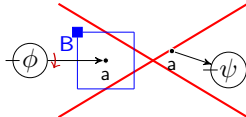
C1: An edge entering a !-box can't be on a node already in that

!-Box: $[\phi_{[\hat{a}]^B}]^B =$ 

C2: Nested !-Boxes around an edge must be nested in the same way

in the rest of the tensor: $\phi_{[[\hat{a}]^A]^B} [\psi]^A =$ 

C3: Bound edges must be in compatible !-boxes:

$\phi_{[\hat{a}]^B} \psi_{\hat{a}}$ = 

Operations

The operation Kill removes a !-box and all nodes and edges in it:

Operations

The operation Kill removes a !-box and all nodes and edges in it:

$$\text{Kill}_B := [[G]^B \mapsto 1, [e]^B \mapsto \epsilon, \langle e \rangle^B \mapsto \epsilon]$$

Operations

The operation Kill removes a !-box and all nodes and edges in it:

$$\text{Kill}_B := [[G]^B \mapsto 1, [e]^B \mapsto \epsilon, \langle e \rangle^B \mapsto \epsilon]$$

Expanding a !-box creates a new copy of its contents with new names for all new edges/boxes. Write $\mathbf{fr}(G)$ for G with all names replaced by new ones (chosen by predetermined function \mathbf{fr}).

Operations

The operation *Kill* removes a !-box and all nodes and edges in it:

$$Kill_B := [[G]^B \mapsto 1, [e]^B \mapsto \epsilon, \langle e \rangle^B \mapsto \epsilon]$$

Expanding a !-box creates a new copy of its contents with new names for all new edges/boxes. Write $\mathbf{fr}(G)$ for G with all names replaced by new ones (chosen by predetermined function \mathbf{fr}).

$$Exp_B := [[G]^B \mapsto [G]^B \mathbf{fr}(G), [e]^B \mapsto [e]^B \mathbf{fr}(e), \langle e \rangle^B \mapsto \mathbf{fr}(e) \langle e \rangle^B]$$

Table of Contents

Introduction

Tensor notation

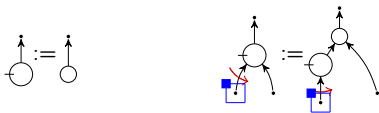
Definitions

Induction

Summary

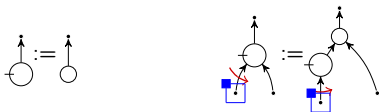
Spider Node

Definition

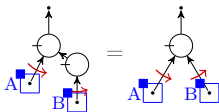


Spider Node

Definition

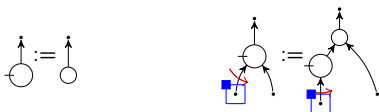


Theorem

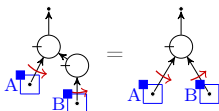


Spider Node

Definition



Theorem



We would like to do induction on !-box B splitting into a base case (after $Kill_B$) and an inductive step (proving Exp_B from original).

Induction

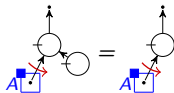
$$\frac{Kill_B(G = H) \quad (G = H) \implies Exp_B(G = H)}{G = H} \text{ (Induction)}$$

Induction

$$\frac{Kill_B(G = H) \quad Fix_B(G = H) \implies Exp_B(G = H)}{G = H} \text{ (Induction)}$$

Spider Theorem ($Kill_B$)

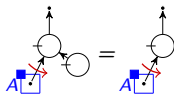
Theorem



(base)

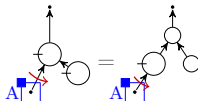
Spider Theorem ($Kill_B$)

Theorem



(base)

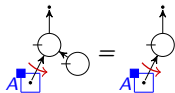
Proof.



□

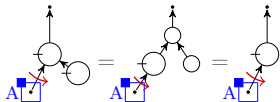
Spider Theorem ($Kill_B$)

Theorem



(base)

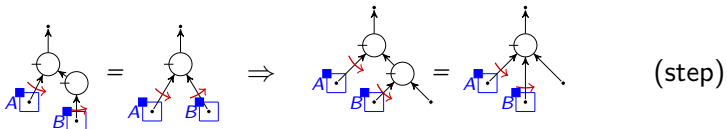
Proof.



□

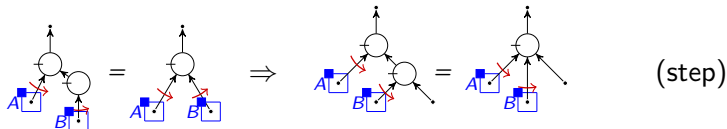
Spider Theorem ($Fix_B \implies Exp_B$)

Theorem

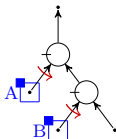


Spider Theorem ($Fix_B \implies Exp_B$)

Theorem



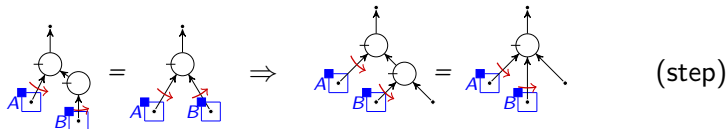
Proof.



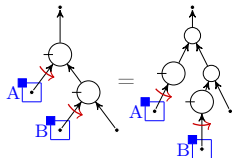
□

Spider Theorem ($Fix_B \implies Exp_B$)

Theorem



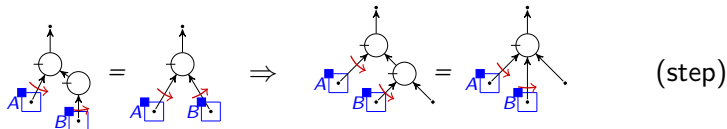
Proof.



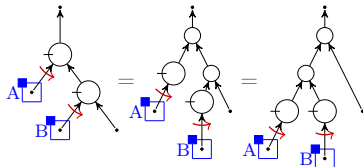
□

Spider Theorem ($Fix_B \implies Exp_B$)

Theorem



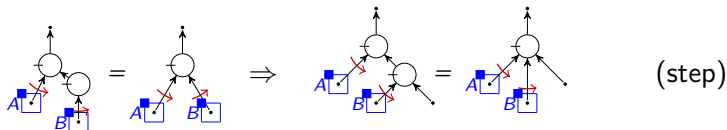
Proof.



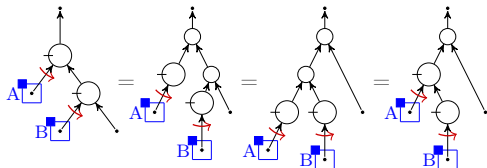
□

Spider Theorem ($Fix_B \implies Exp_B$)

Theorem



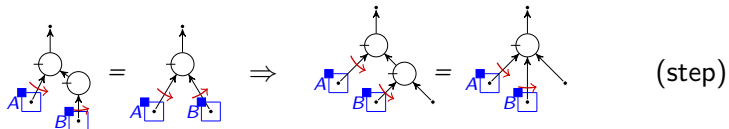
Proof.



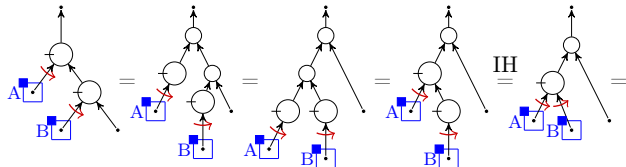
□

Spider Theorem ($Fix_B \implies Exp_B$)

Theorem

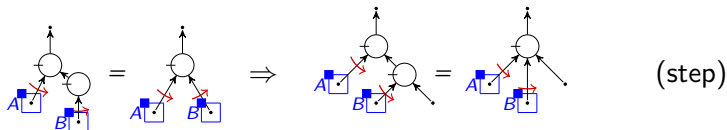


Proof.

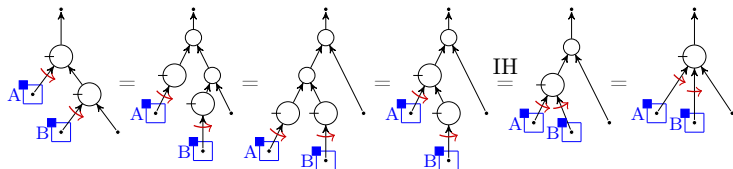


Spider Theorem ($Fix_B \implies Exp_B$)

Theorem

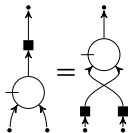


Proof.

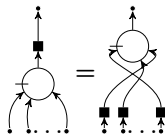
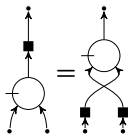


□

Anti-Homomorphism



Anti-Homomorphism



Anti-Homomorphism

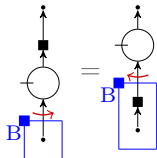
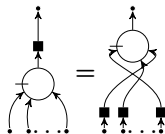
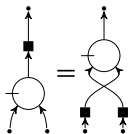


Table of Contents

Introduction

Tensor notation

Definitions

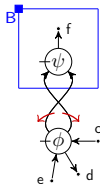
Induction

Summary

Summary

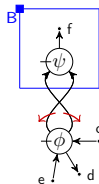
Summary

- Diagrammatic language:



Summary

- Diagrammatic language:

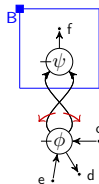


- Tensor notation:

$$[\psi_{\hat{f}\check{a}\check{b}}]^B \phi_{\langle \hat{a} \rangle^B [\hat{b}]^B \check{c} \hat{d} \check{e}}$$

Summary

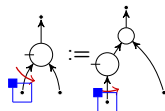
- Diagrammatic language:



- Tensor notation:

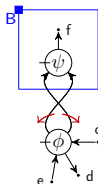
$$[\psi \hat{f} \check{a} \check{b}]^B \phi \langle \hat{a} \rangle^B [\hat{b}]^B \check{c} \hat{d} \check{e}$$

- Recursive definitions:



Summary

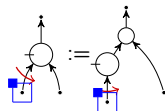
- Diagrammatic language:



- Tensor notation:

$$[\psi \hat{f} \check{a} \check{b}]^B \phi \langle \hat{a} \rangle^B [\hat{b}]^B \check{c} \hat{d} \check{e}$$

- Recursive definitions:



- !-Box Induction:

$$\frac{\text{Kill}_B(G = H) \quad \text{Fix}_B(G = H) \implies \text{Exp}_B(G = H)}{G = H} \quad (\text{Induction})$$