In[1]:=

```
<< Notation`
```

Some code in this cell, for the Hope Algebra, please run in order as follows below, use Cell>Cell Properties>Open to see the code:

Some Notations for Twister Groups

⟶ Morphism code

Modules for twisted goups

Braid Groups modules

Qausialgebra Structure of Octonions

Braided group algebra theories of Prof. Majid Shahn has been one of the most motivating conceptual endeavors in this author's life. However the learning curve is steep and nomenclature quite varied between the researchers and the subject matter is presented, almost exclusively, to theoretical physicists. Therefore the author attempted to develop a package in *Mathematica* 9.0 for symbolic and numeric computation of Hopf Algebras, Twisted Groups, Braid Groups and in part Cateogorial evaluation of their expressions.

The braided algebras actually belong to the computing theories and in specific to the complexity theory. The author firmly believes that the braided algebras configure memory systems wherein the information about physical entities are stored: Absorption of energy forms the memory system, emission of energy erases the memory.

In order to test the code, the author took selections from Prof. Shahn's work [1] and [2] and some from Dixon [3] and coded their symbolic derivation and manipulation.

Remark: *Write up below is unstructured, the purpose is to make sure the code does what Prof. Shah's theory requires.*

Twisted Group Algebra

The author always found the concept of groups awkward, namely special case of something else. Twisted Groups are much better starting point, indeed regular groups are twisted group with null twist. Therefore in the code we assume first there is twisted group and then special case with nullTwist [ ] is the traditional groups:

In[249]:=
```
ℤ2 = ℤnm[2, 1, nullTwist, ""];
tableGroupTwist[ℤ2]
```

Out[250]//TableForm=

	1	e1
1	1	e1
e1	e1	1

In general the code can generate any $\mathbb{Z}_n^m$ as null twisted group:

In[280]:=
```
ℤ2^3 = ℤnm[2, 3, nullTwist, ""];
tableGroupTwist[ℤ2^3]
```

Out[281]//TableForm=

	1	e1	e2	e3	e4	e5	e6	e7
1	1	e1	e2	e3	e4	e5	e6	e7
e1	e1	1	e3	e2	e5	e4	e7	e6
e2	e2	e3	1	e1	e6	e7	e4	e5
e3	e3	e2	e1	1	e7	e6	e5	e4
e4	e4	e5	e6	e7	1	e1	e2	e3
e5	e5	e4	e7	e6	e1	1	e3	e2
e6	e6	e7	e4	e5	e2	e3	1	e1
e7	e7	e6	e5	e4	e3	e2	e1	1

Using concepts from the recent works of Prof. Shahn on Quasialgebras, the code can easily and successfully generate the well known arithmatic/algebraic systems:

2 QUASIALGEBRAS $k_F G$

First of all, we know that if we consider the set of complex numbers, the quaternions or the octonions, all these algebras have something in common: If we choose a suitable basis and remove the $\pm$ signs from the multiplication tables of these algebras, we have the tables of the additive groups $G = \boldsymbol{Z}_2$ (for complex numbers), $G = \boldsymbol{Z}_2 \times \boldsymbol{Z}_2$ (for quaternions) and $G = \boldsymbol{Z}_2 \times \boldsymbol{Z}_2 \times \boldsymbol{Z}_2$ (for octonions) [5]. We view the signs in the multiplication tables as an invertible 2-cochain $F : G \times G \to k$ (a nowhere-zero function which is 1 when either argument is the group identity $e \in G$). Writing $F(x, y) = (-1)^{f(x,y)}$, one has explicitly [1],

$$G = \boldsymbol{Z}_2, \quad f(x, y) = xy, \quad x, y \in \boldsymbol{Z}_2 \qquad \text{(Complex numbers)},$$

$$G = (\boldsymbol{Z}_2)^2, \quad f(\vec{x}, \vec{y}) = x_1 y_1 + (x_1 + x_2) y_2 \qquad \text{(Quaternions)},$$

where $\vec{x} = (x_1, x_2) \in G$ is a vector notation and the components x_1, x_2 are viewed in the field $\boldsymbol{Z}_2$.

$$G = (\boldsymbol{Z}_2)^3, \quad f(\vec{x}, \vec{y}) = \sum_{i \le j} x_i y_j + y_1 x_2 x_3 + x_1 y_2 x_3 + x_1 x_2 y_3 \quad \text{(Octonions)},$$

In[282]:=
```
tableGroupTwist[ℤnm[2, 1, αℂ, ""]]
```

Out[282]//TableForm=

	1	e1
1	1	e1
e1	e1	−1

$\alpha\mathbb{C}$ [ ] is a *Mathematica* module that computes the corresponding $-1^{f(x,y)}$ for Complex Numbers. Similarly Quaternions

In[283]:=
```
tableGroupTwist[ℤnm[2, 2, αℍ, ""]]
```

Out[283]//TableForm=

	1	e1	e2	e3
1	1	e1	e2	e3
e1	e1	−1	e3	−e2
e2	e2	−e3	−1	e1
e3	e3	e2	−e1	−1

And of course Octonions:

In[285]:=
```
tableGroupTwist[ℤnm[2, 3, αO, ""]]
```

Out[285]//TableForm=

	1	e1	e2	e3	e4	e5	e6	e7
1	1	e1	e2	e3	e4	e5	e6	e7
e1	e1	−1	e3	−e2	e5	−e4	−e7	e6
e2	e2	−e3	−1	e1	e6	e7	−e4	−e5
e3	e3	e2	−e1	−1	e7	−e6	e5	−e4
e4	e4	−e5	−e6	−e7	−1	e1	e2	e3
e5	e5	e4	−e7	e6	−e1	−1	−e3	e2
e6	e6	e7	e4	−e5	−e2	e3	−1	−e1
e7	e7	−e6	e5	e4	−e3	−e2	e1	−1

Note that the original table was the following without the twist $\alpha\mathbb{O}$ [ ] i.e. null Twist:

In[286]:=
```
tableGroupTwist[ℤnm[2, 3, nullTwist, ""]]
```

Out[286]//TableForm=

	1	e1	e2	e3	e4	e5	e6	e7
1	1	e1	e2	e3	e4	e5	e6	e7
e1	e1	1	e3	e2	e5	e4	e7	e6
e2	e2	e3	1	e1	e6	e7	e4	e5
e3	e3	e2	e1	1	e7	e6	e5	e4
e4	e4	e5	e6	e7	1	e1	e2	e3
e5	e5	e4	e7	e6	e1	1	e3	e2
e6	e6	e7	e4	e5	e2	e3	1	e1
e7	e7	e6	e5	e4	e3	e2	e1	1

isCommutative[ ], isAssociative[ ]

In order to determine if a twisted algebra is commutative or associative two *Mathematica* modules were developed, which do brute-force check on the multiplication table.

Let's make some Hadamard matrices:

In[287]:=
```
2 * HadamardMatrix[4] // MatrixForm
ArrayPad[2 * HadamardMatrix[4], {1, 0}, 1] // MatrixForm
```

Out[287]//MatrixForm=

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{pmatrix}$$

Out[288]//MatrixForm=

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & -1 & -1 \\ 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & 1 & -1 \end{pmatrix}$$

αMatrix [ ] is a matrix based twist, in this case the matrix is the Hadamard:

In[293]:=
```
grp = ℤnm[2, 4, αMatrix, 4 * HadamardMatrix[16] ];
tableGroupTwist[grp]
isCommutative[grp]
isAssociative[grp]
```

Out[294]//TableForm=

	1	e1	e2	e3	e4	e5	e6	e7
1	1	e1	e2	e3	e4	e5	e6	e7
e1	e1	1	e3	e2	e5	e4	e7	e6
e2	e2	e3	1	e1	−e6	−e7	−e4	−e5
e3	e3	e2	e1	1	−e7	−e6	−e5	−e4
e4	e4	e5	−e6	−e7	−1	−e1	e2	e3
e5	e5	e4	−e7	−e6	−e1	−1	e3	e2
e6	e6	e7	−e4	−e5	e2	e3	−1	−e1
e7	e7	e6	−e5	−e4	e3	e2	−e1	−1
e8	e8	−e9	−e10	e11	e12	−e13	−e14	e15
e9	e9	−e8	−e11	e10	e13	−e12	−e15	e14
e10	e10	−e11	−e8	e9	−e14	e15	e12	−e13
e11	e11	−e10	−e9	e8	−e15	e14	e13	−e12
e12	e12	−e13	e14	−e15	−e8	e9	−e10	e11
e13	e13	−e12	e15	−e14	−e9	e8	−e11	e10
e14	e14	−e15	e12	−e13	e10	−e11	e8	−e9
e15	e15	−e14	e13	−e12	e11	−e10	e9	−e8

Out[295]= True

Out[296]= True

This twisted algebra is commutative but not associative and the indices where the associativity fails are issued.

256×256 Hadamard matrices where formed and

```
(* this might take a long while to calculate *)
n = 8;
grp = ℤnm[2, n, αMatrix, 2^(n / 2) * HadamardMatrix[2^n] ];
isCommutative[grp]
isAssociative[grp]
```

```
True
True
```

Commutative Non-Associative Algebra

‘1’ Padded Hadamard matrices produce commutative but non-associative algebras:

In[301]:=
```
2 * HadamardMatrix[4] // MatrixForm
```
Out[301]//MatrixForm=

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{pmatrix}$$

In[302]:=
```
ArrayPad[2 * HadamardMatrix[4], {1, 0}, 1] // MatrixForm
```
Out[302]//MatrixForm=

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & -1 & -1 \\ 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & 1 & -1 \end{pmatrix}$$

In[297]:=

```
grp = ℤnm[5, 1, αMatrix, ArrayPad[2 * HadamardMatrix[4], {1, 0}, 1] ];
tableGroupTwist[ℤnm[5, 1, αMatrix, ArrayPad[2 * HadamardMatrix[4], {1, 0}, 1] ]]
isCommutative[grp]
isAssociative[grp]
```

Out[297]//TableForm=

	1	e1	e2	e3	e4
1	1	e1	e2	e3	e4
e1	e1	e2	e3	e4	1
e2	e2	e3	e4	−1	−e1
e3	e3	e4	−1	−e1	e2
e4	e4	1	−e1	e2	−e3

Out[298]= True

Out[299]= {False, 4, 4, 3}

This twisted algebra is commutative but not associative and the indices where the associativity fails are issued.

Let's see which number systems are commutative or associative:

```
(* Complex Numbers, commutative and associative *)
grp = ℤnm[2, 1, αℂ, ""] ;
isCommutative[grp]
isAssociative[grp]
```

Out[307]= True

Out[308]= True

```
(* Quaternions, not commutative but associative *)
grp = ℤnm[2, 2, αℍ, ""] ;
isCommutative[grp]
isAssociative[grp]
```

Out[313]= False

Out[314]= True

In[315]:=
```
(* Octonions, not commutative and not associative *)
grp = ℤnm[2, 3, αO, ""] ;
isCommutative[grp]
isAssociative[grp]
```

Out[316]= False

Out[317]= {False, 7, 6, 5}

Hopf Algebra

Overview: Operators and Tensor Products

Notations Package in *Mathematica* was used to model the operators' syntax as they appear in mathematics books.

Convention: $\underset{b}{a}$ *means* $a \in b$ unless otherwise specified.

op [ ] and op ( ) are the same to support seamlessly interface with Mathematica functions and modules.

Here a sample of Hopf Algebra types of expressions and excerpts from Prof. Majid Shahn's Book [1]:

In[336]:=
```
hopfAlgebraHelp[]
```

⊗ tensor product, mapped to tensorProduct[]

⊙ Majid Shahn's '.' product between the algebra members(vectors),
mapped to vectorProduct[].It is written as xy and defined as .:A⊗A ⟶ A

$\odot^{*}$ product between the dual vectors

+ overloaded operator, in the algebra it is vector addition (or -) mapped
to vectorAdd[], between the tensor products is commutative concatenation

◇ is the scalar product of the ground Ring/Field to a vector, mapped to scalarMult[]

$\underset{k}{\diamond}$ Ring k product, mapped to ringMult[]

$\underset{k}{+}$ Ring k add, mapped to ringAdd[]

one is the symbolic 1, l_{one} = 1. x^{-one} does not map to FractionBox[1\, x] or 1/x

two is the symbolic 2, l_{two} = 1.

ϵ: H⟶k Counit map

η: k⟶H

$\underset{m}{\Delta}(h)$ = $c_{one1}\otimes c_{two1}$ + $c_{one2}\otimes c_{two2}$ + ... + $c_{onem}\otimes c_{twom}$

$\underset{n,m}{\Delta}$ breaks up to summation of n and m parts:

$\underset{2,3}{\Delta}(h\odot g)$ = $(c_{one1}\otimes c_{two1}$ + $c_{one2}\otimes c_{two2})\odot(c_{one1}\otimes c_{two1}$
+ $c_{one2}\otimes$SubscriptBox[SubscriptBox[c, two], 2\]+ $c_{one3}\otimes c_{two3})$

τ Transposition map

id the identity map

$\underset{y}{x}$ means x∈y, same for operators $\underset{\mathbb{R}}{\diamond}$ means product involving ℝ

$\odot$

Due to overloading of '.' Algebra $(H,\ +,\ .,\ \eta,\ \Delta,\ \epsilon,\ k)$ is denoted by $(H,\ +,\ \odot,\ \eta,\ \Delta,\ \epsilon,\ k)$.

FIXME: *In some occasions ⊙ has to be used as a String in Mathematica i.e. "⊙", this has to do with the preset of ⊙ as an operator.*

Arrows or Morphisms

Arrows have been coded to provide Categorial interpretations of Morphisms between objects. However only the Domain object is specified and Codomain object is printed as output:

$\mathbf{c}\xrightarrow{\mathbf{id}}$

c

Composition of arrows have the usual semantics:

$$\mathbf{c} \xrightarrow{\mathbf{id}} \xrightarrow{\mathbf{id}} \xrightarrow{\mathbf{id}}$$

c

Coproduct Δ

Number 1 is overloaded in *Mathematica* as integer 1 and might be removed from outputs of multiplications and (2) might be simplified to 2. In order to follow the Sweedler's notion numbers 1 and 2 and 3 are turned into symbolic English one, two and three, this avoiding the operations by number 1 and parenthesis simplifications. Therefore $c_{\text{one}} \otimes c_{\text{two}} = c_{(1)} \otimes c_{(2)}$:

$$\mathbf{c} \xrightarrow{\Delta}$$

$c_{one} \otimes c_{two}$

Remark: *This works for small number of tensor products, for larger tensor products this convention will not work and numeric representation must be used.*

FIXME: $c_{\text{one}} \otimes c_{\text{twoone}} = (c_{\text{one}} \otimes c_{\text{two}})_{\text{one}} = (c_{(1)} \otimes c_{(2)})_{(1)}$ unfortunately *Mathematica* omits parenthesis which makes the expressions a bit difficult to match to the math textbooks.

In case no Domain was specified:

$$\xrightarrow{\Delta}$$

$\heartsuit_{one} \otimes \heartsuit_{two}$

$$\mathbf{c} \xrightarrow{\Delta} \xrightarrow{\Delta}$$

$(c_{one} \otimes c_{twoone}) \otimes (c_{one} \otimes c_{twotwo})$

Δ_n specifies the summation upper bound, it is necessary to be added as a Subscript since the math books assume that upper bound implicitly, but for programming reasons we need to specify it explicitly.

If n is a constant e.g. 5:

$$\underset{\mathbf{5}}{\Delta}(\mathbf{c})$$

$c_{one1} \otimes c_{two1} + c_{one2} \otimes c_{two2} + c_{one3} \otimes c_{two3} + c_{one4} \otimes c_{two4} + c_{one5} \otimes c_{two5}$

If n variable:

$$\Delta_{\mathbf{n}}(\mathbf{c})$$

$$\sum_{i=1}^{n} c_{onei} \otimes c_{twoi}$$

Infinite series:

In[197]:=

$$\Delta_{\infty}(\mathbf{c})$$

Out[197]= $$\sum_{i=1}^{\infty} c_{onei} \otimes c_{twoi}$$

Properties:

```
Δ (h ⊙ g) === (Δ (h )) ⊙ (Δ (g))
True
```

Verify by direct calculation:

$$\Delta(\mathbf{h} \odot \mathbf{g})$$

$$(\mathbf{h}_{one} \otimes \mathbf{h}_{two}) \odot (\mathbf{g}_{one} \otimes \mathbf{g}_{two})$$

$$h_{one} \odot g_{one} \otimes h_{two} \odot g_{two}$$

$$(\Delta(\mathbf{h})) \odot (\Delta(\mathbf{g}))$$

$$(\mathbf{h}_{one} \otimes \mathbf{h}_{two}) \odot (\mathbf{g}_{one} \otimes \mathbf{g}_{two})$$

$$h_{one} \odot g_{one} \otimes h_{two} \odot g_{two}$$

Linear Maps on Δ

f_{linear} means a function that is a linear map. In most cases the bottom subscript specifies the Domain or type of indicated entity f.

FIXME: *We had to use f [ ] , since f ( ) does not work. Mathematica omits the parenthesis.*

$$f_{\mathbf{linear}}[\mathbf{c}_{one} \otimes \mathbf{c}_{two}]$$

$$f_{linear}[c]_{one} \otimes f_{linear}[c]_{two}$$

or

```
f
linear [Δ (c)]
```

```
f
linear [c_one ⊗ c_two]
```

```
f [c]_one ⊗ f [c]_two
linear        linear
```

If f is not typed as linear then evaluation halts:

```
f[Δ (c)]
```

```
f[c_one ⊗ c_two]
```

```
f[c_one ⊗ c_two]
```

```
f[c_one ⊗ c_two]
```

Associativity

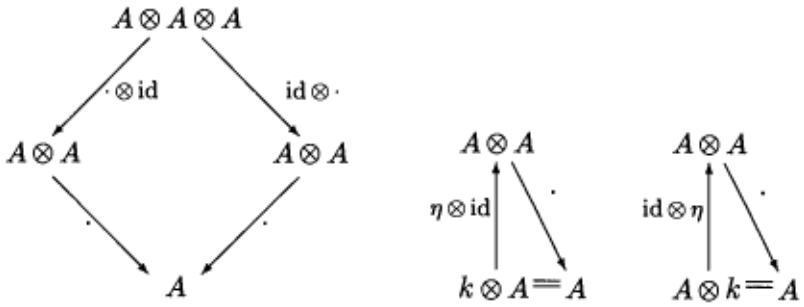


Fig. 1.1. Associativity and unit element expressed as commutative diagrams.

"⊙"⊗id was actually ⊙⊗id but do to some complication in *Mathematica* "⊙" used as String:

$x \otimes y \otimes z \in A \otimes A \otimes A$ (recall ⊙ is actually '.')

Below are the evaluation of the 4 arrows in FIG 1.1.

FIXME: *Mathematica does not do nested evaluation in full, so sometimes we need to manually evaluate the interim expressions i.e. SHIFT+RETURN*

$$(x \otimes y \otimes z) \xrightarrow{"\odot" \otimes id}$$

$$("\odot" \otimes id)[x \otimes y \otimes z]$$

$$x \odot y \otimes z$$

$$(x \otimes y \otimes z) \xrightarrow{id \otimes "\odot"}$$

$$(id \otimes "\odot")[x \otimes y \otimes z]$$

$$x \otimes y \odot z$$

$$(x \otimes y \otimes z) \xrightarrow{"\odot" \otimes id} \xrightarrow{"\odot"}$$

$$"\odot"[("\odot" \otimes id)[x \otimes y \otimes z]]$$

$$"\odot"[x \odot y \otimes z]$$

$$(x \odot y) \odot z$$

$$(x \otimes y \otimes z) \xrightarrow{id \otimes "\odot"} \xrightarrow{"\odot"}$$

$$"\odot"[(id \otimes "\odot")[x \otimes y \otimes z]]$$

$$"\odot"[x \otimes y \odot z]$$

$$x \odot (y \odot z)$$

$\cong_{()}$ is an isomorphism based Boolean to check to see if the expressions are the same, in this case $\cong_{()}$ just checks where the parenthesis are:

$$x \odot (y \odot z) \cong_{()} (x \odot y) \odot z$$

```
True
```

Remark: *More of these so called isomorphisms should be coded to avoid the concept of equality.*

Associativity

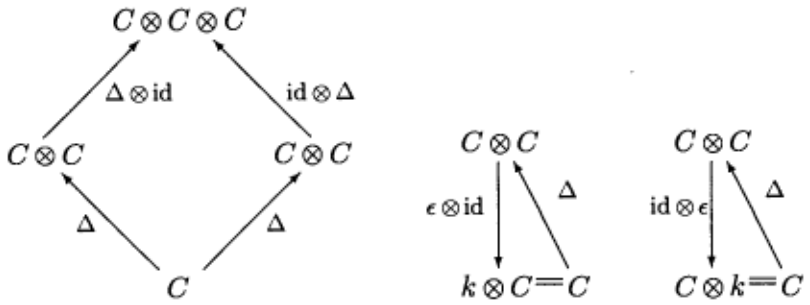


Fig. 1.2. Coassociativity and counit element expressed as commutative diagrams.

$\mathbf{c} \xrightarrow{\Delta} \xrightarrow{\mathbf{id}\otimes\Delta}$

$(\mathbf{id}\otimes\Delta)\,[\mathbf{c_{one}}\otimes\mathbf{c_{two}}]$

$\mathbf{c_{one}}\otimes(\mathbf{c_{twoone}}\otimes\mathbf{c_{twotwo}})$

$c_{one}\otimes(c_{twoone}\otimes c_{twotwo})$

$\mathbf{c} \xrightarrow{\Delta} \xrightarrow{\Delta\otimes\mathbf{id}}$

$(\Delta\otimes\mathbf{id})\,[\mathbf{c_{one}}\otimes\mathbf{c_{two}}]$

$(c_{oneone}\otimes c_{onetwo})\otimes c_{two}$

To appreciate the complexity of the notations, use the function $\underset{\Delta,n}{\text{Expand}}$ which turns the Sweedler's notation into summation:

$\mathbf{c_{one}}\otimes\underset{\Delta,\mathbf{n}}{\mathbf{Expand}}\,[\mathbf{c_{twoone}}\otimes\mathbf{c_{twotwo}}]$

$\mathbf{c_{one}}\otimes\sum_{\mathbf{i=1}}^{\mathbf{n}}\mathbf{c_{twooneI}}\otimes\mathbf{c_{twotwoi}}$

$\sum_{i=1}^{n} c_{one}\otimes c_{twooneI}\otimes c_{twotwoi}$

$\left(\underset{\Delta,\mathbf{n}}{\mathbf{Expand}}\,[\mathbf{c_{oneone}}\otimes\mathbf{c_{onetwo}}]\right)\otimes\mathbf{c_{two}}$

$\left(\sum_{\mathbf{i=1}}^{\mathbf{n}}\mathbf{c_{oneoneI}}\otimes\mathbf{c_{onetwoi}}\right)\otimes\mathbf{c_{two}}$

$\sum_{i=1}^{n} c_{oneoneI}\otimes c_{onetwoi}\otimes c_{two}$

In summary:

$$c \xrightarrow{\Delta} \xrightarrow{\text{id}\otimes\Delta} = c_{\text{one}} \otimes (c_{\text{twoone}} \otimes c_{\text{twotwo}}) = \sum_{i=1}^{n} c_{\text{one}} \otimes c_{\text{twoone}i} \otimes c_{\text{twotwo}i}$$

$$c \xrightarrow{\Delta} \xrightarrow{\Delta\otimes\text{id}} = (c_{\text{oneone}} \otimes c_{\text{onetwo}}) \otimes c_{\text{two}} = \sum_{i=1}^{n} c_{\text{oneone}i} \otimes c_{\text{onetwo}i} \otimes c_{\text{two}}$$

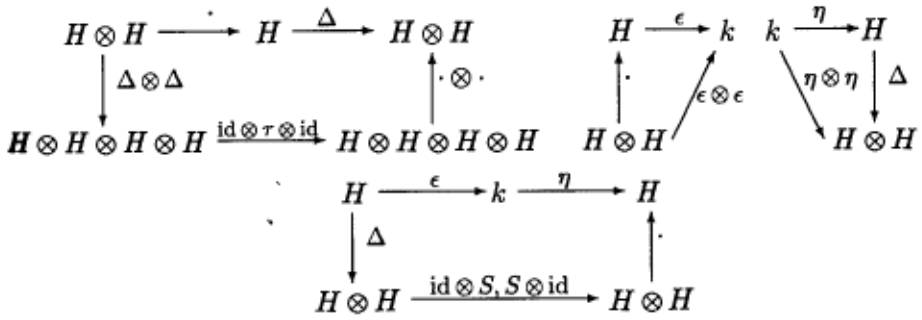


Fig. 1.3. Additional axioms that make the algebra and coalgebra H into a Hopf algebra.

$$(\mathbf{h1} \otimes \mathbf{h2}) \xrightarrow{\Delta\otimes\Delta}$$

$$(\Delta \otimes \Delta)\,[\mathbf{h1} \otimes \mathbf{h2}]$$

$$(\text{h1}_{\text{one}} \otimes \text{h1}_{\text{two}}) \otimes (\text{h2}_{\text{one}} \otimes \text{h2}_{\text{two}})$$

The latter is deceptively simple, but let's expand it to see how it is formed:

$$\left(\underset{\Delta,\text{n},\text{i}}{\mathbf{Expand}}\,[\mathbf{h1}_{\mathbf{one}} \otimes \mathbf{h1}_{\mathbf{two}}]\right) \otimes \left(\underset{\Delta,\text{m},\text{j}}{\mathbf{Expand}}\,[\mathbf{h2}_{\mathbf{one}} \otimes \mathbf{h2}_{\mathbf{two}}]\right)$$

$$\left(\sum_{\mathbf{i}=1}^{\text{n}} \mathbf{h1}_{\mathbf{one}\text{i}} \otimes \mathbf{h1}_{\mathbf{two}\text{i}}\right) \otimes \sum_{\mathbf{j}=1}^{\text{m}} \mathbf{h2}_{\mathbf{one}\text{j}} \otimes \mathbf{h2}_{\mathbf{two}\text{j}}$$

$$\sum_{\mathbf{i}=1}^{\text{n}} \mathbf{h1}_{\mathbf{one}\text{i}} \otimes \mathbf{h1}_{\mathbf{two}\text{i}} \otimes \sum_{\mathbf{j}=1}^{\text{m}} \mathbf{h2}_{\mathbf{one}\text{j}} \otimes \mathbf{h2}_{\mathbf{two}\text{j}}$$

$$\sum_{\text{i}=1}^{\text{n}} \sum_{\text{j}=1}^{\text{m}} \text{h1}_{\text{one}\text{i}} \otimes \text{h1}_{\text{two}\text{i}} \otimes \text{h2}_{\text{one}\text{j}} \otimes \text{h2}_{\text{two}\text{j}}$$

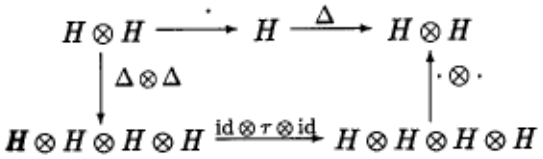
Now let's take its middle part $\text{h1}_{\text{two}\text{i}} \otimes \text{h2}_{\text{one}\text{j}}$ and swap them using τ :

$$(\mathbf{h1} \otimes \mathbf{h2}) \xrightarrow{\Delta\otimes\Delta} \xrightarrow{\mathbf{id}\otimes\text{"}\tau\text{"}\otimes\mathbf{id}}$$

$$(\mathbf{id} \otimes \text{"}\tau\text{"} \otimes \mathbf{id})\,[\,(\Delta \otimes \Delta)\,[\mathbf{h1} \otimes \mathbf{h2}]\,]$$

$$(\mathbf{id} \otimes \text{"}\tau\text{"} \otimes \mathbf{id})[(\mathbf{h1}_{\mathbf{one}} \otimes \mathbf{h1}_{\mathbf{two}}) \otimes (\mathbf{h2}_{\mathbf{one}} \otimes \mathbf{h2}_{\mathbf{two}})]$$

$$h1_{one} \otimes (h2_{one} \otimes h1_{two}) \otimes h2_{two}$$



$$(\mathbf{h1} \otimes \mathbf{h2}) \xrightarrow{\Delta\otimes\Delta} \xrightarrow{\mathbf{id}\otimes\text{"}\tau\text{"}\otimes\mathbf{id}} \xrightarrow{\text{"}\odot\text{"}\otimes\text{"}\odot\text{"}}$$

$$(\text{"}\odot\text{"} \otimes \text{"}\odot\text{"})[(\mathbf{id} \otimes \text{"}\tau\text{"} \otimes \mathbf{id})[(\Delta \otimes \Delta)[\mathbf{h1} \otimes \mathbf{h2}]]]$$

$$(\text{"}\odot\text{"} \otimes \text{"}\odot\text{"})[(\mathbf{id} \otimes \text{"}\tau\text{"} \otimes \mathbf{id})[(\mathbf{h1}_{\mathbf{one}} \otimes \mathbf{h1}_{\mathbf{two}}) \otimes (\mathbf{h2}_{\mathbf{one}} \otimes \mathbf{h2}_{\mathbf{two}})]]$$

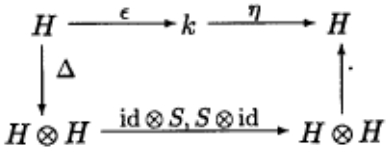
$$(\text{"}\odot\text{"} \otimes \text{"}\odot\text{"})[\mathbf{h1}_{\mathbf{one}} \otimes (\mathbf{h2}_{\mathbf{one}} \otimes \mathbf{h1}_{\mathbf{two}}) \otimes \mathbf{h2}_{\mathbf{two}}]$$

$$h1_{one} \odot h2_{one} \otimes h1_{two} \odot h2_{two}$$

$$(\mathbf{h1} \otimes \mathbf{h2}) \xrightarrow{\text{"}\odot\text{"}} \xrightarrow{\Delta}$$

$$\text{"}\odot\text{"}[\mathbf{h1} \otimes \mathbf{h2}]_{\mathbf{one}} \otimes \text{"}\odot\text{"}[\mathbf{h1} \otimes \mathbf{h2}]_{\mathbf{two}}$$

$$h1_{one} \odot h2_{one} \otimes h1_{two} \odot h2_{two}$$



$$\mathbf{h} \xrightarrow{\Delta} \xrightarrow{\mathbf{id}\otimes\mathbf{S}} \xrightarrow{\text{"}\odot\text{"}}$$

$$\text{"}\odot\text{"}[(\mathbf{id} \otimes \mathbf{S})[\mathbf{h}_{\mathbf{one}} \otimes \mathbf{h}_{\mathbf{two}}]]$$

$$\text{"}\odot\text{"}[\mathbf{h}_{\mathbf{one}} \otimes \mathbf{S}[\mathbf{h}_{\mathbf{two}}]]$$

$$h_{one} \odot S[h_{two}]$$

$$\mathbf{h} \xrightarrow{\Delta} \xrightarrow{\mathbf{S}\otimes\mathbf{id}} \xrightarrow{\text{"}\odot\text{"}}$$

$$\text{"}\odot\text{"}[(\mathbf{S} \otimes \mathbf{id})[\mathbf{h}_{\mathbf{one}} \otimes \mathbf{h}_{\mathbf{two}}]]$$

$$\text{"}\odot\text{"}[\mathbf{S}[\mathbf{h}_{\mathbf{one}}] \otimes \mathbf{h}_{\mathbf{two}}]$$

$$S[h_{one}] \odot h_{two}$$

$$\mathbf{h} \xrightarrow[k]{\epsilon} \xrightarrow{\eta_1}$$

$$\underset{k}{\in}[h]$$

Proposition 1.3.1 *The antipode of a Hopf algebra is unique and obeys* $S(hg) = S(g)S(h)$, $S(1) = 1$ *(i.e. S is an* antialgebra map*) and* $(S \otimes S) \circ \Delta h = \tau \circ \Delta \circ Sh$, $\epsilon Sh = \epsilon h$ *(i.e. S is an* anticoalgebra map*).*

Proof:

Set h = 1

$\mathbf{1} \xrightarrow{\Delta} \xrightarrow{\mathbf{id}\otimes\mathbf{S}} \xrightarrow{\text{"}\odot\text{"}}$

$\mathbf{\text{"}\odot\text{"}[(id \otimes S)[1 \otimes 1]]}$

$\mathbf{\text{"}\odot\text{"}[1 \otimes S[1]]}$

$S[1]$

$\mathbf{1} \xrightarrow{\Delta} \xrightarrow{\mathbf{S}\otimes\mathbf{id}} \xrightarrow{\text{"}\odot\text{"}}$

$\mathbf{\text{"}\odot\text{"}[(S \otimes id)[1 \otimes 1]]}$

$\mathbf{\text{"}\odot\text{"}[S[1] \otimes 1]}$

$S[1]$

Therefore both Arrows give us S[1] and that has to be

$\mathbf{1} \xrightarrow{\underset{k}{\in}} \xrightarrow{\eta_1}$

$\underset{k}{1}$

Therefore S[1] = 1, where 1 is the unit for the Ring.

Proof: If S, S_1 are two antipodes on a bialgebra H then they are equal because $S_1 h = (S_1 h_{(1)})\epsilon(h_{(2)}) = (S_1 h_{(1)})h_{(2)(1)}Sh_{(2)(2)} = (S_1 h_{(1)})h_{(2)}Sh_{(3)} = (S_1 h_{(1)(1)})h_{(1)(2)}Sh_{(2)} = \epsilon(h_{(1)})Sh_{(2)} = Sh$. Here we wrote $h = h_{(1)}\epsilon(h_{(2)})$ by the counity axiom, and then inserted $h_{(2)(1)}Sh_{(2)(2)}$ knowing that it would collapse to $\epsilon(h_{(2)})$. We then used associativity and (the more novel ingredient) coassociativity to be able to collapse $(S_1 h_{(1)(1)})h_{(1)(2)}$ to $\epsilon(h_{(1)})$.

$\mathbf{(h_{one})} \xrightarrow{\Delta} \xrightarrow{\mathbf{S}\otimes\mathbf{id}} \xrightarrow{\text{"}\odot\text{"}}$

$\mathbf{\text{"}\odot\text{"}[(S \otimes id)[h_{oneone} \otimes h_{onetwo}]]}$

$\mathbf{\text{"}\odot\text{"}[S[h_{oneone}] \otimes h_{onetwo}]}$

$S[h_{oneone}] \odot h_{onetwo}$

$\mathbf{(h_{one})} \xrightarrow{\underset{k}{\in}} \xrightarrow{\eta_1}$

$\underset{k}{\in}[h_{one}]$

$S[h_{oneone}] \odot h_{onetwo}$ collapses to $\underset{k}{\in}[h_{one}]$

now to the proof of the proposition. Applying the antipode axiom to hg, we have $(S(h_{(1)}g_{(1)}))h_{(2)}g_{(2)} = \epsilon(hg) = \epsilon(h)\epsilon(g)$. This is not yet what we

$(\mathbf{h} \odot \mathbf{g_{one}}) \xrightarrow{\Delta} \xrightarrow{\mathbf{S \otimes id}} \xrightarrow{\text{"}\odot\text{"}}$

$\text{"}\odot\text{"}[(\mathbf{S} \otimes \mathbf{id})[(\mathbf{h_{one}} \otimes \mathbf{h_{two}}) \odot (\mathbf{g_{one_{one}}} \otimes \mathbf{g_{one_{two}}})]]$

$\text{"}\odot\text{"}[(\mathbf{S} \otimes \mathbf{id})[\mathbf{h_{one}} \odot \mathbf{g_{one_{one}}} \otimes \mathbf{h_{two}} \odot \mathbf{g_{one_{two}}}]]$

$\text{"}\odot\text{"}[\mathbf{S}[\mathbf{h_{one}} \odot \mathbf{g_{one_{one}}}] \otimes \mathbf{h_{two}} \odot \mathbf{g_{one_{two}}}]$

$S[h_{one} \odot g_{one_{one}}] \odot (h_{two} \odot g_{one_{two}})$

By the diagram above the latter should be same as following:

$(\mathbf{h} \odot \mathbf{g_{one}}) \xrightarrow[\mathbf{k}]{\in} \xrightarrow{\eta_1}$

$\underset{\mathbf{k}}{\in}[\mathbf{h} \odot \mathbf{g_{one}}]$

$\texttt{ringMult}\left[\underset{k}{\in}[h], \underset{k}{\in}[g_{one}], k\right]$

Note that $\underset{k}{\in}[h \odot g_{one}]$ is same as $\underset{k}{\in}[h] \odot \underset{k}{\in}[g_{one}]$ or $\texttt{ringMult}\left[\underset{k}{\in}[h], \underset{k}{\in}[g_{one}], k\right]$.

Ok I am running out of gas here, I will add some more computations for S and give up:

In[344]:=
$(\mathbf{S} \otimes \mathbf{S}) \circ (\Delta\ (\mathbf{h}))$

Out[344]= $(S \otimes S)[h_{one} \otimes h_{two}]$

In[345]:= $(\mathbf{S} \otimes \mathbf{S})[\mathbf{h_{one}} \otimes \mathbf{h_{two}}]$

Out[345]= $S[h_{one}] \otimes S[h_{two}]$

In[346]:=
$\Delta \circ (\mathbf{S}[\mathbf{h}])$

Out[346]= $S[h]_{one} \otimes S[h]_{two}$

Q: $S[h_{one}] \otimes S[h_{two}]$ is same as $S[h]_{one} \otimes S[h]_{two}$ due to linearity of S?

Braid Groups

Straightforward implementation of braid groups, below B_3 to B_5 with group relations in check:

```
(* For braid groups an init is required *)
bgInit[];
```

In[333]:=

```
braidGroup3 = bgConstruct[3];
```

```
braidGroup3[[6]][[1]] // MatrixForm
braidGroup3[[6]][[2]] // MatrixForm
```

Out[334]//MatrixForm=

$$\begin{pmatrix} q^2 t & (-1+q)\, q\, t & 0 \\ 0 & 0 & 1 \\ 0 & q & 1-q \end{pmatrix}$$

Out[335]//MatrixForm=

$$\begin{pmatrix} 1-q & 1 & 0 \\ q & 0 & 0 \\ 0 & (-1+q)\, q^2 t & q^2 t \end{pmatrix}$$

```
braidGroup4 = bgConstruct[4];
```

In[320]:=

```
braidGroup4[[6]][[1]] // MatrixForm
braidGroup4[[6]][[2]] // MatrixForm
braidGroup4[[6]][[3]] // MatrixForm
```

Out[320]//MatrixForm=

$$\begin{pmatrix} q^2 t & (-1+q)\, q\, t & (-1+q)\, q\, t & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & q & 0 & 1-q & 0 & 0 \\ 0 & 0 & q & 0 & 1-q & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Out[321]//MatrixForm=

$$\begin{pmatrix} 1-q & 1 & 0 & 0 & 0 & 0 \\ q & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & (-1+q)\, q^2 t & (-1+q)^2\, q\, t & q^2 t & (-1+q)\, q\, t & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & q & 1-q \end{pmatrix}$$

Out[322]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1-q & 1 & 0 & 0 & 0 \\ 0 & q & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1-q & 1 & 0 \\ 0 & 0 & 0 & q & 0 & 0 \\ 0 & 0 & (-1+q)\, q^3 t & 0 & (-1+q)\, q^2 t & q^2 t \end{pmatrix}$$

In[323]:=
```
Simplify[braidGroup4[[6]][[2]].braidGroup4[[6]][[1]].braidGroup4[[6]][[2]]] ===
 Simplify[braidGroup4[[6]][[1]].braidGroup4[[6]][[2]].braidGroup4[[6]][[1]]]
```
Out[323]= True

In[324]:=
```
Simplify[ArtinBasis[[1]].ArtinBasis[[3]]] // MatrixForm
Simplify[ArtinBasis[[3]].ArtinBasis[[1]]] // MatrixForm

Simplify[ArtinBasis[[1]].ArtinBasis[[3]]] ===
 Simplify[ArtinBasis[[3]].ArtinBasis[[1]]]
```
Out[324]//MatrixForm=

$$\begin{pmatrix} q^2 t & (-1+q) q t & (-1+q) q t & 0 & 0 & 0 \\ 0 & 0 & 0 & 1-q & 1 & 0 \\ 0 & 0 & 0 & q & 0 & 0 \\ 0 & -(-1+q) q & q & (-1+q)^2 & 1-q & 0 \\ 0 & q^2 & 0 & -(-1+q) q & 0 & 0 \\ 0 & 0 & (-1+q) q^3 t & 0 & (-1+q) q^2 t & q^2 t \end{pmatrix}$$

Out[325]//MatrixForm=

$$\begin{pmatrix} q^2 t & (-1+q) q t & (-1+q) q t & 0 & 0 & 0 \\ 0 & 0 & 0 & 1-q & 1 & 0 \\ 0 & 0 & 0 & q & 0 & 0 \\ 0 & -(-1+q) q & q & (-1+q)^2 & 1-q & 0 \\ 0 & q^2 & 0 & -(-1+q) q & 0 & 0 \\ 0 & 0 & (-1+q) q^3 t & 0 & (-1+q) q^2 t & q^2 t \end{pmatrix}$$

Out[326]= True

In[327]:=
```
braidGroup5 = bgConstruct[5];

braidGroup5[[6]][[1]] // MatrixForm
braidGroup5[[6]][[2]] // MatrixForm
braidGroup5[[6]][[3]] // MatrixForm
braidGroup5[[6]][[4]] // MatrixForm


Simplify[braidGroup5[[6]][[2]].braidGroup5[[6]][[1]].braidGroup5[[6]][[2]]] ===
 Simplify[braidGroup5[[6]][[1]].braidGroup5[[6]][[2]].braidGroup5[[6]][[1]]]
```
Out[328]//MatrixForm=

$$\begin{pmatrix} q^2 t & (-1+q) q t & (-1+q) q t & (-1+q) q t & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & q & 0 & 0 & 1-q & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & q & 0 & 0 & 1-q & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & q & 0 & 0 & 1-q & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Out[329]//MatrixForm=

$$\begin{pmatrix} 1-q & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ q & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & (-1+q)\,q^2\,t & (-1+q)^2\,q\,t & (-1+q)^2\,q\,t & q^2\,t & (-1+q)\,q\,t & (-1+q)\,q\,t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & q & 0 & 1-q & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & q & 0 & 1-q & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Out[330]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1-q & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & q & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1-q & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & q & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & (-1+q)\,q^3\,t & (-1+q)^2\,q^2\,t & 0 & (-1+q)\,q^2\,t & (-1+q)^2\,q\,t & q^2\,t & (-1+q)\,q\,t & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & q & 1-q \end{pmatrix}$$

Out[331]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1-q & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & q & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1-q & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & q & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1-q & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & q & 0 & 0 \\ 0 & 0 & 0 & (-1+q)\,q^4\,t & 0 & 0 & (-1+q)\,q^3\,t & 0 & (-1+q)\,q^2\,t & q^2\,t \end{pmatrix}$$

Out[332]= True