Network Rewriting II: Bi- and Hopf Algebras

Lars Hellström

Division of Applied Mathematics, The School of Education, Culture and Communication, Mälardalen University, Box 883, 721 23 Västerås, Sweden
lars.hellstrom@residenset.net

Abstract. Bialgebras and their specialisation Hopf algebras are algebraic structures that challenge traditional mathematical notation, in that they sport two core operations that defy the basic functional paradigm of taking zero or more operands as input and producing one result as output. On the other hand, these peculiarities do not prevent studying them using rewriting techniques, if one works within an appropriate network formalism. This paper restates the traditional axioms as rewriting systems, demonstrating confluence in the case of bialgebras and finding the (infinite) completion in the case of Hopf algebras. A noteworthy minor problem solved along the way is that of constructing a quasi-order with respect to which the rules are compatible.

1 Introduction

Bialgebras and Hopf algebras are rarely mentioned in first (or second) abstract algebra courses, but many familiar algebraic and combinatorial [3] structures possess a Hopf algebra structure, which may be viewed as giving a more complete picture of the basic thing than the mere algebra would. For polynomials in one variable x over a field $\mathcal{K}$, one may define the coproduct $\Delta\colon \mathcal{K}[x] \longrightarrow \mathcal{K}[x] \otimes \mathcal{K}[x]$, the counit $\varepsilon\colon \mathcal{K}[x] \longrightarrow \mathcal{K}$, and the antipode $S\colon \mathcal{K}[x] \longrightarrow \mathcal{K}[x]$ as the linear maps which satisfy

$$\Delta(x^n) = \sum_{k=0}^{n} \binom{n}{k} x^k \otimes x^{n-k}, \quad \varepsilon(x^n) = \begin{cases} 1 & \text{if } n = 0, \\ 0 & \text{otherwise,} \end{cases} \quad S(x^n) = (-1)^n x^n$$

for all $n \geqslant 0$; this turns $\mathcal{K}[x]$ into a Hopf algebra. For any group G, the corresponding group algebra $\mathcal{K}[G]$ is similarly endowed with coproduct $\Delta\colon \mathcal{K}[G] \longrightarrow \mathcal{K}[G] \otimes \mathcal{K}[G]$, counit $\varepsilon\colon \mathcal{K}[G] \longrightarrow \mathcal{K}$, and antipode $S\colon \mathcal{K}[G] \longrightarrow \mathcal{K}[G]$ defined by

$$\Delta(g) = g \otimes g, \qquad \varepsilon(g) = 1, \qquad S(g) = g^{-1}$$

for all $g \in G$ and then extended to the whole of $\mathcal{K}[G]$ by linearity, that turn $\mathcal{K}[G]$ into a Hopf algebra. If G is finite, then the linear dual of $\mathcal{K}[G]$ will moreover also be a Hopf algebra. Hopf algebras are thus close at hand, but they can for *syntactic* reasons be awkward to work with abstractly.

The simplest way to fully formalise the Hopf algebra concept of coproduct in a classical computational context would be that it is a function which returns a generator object for a finite sequence of pairs of algebra elements, because the basic way to encode a general element in a vector space tensor product $V \otimes V$ is as a finite sum $x_1 \otimes y_1 + \cdots + x_n \otimes y_n$ where $x_i, y_i \in V$ for all $i = 1, \ldots, n$; note however that neither the length of this sum nor any particular term of it is uniquely determined by the element that the sum encodes. That the overall result of a computation should be independent of how such a tensor product element happened to get encoded also places far-reaching constraints on what one may do to the x_is and y_is; in particular, all the x_i must be processed in the same way, and all the y_i must be processed in the same way, although x_is need *not* be processed in the same way as the y_i. Hence a more intuitive syntactic interpretation of the coproduct Δ is that it takes one operand as input but produces *two* results as output (one of which is the "sequence" of x_i and the other being the corresponding "sequence" of y_i). In a composite expression, the left and the right results of a coproduct may then be used in quite separate places of the expression as a whole.

The counit ε is syntactically even stranger, as it also takes one operand as input, but produces *no* result as output, although it contributes a global factor to the final result of any composite expression of which it is part. Getting a grip on Δ and ε is difficult, and the main reason for this is precisely that they in the natural interpretation go beyond one of the fundamental principles of mathematical notation, namely that each expression is either atomic or a combination of *independent* subexpressions that each contributes one intermediate result to the final combining operation, thus giving every expression an underlying rooted tree structure. The two output results of a coproduct can instead create a syntactic dependence between what from the root looks like separate subexpressions, and the no output results of a counit can leave an expression syntactically disconnected, in both cases invalidating the traditional presumption that an expression is structured like a tree.

One approach for working with bialgebras has been to device special notational extensions to traditional notation, such as the Sweedler [4] notation which however has the drawback of having the bialgebra axioms built in; it cannot be used if one wishes to study the bialgebra axioms themselves. Another approach has been to give up on traditional expressions and equational reasoning, to rather work in the formalism of category theory: instead of an equational proof, one has a huge commutative diagram, where the various paths correspond to expressions, and the facets correspond to applications of axioms. An awkward trait of this approach is that it places considerable emphasis on such elementary issues as the domains of intermediate results at the expense of more structural aspects (like saying

$$\mathbb{R} \xrightarrow{\sin} [-1,1] \xrightarrow{\mathrm{sqr}} [0,1] \xrightarrow{t \mapsto 1-t} [0,1] \xrightarrow{\mathrm{sqrt}} [0,1]$$

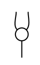
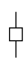
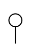
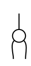
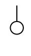
instead of $\sqrt{1 - \sin^2 x}$ while aiming to do basic calculus) and a significant disadvantage is that it requires many steps for trivial rearrangements of parentheses.

The approach followed here, to the end of examining bi- and Hopf algebras using techniques of rewriting, is however instead to adopt a more general expression (formal term) concept, where the underlying structure is a DAG rather than a simple tree. This more general expression concept, the *network*, can be transcribed in terms of the categorical primitives of morphism composition, tensor product, and component permutation, but it is graphical and thus more accessible to the human eye. Even better, the matter of whether two categorical expressions are equal modulo the axioms of a symmetric monoidal category (the "rearrangement of parentheses" mentioned above) turns out to be exactly the same as whether the corresponding networks are isomorphic (as graphs with some extra structure). There is much technical nonsense, but once the many minutiae of establishing the more general expression concept have been taken care of [2], rewriting behaves *very much* as we're used to, even if some new phenomena pop up.

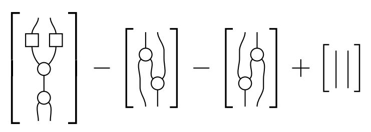
Section 2 gives an introduction to the network formalism for bi- and Hopf algebras. Section 3 presents the axiom system for bialgebras and shows that it constitutes a confluent system of rewrite rules. Section 4 presents the axiom system for Hopf algebras and derives the additional rules needed to make the rewrite system complete. Section 5 shows that the system of the previous section indeed is confluent. Section 6 takes care of a technical detail left aside in the earlier critical pairs/completion oriented sections, namely that of how to construct a compatible order on the set of networks.

2 Network formalism for bi- and Hopf algebras

In a Hopf algebra $\mathcal{H}$ over a field $\mathcal{K}$, there are five multilinear operations:

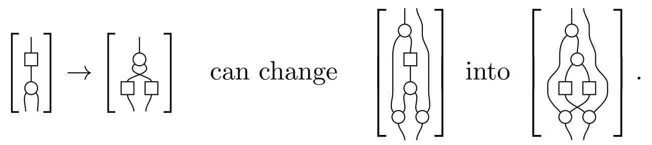
	multiplication $\mu\colon \mathcal{H}\otimes\mathcal{H} \longrightarrow \mathcal{H}$		antipode $S\colon \mathcal{H} \longrightarrow \mathcal{H}$		unit $u\colon \mathcal{K} \longrightarrow \mathcal{H}$
	coproduct $\Delta\colon \mathcal{H} \longrightarrow \mathcal{H}\otimes\mathcal{H}$				counit $\varepsilon\colon \mathcal{H} \longrightarrow \mathcal{K}$

A bialgebra is not required to have an antipode. The graphic symbols shown are used to denote these operations in network notation expressions (see [2, Sec. 5] for the formal definition). These networks will be directed acyclic graphs where each inner vertex is one of the above five, and all edges by convention are directed downwards; no arrowheads are drawn. Edges beginning at the top of the network correspond to inputs and edges ending at the bottom correspond to outputs; together, these constitute the *legs* of the network. A network may be interpreted as a "circuit" performing Hopf algebra operations; any antichain k-edge-cut separating input side from output side then corresponds to an intermediate result of the circuit; technically such an intermediate result is an element of the tensor power $\mathcal{H}^{\otimes k}$. When occurring as parts of a larger mathematical formula, network expressions are for clarity framed in brackets, like so:



The rightmost of these networks is the identity map $\mathcal{H}^{\otimes 2} \longrightarrow \mathcal{H}^{\otimes 2}$, whereas the first three in categorical notation rather would be $\Delta \circ \mu \circ (S \otimes S)$, $(\mathrm{id} \otimes \mu) \circ (\Delta \otimes \mathrm{id})$, and $(\mu \otimes \mathrm{id}) \circ (\mathrm{id} \otimes \Delta)$. The rewriting formalism applied operates on linear combinations of networks, but the bialgebra and Hopf algebra axioms are all binomial, so the reader may for this paper ignore that aspect. (Rewrite rules always have simple networks as left hand sides. Critical pairs/ambiguities thus only arise at simple networks, even though their resolutions might involve linear combinations if there are rules introducing such.)

Rewrite rules act on networks in the pictorially intuitive way of removing a subnetwork isomorphic to the left hand side and instead splicing in a subnetwork isomorphic to the right hand side, making sure that corresponding legs of the left and right hand sides are spliced into the same edge of the network being rewritten. Thus the rule

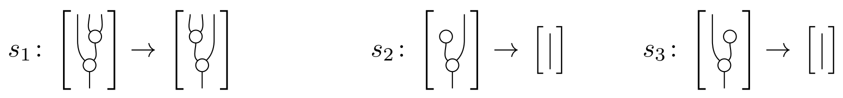


Critical pairs (the formal term used in [2] is *decisive ambiguities*) arise when the left hand sides of two rules occur as overlapping subnetworks of some network that they cover completely, at least in the case of the rewrite systems considered here. (One of the new phenomena hinted at above concerns critical pairs that are neither overlaps nor inclusions; see Example 10.28 of [2] and the discussion there.)

3 The bialgebra axioms and rewriting system

The bialgebra axioms are straightforward to state as network rewrite rules. First, there are the axioms for an associative unital algebra

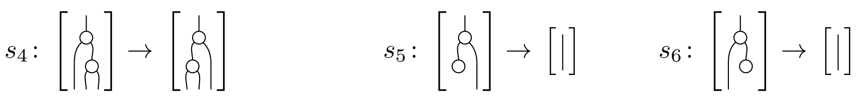
associativity   left unit   right unit



$\mu \circ (\mathrm{id} \otimes \mu) \to \mu \circ (\mu \otimes \mathrm{id})$ $\mu \circ (u \otimes \mathrm{id}) \to \mathrm{id}$ $\mu \circ (\mathrm{id} \otimes u) \to \mathrm{id}$

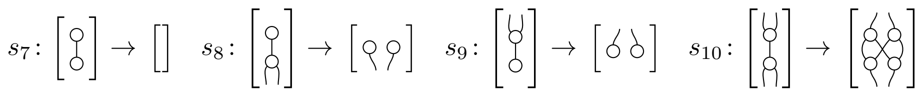
then the dual axioms (obtained from the above by exchanging the roles of inputs and outputs) for a coassociative counital coalgebra

coassociativity   left counit   right counit



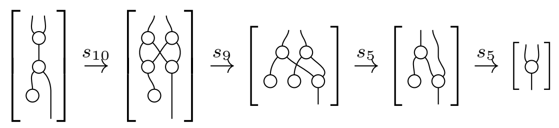
$(\mathrm{id} \otimes \Delta) \circ \Delta \to (\Delta \otimes \mathrm{id}) \circ \Delta$ $(\varepsilon \otimes \mathrm{id}) \circ \Delta \to \mathrm{id}$ $(\mathrm{id} \otimes \varepsilon) \circ \Delta \to \mathrm{id}$

and finally the axioms relating co- and non-co operations

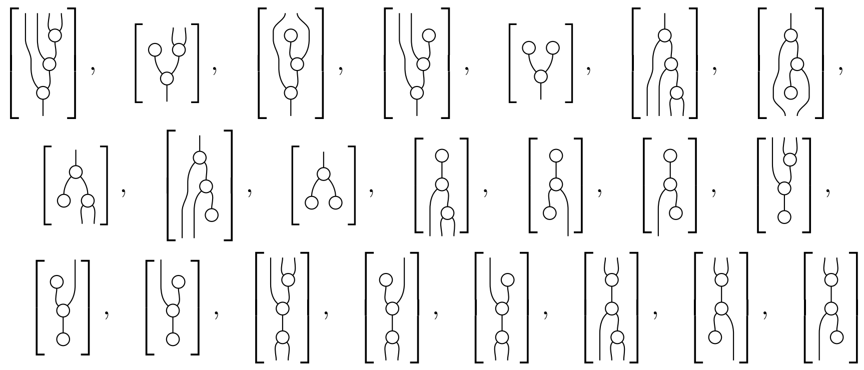


In more traditional formalisms, axioms s_7 and s_9 may be combined into the claim that ε is a unital algebra homomorphism, whereas axioms s_8 and s_{10} combine into the same claim about Δ. The crossing of edges in the right hand side of s_{10} is then swept under the rug as a detail of how the multiplication operation of a tensor product algebra $\mathcal{H} \otimes \mathcal{H}$ is defined, but it is an important feature which deserves to be made explicit.

A sequence of rewriting steps modulo the system $\{s_1, s_2, \dots, s_{10}\}$ is



and that is also half of the resolution of the critical pair formed by rules s_{10} and s_5; the other half amounts to just one application of s_5. The full list of networks being sites of critical pairs for this rewriting system is



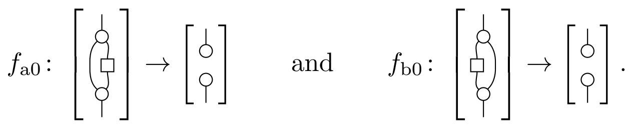
and these all resolve in a quite straightforward manner. This list was compiled by enumerating all networks that satisfy the conditions of [2, Lemma 10.15]. Together with the quasi-order discussed in Section 6, this meets the conditions of the network rewriting diamond lemma [2, Th. 10.24], and so it follows that:

Theorem 1. *The rewriting system $\{s_k\}_{k=1}^{10}$ is terminating and confluent.*

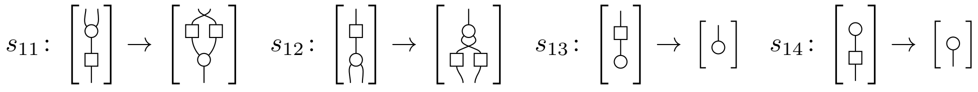
Networks which are built from μ, u, Δ, and ε vertices and moreover are on normal form with respect to $\{s_k\}_{k=1}^{10}$ consist of three layers, and have the overall form $A \circ B \circ C$. The middle B part is a permutation, whereas $A = \bigotimes_{k=1}^{m} M_{p_k}$ and $C = \bigotimes_{k=1}^{n} D_{q_k}$ where $\{p_k\}_{k=1}^{m}, \{q_k\}_{k=1}^{n} \subseteq \mathbb{N}$ are some numbers, $M_0 = u$, $M_1 = \mathrm{id}$, $M_{i+2} = M_{i+1} \circ (\mu \otimes \mathrm{id}^{\otimes i})$ for $i \geqslant 0$, $D_0 = \varepsilon$, $D_1 = \mathrm{id}$, and $D_{i+2} = (\Delta \otimes \mathrm{id}^{\otimes i}) \circ D_{i+1}$ for $i \geqslant 0$. In other words, the A part contain all the μ and u, whereas the C part contain all the Δ and ε, and both the A part and the C part are written on left-leaning form.

4 The Hopf algebra axioms and rewriting system

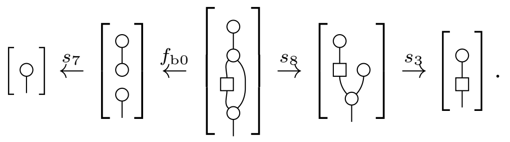
The situation for Hopf algebras is far more complicated. The traditional axiom system for these adds just two axioms to the ten of a bialgebra, namely



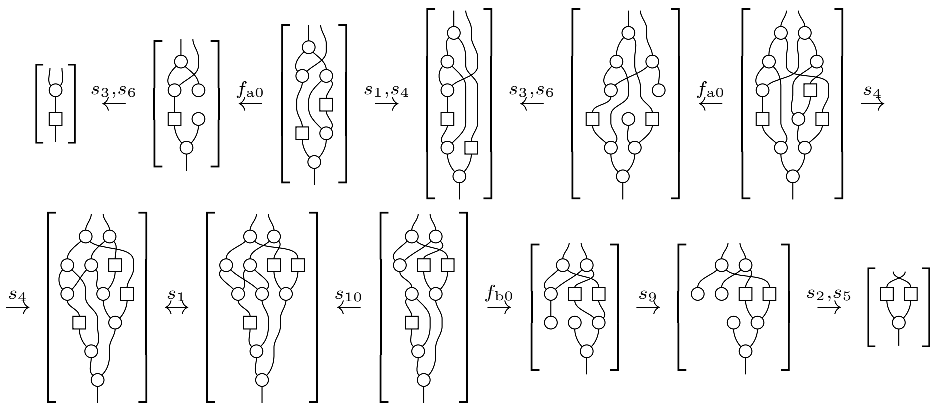
Logically, these two are all that is needed, but in practical calculations one need to employ a number of derived rules. In particular, there are four rules describing interaction of an antipode with one of the four bialgebra operations:



The rules s_{13} and s_{14} for how an antipode interacts with a counit and unit are fairly straightforward; they are among the first things an automated completion procedure discovers when given the Hopf algebra axioms as input, and the derivation of s_{14} is merely

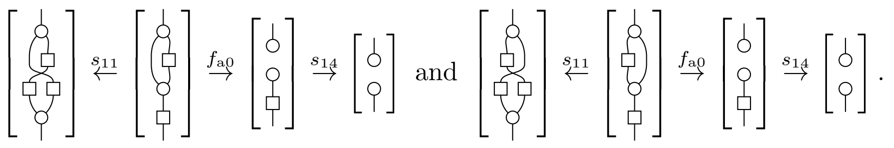


The rules s_{11} and s_{12} for how an antipode interacts with the multiplication and coproduct are on the other hand among the last spurious rules such a procedure discovers; a derivation of s_{11} with some steps combined is

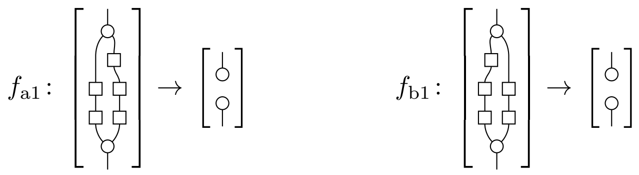


and the derivation of s_{12} is just the vertical flip (exchanging inputs and outputs, multiplication and coproduct, and unit and counit) of this one.

Given rules s_{11} and s_{12}, it is easy to see that these will form critical pairs with the axioms $f_{\mathrm{a}0}$ and $f_{\mathrm{b}0}$ that lead to the failed resolutions

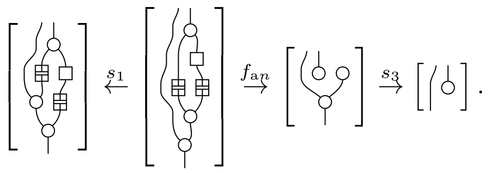


This thus calls for the introduction of two derived rules $f_{\mathrm{c}0}$ and $f_{\mathrm{d}0}$, which are themselves involved in similar critical pairs, that in turn call for another two derived rules with an extra pair of antipodes and an extra crossing. Since crossing twice takes one back to the original uncrossed state, this second pair of derived rules may be called $f_{\mathrm{a}1}$ and $f_{\mathrm{b}1}$ as they look just like $f_{\mathrm{a}0}$ and $f_{\mathrm{b}0}$, except with two extra antipodes on each of the two paths between coproduct and multiplication:



Continuing this way, one will generate four infinite families of rules, where the members of a family differ only in how many extra antipodes are inserted between the coproduct and the multiplication in the left hand side, but all rules in a family have the same right hand side. To state this succinctly in network notation, it becomes convenient to introduce a special *double antipode sequence* vertex ⊞, that denotes a path of some $2n$ antipode vertices; the number n will appear as an index in the rule name. Note especially that all double antipode sequence vertices in a single network denote the *same* even number of antipodes. Using this, the $f_{\mathrm{a}n}$, $f_{\mathrm{b}n}$, $f_{\mathrm{c}n}$, and $f_{\mathrm{d}n}$ families of rules are what is shown in the top row of Figure 1.

Families a–d also form critical pairs with rules s_1 and s_4, that do not resolve using the rules mentioned so far; one example is



These failed resolutions thus give rise to additional families of derived rules; with s_1 one gets $f_{\mathrm{e}n}$ through $f_{\mathrm{h}n}$ and with s_4 one gets $f_{\mathrm{i}n}$ through $f_{\mathrm{l}n}$, also defined in Figure 1. Furthermore families e–h form critical pairs with s_4 that give rise to another four families m–p, and those same four families also arise from critical pairs of s_1 and a member of families i–l. But after these last four that bring the total up to sixteen families $\{f_{\mathrm{a}n}\}_{n=0}^{\infty}$ through $\{f_{\mathrm{p}n}\}_{n=0}^{\infty}$, there are no more

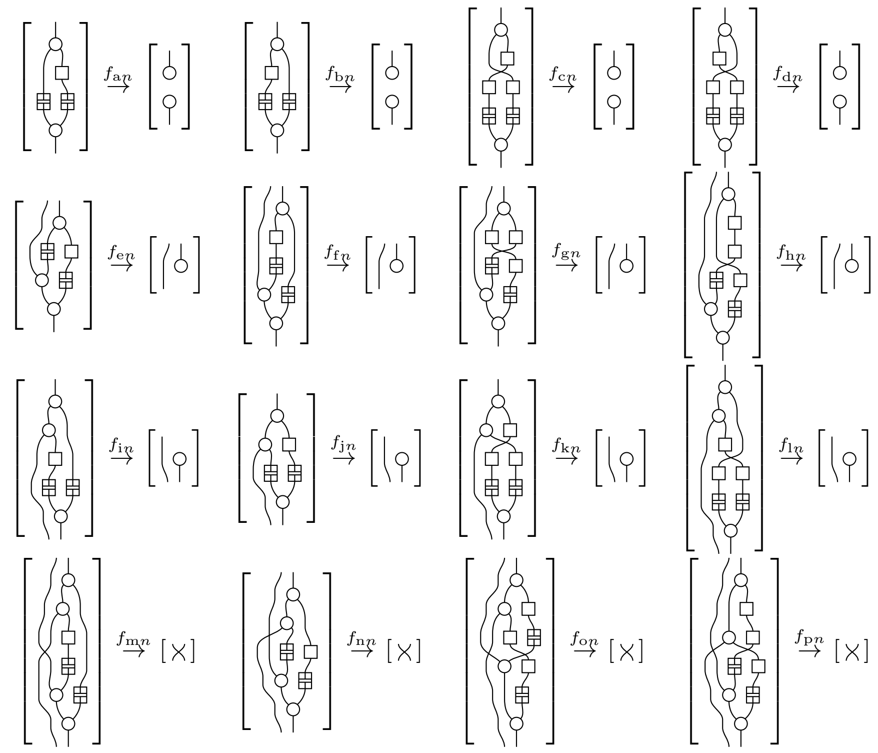
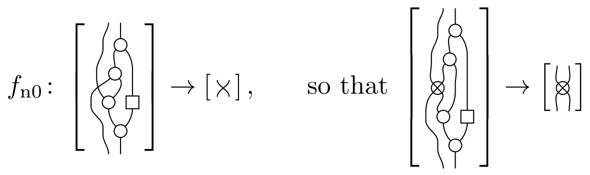


Fig. 1. The sixteen infinite families of rewrite rules for Hopf algebras. The letters a–p in the index correspond to the subequation labels in [2, Eq. 1.2], where this system of rewrite rules was first announced.

derived rules to discover; the rewrite system is, as shall be shown in Section 5, complete.

The last four families, the simplest member of which is



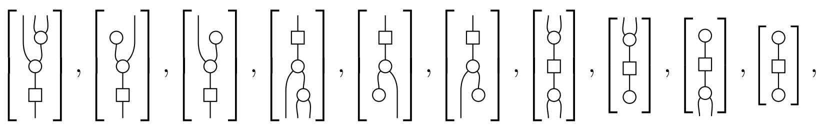
(the $\otimes$ vertex in that formula is intended as a placeholder for an arbitrary network expression) do however exhibit an interesting property: they apply even in places where the first input is reachable from the first output. Note that since networks are by definition DAGs, a rewrite formalism for networks may not perform any surgery that would introduce cycles. A simple condition to that effect would be that left hand sides of rules may only be identified with networks in such a way that no directed path exists from a left hand side output to a left hand side input, because that ensures the result of applying the rule is also acyclic no matter what the right hand side looks like. Though simple, this condition turns out to be a bit too restrictive in practice, and a more appropriate condition is that a rewrite rule should not introduce any *new* connections; the right hand side should not have a path from input j to output i unless there was such a path in the left hand side.

Remember that a derived rule can be considered a bottled sequence of proof steps exercising more elementary rules; any application of a derived rule can, as far as equational reasoning is concerned, be replaced by a sequence of more elementary steps. When considering such a sequence of steps in the context of a larger network, there is no a priori reason why the union of the subnetworks operated upon in the more elementary steps should always be convex, and in general it will indeed not be. What matters for the validity of a derived rule is merely that the elementary steps it is a parcel for can be carried out in every context where the rule is claimed to apply, and that is certainly the case with rule families m–p.

Experience with completing a modification of the Hopf axiom system suggests that rules will typically become nonconvex as soon as they grow complicated enough. An open problem in the more general case is however that there may exist more connections in the intermediate steps of a rule derivation than there are in either the final left or right hand sides. In this case the rule is *non-sharp* [2, Def. 10.3], and it may be involved in critical pairs other than the decisive ambiguities, a matter which requires further research. The rewrite rules considered in this paper are however all sharp, so that is not a concern for the results stated here.

5 Confluence of the Hopf system

For the matter of proving confluence of the system of Hopf algebra rules derived in the previous section, one may begin with the system of the fourteen spurious rules s_1 through s_{14}. Since this is a superset of the bialgebra system, all the critical pairs of that system arise again, but they can also be resolved in exactly the same way as there. The additional critical pairs that arise are at the sites



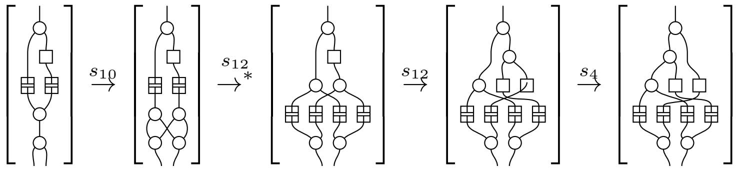
and these also resolve through straightforward calculations. Putting aside until Section 6 the technical details concerning the construction of a compatible order of the set of networks, it now follows that:

Theorem 2. *The rewriting system $\{s_k\}_{k=1}^{14}$ is terminating and confluent.*

The normal form modulo $\{s_k\}_{k=1}^{14}$ of a Hopf algebra expression is a three-layered $A \circ B \circ C$ as in the case of a bialgebra, but with the difference that B in this case may contain antipodes. Hence rather than being a simple matching (as permutations are), the B network is in general a disjoint union of paths where each path may contain any number (including 0) of antipode vertices. Adding the sixteen infinite families to the system will reduce that slightly, but not very much.

Continuing with that full rewriting system $F = \{s_k\}_{k=1}^{14} \cup \{f_{\mathrm{a}n}\}_{n=0}^{\infty} \cup \cdots \cup \{f_{\mathrm{p}n}\}_{n=0}^{\infty}$, one may first observe that the spurious rules s_3, s_6, s_7, s_{13}, and s_{14} do not form any critical pairs with the family rules. Rules s_8 and s_9 form critical pairs, but these all resolve very easily as a unit or counit will effectively gobble any vertex to which it becomes adjacent. Rules s_1, s_2, s_4, s_5, s_{11}, and s_{12} are another matter, as they get involved in a rather complicated dance of transforming rules of one family into rules of another family (or sometimes the same family); Figure 2 gives an overview of how the families connect. If not for the fact that all rules in a family form the same kind of critical pair with a spurious rule, and also that the resolutions are all trivial variations on each other, it would be very hard to see that the resolutions all succeed.

The final spurious rule s_{10} is not only the one most prolific in forming critical pairs with family rules (once for the coproduct at the top, once for the multiplication at the bottom), but also the one which creates the most complicated intermediate steps in their resolutions. A typical sequence is



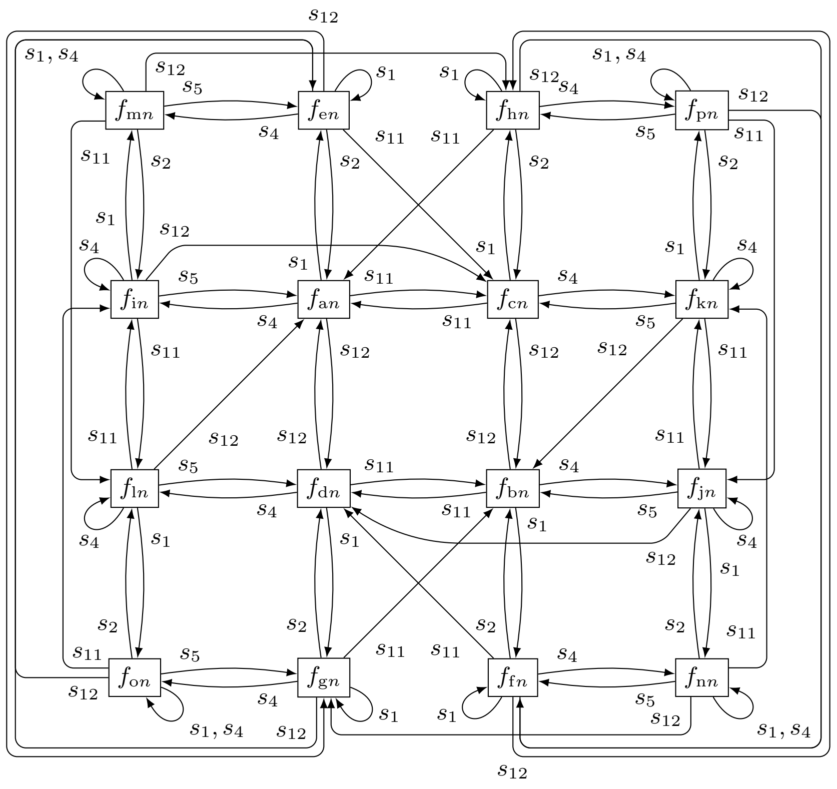


Fig. 2. The effect of spurious rules on family rules. Critical pairs formed by one spurious rule from $\{s_1, s_2, s_4, s_5, s_{11}, s_{12}\}$ and some rule f_{xn} from one of the sixteen infinite families resolve in a number of spurious rule steps and one family rule step; in several but not all cases, these resolutions can alternatively be used as derivations of that family rule. The head of an arrow point at the family of which a member might be derived, whereas the family at the tail and the arrow label correspond to the rules that would be involved in the critical pair.

where now a rule from family $f_{\mathrm{j}n}$ applies on the subnetwork containing the right multiplication, and then a rule from family $f_{\mathrm{a}n}$ resolves the rest. In the middle step it looks unlikely that any family rule can apply, because the crossing introduced by rule s_{10} means that two paths of antipodes that are adjacent on the coproduct side are not adjacent on the multiplication side, but what makes everything fit together is that one of the original paths between coproduct and multiplication has an even number of antipodes whereas the other has an odd number, and thus there is an extra twist at the end which makes one pair of paths adjacent on both sides, after which the resolution becomes straightforward. So even though rule s_{10} creates a lot of noise as far as critical pairs are concerned, it does not really contribute anything interesting here. (Note however that rule s_{10} plays a crucial role at one point in the derivation of rule s_{11}.)

The final case of critical pair would be one formed by two family rules, and although that happens (for example between $f_{\mathrm{a}n}$ and $f_{\mathrm{m}n}$), it does not happen very often. The main reason is that the overlap has to take the form of a path starting in a coproduct, passing some number of antipodes, and ending in a multiplication; this places a strong restriction on the n values that might be involved, as both rules must have such paths with the same number of antipode vertices. Considering in addition that twisted families (c, d, g, h, k, l, o, and p) cannot form overlaps with the straight families (a, b, e, f, i, j, m, and n), one ends up with the conclusion that the n values of both rules involved must in fact be equal, and then it follows that all these critical pairs have trivial resolutions. Thus we have:

Theorem 3. *The full Hopf rewriting system F is terminating and confluent.*

As before, networks on normal form with respect to F can be written as $A \circ B \circ C$ where all the μ and u are in A, all the antipodes S are in B, and all Δ and ε are in C. What is new in the full system is that those arrangements of coproduct, antipodes, and multiplication upon which one of the family rules would act are forbidden, but which arrangements are those? Define a *mid-section path* of a network to be one that begins in a coproduct, have antipodes as inner vertices, and ends in a multiplication. Two mid-section paths are said to be *adjacent* on the coproduct or multiplication side if their outermost vertices on that side are adjacent or coincide. Clearly, the family rules may only apply to pairs of paths that are adjacent on both sides. Moreover, the number of antipodes on the paths in the pair must differ by 1, and the paths must cross (or not cross) depending on whether it is the path with the even number of antipodes that is the longer (or shorter, respectively).

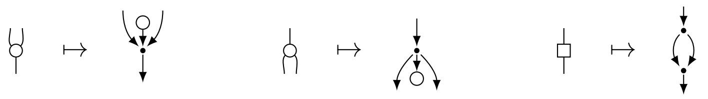
6 Compatible ordering of networks

The diamond lemma in [2] is a descendant of Bergman's diamond lemma for associative algebras [1], so it requires a well-founded quasi-order P on the set of networks, that on one hand is compatible with the rules of the rewriting system, and on the other is strictly preserved under composition of networks. This turns out to not be entirely trivial to construct in this setting.

The main complication is the coproduct–multiplication rule s_{10}, since this has the unhelpful property of *increasing* the number of vertices in a network; were it not for this (and to a lesser extent rules s_{11} and s_{12}), one could have ensured well-foundedness simply by first ordering networks by the number of vertices in them. A generalisation to weighted vertex counts achieves nothing, since one would need $w_\mu + w_u \geqslant 0$ for rules s_2 and s_3, $w_\Delta + w_\varepsilon \geqslant 0$ for rules s_5 and s_6, $w_\mu \geqslant w_\varepsilon$ for rule s_9, $w_\Delta \geqslant w_u$ for rule s_8, and $0 \geqslant w_\mu + w_\Delta$ for rule s_{10}, all of which merely implies that we have equality in all those inequalities. Beyond weighted vertex counts, it is not easy to come up with an ordering principle that is preserved under composition; most elementary suggestions of orders one can make up that do take the structure of an expression into account tend to fail at being preserved under composition.

Might it be better to orient some rules the other way? But no, this is the natural orientation; rules s_8, s_9, s_{10}, s_{11} and s_{12} expand things, whereas most of the others remove superfluous operations, and only the orientations of s_1 and s_4 are really arbitrary. It is natural that s_{11} changes expressions so that antipodes are applied before the multiplication rather than after, so how would one formalise this intuition? Obviously the order in which operations are performed matters, but how does one express that when comparing networks, as the structure of one network can be quite different from the structure of another? One possibility is to compare the sequence in which different vertex types occur along paths from input to output, because in the left hand side of s_{10} each path passes first a μ vertex and then a Δ vertex, whereas in the right hand side it is Δ first and μ second; the same kind of condition works for s_{11} and s_{12}. The only catch is that in order to be preserved under composition of networks, these comparisons must be performed separately for each pair of input and output, and also separately for paths that begin or end within a network.

In the end, it turns out to be sufficient to compare the *number* of paths through a network, provided that vertices are replaced by suitable gadgets, as follows:

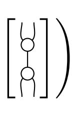
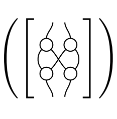


Unfilled circles (like for the unit u and counit ε) may serve as start or end point of a path, but the filled dots may only occur as inner vertices on a path; hence a multiplication μ counts as having three types of paths: paths from the left input passing through, paths from the right input passing through, and paths beginning here and continuing through the output. With these substitutions, it turns out that both the left and right hand sides of s_{10} contribute the same number of paths reaching the boundary of the network (despite the right hand side having 3 edges more), but the left hand side in addition has a path that both starts and ends within the network (starting at the μ, ending at the Δ), which the right hand side does not (since every μ there, where such a path might start, comes after the Δs where it would have to end), and *therefore* the left hand side achieves a greater number of paths than the right hand side. Similarly in rule s_{11}, the antipode doubles the number of paths passing through it, but it is only in the left hand side that the antipode also doubles the number of paths beginning at the multiplication. This is how these rules can be oriented from greater to smaller, even though there are more vertices in the right hand side than in the left hand side.

That is however the intuitive explanation. The technical nonsense is rather that a suitable quasi-order is constructed by pulling back the standard [2, Constr. 6.1] PROP quasi-order [2, Def. 3.1] on the biaffine PROP $\mathrm{Baff}(\mathbb{N})$ [2, Ex. 2.15] along a cleverly chosen PROP homomorphism, namely that g which satisfies

$$g(\mu) = \begin{pmatrix} 1 & \mathit{0} & \mathit{0} & \mathit{0} \\ 0 & 1 & 0 & 0 \\ 0 & \mathit{1} & \mathit{1} & \mathit{1} \end{pmatrix} \quad g(u) = \begin{pmatrix} 1 & \mathit{0} \\ 0 & 1 \\ 0 & \mathit{1} \end{pmatrix} \quad g(\Delta) = \begin{pmatrix} 1 & \mathit{0} & \mathit{1} \\ 0 & 1 & 0 \\ 0 & \mathit{0} & \mathit{1} \\ 0 & \mathit{0} & \mathit{1} \end{pmatrix} \quad g(S) = \begin{pmatrix} 1 & \mathit{0} & \mathit{0} \\ 0 & 1 & 0 \\ 0 & \mathit{0} & \mathit{2} \end{pmatrix}$$
$$g(\varepsilon) = \begin{pmatrix} 1 & \mathit{0} & \mathit{1} \\ 0 & 1 & 0 \end{pmatrix};$$

these conditions uniquely determine a homomorphism, because the set of all networks is the free PROP [2, Th. 8.4]. The italic matrix entries are those that are not fixed for elements of the biaffine PROP and thus may be chosen, although for the resulting PROP order to be strict it is necessary that there is at least one positive element in each row and at least one positive element in each column [2, Cor. 6.5]. The relevant interpretation of an element of $\mathrm{Baff}(\mathbb{N})$ is that the $(i+2, j+2)$ entry keeps track of the number of paths going from input j to output i of a network, whereas entry $(i+2, 2)$ keeps track of the number of paths which begin inside the network and reach output i, entry $(1, j+2)$ keeps track of the number of paths which come from input j but end inside the network, and entry $(1, 2)$ keeps track of the number of paths which both begin and end inside the network. Thus the above argument about counting paths in a gadgetified s_{10} corresponds to the observation that

$g\Bigg($  $\Bigg) = \begin{pmatrix} 1&1&1&1 \\ 0&1&0&0 \\ 0&1&1&1 \\ 0&1&1&1 \end{pmatrix} > \begin{pmatrix} 1&0&1&1 \\ 0&1&0&0 \\ 0&1&1&1 \\ 0&1&1&1 \end{pmatrix} = g\Bigg($  $\Bigg).$

What is not distinguished by the order pulled back over that g are the left and right hand sides of rules s_1 and s_4; since these impose a left–right asymmetry, they would interact poorly with the left–right swaps introduced by rules s_{11} and s_{12}. To orient also these, one introduces a secondary comparison criterion (technically makes a lexicographic composition [2, Constr. 3.7] of quasi-orders) by pulling back along a second cleverly chosen homomorphism g_2, for example that which has

$$g_2(\mu) = \begin{pmatrix} 1 & \mathit{0} & \mathit{0} & \mathit{1} \\ 0 & 1 & 0 & 0 \\ 0 & \mathit{1} & \mathit{1} & \mathit{1} \end{pmatrix} \qquad\qquad g_2(\Delta) = \begin{pmatrix} 1 & \mathit{0} & \mathit{1} \\ 0 & 1 & 0 \\ 0 & \mathit{0} & \mathit{1} \\ 0 & \mathit{1} & \mathit{1} \end{pmatrix}$$

In the path-counting interpretation, this amounts to adding the opportunity to end a path entering through the right input of a μ and begin a path leaving through the right output of a Δ. This causes the third input of a right-leaning $\mu \circ (\mathrm{id} \otimes \mu)$ to offer two chances for a path to end, whereas the third input of a left-leaning $\mu \circ (\mu \otimes \mathrm{id})$ only sees one; this suffices for making the left hand side of s_1 strictly larger than the right hand side.

Two additional results of [2] that are important in verifying that the quasi-order constructed as described above meet the conditions for the diamond lemma (Th. 10.24) are Corollary 9.16 and Lemma 9.18. Arguably also Lemma 3.5 on well-foundedness, but that result is on the other hand quite standard.

Acknowledgements

This work was begun in the spring of 2004, during my postdoc stay at the Mittag-Leffler institute as part of the NOG (Noncommutative Geometry) programme, funded by the European Science Foundation and the Royal Swedish Academy of Sciences. Much research has also been done while the author was associated with the Department of Mathematics and Mathematical Statistics at Umeå University. Final write-up was supported by The School of Education, Culture and Communication at Mälardalen University.

References

1. Bergman, G. M.: The Diamond Lemma for Ring Theory, *Adv. Math.* **29** (1978), 178–218.
2. Hellström, L.: *Network Rewriting I: The Foundation*, 2012; arXiv:1204.2421v1 [math.RA].
3. Joni, S. A. and Rota, G.-C.: Coalgebras and bialgebras in combinatorics. *Stud. Appl. Math.* **61** (1979), no. 2, 93–139.
4. Sweedler, M. E.: *Hopf Algebras*, Benjamin New York, 1969.