

String Diagram Rewriting Modulo Commutative (Co)Monoid Structure

Aleksandar Milosavljević ✉

University College London, UK

Robin Piedeleu ✉ 

University College London, UK

Fabio Zanasi ✉ 

University College London, UK

University of Bologna, Italy

Abstract

String diagrams constitute an intuitive and expressive graphical syntax that has found application in a very diverse range of fields including concurrency theory, quantum computing, control theory, machine learning, linguistics, and digital circuits. Rewriting theory for string diagrams relies on a combinatorial interpretation as double-pushout rewriting of certain hypergraphs. As previously studied, there is a “tension” in this interpretation: in order to make it sound and complete, we either need to add structure on string diagrams (in particular, Frobenius algebra structure) or pose restrictions on double-pushout rewriting (resulting in “convex” rewriting). From the string diagram viewpoint, imposing a full Frobenius structure may not always be natural or desirable in applications, which motivates our study of a weaker requirement: commutative monoid structure. In this work we characterise string diagram rewriting modulo commutative monoid equations, via a sound and complete interpretation in a suitable notion of double-pushout rewriting of hypergraphs.

2012 ACM Subject Classification Theory of computation → Rewrite systems

Keywords and phrases String diagrams, Double-pushout rewriting, Commutative monoid

Digital Object Identifier 10.4230/LIPIcs.CALCO.2023.9

Related Version *Full Version:* <https://arxiv.org/abs/2204.04274>

Funding RP and FZ acknowledge support from EPSRC grant EP/V002376/1.

Acknowledgements We thank Tobias Fritz for helpful discussion and the anonymous reviewers of CALCO for their suggestions.

1 Introduction

String diagrams [28] are a diagrammatic syntax for reasoning algebraically about component-based systems, which in the last few years have found application across diverse fields, including quantum computation [23], digital [20] and electrical circuits [3, 10], machine learning [15], concurrency theory [9], control theory [1, 12], and linguistics [29] amongst others. Compared to traditional syntax, the use of string diagrams allows to neatly visualise resource exchange and message passing between different parts of a system, which is pivotal in studying subtle interactions such as those arising in concurrent processes and quantum computation. Moreover, we can reason with string diagrams both combinatorially and as syntactic, inductively defined objects, which enables forms of compositional analysis typical of programming language semantics.

A cornerstone of string diagrammatic approaches is the possibility of performing *diagrammatic reasoning*: transforming a string diagram according to a certain rewrite rule, which replaces a sub-diagram with a different one. A set of such rules, which typically preserve the semantics of the model, may represent for instance a compilation procedure [27], the realisation of a specification [12], or a refinement of system behaviour [8].



© Aleksandar Milosavljević, Robin Piedeleu, and Fabio Zanasi;
licensed under Creative Commons License CC-BY 4.0

10th Conference on Algebra and Coalgebra in Computer Science (CALCO 2023).

Editors: Paolo Baldan and Valeria de Paiva; Article No. 9; pp. 9:1–9:17

Leibniz International Proceedings in Informatics



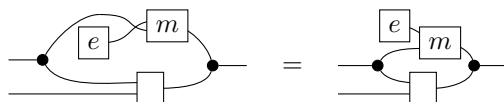
LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

9:2 String Diagram Rewriting Modulo Commutative (Co)Monoid Structure

Compared to traditional term rewriting, a mathematical theory of string diagram rewriting poses new challenges. Formally, string diagrams are graphical representations of morphisms in a category, typically assumed in applications to be a symmetric monoidal category (SMC). In order to perform a rewrite step, we need to match the left-hand side of a rewrite rule to a sub-diagram of a given string diagram. For instance, consider the rewrite rule as on the left below, and the string diagram on the right.



Morally, there is a match for the rule in the string diagram. The issue is that, strictly speaking, such a match does not happen on the nose: we need first to apply the laws of SMCs in order to transform the string diagram into an equivalent one, with the wires into m uncrossed. At this point we have clearly isolated the sub-diagram and are able to perform the rewrite step.



As seen in this example, string diagram rewriting is performed modulo certain structural laws, which reflect the categorical structure in which the string diagrams live. However, from a practical viewpoint, this form of rewriting is not really feasible, as each rewrite step would require us to inspect all string diagrams equivalent to a given one looking for redexes.

This issue can be solved via an interpretation of string diagrams as certain hypergraphs, and of string diagram rewriting as *double-pushout rewriting* (DPO) [14] of such hypergraphs. We refer to [5, 6, 7] for a systematic introduction to this approach. In a nutshell, the benefit of working with such an interpretation is that an equivalence class of string diagrams corresponds to just one hypergraph, meaning that our search for redexes is drastically simplified. However, there is a mismatch: if we want to rewrite string diagrams in a SMC, then soundness is only ensured by adopting a restricted notion of DPO rewriting, called *convex* DPO rewriting [4]. Conversely, if we want to work with arbitrary DPO rewriting steps, then the corresponding notion of string diagram rewriting does not rewrite only modulo the laws of SMCs, but requires a special commutative *Frobenius algebra* on each object of the category. Recall that a Frobenius algebra consists of a commutative monoid and a commutative comonoid, interacting with each other via the so-called Frobenius law [13].

When modelling a certain class of systems with string diagrams, assuming that such Frobenius structure exists is not always reasonable, or desirable. A first class of such examples are matrix-like semantic structures, which are axiomatised by bialgebra equations – see *e.g.* [30] for a survey. It is known that if the monoid and the comonoid both obey the Frobenius and the bialgebra laws, then the equational theory trivialises, *cf.* [17, Ex. 4.3]. A second important class are semantic structures for probability theory, which usually feature a commutative comonoid structure, but no Frobenius equations – introducing Frobenius structure amounts to allowing unnormalised probabilities, *cf.* [22, 18]. These categories, sometimes called *CD-categories*, also play a special role in the study of algebraic theories, because they model the cartesian handling of variables [11].

All these models motivate the study of rewriting for structures intermediate between plain symmetric monoidal and equipped with Frobenius algebras. More specifically, we focus on string diagrams in categories where each object comes with a *commutative monoid* structure.

$$\begin{aligned}
(s; t); u &\equiv s; (t; u), \quad id_n; s \equiv s \equiv s; id_m, \quad (s \oplus t) \oplus u \equiv s \oplus (t \oplus u), \quad id_0 \oplus s \equiv s \equiv s \oplus id_0, \\
id_m \oplus id_n &\equiv id_{m+n}, \quad \sigma_{m,n}; \sigma_{n,m} \equiv id_{m+n}, \quad (s \oplus id_m); \sigma_{m,n} \equiv \sigma_{m,o}; (id_m \oplus s) \\
(s; u) \oplus (t; v) &\equiv (s \oplus t); (u \oplus v), \quad (\sigma_{m,n} \oplus id_o); (id_n \oplus \sigma_{m,o}) \equiv \sigma_{m,n+o},
\end{aligned}$$

■ **Figure 1** Laws of symmetric monoidal categories in a prop, where n, m, o range over \mathbb{N} .

From a rewriting viewpoint, this case is particularly significant because symmetries in a SMC may always create redexes for the commutativity axiom of the monoid multiplication, yielding a non-terminating rewrite system:



Therefore, rather than taking commutativity as a rewrite rule, we need to find an alternative representation of string diagrams (and of string diagram rewriting) that is invariant modulo the axioms of commutative monoids (and the laws of SMCs), which is the focus of this paper. Our contribution is two-fold:

- we identify which class of hypergraphs provides an adequate interpretation of string diagrams in a SMC with commutative monoid structure, and organise them into a SMC. This characterisation will take the form of an isomorphism between the SMC of string diagrams and the SMC of hypergraphs.¹
- We identify which notion of double-pushout hypergraph rewriting interprets string diagram rewriting modulo the axioms of commutative monoids in a sound and complete way.

Note that all of the theory developed in this work can be easily dualised to obtain a framework for rewriting modulo commutative *comonoid* structure, which justifies the title and makes it relevant also for the aforementioned CD categories.

Synopsis. Section 2 recalls background on string diagrams and hypergraphs. Section 3 shows the hypergraph characterisation of string diagrams with a chosen commutative monoid structure. Section 4 shows how string diagram rewriting may be characterised in terms of DPO hypergraph rewriting. We summarise our work and suggest future directions in Section 5. Additional details and missing proofs can be found in [26].

2 Preliminaries

We recall some basic definitions, using the same terminology as [5].

► **Definition 1** (Theories and Props). *A symmetric monoidal theory is a pair (Σ, ε) , where Σ is a monoidal signature i.e. a set of operations $o : m \rightarrow n$ with a fixed arity m and coarity n , and ε is a set of equations, i.e. pairs $\langle l, r \rangle$ of Σ -terms $l, r : v \rightarrow w$ with the same arity and coarity. Σ -terms are constructed by combining the operations in Σ , identities $id_n : n \rightarrow n$ and symmetries $\sigma_{m,n} : m + n \rightarrow n + m$ for each $m, n \in \mathbb{N}$, by sequential $(;)$ and parallel (\oplus) composition. A prop is a symmetric strict monoidal category $(\mathcal{C}, +, 0)$ for which $\text{Ob}(\mathcal{C}) = \mathbb{N}$, the monoidal unit is $0 \in \mathbb{N}$, and the monoidal product on objects is given by addition. The prop freely generated from a symmetric monoidal theory (Σ, ε) has, as*

¹ Simultaneously to a preprint of our work (arXiv:2204.04274), a preprint showing a result closely related to this first contribution also appeared on ArXiv. We comment on their relation in Section 5.

morphisms, the Σ -terms modulo ε and the laws of symmetric monoidal categories recalled in Fig. 1. Given two props \mathcal{C} and \mathcal{D} , a functor $F : \mathcal{C} \rightarrow \mathcal{D}$ is called a prop-morphism from \mathcal{C} to \mathcal{D} if it is an identity-on-objects strict symmetric monoidal functor. Props and prop-morphisms form a category we call PROP.

► **Example 2** (Monoids and Functions). Particularly relevant to our development are the prop \mathbb{F} of functions and the prop **CMon** of commutative monoids. \mathbb{F} has morphisms $f : m \rightarrow n$ the functions from the set $\{0, \dots, m - 1\}$ to $\{0, \dots, n - 1\}$, with monoidal product on functions being their disjoint union. The prop **CMon** is freely generated by the signature consisting of $\mu : 2 \rightarrow 1$ (multiplication) and $\eta : 0 \rightarrow 1$ (unit), and equations expressing commutativity, unitality and associativity of μ .

The theory of commutative monoids presents the prop of functions, in the sense that $\mathbf{CMon} \cong \mathbb{F}$ – see e.g. [24].

► **Example 3** (Cospans). A second prop important for our purposes is the one of cospans. When interpreting string diagrams as hypergraphs, it is fundamental to record the information of what wires are available for composition on the left and right hand side of the diagram: this is achieved by considering cospans of hypergraphs, with the cospan structure indicating which nodes constitute the left and the right interface of the hypergraph.

Given a category \mathbb{C} with finite colimits, let $\mathbf{Csp}(\mathbb{C})$ be the category with the same objects as \mathbb{C} and morphisms $X \rightarrow Y$ the cospans from X to Y , that is, pairs of arrows $X \rightarrow A \leftarrow Y$, for any object A (called the carrier of the cospan). Composition of cospans $X \xrightarrow{f} A \xleftarrow{h} Z$ and $Z \xrightarrow{b} B \xleftarrow{i} Y$ is defined by pushout of the span formed by the middle legs, i.e., it is the² cospan $X \xrightarrow{f:p} Q \xleftarrow{i;q} Y$ where $A \xrightarrow{p} Q \xleftarrow{q} B$ is the pushout of $A \xleftarrow{h} Z \xrightarrow{i} B$. $\mathbf{Csp}(\mathbb{C})$ is symmetric monoidal with the monoidal unit being the initial object $0 \in \mathbb{C}$ and the monoidal product given by the coproduct in \mathbb{C} of the two maps of each cospan.

Hypergraphs [2] generalise graphs by replacing edges with ordered and directed hyperedges, which may have lists of source and target nodes instead of just individual ones. Hypergraphs and hypergraph homomorphisms form a category **Hyp**. As observed in [5], this category may also be defined as a presheaf topos – this is particularly convenient for calculating (co)limits and to ensure that it is adhesive [25], a fundamental property for DPO rewriting. More precisely, **Hyp** is isomorphic to the functor category $\mathbb{F}^{\mathbf{I}}$, where \mathbf{I} has objects the pairs of natural numbers $(k, l) \in \mathbb{N} \times \mathbb{N}$ and an extra object \star , with $k + l$ arrows from (k, l) to \star , for all $k, l \in \mathbb{N}$. A hypergraph G is therefore given by a set G_\star of nodes, and sets $G_{k,l}$ of hyperedges for each $(k, l) \in \mathbb{N} \times \mathbb{N}$, with source maps $s_i : G_{k,l} \rightarrow G_\star$ for $1 \leq i \leq k$ and target maps $t_j : G_{k,l} \rightarrow G_\star$, $1 \leq j \leq l$. A monoidal signature Σ yields a directed hypergraph G_Σ with only a single node and a hyperedge for every Σ -operation $o : k \rightarrow l$ with k sources and l targets (i.e., in $G_{k,l}$). We can use this observation to define the category of Σ -labelled hypergraphs as follows.

► **Definition 4.** The slice category $\mathbf{Hyp} \downarrow G_\Sigma$ is called the category of Σ -labelled hypergraphs and denoted by \mathbf{Hyp}_Σ .

As proven in [5], morphisms in a prop freely generated by a signature Σ may be faithfully interpreted as discrete cospans of Σ -labelled hypergraphs, where the cospan structure represents the interfaces (left and right) of the string diagram. Motivated by this characterisation,

² Note composition is only defined up-to-isomorphism. Strictly speaking, to obtain a (1-)category, we take as morphisms isomorphism classes of cospans (isomorphisms are invertible maps between the carriers that make the obvious diagram commute). We will gloss over this aspect to keep notation light and because the bicategorical aspects do not feature in our development.

we recall from [5] the faithful, coproduct-preserving functor $D : \mathbb{F} \rightarrow \mathbf{Hyp}_\Sigma$ mapping every object $i \in \text{Ob}(\mathbb{F}) = \mathbb{N}$ to the discrete hypergraph with set of nodes $i = \{0, \dots, i-1\}$ and mapping each function to the induced hypergraph homomorphism. We can define the category $\text{Csp}_D(\mathbf{Hyp}_\Sigma)$ of discrete cospans of hypergraphs as the full subcategory of $\text{Csp}(\mathbf{Hyp}_\Sigma)$ (cf. Example 3) on discrete hypergraphs.

3 The Combinatorial Interpretation

In this section we prove that a freely generated prop with a chosen commutative monoid structure is isomorphic to a category of cospans of hypergraphs with certain restrictions (Theorem 21 below).

As shown in [6], the standard interpretation of string diagrams in a prop as cospans of hypergraphs is not full. In order to characterise the image of the interpretation, it is necessary to restrict ourselves to a class of so-called *acyclic* and *monogamous* cospans. In order to prove our result for props with a chosen commutative monoid structure, we may relax this notion to *right-monogamous* cospans, which we now introduce.

► **Definition 5** (Degree of a node [6]). *The in-degree of a node v in hypergraph H is the number of pairs (h, i) where h is a hyperedge with v as its i^{th} target. Similarly, the out-degree of v is the number of pairs (h, j) where h is a hyperedge with v as its j^{th} source.*

► **Definition 6** (Terminal node). *We say that a node v of a hypergraph H is terminal if its out-degree is 0, i.e., if there are no hyperedges of H with source v .*

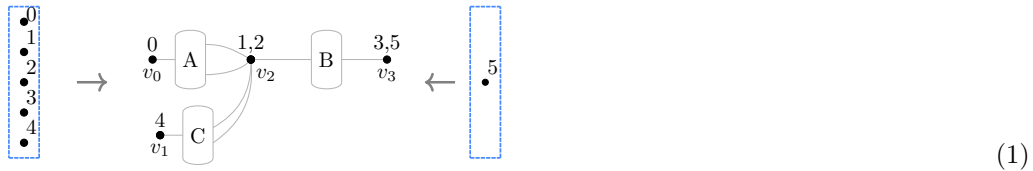
Given $m \xrightarrow{f} H \xleftarrow{g} n$ in $\text{Csp}_D(\mathbf{Hyp}_\Sigma)$, we call inputs of H the set $\text{in}(H)$, defined as the image of f and outputs, the set $\text{out}(H)$ defined as the image of g .

► **Definition 7** (Right-monogamy). *We say that a cospan $m \xrightarrow{f} H \xleftarrow{g} n$ is right-monogamous if g is mono and $\text{out}(H)$ is the set of terminal nodes of H .*

Compared to monogamy [6], right-monogamy does not impose any requirement on f , and only constraints the out-degree of nodes (not the in-degree).

Acyclicity is a standard condition which forbids (directed) loops in a hypergraph, cf. [6, Definition 20] Similarly to monogamous cospans [6], one may verify that acyclic right-monogamous cospans in $\text{Csp}_D(\mathbf{Hyp}_\Sigma)$ form a sub-prop of $\text{Csp}_D(\mathbf{Hyp}_\Sigma)$, denoted by $\text{RMACsp}_D(\mathbf{Hyp}_\Sigma)$.

► **Example 8.** Let us use blue frames to indicate the left and the right interface of a cospan, natural numbers to indicate how the cospan legs are defined, and rounded rectangles to represent hyperedges. The cospan depicted below is right-monogamous.



The notion of right-monogamy is justified by its connection to commutative monoids, and crystallised in the following result. Below, the empty set in $\text{RMACsp}_D(\mathbf{Hyp}_\emptyset)$ refers to the empty signature $\Sigma = \emptyset$.

► **Proposition 9.** *There is an isomorphism of props $\mathbf{CMon} \cong \text{RMACsp}_D(\mathbf{Hyp}_\emptyset)$.*

In particular, the isomorphism interprets the comultiplication and unit as follows:



The fundamental observation leading to Proposition 9 is that the prop of \emptyset -labelled hypergraphs is isomorphic to \mathbb{F} . Right-monogamous cospans in this category coincide with cospans of the form $m \xrightarrow{f} n \xleftarrow{id} n$, and can thus be thought of as morphisms in \mathbb{F} . Paired with the fact that $\mathbf{CMon} \cong \mathbb{F}$ (cf. Example 2), we obtain the above result – see [26] for more details. Following this result, we will sometimes refer abusively to certain functions f as cospans, assuming implicitly that we mean the cospan $m \xrightarrow{f} n \xleftarrow{id} n$.

When referring to “string diagrams with a chosen commutative monoid structure”, we mean morphisms of the prop $\mathbf{S}_\Sigma + \mathbf{CMon}$, the coproduct of the free props over signature Σ and \mathbf{CMon} . Intuitively, such morphisms are obtained by freely combining Σ -terms with terms of the theory of commutative monoids, then quotienting by the laws of symmetric monoidal categories and those of \mathbf{CMon} . For a formal definition of the coproduct of props, see [30]. Our next goal, and the core result of this section, is extending Proposition 9 to the case where Σ is non-empty, *i.e.*, an isomorphism between $\mathbf{S}_\Sigma + \mathbf{CMon}$ and $\mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma)$. This will allow us to refer to $\mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma)$ as the combinatorial characterisation of string diagrams in $\mathbf{S}_\Sigma + \mathbf{CMon}$, and study their rewriting as DPO-rewriting in $\mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma)$.

In order to relate $\mathbf{S}_\Sigma + \mathbf{CMon}$ and $\mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma)$, we will use a strategy analogous to the one used in [6] for theories with symmetric monoidal structure only. In essence, we want to show that $\mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma)$ has the universal property of the coproduct. Consider

$$\mathbf{S}_\Sigma \xrightarrow{[\cdot]} \mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma) \xleftarrow{|\cdot|} \mathbf{CMon}$$

where $[\cdot] : \mathbf{S}_\Sigma \rightarrow \mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma)$ is the faithful prop morphism defined in [6], and $|\cdot| : \mathbf{CMon} \rightarrow \mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma)$ is defined by composing the isomorphism of Proposition 9 with the obvious faithful prop morphism $\mathbf{RMACsp}_D(\mathbf{Hyp}_\emptyset) \rightarrow \mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma)$. To show that $\mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma)$ has the universal property of the coproduct, the fundamental step is investigating how right-monogamous acyclic cospans can be factorised into a composite cospan $\mathcal{M}_0; \mathcal{D}_0; \dots; \mathcal{M}_l; \mathcal{D}_l$ that alternates between monogamous acyclic cospans that are in the image $[\cdot] : \mathbf{S}_\Sigma \rightarrow \mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma)$ and discrete right-monogamous acyclic cospans that are in the image of $|\cdot| : \mathbf{CMon} \rightarrow \mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma)$.

3.1 Weak decomposition

First, we need to show how to decompose right-monogamous cospans of hypergraphs in the same way that we can take sub-diagrams of string diagrams.

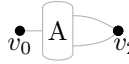
Formally, a sub-diagram c of some larger string diagram d can be defined as a sub-term (modulo the laws of symmetric monoidal categories) of d . It is not difficult to show (by induction) that we can always find some $k \in \mathbb{N}$ and diagrams c_1, c_2 such that $d = c_1; (id_k \oplus l); c_2$, that is, such that d decomposes as

$$\begin{array}{c} n \\ \boxed{d} \\ m \end{array} = \begin{array}{c} n \\ \boxed{c_1} \\ i \end{array} \begin{array}{c} k \\ \boxed{c} \\ j \end{array} \begin{array}{c} \\ \boxed{c_2} \\ m \end{array} \quad (2)$$

In fact, we could also take this decomposition as a definition of sub-diagrams. We turn to the corresponding notion of sub-structure for cospans of hypergraphs.

In the plain symmetric monoidal case, not all sub-hypergraphs of the cospan representation of a string diagram d correspond to sub-diagrams of d . Those that do have the additional properties of being *convex* [6].

► **Definition 10** (Convex sub-hypergraph). *A sub-hypergraph $H \subseteq G$ is convex if, for any nodes v, v' in H and any path p from v to v' in G , every hyperedge in p is also in H .*

► **Example 11.**  is a convex sub-hypergraph of the hypergraph in Example 8.

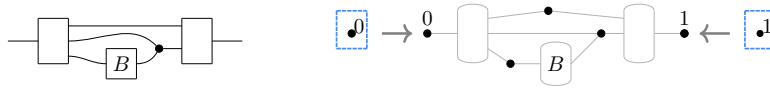
The following lemma shows that convex sub-hypergraphs is the right counterpart of that of sub-diagram, in the hypergraph world. Note the correspondence between (3) below and (2) above.

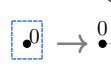

► **Lemma 12** (Weak decomposition). *Let $\mathcal{G} = m \rightarrow G \leftarrow n$ be a right-monogamous acyclic cospan and L be a convex sub-hypergraph of G . We can decompose \mathcal{G} as*

$$(m \rightarrow \tilde{C}_1 \leftarrow i + k); \left(\begin{array}{c} k \xrightarrow{id} k \xleftarrow{id} k \\ \oplus \\ i \rightarrow L \leftarrow j \end{array} \right); (j + k \rightarrow C_2 \leftarrow n) \quad (3)$$

for some $k \in \mathbb{N}$ and where all the above cospans are right-monogamous acyclic.

► **Example 13.** Consider the diagram below with its corresponding cospan representation:



For the convex sub-hypergraph $L := \bullet \text{---} B \text{---} \bullet$, there are two possible choices of weak decomposition, depending on where we attach the second leg of the monoid multiplication that appears in the corresponding string diagram:  or .

Note that this situation differs from the plain symmetric monoidal case [6], where $i \rightarrow L \leftarrow j$ is unique, given L . With commutative monoids, the non-uniqueness comes from having to choose whether we include the monoid structure nodes in the cospan $i \rightarrow L \leftarrow j$ or in the surrounding two cospans. Of course there is a minimal such cospan, that corresponds to the sub-diagram in the diagrammatic decomposition, but we sometimes need the flexibility to choose another decomposition.

3.2 Factorisation into levels

Now we tackle the factorisation of cospans in $\text{RMACsp}_D(\mathbf{Hyp}_\Sigma)$ into alternating monogamous cospans and right-monogamous discrete cospans, to prove that $\text{RMACsp}_D(\mathbf{Hyp}_\Sigma) \cong \mathbf{S}_\Sigma + \mathbf{CMon}$, our main characterisation theorem (Theorem 21).

As we saw, the hypergraphs that correspond to plain string diagrams (in symmetric monoidal categories) are monogamous: nodes are precisely the target and source of one hyperedge. The commutative monoid structure relaxes this requirement for targets. For our last decomposition, we would like to identify nodes that can only appear in the hypergraph representation of diagrams that contain some occurrence of the commutative monoid structure (multiplication or unit), that is, nodes that do not simply represent plain wires. The following definition formalises this idea.

► **Definition 14.** *Let $m \rightarrow G \leftarrow n$ be a right-monogamous acyclic cospan. We say that the node v in G is left-amonogamous if:*

- *it is in $\text{in}(G)$ and its in-degree is not equal to 0, or*
- *it is not in $\text{in}(G)$ and its in-degree is not equal to 1*

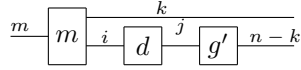
► **Example 15.** In the cospan (1), v_2 and v_3 are left-amonogamous, while v_0 and v_1 are not.

► **Definition 16** (Order of nodes and level of hyperedges). *Let $m \rightarrow G \leftarrow n$ be a right-monogamous acyclic cospan. We define the order of a node v to be the largest number of left-amonogamous nodes preceding it (including itself) on a path leading to v . The level of a hyperedge is the largest number of left-amonogamous nodes on a path ending with v .*

► **Example 17.** In the cospan (1), hyperedges A and C are level-0 hyperedges, and hyperedge B is a level-1 hyperedge.

Recall that we want to obtain a factorisation of any cospan into an alternating composition of discrete right-monogamous cospans – corresponding to diagrams with no generating boxes from the chosen signature – and monogamous cospans – corresponding to plain string diagrams over the chosen signature. We will do this by induction on the maximum level of hyperedges, effectively stripping the necessary cospans (discrete right-monogamous and monogamous) at each level as we move from left to right.

The following lemma will be used at each induction step: here, we require the decomposition to not only alternate between monogamous and discrete right-monogamous cospans, but to also keep track of the order of terminal nodes. Diagrammatically, we want a decomposition of a right-monogamous cospan into the following form:



with m corresponding to a monogamous cospan, and d to a discrete right-monogamous one, and g' is the rest of the decomposition. Here, the first k wires are the terminal order-0 nodes of the overall composite. Keeping track of where terminal nodes of each order are located is an important technical complication that will be needed to prove that the map that we construct out of $\text{RMACsp}_D(\mathbf{Hyp}_\Sigma)$ to show it satisfies the universal property of the coproduct $\mathbf{S}_\Sigma + \mathbf{CMon}$, is a monoidal functor. Recall that, following Proposition 9, we refer abusively to permutations $\pi : n \rightarrow n$ below as cospans, assuming implicitly that we mean the cospan $n \xrightarrow{\pi} n \xleftarrow{id} n$.

► **Lemma 18** (Level-0 decomposition). *Let $m \rightarrow G \leftarrow n$ be a right-monogamous acyclic cospan whose order-0 terminal nodes are the first k nodes of n . Then there exists a unique decomposition of G as $\mathcal{M}; (id_k \oplus (\mathcal{D}; \mathcal{G}'))$, where (A) \mathcal{M} is monogamous acyclic and contains precisely all the level-0 hyperedges; (B) \mathcal{D} is discrete right-monogamous and contains precisely all order-1 left-amonogamous nodes; (C) \mathcal{G}' is right-monogamous acyclic and has no left-amonogamous nodes without any in-connections.*

Moreover, any two such factorisations differ only by permutations of the terminal nodes of the factors, i.e., if $\mathcal{G} = \mathcal{M}; (id_k \oplus (\mathcal{D}; \mathcal{G}')) = \mathcal{M}'; (id_k \oplus (\mathcal{D}'; \mathcal{G}'))$, then there exists permutations π, θ such that $\mathcal{M}' = \mathcal{M}; \pi$, $\pi; (id_k \oplus (\mathcal{D}'; \mathcal{G}')) = id_k \oplus (\mathcal{D}; \mathcal{G}')$, and $\mathcal{D}' = \mathcal{D}; \theta$, $\mathcal{G}'' = \mathcal{G}'$.

Note that the non-uniqueness of the decomposition comes from two distinct sources: 1) arbitrary ordering of nodes on the boundaries of cospans, and 2) the commutativity of the monoid multiplication which can absorb any permutation of the wires that it merges.

► **Lemma 19** (Factorisation into levels). *Any right-monogamous acyclic cospan $\mathcal{G} = m \rightarrow G \leftarrow n$ can be factored into $\mathcal{M}_0; (id_{k_0} \oplus (\mathcal{D}_0; \dots; \mathcal{M}_l; (id_{k_l} \oplus \mathcal{D}_l) \dots)); \pi$ for some permutation π and where, for each i , (A) \mathcal{M}_i is monogamous acyclic and contains precisely the level i hyperedges of \mathcal{G} , and (B) \mathcal{D}_i is discrete right-monogamous, contains all order $i + 1$ left-amonogamous nodes and all order- i terminal nodes of \mathcal{G} .*

Proof. First, let k_i be the set of order- i terminal nodes of \mathcal{G} . We then define the permutation π to be the reordering of n such that the order-0 terminal nodes are the first k_0 nodes, the order-1 are the next k_1 terminal nodes and, more generally, the order- $i + 1$ nodes are the first k_{i+1} terminal nodes after the first $\sum_{j=0}^i k_j$ nodes.

We can now prove the lemma using induction on the highest order of left-amonogamous nodes in $\mathcal{G}; \pi^{-1}$, using Lemma 18.

For the base case we note that any right-monogamous acyclic cospan without any left-amonogamous nodes is simply monogamous acyclic.

For the induction hypothesis, we assume that the statement holds for all the right-monogamous acyclic cospans with the maximum order of left-amonogamous nodes strictly less than r , where r is a positive integer. For the inductive case, suppose that $\mathcal{G}; \pi^{-1}$ is a cospan whose highest order of left-amonogamous nodes is r . Then, by Lemma 18 (Level-0 decomposition), it can be factored into $\mathcal{G}; \pi^{-1} = \mathcal{M}_0; (id_{k_0} \oplus (\mathcal{D}_0; \mathcal{G}'))$ where the first cospan is monogamous acyclic and contains all level 0 hyperedges of \mathcal{G} , and \mathcal{D}_0 is discrete right monogamous with all order 1 left-amonogamous nodes of $\mathcal{G}; \pi^{-1}$ and k_0 all order 0 terminal nodes of $\mathcal{G}; \pi^{-1}$. Now, every node in \mathcal{G}' corresponding to an order i left-amonogamous node in $\mathcal{G}; \pi^{-1}$, is now an order $i - 1$ left-amonogamous. Thus, the highest order of left-amonogamous nodes in \mathcal{G}' is $r - 1$ and, by the induction hypothesis, \mathcal{G}' can be factored into $\mathcal{M}_1; (id_{k_1} \oplus (\mathcal{D}_1; \dots; \mathcal{M}_l; (id_{k_l} \oplus \mathcal{D}_l) \dots))$ as in the statement of the lemma. Therefore, the composite $\mathcal{M}_0; (id_{k_0} \oplus (\mathcal{D}_0; \dots; \mathcal{M}_l; (id_{k_l} \oplus \mathcal{D}_l) \dots)) = \mathcal{M}_0; (id_{k_0} \oplus (\mathcal{D}_0; \mathcal{G}')) = \mathcal{G}; \pi^{-1}$ satisfies conditions (A) and (B) of the lemma and $\mathcal{M}_0; (id_{k_0} \oplus (\mathcal{D}_0; \dots; \mathcal{M}_l; (id_{k_l} \oplus \mathcal{D}_l) \dots)); \pi$ is the factorisation we are looking for. ◀

We will also need the following simpler form of the factorisation into levels, which matches closely the leading intuition of a factorisation into an alternating composition of monogamous and discrete right-monogamous cospans.

► **Corollary 20.** *Any right-monogamous acyclic cospan $\mathcal{G} = m \rightarrow G \leftarrow n$ can be factorised into an alternating sequence of monogamous cospans and discrete right-monogamous cospans, i.e., as $\mathcal{M}_0; \mathcal{D}_0; \dots; \mathcal{M}_l; \mathcal{D}_l$.*

Moreover any two such factorisations differ only by permutations of the terminal nodes of each factor, i.e., if $\mathcal{M}_0; \mathcal{D}_0; \dots; \mathcal{M}_l; \mathcal{D}_l = \mathcal{M}'_0; \mathcal{D}'_0; \dots; \mathcal{M}'_l; \mathcal{D}'_l$, there exists permutations π_i, θ_i such that $\mathcal{M}'_i = \mathcal{M}_i; \pi_i$, $\pi \mathcal{D}'_i = \mathcal{D}_i$ and $\mathcal{D}'_i = \mathcal{D}_i; \theta_i$, $\theta_i \mathcal{M}'_{i+1} = \mathcal{M}_{i+1}$.

Proof. Since identities can be seen as monogamous cospans or discrete right-monogamous, and a permutation can be seen as discrete right-monogamous cospan, if we can get a factorisation of \mathcal{G} into levels as in Lemma 19, we also obtain a factorisation as in the statement of this lemma.

Finally, we can prove by induction, using from the second part of the statement of Lemma 18 that any two such factorisations differ only by some permutation of the factors. ◀

We are now able to conclude with our characterisation theorem.

► **Theorem 21.** *There exists an isomorphism $\langle \cdot \rangle : \mathbf{S}_\Sigma + \mathbf{CMon} \rightarrow \mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma)$.*

Proof. Let us define $\langle \cdot \rangle$ as a copairing (in PROP) of the faithful functors $\llbracket \cdot \rrbracket : \mathbf{S}_\Sigma \rightarrow \mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma)$ and $|\cdot| : \mathbf{CMon} \rightarrow \mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma)$. It suffices to show that the prop $\mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma)$ satisfies the universal property of the coproduct $\mathbf{S}_\Sigma + \mathbf{CMon}$ in PROP:

$$\begin{array}{ccc}
 \mathbf{S}_\Sigma & \xrightarrow{\llbracket \cdot \rrbracket} & \mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma) & \xleftarrow{|\cdot|} & \mathbf{CMon} \\
 & \searrow \alpha & \downarrow \exists! \gamma & \swarrow \beta & \\
 & & \mathbb{A} & &
 \end{array}$$

Given a prop \mathbb{A} and prop-morphisms $\alpha : \mathbf{S}_\Sigma \rightarrow \mathbb{A}$, $\beta : \mathbf{CMon} \rightarrow \mathbb{A}$, we need to prove there exists a unique prop-morphism $\gamma : \mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma) \rightarrow \mathbb{A}$, such that the diagram above commutes. Now, since prop-morphisms are identity-on-objects functors, it is sufficient to consider what happens to the arrows of the above props. Since the diagram needs to commute, for any arrow s in \mathbf{S}_Σ and for any arrows c in \mathbf{CMon} we want $\gamma(\llbracket s \rrbracket) = \alpha(s)$ and $\gamma(|c|) = \beta(c)$. But, by Corollary 20, any cospan \mathcal{G} in $\mathbf{RMACsp}_D(\mathbf{Hyp}_\Sigma)$ can be factorised as an alternating sequence of monogamous cospans and discrete right-monogamous cospans, *i.e.*, as $\mathcal{G} = \mathcal{M}_0; \mathcal{D}_0 \dots; \mathcal{M}_l; \mathcal{D}_l$ where $\mathcal{M}_i = \llbracket s_i \rrbracket$ for some s_i in \mathbf{S}_Σ and $\mathcal{D}_i = |c_i|$ for some c_i in \mathbf{CMon} . Then $\gamma(h_i)$ is uniquely defined by the conditions $\gamma(\llbracket s_i \rrbracket) = \alpha(s_i)$ and $\gamma(|c_i|) = \beta(c_i)$: let $\gamma(\mathcal{G}) = \gamma(\mathcal{M}_0; \mathcal{D}_0; \dots; \mathcal{M}_l; \mathcal{D}_l) = \alpha(s_0)\beta(c_0) \dots \alpha(s_l)\beta(c_l)$. We now verify that γ is well-defined and functorial.

Well-definedness. Since the factorisation of \mathcal{G} into levels is not unique, we need to show that γ is well-defined, *i.e.*, that any two such factorisations will define the same value of $\gamma(\mathcal{G})$. Consider another factorisation $\mathcal{G} = \mathcal{M}'_0; \mathcal{D}'_0; \dots; \mathcal{M}'_l; \mathcal{D}'_l$ obtained from Corollary 20. Then there exists permutations π_i, θ_i such that $\mathcal{M}'_i = \mathcal{M}_i; \pi_i$, $\pi_i \mathcal{D}'_i = \mathcal{D}_i$ and $\mathcal{D}'_i = \mathcal{D}_i; \theta_i$, $\theta_i \mathcal{M}'_{i+1} = \mathcal{M}_{i+1}$. In addition, $\mathcal{M}'_i = \llbracket s'_i \rrbracket$ for some s'_i in \mathbf{S}_Σ , $\mathcal{D}'_i = |c'_i|$ for some c'_i in \mathbf{CMon} . To show well-definedness of γ , we will use the following facts:

1. since \mathbf{S}_Σ , \mathbf{CMon} and \mathbb{A} are props, they all contain a copy of the prop of permutations so we will abuse notation slightly and use the same names to refer to the same permutation in all of them;
2. prop morphisms preserve permutations so that $\alpha(\pi) = \beta(\pi) = \gamma(\pi) = \llbracket \cdot \rrbracket \pi = | \cdot | \pi = \pi$ for any permutation π ;
3. by definition of γ it is clear that $\gamma(\mathcal{G}; \pi) = \gamma(\mathcal{G}); \pi$ and $\gamma(\pi; \mathcal{G}) = \pi; \gamma(\mathcal{G})$.

Now, we have

$$\begin{aligned}
 \gamma(\mathcal{M}_0; \mathcal{D}_0; \dots; \mathcal{M}_l; \mathcal{D}_l) &= \gamma(\mathcal{M}_0); \gamma(\mathcal{D}_0); \dots; \gamma(\mathcal{M}_l); \gamma(\mathcal{D}_l) \\
 &= \gamma(\mathcal{M}_0); \gamma(\pi_0; \mathcal{D}'_0); \dots; \gamma(\mathcal{M}_l); \gamma(\pi_l; \mathcal{D}'_l) \\
 &= \gamma(\mathcal{M}_0); \pi_0; \gamma(\mathcal{D}'_0); \dots; \gamma(\mathcal{M}_l); \pi_l; \gamma(\mathcal{D}'_l) \\
 &= \gamma(\mathcal{M}_0; \pi_0); \gamma(\mathcal{D}'_0); \dots; \gamma(\mathcal{M}_l; \pi_l); \gamma(\mathcal{D}'_l) \\
 &= \gamma(\mathcal{M}'_0); \gamma(\mathcal{D}'_0); \dots; \gamma(\mathcal{M}'_l); \gamma(\mathcal{D}'_l) = \gamma(\mathcal{M}'_0; \mathcal{D}'_0; \dots; \mathcal{M}'_l; \mathcal{D}'_l)
 \end{aligned}$$

Monoidal functoriality. First, γ preserves monoidal products, as the decomposition of a monoidal product is obtained by taking a monoidal product of monogamous acyclic cospans, and a monoidal product of discrete right monogamous cospans, for each level separately. Second, consider two cospans $\mathcal{G} = m \rightarrow G \leftarrow n$ and $\mathcal{H} = n \rightarrow G \leftarrow o$. We can factorise \mathcal{H} as $\mathcal{M}_0; \mathcal{D}_0; \dots; \mathcal{M}_l; \mathcal{D}_l$. Hence, if we can show that $\gamma(\mathcal{G}; \mathcal{M}; \mathcal{D}) = \gamma(\mathcal{G}); \gamma(\mathcal{M}; \mathcal{D})$, for \mathcal{M} monogamous and \mathcal{D} discrete right-monogamous, a simple induction will allow us to conclude that $\gamma(\mathcal{G}; \mathcal{H}) = \gamma(\mathcal{G}); \gamma(\mathcal{H})$. In fact, to show the induction step, it is enough to show that $\gamma(\mathcal{G}; (\mathcal{M} \oplus id_{n-k+l})) = \gamma(\mathcal{G}); \gamma(\mathcal{M} \oplus id_{n-k+l})$ where \mathcal{M} consists of a single hyperedge h , with k source nodes and $l \leq n$ target nodes – we can recover the general case of all monogamous cospans by performing another induction on the number of hyperedges in \mathcal{M} .

Now, we need to understand to what level in $\mathcal{G}; (\mathcal{M} \oplus id_{n-k+l})$ the single hyperedge h of \mathcal{M} belongs. By the definition of the level of hyperedges (Definition 16), h will belong to level i of in $\mathcal{G}; (\mathcal{M} \oplus id_{n-k+l})$ if the node with the largest order in the first k terminal nodes of \mathcal{G} is i . If we assume without loss of generality (as we can always post-compose with a permutation to achieve this), that the terminal nodes of \mathcal{G} are ordered by order size, this implies that the factorisation of $\mathcal{G}; (\mathcal{M} \oplus id_{n-k+l})$ into levels is $\mathcal{G}_{\leq i}; (\mathcal{M} \oplus \mathcal{G}_{> i})$

where $\mathcal{G}_{>i}$ and $\mathcal{G}_{\leq i}$ are obtained from the factorisation $\mathcal{M}_0; (id_{k_0} \oplus (\mathcal{D}_0; \dots; \mathcal{M}_l; (id_{k_l} \oplus \mathcal{D}_l) \dots))$ of \mathcal{G} (from Lemma 19) as follows: $\mathcal{G}_{\leq i} := \mathcal{M}_0; (id_{k_0} \oplus (\mathcal{D}_0; \dots; \mathcal{M}_i) \dots)$ and $\mathcal{G}_{>i} := \mathcal{D}_i; \mathcal{M}_{i+1}; (id_{k_{i+1}} \oplus (\mathcal{D}_{i+1}; \dots; \mathcal{M}_l; (id_{k_l} \oplus \mathcal{D}_l) \dots))$. Note that, by construction, we have $\mathcal{G} = \mathcal{G}_{\leq i}; (id_{k_i} \oplus \mathcal{G}_{>i})$. Thus

$$\begin{aligned} \mathcal{G}; (\mathcal{M} \oplus id_{n-k+l}) &= \mathcal{G}_{\leq i}; (id_{k_i} \oplus \mathcal{G}_{>i}); (\mathcal{M} \oplus id_{n-k+l}) \\ &= \mathcal{G}_{\leq i}; ((id_{k_i}; \mathcal{M}) \oplus (\mathcal{G}_{>i}; id_{n-k+l})) \\ &= \mathcal{G}_{\leq i}; (\mathcal{M} \oplus \mathcal{G}_{>i}) \end{aligned}$$

by the interchange and unitality axioms of symmetric monoidal categories (see Fig. 1).

The intuition now is that we are able to slide the hyperedge h back to level i into the decomposition of \mathcal{G} and that the operation of sliding back – which only uses the monoidal product and composition with identities – is preserved by γ . This will be sufficient to prove functoriality of γ . We have

$$\begin{aligned} \gamma(\mathcal{G}; (\mathcal{M} \oplus id_{n-k+l})) &= \gamma(\mathcal{G}_{\leq i}; (\mathcal{M} \oplus \mathcal{G}_{>i})) \\ &= \gamma(\mathcal{G}_{\leq i}); \gamma(\mathcal{M} \oplus \mathcal{G}_{>i}) \\ &= \gamma(\mathcal{G}_{\leq i}); (\gamma(\mathcal{M}) \oplus \gamma(\mathcal{G}_{>i})) \\ &= \gamma(\mathcal{G}_{\leq i}); (\gamma(id_{k_i}) \oplus \gamma(\mathcal{G}_{>i})); (\gamma(\mathcal{M}) \oplus \gamma(id_{n-k+l})) \end{aligned}$$

where the second equality holds because $\mathcal{G}_{\leq i}; (\mathcal{M} \oplus \mathcal{G}_{>i})$ is the factorisation of $\mathcal{G}; \mathcal{M}$ through which we define γ ; the third equality holds because γ preserves monoidal products and the remaining equalities use the interchange and unitality laws of symmetric monoidal categories as above. Finally, by definition of γ , $\gamma(\mathcal{G}_{\leq i}); (\gamma(id_{k_i}) \oplus \gamma(\mathcal{G}_{>i})) = \gamma(\mathcal{G})$ and, since γ also preserves monoidal products, we can conclude that $\gamma(\mathcal{G}; (\mathcal{M} \oplus id_{n-k+l})) = \gamma(\mathcal{G}); \gamma(\mathcal{M} \oplus id_{n-k+l})$ as we wanted to show. ◀

4 Characterisation of String Diagram Rewriting

Now that we have a characterisation theorem for $\mathbf{S}_\Sigma + \mathbf{CMon}$, we are ready to interpret rewriting modulo commutative monoid structure as DPO rewriting, and to show that such a correspondence is sound and complete. We first recall formally the former notion of rewriting.

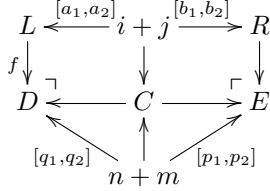
► **Definition 22** (String Diagram Rewriting Modulo \mathbf{CMon}). *Let $d, e: n \rightarrow m$ and $l, r: i \rightarrow j$ be pairs of morphisms in $\mathbf{S}_\Sigma + \mathbf{CMon}$. We say that d rewrites into e modulo commutative monoid structure according to the rewrite rule $\mathcal{R} = \langle l, r \rangle$, notation $d \Rightarrow_{\mathcal{R}} e$, if, in $\mathbf{S}_\Sigma + \mathbf{CMon}$, we have:*

$$\begin{array}{c} n \\ \downarrow \\ \boxed{d} \\ \downarrow \\ m \end{array} = \begin{array}{c} n \\ \downarrow \\ \boxed{c_1} \begin{array}{c} \xrightarrow{k} \\ \boxed{i} \quad \boxed{j} \\ \downarrow \quad \downarrow \\ \boxed{l} \end{array} \boxed{c_2} \\ \downarrow \\ m \end{array} \quad \begin{array}{c} n \\ \downarrow \\ \boxed{e} \\ \downarrow \\ m \end{array} = \begin{array}{c} n \\ \downarrow \\ \boxed{c_1} \begin{array}{c} \xrightarrow{k} \\ \boxed{i} \quad \boxed{j} \\ \downarrow \quad \downarrow \\ \boxed{r} \end{array} \boxed{c_2} \\ \downarrow \\ m \end{array} \quad (4)$$

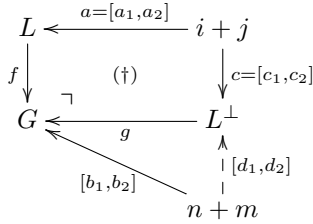
As studied in [5, 6], rewriting of string diagrams may be interpreted as DPO rewriting of the corresponding hypergraphs. The relevant notion is the one of DPO rewriting “with interfaces” (originally used for a single interface in [16], and adapted for two interfaces in [4]), which ensures preservation of the interfaces described by the cospan structure.

9:12 String Diagram Rewriting Modulo Commutative (Co)Monoid Structure

► **Definition 23** (DPO Rewriting (with interfaces)). Consider a DPO rewrite rule $\mathcal{R} = L \xleftarrow{[a_1, a_2]} i + j \xrightarrow{[b_1, b_2]} R$ given by cospans $i \xrightarrow{a_1} L \xleftarrow{a_2}$ and $i \xrightarrow{b_1} R \xleftarrow{b_2}$ in \mathbf{Hyp}_Σ . We say that cospan $n \xrightarrow{q_1} D \xleftarrow{q_2} m$ rewrites into $n \xrightarrow{p_1} E \xleftarrow{p_2} m$ with rule \mathcal{R} , written $(n \rightarrow D \leftarrow m) \xrightarrow{\mathcal{R}} (n \rightarrow E \leftarrow m)$, if there is a cospan $i + j \rightarrow C \leftarrow n + m$ (called the pushout complement) making the diagram below commutes with the two squares being pushouts.

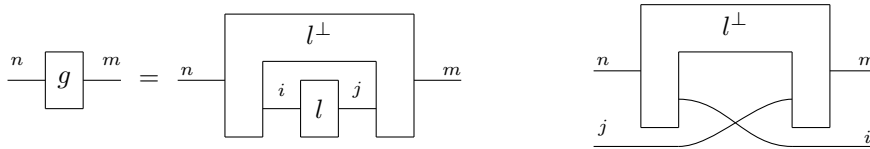


However, unless string diagram rewriting happens modulo the laws of Frobenius algebras, not all DPO rewrites are sound for string diagram rewriting: some pushout complements may yield as outcome of the rewriting hypergraphs that are not in the image of any string diagram [6]. To avoid these situations, [6] introduced the notion of boundary complements and convex matching. The former guarantees that inputs can only be connected to outputs and vice-versa, while the latter are matches that do not contain directed paths from outputs to inputs, *i.e.*, monomorphisms whose image is convex. However, these notions were designed for monogamous hypergraphs, and string diagram rewriting modulo symmetric monoidal structure. In order to capture the correct notion of DPO rewriting for right-monogamous hypergraphs, and rewriting modulo commutative monoid structure, we need to relax the first slightly to that of *weak boundary complements*.



► **Definition 24** (Weak boundary complement). For right-monogamous acyclic cospans $i \xrightarrow{a_1} L \xleftarrow{a_2} j$ and $n \xrightarrow{b_1} G \xleftarrow{b_2} m$ and a morphism $f : L \rightarrow G$, a pushout complement as on the right above is called a *weak boundary complement* if: (A) given two nodes in L that are mapped to the same node in G by f , they must be in the image of a_2 ; (B) c_1 is mono; (C) no two nodes are both in the image of c_1 and c_2 ; (D) there exist $d_1 : n \rightarrow L^\perp$ and $d_2 : m \rightarrow L^\perp$ making the above diagram commute and such that $n + j \xrightarrow{[c_2, d_1]} L^\perp \xleftarrow{[c_1, d_2]} m + i$ is right-monogamous.

Intuitively, the complement L^\perp is G with an L -shaped hole:



where $g : n \rightarrow m$, $l : i \rightarrow j$, and $l^\perp : n + j \rightarrow m + i$ are diagrams for the cospans $n \rightarrow G \leftarrow m, i \rightarrow L \leftarrow j$, and $n + j \rightarrow L^\perp \leftarrow m + i$ respectively, *i.e.* such that $\langle\langle g \rangle\rangle = (n \rightarrow G \leftarrow m), \langle\langle l \rangle\rangle = i \rightarrow L \leftarrow j$, and $\langle\langle l^\perp \rangle\rangle = n + j \rightarrow L^\perp \leftarrow m + i$. (Recall that

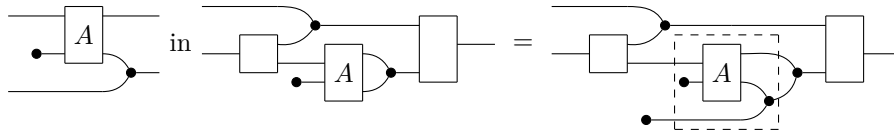
$\langle\langle \cdot \rangle\rangle : \mathbf{S}_\Sigma + \mathbf{CMon} \rightarrow \mathbf{RMAC}_{\text{sp}_D}(\mathbf{Hyp}_\Sigma)$ is the isomorphism established by Theorem 21; we will use it quite liberally from now on in order to manipulate cospans as string diagrams when convenient). Boundary complements restrict the shape that these can take. Let us explain the conditions of Definition 24 in plainer language.

- Condition (A) allows matches to occur in a diagram G that contains the sub-diagram L potentially with some nodes identified, *i.e.* wires connected by the monoid multiplication (see Example 26 below. However, these can only occur as terminal nodes, that is, in the image of a_2 , the right boundary of the subdiagram L).
- Plain boundary complements [6] require c_1, c_2 to be jointly monic. This enforces two distinct properties: it prevents nodes from the left and right boundaries of the match to be identified, and it prevents nodes from within each of the two boundary sides to be identified. Here, we need to relax the second condition to allow nodes in the right boundary of the match to be identified. This is what conditions (B) and (C) give us.
- Condition (D) forces the boundary of the complement, both with the subdiagram L and those of the larger diagram G , to be right-monogamous. In other words, we want the cospan $n + j \xrightarrow{[c_2, d_1]} L^\perp \xleftarrow{[c_1, d_2]} m + i$ depicted above to be right-monogamous.

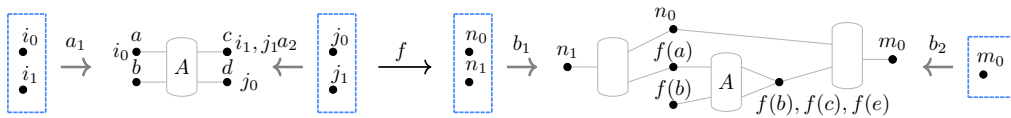
The last ingredient we require is the same as in [6]: we require the match to be *convex*.

► **Definition 25** (Convex matching [6]). $f : L \rightarrow G$ in \mathbf{Hyp}_Σ is a convex match if its image is a convex sub-hypergraph of G .

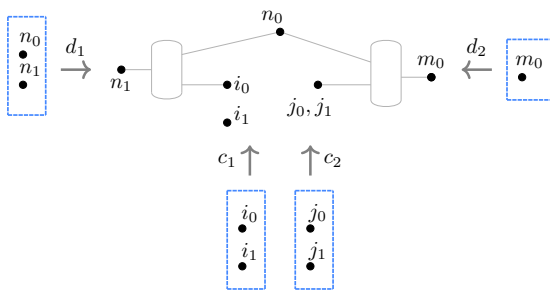
► **Example 26.** Consider the diagram below



As cospans of hypergraphs, this corresponds to the convex matching below



with the following weak boundary complement:



► **Definition 27** (Weakly Convex DPO Rewriting). We call a DPO rewriting step as in Definition 23 weakly convex if $f : L \rightarrow D$ is a convex matching and $i + j \rightarrow C$ is a weak boundary complement in the leftmost pushout square.

Note that, contrary to boundary complements in the symmetric monoidal case [6], weak boundary complements are not necessarily unique if they exist. We can now conclude the soundness and completeness of weakly convex DPO rewriting for string diagrams with commutative monoid structure.

9:14 String Diagram Rewriting Modulo Commutative (Co)Monoid Structure

► **Theorem 28.** *Let $\mathcal{R} = \langle l, r \rangle$ be a rewrite rule on $\mathbf{S}_\Sigma + \mathbf{CMon}$. Then,*

$$d \Rightarrow_{\mathcal{R}} e \text{ iff } \langle\langle d \rangle\rangle \hookrightarrow_{\langle\langle \mathcal{R} \rangle\rangle} \langle\langle e \rangle\rangle \quad (5)$$

Proof. For the direction from left to right we proceed as follows. From the definition of rewriting, and given the assumption $d \Rightarrow_{\mathcal{R}} e$, we have equalities as in (4). We now interpret the string diagrams involved, obtaining right-monogamous cospans:

$$\begin{aligned} \left(n \xrightarrow{q_1} D \xleftarrow{q_2} m \right) &:= \langle\langle d \rangle\rangle & \left(n \xrightarrow{p_1} E \xleftarrow{p_2} m \right) &:= \langle\langle e \rangle\rangle \\ \left(i \xrightarrow{a_1} L \xleftarrow{a_2} j \right) &:= \langle\langle l \rangle\rangle & \left(i \xrightarrow{b_1} R \xleftarrow{b_2} j \right) &:= \langle\langle r \rangle\rangle \\ \left(n \xrightarrow{x_1} C_1 \xleftarrow{x_2} k + i \right) &:= \langle\langle c_1 \rangle\rangle & \left(k + j \xrightarrow{y_1} C_2 \xleftarrow{y_2} m \right) &:= \langle\langle c_2 \rangle\rangle \end{aligned} \quad (6)$$

From the last two cospans above, by simply rearranging nodes on the interface from the left to the right and viceversa, we obtain:

$$i + k \xrightarrow{\tilde{x}_1} \tilde{C}_1 \xleftarrow{\tilde{x}_2} n \quad k + j \xrightarrow{\tilde{y}_1} \tilde{C}_2 \xleftarrow{\tilde{y}_2} m$$

We now define a cospan $i + j \rightarrow C \leftarrow n + m$ as:

$$\left(i + j \xrightarrow{z_1} i + k + j \xleftarrow{z_2} i + k + k + j \right); \left(\begin{array}{c} i + k \xrightarrow{\tilde{x}_1} \tilde{C}_1 \xleftarrow{\tilde{x}_2} n \\ \oplus \\ k + j \xrightarrow{\tilde{y}_1} \tilde{C}_2 \xleftarrow{\tilde{y}_2} m \end{array} \right) \quad (7)$$

where $z_1: (i + j) \rightarrow (i + k + j)$ is the inclusion map, $z_2: (i + k + k + j) \rightarrow (i + k + j)$ is defined as $id_i + \mu_k + id_j$, with $\mu_k: k + k \rightarrow k$ mapping both copies of node $i \in \{0, \dots, k - 1\}$ to i . Intuitively, $i + j \rightarrow C \leftarrow n + m$ represents the string diagram where we have rearranged interface nodes n and i on the opposite interface. One may verify that:

$$\begin{aligned} \left(0 \rightarrow D \xleftarrow{[q_1, q_2]} n + m \right) &= \left(0 \rightarrow L \xleftarrow{[a_1, a_2]} i + j \right); (i + j \rightarrow C \leftarrow n + m) \\ \left(0 \rightarrow E \xleftarrow{[p_1, p_2]} n + m \right) &= \left(0 \rightarrow R \xleftarrow{[b_1, b_2]} i + j \right); (i + j \rightarrow C \leftarrow n + m) \end{aligned} \quad (8)$$

Recall that composition of cospans is obtained via pushouts, hence the two equalities of (8) yield a DPO rewriting step $\langle\langle d \rangle\rangle \hookrightarrow_{\langle\langle \mathcal{R} \rangle\rangle} \langle\langle e \rangle\rangle$ as in Definition 23. Since l is simply a sub-string diagram of d , the mapping from L to D is a convex match. Furthermore, note that no two nodes from $i + k$ can be identified with each other, hence $i \rightarrow C$ is mono, and no node from i can be identified with any node in j or m . As $m \rightarrow C$ is trivially mono, we have that C is indeed a weak boundary complement.

Now we deal with the converse implication. Assume $\langle\langle d \rangle\rangle \hookrightarrow_{\langle\langle \mathcal{R} \rangle\rangle} \langle\langle e \rangle\rangle$, where $\langle\langle d \rangle\rangle$, $\langle\langle e \rangle\rangle$, $\langle\langle l \rangle\rangle$, and $\langle\langle r \rangle\rangle$ are defined as the cospans in (6). By assumption, and since composition of cospans is performed via pushouts, there exists a weak boundary complement $i + j \xrightarrow{[c_1, c_2]} L^\perp \xleftarrow{[d_1, d_2]} n + m$ such that

$$\begin{aligned} \langle\langle d \rangle\rangle &= (0 \rightarrow i \xleftarrow{\mu_i} i + i); (id_i \oplus \langle\langle l \rangle\rangle); (i + j \xrightarrow{[c_1, c_2]} L^\perp \xleftarrow{[d_1, d_2]} n + m) \\ \langle\langle e \rangle\rangle &= (0 \rightarrow i \xleftarrow{\mu_i} i + i); (id_i \oplus \langle\langle r \rangle\rangle); (i + j \xrightarrow{[c_1, c_2]} L^\perp \xleftarrow{[d_1, d_2]} n + m) \end{aligned}$$

We can now apply Lemma 12 (weak decomposition) to $n + j \xrightarrow{[c_2, d_1]} L^\perp \xleftarrow{[c_1, d_2]} m + i$, with the convex sub-hypergraph of L^\perp given by the image of i and j , to obtain a decomposition of $n + j \xrightarrow{[c_2, d_1]} L^\perp \xleftarrow{[c_1, d_2]} m + i$ as

$$(n + j \rightarrow C_1 \leftarrow k + i + j); (k + i + j \xrightarrow{id_k \oplus \sigma_i^j} k + j + i \leftarrow k + j + i); (k + j + i \rightarrow C_2 \leftarrow m + i)$$

for some $k \in \mathbb{N}$, right monogamous cospans $n \rightarrow C_1 \leftarrow i + k$ and $j + k \rightarrow C_2 \leftarrow m$, and where $\sigma_i^j : i + j \rightarrow j + i$ the map that swaps the two components i and j . By fullness of $\langle\langle \cdot \rangle\rangle$ we have c_1, c_2 such that $\langle\langle c_1 \rangle\rangle = n \rightarrow C_1 \leftarrow i + k$ and $\langle\langle c_2 \rangle\rangle = j + k \rightarrow C_2 \leftarrow m$; moreover we have, by construction:

$$\begin{aligned} \langle\langle d \rangle\rangle &= (id_n \oplus (0 \rightarrow i \xleftarrow{\mu_i} i + i)); \langle\langle \begin{array}{c} n \\ \xrightarrow{\quad} \boxed{l} \xrightarrow{j} \\ \xleftarrow{i} \end{array} \rangle\rangle; \langle\langle \begin{array}{c} k \\ \xrightarrow{\quad} \boxed{c_1} \xrightarrow{\quad} \boxed{c_2} \xrightarrow{m} \\ \xleftarrow{i} \end{array} \rangle\rangle; (id_m \oplus i + i \xrightarrow{\mu_i} i \leftarrow 0) \\ \langle\langle e \rangle\rangle &= (id_n \oplus (0 \rightarrow i \xleftarrow{\mu_i} i + i)); \langle\langle \begin{array}{c} n \\ \xrightarrow{\quad} \boxed{r} \xrightarrow{j} \\ \xleftarrow{i} \end{array} \rangle\rangle; \langle\langle \begin{array}{c} k \\ \xrightarrow{\quad} \boxed{c_1} \xrightarrow{\quad} \boxed{c_2} \xrightarrow{m} \\ \xleftarrow{i} \end{array} \rangle\rangle; (id_m \oplus (i + i \xrightarrow{\mu_i} i \leftarrow 0)) \end{aligned}$$

Computing these cospans, we obtain $\langle\langle d \rangle\rangle = \langle\langle c_1 \rangle\rangle; (\langle\langle id_k \rangle\rangle \oplus \langle\langle l \rangle\rangle); \langle\langle c_2 \rangle\rangle$ and $\langle\langle e \rangle\rangle = \langle\langle c_1 \rangle\rangle; (\langle\langle id_k \rangle\rangle \oplus \langle\langle r \rangle\rangle); \langle\langle c_2 \rangle\rangle$. By monoidal functoriality of $\langle\langle \cdot \rangle\rangle$, we thus have $\langle\langle d \rangle\rangle = \langle\langle (c_1; (id_k \oplus l); c_2) \rangle\rangle$ and $\langle\langle e \rangle\rangle = \langle\langle (c_1; (id_k \oplus r); c_2) \rangle\rangle$. Finally, since $\langle\langle \cdot \rangle\rangle$ is faithful, we can conclude that $d = c_1; (id_k \oplus l); c_2$ and $e = c_1; (id_k \oplus r); c_2$. This is precisely what it means to apply the rule $\langle l, r \rangle$ to d , so that $d \Rightarrow_{\emptyset} e$ as we wanted to prove. \blacktriangleleft

5 Conclusions and Future Work

The main contribution of this work is twofold. First, with Theorem 21, we identified a combinatorial representation of string diagrams modulo commutative monoid structure. This correspondence relies on introducing a notion of right monogamous cospans, which is intermediate between the “vanilla” cospans characterising string diagrams modulo Frobenius structure, and the monogamous cospans characterising string diagrams modulo symmetric monoidal structure. The characterisation result relies on a factorisation result for right monogamous cospans, which requires some ingenuity: compared with similar theorems in [5, 6] the increased sophistication is due to the fact that there is additional structure to consider both on the side of string diagrams (in contrast with [6, Theorem 25], which only accounts for symmetric monoidal structure) and of hypergraphs (in contrast with [5, Theorem 4.1], which accounts for generic hypergraph without monogamicity conditions). Note the work [19], which appeared at the same time as a preprint of our work [26], provides a result dual to Theorem 21: instead of monoids, they consider props with a chosen commutative *comonoid* structure – called “CD-categories” or “gs-categories”. On the side of hypergraphs, instead of restricting monogamicity to right-monogamicity, they consider *left-monogamicity*, which is essentially the dual notion.

The second contribution of our paper, Theorem 28, showed a correspondence between string diagrams rewriting modulo commutative monoid structure and a certain variant of DPO hypergraph rewriting. In order to ensure soundness and completeness, we introduced a suitable restriction of DPO rewriting, called *weakly convex* to echo the convex rewriting characterising string diagrams in a symmetric monoidal category [6]. A subtlety in this result was identifying a notion of boundary complement which, even though could not be unique “on-the-nose” as the one considered in the more restrictive convex rewriting, it was sufficiently well-behaved for the purpose of showing the correspondence with string diagram rewriting.

Going forward, we believe our approach could be extended to coloured props in a rather straightforward way, following the analogous developments in [31, 6]. Other interesting directions to pursue are the study of confluence [7] and the characterisations of notions of rewriting modulo structures intermediate between commutative monoid and Frobenius algebra – comparison with the very recent work on rewriting in traced comonoid structure [21] seems particularly promising in this regard. In terms of case studies, as mentioned in the introduction, our work paves the way for studying rewriting of theories which do not host a Frobenius structure, but at the same time include commutative (co)monoid equations, which would immediately lead to non-termination if taken as rewrite rules. Categories of matrix-like structures, based on Hopf algebras which would become degenerate if added Frobenius equations (see *eg.* [30] for an overview), seem a particularly fitting candidate for investigation.

References

- 1 John Baez and Jason Erbele. Categories in control. *Theory and Applications of Categories*, 30:836–881, 2015. URL: <http://arxiv.org/abs/1405.6881>.
- 2 C. Berge. *Graphs and Hypergraphs*. Elsevier Science Ltd., GBR, 1985.
- 3 Guillaume Boisseau and Pawel Sobocinski. String diagrammatic electrical circuit theory. In *ACT*, volume 372 of *EPTCS*, pages 178–191, 2021.
- 4 Filippo Bonchi, Fabio Gadducci, Aleks Kissinger, Pawel Sobocinski, and Fabio Zanasi. Rewriting modulo symmetric monoidal structure. In Martin Grohe, Eric Koskinen, and Natarajan Shankar, editors, *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016*, pages 710–719. ACM, 2016. doi:10.1145/2933575.2935316.
- 5 Filippo Bonchi, Fabio Gadducci, Aleks Kissinger, Pawel Sobocinski, and Fabio Zanasi. String diagram rewrite theory I: Rewriting with Frobenius structure, 2020. doi:10.48550/ARXIV.2012.01847.
- 6 Filippo Bonchi, Fabio Gadducci, Aleks Kissinger, Pawel Sobocinski, and Fabio Zanasi. String diagram rewrite theory II: Rewriting with symmetric monoidal structure, 2021. doi:10.48550/ARXIV.2104.14686.
- 7 Filippo Bonchi, Fabio Gadducci, Aleks Kissinger, Pawel Sobociński, and Fabio Zanasi. String diagram rewrite theory III: Confluence with and without Frobenius, 2021. doi:10.48550/ARXIV.2109.06049.
- 8 Filippo Bonchi, Joshua Holland, Dusko Pavlovic, and Pawel Sobocinski. Refinement for signal flow graphs. In *CONCUR*, volume 85 of *LIPICs*, pages 24:1–24:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.
- 9 Filippo Bonchi, Joshua Holland, Robin Piedeleu, Pawel Sobocinski, and Fabio Zanasi. Diagrammatic algebra: from linear to concurrent systems. *Proc. ACM Program. Lang.*, 3(POPL):25:1–25:28, 2019. doi:10.1145/3290338.
- 10 Filippo Bonchi, Robin Piedeleu, Pawel Sobocinski, and Fabio Zanasi. Graphical affine algebra. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019*, pages 1–12. IEEE, 2019. doi:10.1109/LICS.2019.8785877.
- 11 Filippo Bonchi, Pawel Sobocinski, and Fabio Zanasi. Deconstructing Lwvere with distributive laws. *J. Log. Algebraic Methods Program.*, 95:128–146, 2018.
- 12 Filippo Bonchi, Pawel Sobocinski, and Fabio Zanasi. A survey of compositional signal flow theory. In *IFIP's Exciting First 60+ Years*, volume 600 of *IFIP Advances in Information and Communication Technology*, pages 29–56. Springer, 2021.
- 13 Aurelio Carboni and R. F. C. Walters. Cartesian bicategories I. *J Pure Appl Algebra*, 49:11–32, 1987.

- 14 A. Corradini, Ugo Montanari, Francesca Rossi, H. Ehrig, Reiko Heckel, and Michael Löwe. Basic concepts and double pushout approach. *Algebraic Approaches to Graph Transformation*, pages 163–246, January 1997.
- 15 Geoffrey S. H. Cruttwell, Bruno Gavranovic, Neil Ghani, Paul W. Wilson, and Fabio Zanasi. Categorical foundations of gradient-based learning. In *ESOP*, volume 13240 of *Lecture Notes in Computer Science*, pages 1–28. Springer, 2022.
- 16 Hartmut Ehrig and Barbara König. Deriving bisimulation congruences in the DPO approach to graph rewriting. In Igor Walukiewicz, editor, *Foundations of Software Science and Computation Structures*, pages 151–166, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.
- 17 Brendan Fong and Fabio Zanasi. Universal constructions for (co)relations: categories, monoidal categories, and props. *Log. Methods Comput. Sci.*, 14(3), 2018.
- 18 Tobias Fritz. A synthetic approach to markov kernels, conditional independence and theorems on sufficient statistics. *Advances in Mathematics*, 370:107239, August 2020. doi:10.1016/j.aim.2020.107239.
- 19 Tobias Fritz and Wendong Liang. Free gs-monoidal categories and free Markov categories, 2022. doi:10.48550/ARXIV.2204.02284.
- 20 Dan R. Ghica, Achim Jung, and Aliaume Lopez. Diagrammatic semantics for digital circuits. In Valentin Goranko and Mads Dam, editors, *26th EACSL Annual Conference on Computer Science Logic, CSL 2017, August 20-24, 2017, Stockholm, Sweden*, volume 82 of *LIPICs*, pages 24:1–24:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPICs.CSL.2017.24.
- 21 Dan R. Ghica and George Kaye. Rewriting modulo traced comonoid structure. *CoRR*, abs/2302.09631, 2023.
- 22 Bart Jacobs, Aleks Kissinger, and Fabio Zanasi. Causal inference via string diagram surgery: A diagrammatic approach to interventions and counterfactuals. *Math. Struct. Comput. Sci.*, 31(5):553–574, 2021.
- 23 Aleks Kissinger, John van de Wetering, and Renaud Vilmart. Classical simulation of quantum circuits with partial and graphical stabiliser decompositions, 2022. doi:10.48550/ARXIV.2202.09202.
- 24 Stephen Lack. Composing props. *Theory and Applications of Categories*, 13(9):147–163, 2004.
- 25 Stephen Lack and Paweł Sobociński. Adhesive and quasiadhesive categories. *RAIRO - Theoretical Informatics and Applications - Informatique Théorique et Applications*, 39(3):511–545, 2005. doi:10.1051/ita:2005028.
- 26 Aleksandar Milosavljevic, Robin Piedeleu, and Fabio Zanasi. String diagram rewriting modulo commutative (co)monoid structure. *arXiv:2204.04274*, 2023.
- 27 Koko Muroya and Dan R. Ghica. The dynamic geometry of interaction machine: A token-guided graph rewriter. *Log. Methods Comput. Sci.*, 15(4), 2019.
- 28 Robin Piedeleu and Fabio Zanasi. An introduction to string diagrams for computer scientists. *arXiv:2305.08768*, 2023.
- 29 Mehrnoosh Sadrzadeh, Stephen Clark, and Bob Coecke. The Frobenius anatomy of word meanings I: subject and object relative pronouns. *J. Log. Comput.*, 23(6):1293–1317, 2013.
- 30 Fabio Zanasi. *Interacting Hopf Algebras- the Theory of Linear Systems. (Interacting Hopf Algebras - la théorie des systèmes linéaires)*. PhD thesis, École normale supérieure de Lyon, France, 2015. URL: <https://tel.archives-ouvertes.fr/tel-01218015>.
- 31 Fabio Zanasi. Rewriting in free hypergraph categories. In *GaM@ETAPS*, volume 263 of *EPTCS*, pages 16–30, 2017.