Confluence of Graph Rewriting with Interfaces

Filippo Bonchi[1], Fabio Gadducci[2], Aleks Kissinger[3],
Paweł Sobociński[4], and Fabio Zanasi[5]

[1] CNRS, ENS de Lyon, France
[2] University of Pisa, Italy
[3] Radboud University Nijmegen, The Netherlands
[4] University of Southampton, United Kingdom
[5] University College London, United Kingdom

Abstract. For terminating double-pushout (DPO) graph rewriting systems confluence is, in general, undecidable. We show that confluence *is* decidable for an extension of DPO rewriting to graphs *with interfaces*. This variant is important due to it being closely related to rewriting of string diagrams. We show that our result extends, under mild conditions, to decidability of confluence for terminating rewriting systems of string diagrams in symmetric monoidal categories.

Keywords: Confluence, DPO rewriting systems, adhesive categories, PROPs, string diagrams.

1 Introduction

Confluence and termination are some of the most important properties of rewriting systems. For *term* rewriting, both confluence [3] and termination [27] are, in general, undecidable. However, for systems known to be terminating, confluence is decidable. The key, celebrated property observed by Knuth and Bendix [33] is that the system is confluent exactly when *all its critical pairs are joinable*.

In recent years, an increasing amount of attention has been given to rewriting structures that are richer than mere terms, many of which can be seen as various flavours (including higher-dimensional) of graphs. Here, unfortunately, the status of confluence is murky because old certainties of critical pair analysis fail: Plump [42], working in the well-established framework of the double-pushout (DPO) graph rewriting mechanism [20], showed that joinability of critical pairs *does not* entail confluence, and that confluence of terminating DPO rewriting systems is, in general, undecidable.

In this paper we focus on an extension of DPO, called *DPO with interfaces*. This variant has emerged in several research threads, including rewriting with borrowed contexts [19], encodings of process calculi [24,5], connecting DPO rewriting systems with computads in cospans categories [25,44] and, more recently, for checking the equivalence of terms of symmetric monoidal theories [4]. Our key observation is that for DPO rewriting with interfaces, the Knuth-Bendix property holds and therefore confluence of a terminating system can be decided by checking whether its critical pairs are joinable. More precisely, if some mild assumptions related to the computability of performing rewriting steps on the underlying notion of term are satisfied, our result holds for the most general venue available for DPO rewriting, namely, adhesive categories [34].

Our results do not falsify Plump's: in DPO with interfaces, rather than rewriting graphs, one rewrites graph morphisms $J \to G$, thought of as a graph G with interface J. The latter allows one to "glue" G to other graphs, analogously to how variables allow a single term to apply to a variety of contexts via substitution. Plump's result, in the light of our analysis, states that it is undecidable to check whether rewriting is confluent for all morphisms $0 \to G$. Intuitively, the failure of Knuth-Bendix for such morphisms is due to the loss of expressive power of critical pairs, when deprived of an interface.

This reveals an attractive analogy with term rewriting: morphisms $0 \to G$ – representing graphs that cannot be non-trivially attached to other graphs, since they have an empty interface – correspond to *ground terms*, that cannot be extended since they have no variables. Now, the property that Plump showed to be undecidable should be compared to *ground confluence* for term rewriting [40], i.e., confluence with respect to all ground terms. And in fact, this property is undecidable for terminating term rewriting systems [30]. Summarising, for both term and DPO rewriting with interfaces, confluence of terminating rewriting systems is decidable, while ground confluence is not.

	Terminating term rewriting system	Terminating DPO system
Ground confluence	undecidable (Kapur et al. [30])	undecidable (Plump [42])
Confluence	decidable (Knuth and Bendix [33])	decidable (this paper)

Our interest in DPO rewriting with interfaces is motivated by *symmetric monoidal theories* (SMTs) that appear in different fields of computer science, like concurrency theory [37,11,47], quantum information [15,16], and systems theory [6,7,1,23], just to mention a few. The terms of an SMT enjoy an efficient graphical representation by means of *string diagrams* [29,45], in the sense that structural equations are "baked into" the representation. Rewriting at the diagrammatic level can be used to determine equality of terms, i.e. the word problem for an SMT. While rewriting of string diagrams has been broadly studied from a foundational point of view (e.g. using *computads* [48] or *polygraphs* [12]), its *implementation* has thus far received less attention.

In [4] we showed that rewriting of string diagrams, representing terms of an SMT, can be soundly and effectively encoded into DPO rewriting with interfaces. This enables us to reuse the main result of this paper to study confluence of rewriting of string diagrams. This problem is known to be particularly challenging: for example a directed form of the Yang-Baxter equation generates infinitely many critical pairs [35,39].

We show that this issue can be avoided by using DPO with interfaces, and that confluence is decidable. We identify two classes of terminating rewriting systems for which confluence can be decided by means of critical pair analysis. The first one concerns SMTs containing a *special Frobenius structure* [14] (yielding categories alternatively called *well-supported compact closed* [13], *p*- and *dgs-monoidal* [25,9], or recently *hypergraph* categories [22,31]). For arbitrary SMTs, not necessarily equipped with a special Frobenius structure, we identify a second class of rewriting systems for which confluence can be decided. The rules of these systems need to satisfy a simple condition that we call *left-connectedness*. Many rewrite systems arising from SMTs (e.g., [35,26,21]), including aforementioned Yang-Baxter rule, enjoy this property. Amongst these, we consider a rewriting system for *non-commutative bimonoids* that has been shown to be terminating in [4]. We exploit our approach to prove that it is also confluent and thus conclude that equivalence of non-commutative bimonoids is decidable.

Related work. For ordinary DPO rewriting, a variant of the Knuth-Bendix property holds with respect to a stronger notion of joinability for critical pairs [42]. Moreover, confluence is decidable whenever all critical pairs satisfy a certain syntactic condition called coverability [43]. Both these results however refer to confluence for graphs without interfaces, namely ground confluence. Instead, our same notion of confluence has been studied in [8] in the setting of Milner's reactive systems. By instantiating Proposition 22 in [8] to the category of input-linear cospans (of hypergraphs) and by using the results relating borrowed context DPO rewriting with reactive systems over cospans in [46], one obtains a variant of our Theorem 19. One restriction of that approach is that the matches are required to be mono, which rules out our applications to SMTs.

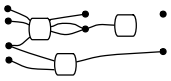
2 Background

Notation. The composition of arrows $f\colon a \to b$, $g\colon b \to c$ in a category $\mathbb{C}$ is written as $f\,;\,g$. For $\mathbb{C}$ symmetric monoidal, $\oplus$ is its monoidal product and $\sigma_{a,b}\colon a \oplus b \to b \oplus a$ is the symmetry for objects $a, b \in \mathbb{C}$.

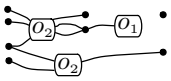
2.1 DPO rewriting

Adhesive categories and (typed) hypergraphs. In order not to restrict ourselves to any one concrete model of graphs, we work with adhesive categories [34]. Adhesive categories are relevant because they have well-behaved pushouts along monomorphisms, and for this reason they are convenient as ambient categories for DPO rewriting.

An important example is the category of finite directed hypergraphs **Hyp**. An object G of **Hyp** is a hypergraph with finite set of *nodes* $G_\star$ and for each $k, l \in \mathbb{N}$ finite set of *hyperedges* $G_{k,l}$ with k (ordered) sources and l (ordered) targets, i.e. for each $0 \leq i < k$ there is the ith source map $s_i\colon G_{k,l} \to G_\star$, and for each $0 \leq j < l$, the jth target map $t_j\colon G_{k,l} \to G_\star$. The arrows of **Hyp** are homomorphisms: functions $G_\star \to H_\star$ such that for each k, l, $G_{k,l} \to H_{k,l}$ they respect the source and target maps in the obvious way. The seasoned reader will recognise **Hyp** as a presheaf topos, and as such, it is adhesive [34]. We shall visualise hypergraphs as follows: $\bullet$ is a node and  is an hyperedge, with ordered tentacles attached to the left boundary linking to sources and those on the right linking to targets. An example is on the right.



A signature Σ consists of a set of *generators* $o\colon n \to m$ with arity n and coarity m where $m, n \in \mathbb{N}$. Any signature Σ can be considered as a hypergraph with a single node, in the obvious way. We can then express Σ-typed hypergraphs as the objects of the slice category $\mathbf{Hyp}/\Sigma$, denoted by $\mathbf{Hyp}_\Sigma$, which is adhesive, since adhesive categories are closed under slice [34]. Σ-typed hypergraphs are drawn by labeling hyperedges with generators in Σ, as on the right.



DPO rewriting. We recall the DPO approach [20] to rewriting in an adhesive category $\mathbb{C}$. A *DPO rule* is a span $L \leftarrow K \rightarrow R$ in $\mathbb{C}$. A *DPO system* $\mathcal{R}$ is a finite set of DPO rules. Given objects G and H in $\mathbb{C}$, we say that G *rewrites* into H —notation $G \Rightarrow_{\mathcal{R}} H$— if

there esist $L \leftarrow K \rightarrow R$ in $\mathcal{R}$, object C and morphisms such that the squares below are pushouts. A *derivation* from G into H is a sequence of such rewriting steps.

$$\begin{tikzcd} L \arrow[d, "m"'] & K \arrow[l] \arrow[r] \arrow[d] & R \arrow[d] \\ G \arrow[ur, phantom, "\urcorner"] & C \arrow[l] \arrow[r] & H \arrow[ul, phantom, "\ulcorner"] \end{tikzcd}$$

The arrow $m\colon L \rightarrow G$ is called a *match* of L in G. A rule $L \leftarrow K \rightarrow R$ is said to be *left-linear* if the morphism $K \rightarrow L$ is mono. In this case, the matching m fully determines the graphs C and H, i.e., for fixed a rule and a matching there is a unique H such that $G \Rightarrow_{\mathcal{R}} H$. Here, by unique, we mean unique up-to isomorphism. More generally, the rewriting steps will always be *up-to iso*: in a step $G \Rightarrow_{\mathcal{R}} H$, G and H should not be thought of as single graphs but rather as equivalence classes of isomorphic graphs.

Undecidability of confluence. In DPO rewriting, the confluence of terminating systems is not decidable, even if we restrict to left-linear rules.

Theorem 1 ([42]). *Confluence of terminating DPO systems over* $\mathbf{Hyp}_{\Sigma}$ *is undecidable.*

Indeed, critical pair analysis for traditional DPO systems fails: for terminating DPO systems, joinability of critical pairs does not necessarily imply confluence.

Definition 2 (Pre-critical pair and joinability). *Let $\mathcal{R}$ be a DPO system with rules $L_1 \leftarrow K_1 \rightarrow R_1$ and $L_2 \leftarrow K_2 \rightarrow R_2$. Consider two derivations with common source S*

$$\begin{tikzcd} R_1 \arrow[d] & K_1 \arrow[l] \arrow[r] \arrow[d] & L_1 \arrow[dr, bend left, "f_1"] & & L_2 \arrow[dl, bend right, "f_2"'] & K_2 \arrow[l] \arrow[r] \arrow[d] & R_2 \arrow[d] \\ H_1 \arrow[urr, phantom, "\urcorner"] & C_1 \arrow[l] \arrow[rr] & \arrow[r, phantom, "\ulcorner"] & S & \arrow[l, phantom, "\urcorner"] & C_2 \arrow[l] \arrow[r] & H_2 \arrow[ull, phantom, "\ulcorner"] \end{tikzcd}$$

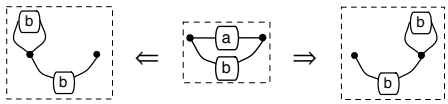
We say that $H_1 \Leftarrow S \Rightarrow H_2$ is a pre-critical pair *if $[f_1, f_2]\colon L_1 + L_2 \rightarrow S$ is epi; it is* joinable *if there exists W such that $H_1 \Rightarrow^* W \,{}^*\!\Leftarrow H_2$.*

Intuitively, in a pre-critical pair S should not be bigger than $L_1 + L_2$. In a critical pair, L_1 and L_2 must additionally overlap in S, so that the two rewriting steps are not *parallel independent* (see e.g. [17]). For the purposes of this paper, this restriction is immaterial. We stick to pre-critical pairs in our results, as proofs are less tedious. However, for the sake of succinctness, most of the examples only display the critical pairs. For a pre-critical pair which is *not* a critical pair, see for instance the first picture of Section 5.

Example 3 ([42]). Consider a DPO system $\mathcal{R}$ consisting of the following two rules, where we labeled nodes with numbers in order to make the graph morphisms explicit.

Amongst the several pre-critical pairs, only the following two have non-trivial overlap.

Both are obviously joinable. However, $\mathcal{R}$ is not confluent, as witnessed by the following



DPO rewriting with interfaces. Morphisms $G \leftarrow J$ will play a special role in our exposition. When $\mathbb{C}$ is $\mathbf{Hyp}_\Sigma$, we will call them *(hyper)graphs with interface*. The intuition is that G is a hypergraph and J is an interface that allows G to be "glued" to a context.

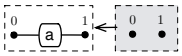
Given $G \leftarrow J$ and $H \leftarrow J$ in $\mathbb{C}$, *G rewrites into H with interface J* — notation $(G \leftarrow J) \Rightarrow_{\mathcal{R}} (H \leftarrow J)$ — if there exist rule $L \leftarrow K \rightarrow R$ in $\mathcal{R}$, object C and morphisms such that the diagram below commutes and the squares are pushouts.

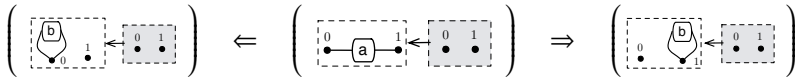
$$\begin{tikzcd}
L \arrow[d, "m"'] & K \arrow[l] \arrow[r] \arrow[d] & R \arrow[d] \\
G & C \arrow[l] \arrow[r] & H \\
 & J \arrow[ul] \arrow[u] \arrow[ur] &
\end{tikzcd}$$

Hence, the interface J is preserved by individual rewriting steps.

When $\mathbb{C}$ has an initial object 0 (for instance, in $\mathbf{Hyp}_\Sigma$ 0 is the empty hypergraph), ordinary DPO rewriting can be considered as a special case, by taking J to be 0.

Like for traditional DPO, rewriting steps are modulo isomorphism: $G_1 \leftarrow J : f_1$ and $G_2 \leftarrow J : f_2$ are isomorphic if there is an isomorphism $\varphi\colon G_1 \rightarrow G_2$ with $f_1 \,;\, \varphi = f_2$.

Example 4. Consider the system $\mathcal{R}$ from Example 3 and  , a graph with interface (henceforth depicted in grey). It is the source of two rewriting steps



(1)

that are *not* joinable. Intuitively, the main difference with Example 3 is that here the interface $\{0, 1\}$ allows one to "look inside" the graph and distinguish between the two nodes. Notice that if (1) were considered as a critical pair, the counterexample of Plump [42] (Example 3) would not work. This is the starting observation for our work: in Section 3 we will introduce pre-critical pairs for rewriting with interfaces and we will show that, as in term rewriting, joinability of pre-critical pairs entails confluence.

2.2 PROP rewriting

SMTs and PROPs. A uniform way to express an algebraic structure within a symmetric monoidal category is with a *symmetric monoidal theory* (SMT). A (one-sorted) SMT is a pair (Σ, E) where Σ is a *signature* defined as in Section 2.1. The set of Σ-terms is obtained by combining generators in Σ, the *unit* $id\colon 1 \rightarrow 1$ and the *symmetry* $\sigma_{1,1} : 2 \rightarrow 2$ with ; and $\oplus$. That means, given Σ-terms $t\colon k \rightarrow l$, $u\colon l \rightarrow m$, $v\colon m \rightarrow n$, one constructs new Σ-terms $t \,;\, u\colon k \rightarrow m$ and $t \oplus v\colon k+m \rightarrow l+n$. The set E of *equations* contains pairs (t, t') of Σ-terms, with the requirement that t and t' have the same arity and coarity.

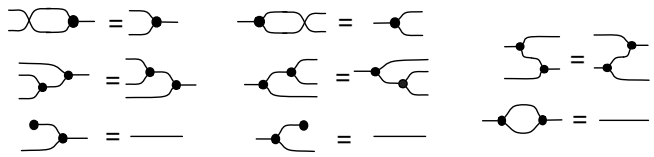


Fig. 1. The equations E_F of special Frobenius monoids.

Just as ordinary (cartesian) algebraic theories have a categorical rendition as Lawvere categories [28], the corresponding (linear[6]) notion for SMTs is a PROP [36] (**pro**duct and **p**ermutation category). A PROP is a symmetric strict monoidal category with objects the natural numbers, where $\oplus$ on objects is addition. Morphisms between PROPs are identity-on-objects strict symmetric monoidal functors. PROPs and their morphisms form a category **PROP**. Any SMT (Σ, E) freely generates a PROP by letting the arrows $n \to m$ be the Σ-terms $n \to m$ modulo the laws of symmetric monoidal categories and the (smallest congruence containing the) equations $t = t'$ for any $(t, t') \in E$.

We write $\mathbf{S}_\Sigma$ to denote the PROP freely generated by $(\Sigma, \varnothing)$. There is a graphical representation of the arrows of $\mathbf{S}_\Sigma$ as string diagrams, which we now sketch, referring to [45] for the details. A Σ-term $n \to m$ is pictured as a box with n ports on the left and m ports on the right, which are ordered and referred to with top-down enumerations $1, \ldots, n$ and $1, \ldots, m$. Compositions via $;$ and $\oplus$ are drawn respectively as horizontal and vertical juxtaposition, that means, $t\,;\,s$ is drawn $\boxed{t}\boxed{s}$ and $t \oplus s$ is drawn $\boxed{t}\,\boxed{s}$. There are specific diagrams for the Σ-terms responsible for the symmetries: these are $id_1 \colon 1 \to 1$, represented as $\boxed{}$, the symmetry $\sigma_{1,1} \colon 1 + 1 \to 1 + 1$, represented as $\boxed{\times}$, and the unit object for $\oplus$, that is, $id_0 \colon 0 \to 0$, whose representation is an empty diagram $\boxed{}$. Graphical representation for arbitrary identities id_n and symmetries $\sigma_{n,m}$ are generated using the pasting rules for $;$ and $\oplus$. It will be sometimes convenient to represent id_n with the shorthand diagram $\boxed{n}$ and, similarly, $t \colon n \to m$ with $\overset{n}{-}\boxed{t}\overset{m}{-}$.

Example 5.

(a) A basic example is the theory (Σ_M, E_M) of *commutative monoids*. The signature Σ_M contains two generators: *multiplication* — which we depict $\boxed{\bullet}$: $2 \to 1$ — and *unit*, represented as $\boxed{\bullet}$: $0 \to 1$. Equations in E_M are given in the leftmost column of Figure 1: they assert commutativity, associativity and unitality.

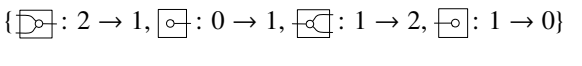
(b) An SMT that plays a key role in our exposition is the theory (Σ_F, E_F) of *special Frobenius monoids*. The signature Σ_F is as follows and E_F is depicted in Figure 1.

$$\{\boxed{\bullet} : 2 \to 1,\ \boxed{\bullet} : 0 \to 1,\ \boxed{\bullet} : 1 \to 2,\ \boxed{\bullet} : 1 \to 0\}$$

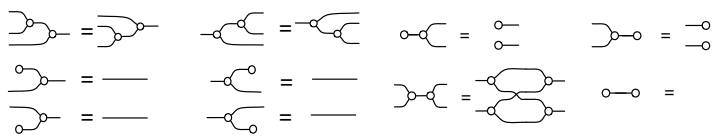
E_F includes the theory of commutative monoids in the leftmost column. Dually, the equations in the middle column assert that $\boxed{\bullet}$ and $\boxed{\bullet}$ form a cocommutative comonoid. Finally, the two rightmost equations describe an interaction between these two structures. We call **Frob** the PROP freely generated by (Σ_F, E_F).

[6] In the sense that variables can neither be copied, nor discarded.

(c) The theory of *non-commutative bimonoids* has signature Σ_{NB}

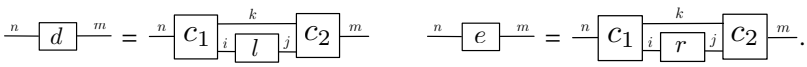


and the following equations E_{NB}.



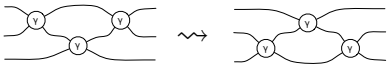
We call **NB** the PROP freely generated from (Σ_{NB}, E_{NB}). In [4], we showed that the rewriting system that is obtained by orienting the equalities from left to right terminates. In this paper, we will show that is also confluent. For this, it will be convenient to use μ, η, ν, ϵ, respectively, to refer to the generators in Σ_{NB}.

Rewriting in a PROP. Fix an arbitrary PROP **X**. A rewriting *rule* is a pair $\langle l, r\rangle$ where $l, r\colon i \to j$ in **X** have the same domain and codomain. We say that $i \to j$ is the rule's *type* and sometimes write $\langle l, r\rangle\colon (i, j)$. A *rewriting system* $\mathcal{R}$ is a finite set of rules. Given two arrows $d, e\colon n \to m$ in **X**, $d \rightsquigarrow_{\mathcal{R}} e$ iff $\exists\langle l, r\rangle\colon (i, j) \in \mathcal{R}, c_1 : n \to k+i, c_2 : k+j \to n$ such that $d = c_1 \,;\, (id_k \oplus l) \,;\, c_2$ and $e = c_1 \,;\, (id_k \oplus r) \,;\, c_2$, i.e., diagrammatically



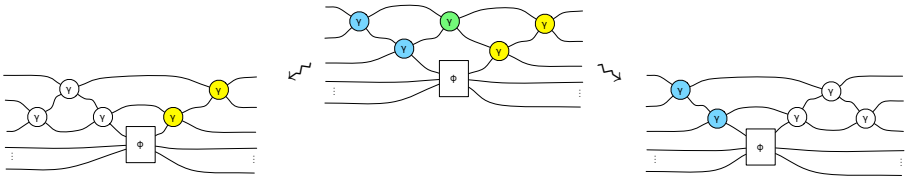
The following well-known example illustrates the subtlety of critical pair analysis when rewriting in monoidal categories.

Example 6 (From [35], see also [39]). Fix $\Sigma = \{\gamma\colon 2 \to 2\}$ and consider the rewriting system on $\mathbf{S}_\Sigma$ consisting of the following rule:



(2)

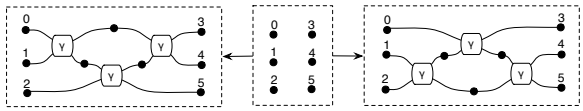
A critical pair analysis yields an infinite number of critical pairs. Indeed, as shown in [35,39], any diagram $\phi : 1 + m \to 1 + n$ that does not decompose non-trivially into $\phi = \mu + \nu$ for some μ, ν yields a critical pair



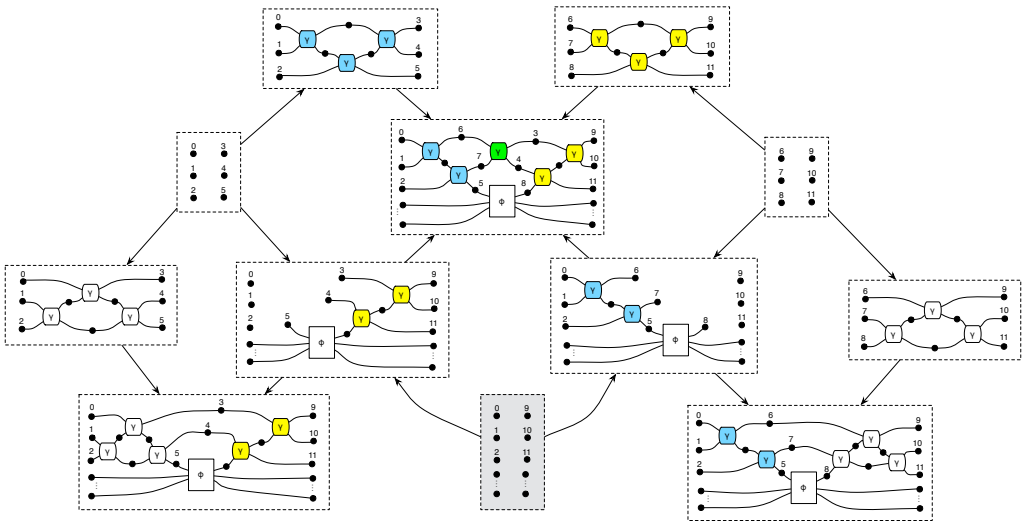
in which clearly there are two embeddings of the left-hand side of (2) (depicted in blue and yellow, respectively, in a colour version of the paper) with an overlap (in green).

In [38] this problem was solved by freely adding duals to monoidal categories. In Section 4, we will show another solution based on our earlier work [4]: a translation from PROPs to DPO rewriting with interfaces. The example below anticipates this encoding. It will be useful as a running example for the next section, which is devoted to critical pair analysis and confluence in DPO rewriting with interfaces.

Example 7. Treating the rewriting system of Example 6 as DPO system over $\mathbf{Hyp}_\Sigma$ with $\gamma : 2 \to 2 \in \Sigma$ yields the following DPO rules.



Below, we give a DPO derivation with interface (in grey), corresponding to a critical pair from the family identified in Example 6.



3 Confluence for DPO rewriting with interfaces

Differently from Definition 2, when considering pre-critical pairs in the setting of DPO with interfaces, the interface of the pre-critical pair plays a crucial role.

Definition 8 (Pre-critical pair with interface). *Let $\mathcal{R}$ be a DPO system with rules $L_1 \leftarrow K_1 \rightarrow R_1$ and $L_2 \leftarrow K_2 \rightarrow R_2$. Consider two derivations with source $S \leftarrow J$*

$$\begin{tikzcd}
R_1 \arrow[d] & K_1 \arrow[l] \arrow[r] \arrow[d] & L_1 \arrow[dr, "f_1"] & & L_2 \arrow[dl, "f_2"'] & K_2 \arrow[l] \arrow[r] \arrow[d] & R_2 \arrow[d] \\
H_1 & C_1 \arrow[l] \arrow[rr] & \arrow[d, phantom, "(\dagger)"] & S & & C_2 \arrow[ll] \arrow[r] & H_2 \\
& & J \arrow[ull, bend left] \arrow[urr, bend right] & & & &
\end{tikzcd} \tag{3}$$

We say that $(H_1 \leftarrow J) \Leftarrow (S \leftarrow J) \Rightarrow (H_2 \leftarrow J)$ *is a* pre-critical pair *if* $[f_1, f_2] \colon L_1 + L_2 \to S$ *is epi and* (†) *is a pullback; it is* joinable *if there exists* $W \leftarrow J$ *such that* $(H_1 \leftarrow J) \Rightarrow^* (W \leftarrow J) \,{}^*\!\Leftarrow\, (H_2 \leftarrow J)$.

Definition 8 augments Definition 2 with the interface J, given by "intersecting" C_1 and C_2. Intuitively, J is the largest interface that allows both the rewriting steps.

Example 9. Consider the pair of rewriting steps (1) in Example 4. This is a pre-critical pair: the reader can check that the interface is indeed a pullback, constructed as in (†). Observe moreover that this pair is *not* joinable.

Plump's Example 3 shows that in ordinary DPO, joinability of pre-critical pairs does not imply confluence. Our Example 9 shows that the argument does not work for DPO with interfaces. Indeed, as we shall see in Theorem 19, in the presence of interfaces joinability suffices for confluence. To prove it, we assume the following.

Assumption 10. *Our ambient category* $\mathbb{C}$ *is assumed (1) to possess an epi-mono factorisation system, (2) to have binary coproducts, pushouts and pullbacks, (3) to be adhesive (4) with all the pushouts stable under pullbacks.*

All of the above hold in any presheaf category. Additionally, all four are closed under slice. It follows that $\mathbf{Hyp}_\Sigma$ is an example of such a category. The final property allows us to treat non left-linear rules: to this aim we need the following simple pushout decomposition lemma (aka "mixed decomposition" from [2]).

Lemma 11. *Suppose in the following diagram* m *is mono,* (†) + (‡) *is a pushout and* (‡) *is a pullback. Then both* (†) *and* (‡) *are pushouts.*

$$\begin{tikzcd} K \arrow[r] \arrow[d] \arrow[dr, phantom, "(\dagger)"] & C' \arrow[r] \arrow[d] \arrow[dr, phantom, "(\ddagger)"] & C \arrow[d] \\ L \arrow[r] & G' \arrow[r, "m"'] & G \end{tikzcd} \tag{4}$$

The following construction mimics [18]. It allows us to restrict –or "clip"– a DPO rewriting step with match $f \colon L \to G$ to any subobject of G' through which f factors.

Construction 12 (One-step clipping). *Suppose we have a DPO rewriting step as below left, together with factorisation* $L \to G' \xrightarrow{m} G$ *where* m *is mono. As shown below right, we get* C' *by pulling back* $G' \to G \leftarrow C$ *and* $K \to C'$ *by the universal property.*

$$\begin{tikzcd} & L \arrow[dl] \arrow[d] & K \arrow[l] \arrow[r] \arrow[d] & R \arrow[d] \\ G' \arrow[dr, "m"'] & & & \\ & G & C \arrow[l] \arrow[r] & H \end{tikzcd} \qquad \begin{tikzcd} & L \arrow[dl] \arrow[d] & K \arrow[l] \arrow[r] \arrow[d] \arrow[dl, dotted] & R \arrow[d] \\ G' \arrow[dr, "m"'] & C' \arrow[l] \arrow[dr] & & \\ & G & C \arrow[l] \arrow[r] & H \end{tikzcd}$$

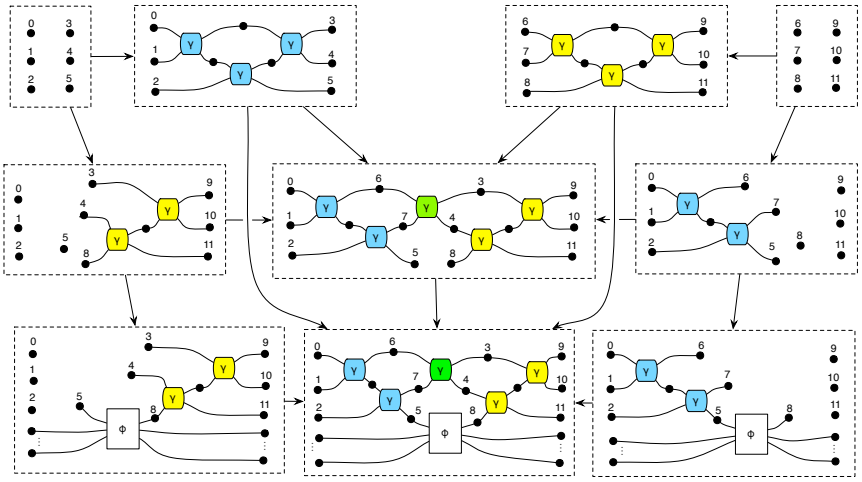
By Lemma 11 the two leftmost squares are both pushouts. Next, H' *is the pushout of* $C' \leftarrow K \to R$ *and* $H' \to H$ *follows from its universal property.*

$$\begin{tikzcd} & L \arrow[dl] \arrow[d] & K \arrow[l] \arrow[r] \arrow[d] \arrow[dl, dotted] & R \arrow[dl] \arrow[d] \\ G' \arrow[dr, "m"'] & C' \arrow[l] \arrow[r] \arrow[dr] & H' \arrow[dr, dotted] & \\ & G & C \arrow[l] \arrow[r] & H \end{tikzcd}$$

By pushout pasting also the bottom-rightmost square is a pushout. Finally, observe that $C' \to C$ is mono since it is the pullback of m along $C \to G$. This means that each of the two squares in diagram below is, as well as being a pushout, also a pullback, since each is a pushout along a mono in an adhesive category.

$$\begin{tikzcd} G' \arrow[d, "m"'] & C' \arrow[l] \arrow[r] \arrow[d] & H' \arrow[d] \\ G & C \arrow[l] \arrow[r] & H \end{tikzcd}$$

Example 13. We use the clipping construction to restrict pairs of derivations with common source into pre-critical pairs. For example, consider the two DPO rewrite rules illustrated in Example 7. We can factorise the two matches through their common image, and clip, as illustrated below.



Note that the clipped derivations result with the two matches being jointly epi, which is one of the properties of a pre-critical pair. This generalises: given two rewriting steps with common source $(G_{1,1} \leftarrow I) \Leftarrow (G_0 \leftarrow I) \Rightarrow (G_{1,2} \leftarrow I)$, next construction produces a pre-critical pair $(G'_{1,1} \leftarrow J') \Leftarrow (G_0{}' \leftarrow J') \Rightarrow (G'_{1,2} \leftarrow J')$ using clipping.

Construction 14 (Pre-critical pair extraction). *Start with two rewrites from $G_0 \leftarrow I$*

$$\begin{tikzcd} R_{1,1} \arrow[d] & K_{1,1} \arrow[l] \arrow[r] \arrow[d] & L_{1,1} \arrow[dr, "f_1"] & & L_{1,2} \arrow[dl, "f_2"'] & K_{1,2} \arrow[l] \arrow[r] \arrow[d] & R_{1,2} \arrow[d] \\ G_{1,1} & C_{1,1} \arrow[l] \arrow[rr] \arrow[drr, bend right] & & G_0 & & C_{1,2} \arrow[ll] \arrow[r] \arrow[dll, bend left] & G_{1,2} \\ & & & I \arrow[u] & & & \end{tikzcd}$$

and factorise $[f_1, f_2]\colon L_{1,1} + L_{1,2} \to G_0$ *to obtain*

$$\begin{tikzcd}
L_{1,1} \arrow[dr] \arrow[ddr, bend right, "f_1"'] & & L_{1,2} \arrow[dl] \arrow[ddl, bend left, "f_2"] \\
& G'_0 \arrow[d, hook] & \\
& G_0. &
\end{tikzcd}$$

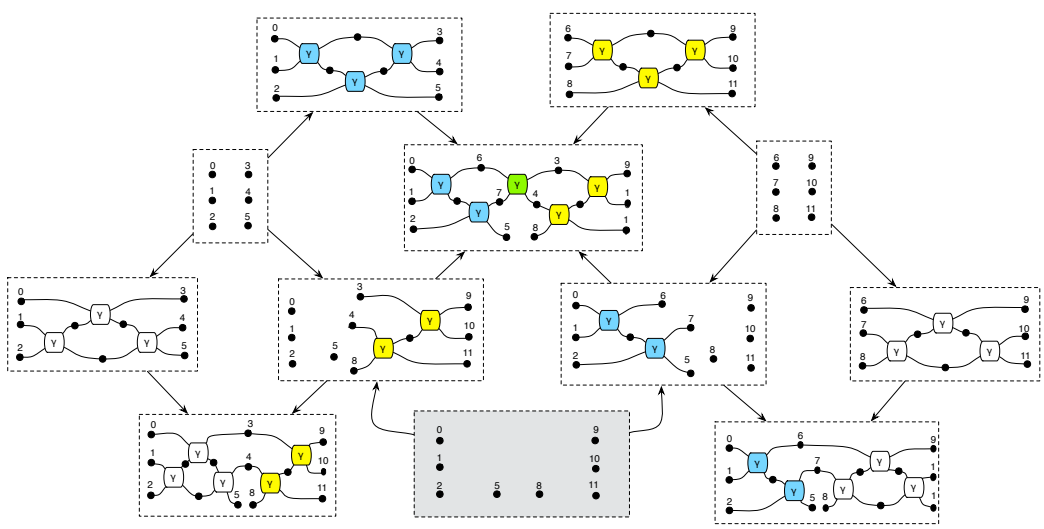
Next apply Construction 12 twice, obtaining

$$\begin{tikzcd}
R_{1,1} \arrow[dr] & K_{1,1} \arrow[l] \arrow[r] \arrow[dr] & L_{1,1} \arrow[dr] & & L_{1,2} \arrow[dl] & K_{1,2} \arrow[l] \arrow[r] \arrow[dl] & R_{1,2} \arrow[dl] \\
H'_{1,1} \arrow[dr] & C'_{1,1} \arrow[l] \arrow[r] \arrow[dr] & & G'_0 \arrow[d] & & C'_{1,2} \arrow[l] \arrow[r] \arrow[dl] & H'_{1,2} \arrow[dl] \\
H_{1,1} & C_{1,1} \arrow[l] \arrow[r] & & G_0 & & C_{1,2} \arrow[l] \arrow[r] & H_{1,2}. \\
& & & I \arrow[ull, bend left] \arrow[uur, bend right] & & &
\end{tikzcd}$$

Finally pull back $C'_{1,1} \to G'_0 \leftarrow C'_{1,2}$ *to obtain pre-critical pair*

$$\begin{tikzcd}
R_{1,1} \arrow[dr] & K_{1,1} \arrow[l] \arrow[r] \arrow[dr] & L_{1,1} \arrow[dr] & & L_{1,2} \arrow[dl] & K_{1,2} \arrow[l] \arrow[r] \arrow[dl] & R_{1,2} \arrow[dl] \\
G'_{1,1} & C'_{1,1} \arrow[l] \arrow[r] & & G'_0 & & C'_{1,2} \arrow[l] \arrow[r] & G'_{1,2}. \\
& & & J' \arrow[ull, bend left] \arrow[uur, bend right] & & &
\end{tikzcd}$$

Example 15. We can now complete the pre-critical pair extraction process, commenced in Example 13, following the steps of Construction 14.



Construction 14 means that we are able to extract a pre-critical pair from two rewriting steps with common source. If the pre-critical pair is joinable, we would then like to embed the joining derivations to the original context.

The following is a useful step in this direction. Assuming a mono $G'_0 \to G_0$, it allows us to extend a derivation from $G'_0 \leftarrow J'$ to a corresponding one from $G_0 \leftarrow J$, if we can obtain G_0 by glueing G'_0 and some context C_0 along J'. Stated more formally, we want the following diagram commute and (†) be a pushout.

$$\begin{tikzcd} & J' \arrow[r] \arrow[d] \arrow[dr, phantom, "(\dagger)"] & G'_0 \arrow[d] \\ J \arrow[r] \arrow[rr, bend right] & C_0 \arrow[r] & G_0 \end{tikzcd} \tag{5}$$

Construction 16 (Embedding). *The extended derivation is constructed as in the commuting diagram below, where each square is a pushout diagram.*

$$\begin{tikzcd}[column sep=small] & & L_1 \arrow[dl] & K_1 \arrow[l] \arrow[r] \arrow[d] & R_1 \arrow[dr] & & L_2 \arrow[dl] & K_2 \arrow[l] \arrow[r] \arrow[d] & R_2 \arrow[dr] & & \dots & & L_n \arrow[dl] & K_n \arrow[l] \arrow[r] \arrow[d] & R_n \arrow[dr] & \\ J' \arrow[r] \arrow[d] & G'_0 \arrow[d] \arrow[dr, phantom, "(\dagger)"] & & C'_1 \arrow[ll] \arrow[r] \arrow[d] \arrow[dr, phantom, "(\gamma_1)"] & G'_1 \arrow[d] \arrow[dr, phantom, "(\delta_1)"] & & C'_2 \arrow[ll] \arrow[r] \arrow[d] \arrow[dr, phantom, "(\gamma_2)"] & G'_2 \arrow[d] \arrow[dr, phantom, "(\delta_2)"] & & \dots & G'_n \arrow[d] \arrow[dr, phantom, "(\gamma_n)"] & & C'_n \arrow[ll] \arrow[r] \arrow[d] \arrow[dr, phantom, "(\delta_n)"] & G'_n \arrow[d] & \\ C_0 \arrow[r] & G_0 & & C_1 \arrow[ll] \arrow[r] & G_1 & & C_2 \arrow[ll] \arrow[r] & G_2 & & \dots & G_n & & C_n \arrow[ll] \arrow[r] & G_n & \\ & & & & & & & & & & & & & & & \\ & & & & & & & & & & & & J' \arrow[uull, bend left, "(\epsilon_1)"] \arrow[uuull] \arrow[uuuull] \arrow[uuuuull] \arrow[uuuuuull] \arrow[uuuuuuull] \arrow[uuuuuuuull] \arrow[uuuuuuuuull] \arrow[d] & & & \\ & & & & & & & & & & & & C_0 \arrow[uull, bend left, "(\epsilon_2)"] \arrow[uuull] \arrow[uuuull] \arrow[uuuuull] \arrow[uuuuuull] \arrow[uuuuuuull] \arrow[uuuuuuuull] \arrow[uuuuuuuuull] & J \arrow[l] & & \end{tikzcd}$$

We shall now explain each of the components. The upper row of pushouts together with morphisms $J' \to C'_i$ witnesses the original derivation $(G'_0 \leftarrow J') \Rightarrow^* (G'_n \leftarrow J')$.

For $i = 1 \dots n$, (ϵ_i) is formed as the pushout of $C_0 \leftarrow J' \to C'_i$ and (δ_i) as the pushout of $C_i \leftarrow C'_i \to G'_i$, as shown in the diagram below.

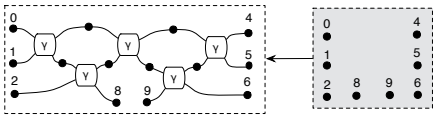
$$\begin{tikzcd} J' \arrow[r] \arrow[d] \arrow[dr, phantom, "(\epsilon_i)"] & C'_i \arrow[r] \arrow[d] \arrow[dr, phantom, "(\delta_i)"] & G'_i \arrow[d] \\ C_0 \arrow[r] & C_i \arrow[r] & G_i \end{tikzcd} \tag{6}$$

It remains to construct pushouts (γ_i), which is done in the following diagram.

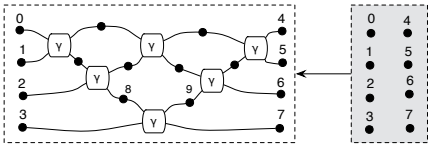
$$\begin{tikzcd} J' \arrow[rr] \arrow[dr] \arrow[dd] & & G'_{i-1} \arrow[dd] \\ & C'_i \arrow[ur] \arrow[d] \arrow[dr, phantom, "(\epsilon_i)"] \arrow[dr, phantom, "(\gamma_i)"] & \\ & C_i \arrow[dr, dotted] & \\ C_0 \arrow[ur] \arrow[rr] & & G_{i-1} \end{tikzcd} \tag{7}$$

The exterior square in (7) *is a pushout: for $i = 1$ it is* (†) *from* (5)*, while for $i \geq 2$ it is obtained by composing (ϵ_{i-1}) and (δ_{i-1}) from* (6)*. The universal property of (ϵ_i) yields the morphism $C_i \to G_{i-1}$. By pushout decomposition, the diagram (γ_i) is a pushout.*

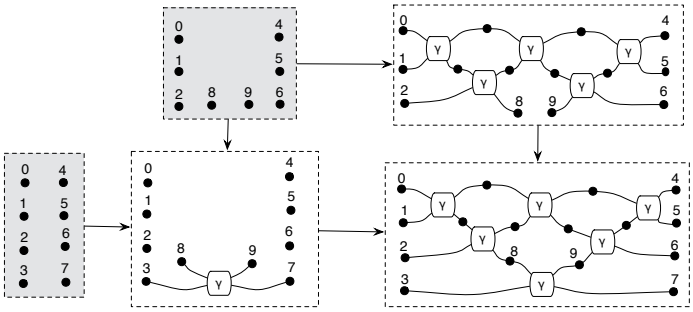
Example 17. In Example 15 we saw two derivations from



These can be extended to



following the steps in Construction 16 because the square in the following is a pushout.



Constructions 14 and 16 are the main ingredients for showing the Knuth-Bendix property for DPO with interfaces. Before we prove it, we need one technical lemma from the theory of adhesive categories.

Lemma 18. *Consider the cube on the right, where the top and bottom faces are pullbacks, the rear faces are both pullbacks and pushouts, and m is mono. Then, the front faces are also pushouts.*

$$\begin{tikzcd}
 & G'_0 & C'_{1,2} \arrow[l] \arrow[dd] \\
C'_{1,1} \arrow[ur] \arrow[dd, "n"'] & J' \arrow[l] \arrow[u, "m"] \arrow[d] & \\
 & G_0 & C_{1,2} \arrow[l] \\
C_{1,1} \arrow[ur] & J \arrow[l] \arrow[u] &
\end{tikzcd}$$

Theorem 19 (Local confluence). *For a DPO system with interfaces, if all pre-critical pairs are joinable then rewriting is locally confluent: given* $(G_{1,1} \leftarrow I) \Leftarrow (G_0 \leftarrow I) \Rightarrow (G_{1,2} \leftarrow I)$*, there exists* $W \leftarrow I$ *such that*

$$\begin{tikzcd}
 & (G_0 \leftarrow I) \arrow[dl, Rightarrow] \arrow[dr, Rightarrow] & \\
(G_{1,1} \leftarrow I) \arrow[dr, Rightarrow, "*"] & & (G_{1,2} \leftarrow I) \arrow[dl, Rightarrow, "*"'] \\
 & (W \leftarrow I). &
\end{tikzcd}$$

Proof. Following the steps of Construction 14, we obtain a pre-critical pair

$$(G'_{1,1} \leftarrow J') \Leftarrow (G'_0 \leftarrow J') \Rightarrow (G'_{1,2} \leftarrow J')$$

Because pre-critical pairs are by assumption joinable we have derivations

$$(G'_{1,1} \leftarrow J') \Rightarrow^* (W' \overset{\beta'}{\leftarrow} J') \,{}^*\!\Leftarrow (G'_{1,2} \leftarrow J').$$

Suppose w.l.o.g. that the leftmost derivation requires n steps and the rightmost m. To keep the notation consistent with Construction 16, we fix notation $G'_{n,1} := W' =: G'_{m,2}$.

Now let J be the pullback object of $C_{1,1} \to G_0 \leftarrow C_{1,2}$. By the universal property, we obtain maps $\iota\colon I \to J$ and $\xi\colon J' \to J$.

$$\begin{tikzcd}
 & G'_0 \arrow[dd] & C'_{1,2} \arrow[l] \arrow[dd] \\
C'_{1,1} \arrow[ur] \arrow[dd] & J' \arrow[l] \arrow[ur] \arrow[dd, dotted, "\xi"] & \\
 & G_0 & C_{1,2} \arrow[l] \\
C_{1,1} \arrow[ur] & J \arrow[l] \arrow[ur] & I \arrow[l, dotted, "\iota"'] \arrow[ul, bend left] \arrow[ull, bend left]
\end{tikzcd} \tag{8}$$

Recall by Construction 12 that the rear faces of (8) are both pullbacks and pushouts. Then, by Lemma 18, the square on the right is a pushout.

$$\begin{tikzcd}
J' \arrow[r] \arrow[d] \arrow[dr, phantom, "(\dagger)"] & G'_0 \arrow[d] \\
J \arrow[r] & G_0
\end{tikzcd}$$

We are now in position to apply Construction 16 by taking $C_0 = J$, which yields

$$(G_0 \leftarrow J) \Rightarrow (G_{1,1} \leftarrow J) \Rightarrow^* (G_{n,1} \overset{\beta_1}{\leftarrow} J)$$

extending $(G'_0 \leftarrow J') \Rightarrow (G'_{1,1} \leftarrow J') \Rightarrow^* (G'_{n,1} \overset{\beta'}{\leftarrow} J')$ and

$$(G_0 \leftarrow J) \Rightarrow (G_{1,2} \leftarrow J) \Rightarrow^* (G_{m,2} \overset{\beta_2}{\leftarrow} J)$$

extending $(G'_0 \leftarrow J') \Rightarrow (G'_{1,1} \leftarrow J') \Rightarrow^* (G'_{m,2} \overset{\beta'}{\leftarrow} J')$.

The next step is to prove that $(G_{n,1} \overset{\beta_1}{\leftarrow} J) \cong (G_{m,2} \overset{\beta_2}{\leftarrow} J)$. To see this, it is enough to observe that both the following squares are pushouts of $J \overset{\xi}{\leftarrow} J' \overset{\beta'}{\rightarrow} W' = G'_{n,1} = G'_{m,2}$.

$$\begin{tikzcd}
J' \arrow[r, "\beta'"] \arrow[d, "\xi"'] & G'_{n,1} \arrow[d] \\
J \arrow[r, "\beta_1"'] & G_{n,1}
\end{tikzcd}
\qquad
\begin{tikzcd}
J' \arrow[r, "\beta'"] \arrow[d, "\xi"'] & G'_{m,2} \arrow[d] \\
J \arrow[r, "\beta_2"'] & G_{m,2}
\end{tikzcd}$$

Indeed, the leftmost is a pushout by composition of squares (ϵ_n) and (δ_n) in the embedding construction and the rightmost by composition of (ϵ_m) and (δ_m).

To complete the proof, it remains to show that, in the above derivations, interface J extends to interface I as in the statement of the theorem. But this trivially holds by precomposing with $\iota\colon I \to J$. □

We are now ready to give our decidability result. To formulate it at the level of generality of adhesive categories we need some additional definitions.

A *quotient* of an object X is an equivalence class of epis with domain X. Two epis $e_1 : X \to X_1$, $e_2 : X \to X_2$ are equivalent when there exists isomorphism $\varphi : X_1 \to X_2$ such that $e_1 ; \varphi = e_2$. Note that quotient is the dual of *subobject*.

A DPO rewriting system with interfaces is *computable* when

- pullbacks are computable,
- for every pairs of rules $L_i \leftarrow K_i \to R_i$, $L_j \leftarrow K_j \to R_j$, the set of quotients of $L_i + L_j$ is finite and computable,
- for all $G \leftarrow J$, it is possible to compute every $H \leftarrow J$ such that $(G \leftarrow J) \Rightarrow (H \leftarrow J)$.

Computability refers to the possibility of effectively computing each rewriting step as well as to have a finite number of pre-critical pairs. More precisely, the first two conditions ensure that the set of all pre-critical pairs is finite (since every objects has finitely many quotients) and each of them can be computed, while the last one ensures that any possible rewriting step can also be computed. Thus, these assumptions rule out the rewriting of infinite structures, singleing out instead those structures where it is reasonable to apply the DPO mechanism, like finite hypergraphs in $\mathbf{Hyp}_\Sigma$, which are exactly what is needed for implementing rewriting of SMTs.

Corollary 20. *For a computable terminating DPO system with interfaces, confluence is decidable.*

Proof. By Theorem 19, if all pre-critical pairs are joinable then the system is confluent. If not all pre-critical pairs are joinable, then at least one pair witnesses the fact that the system is not confluent. Therefore, to decide confluence, it is enough to check that all pre-critical pairs are joinable.

Since the system is computable, there are only finitely many pre-critical pairs and these can be computed. For each pair, one can decide joinability: indeed each rewriting step can be computed (since the system is computable) and there are only finitely many $(H \leftarrow J)$ such that $(G \leftarrow J) \Rightarrow^* (H \leftarrow J)$ (since the system is terminating). □

It is worth to remark that this result is not in conflict with Theorem 1: Corollary 20 refers to confluence of all hypergraphs with interfaces $G \leftarrow J$. The property that Theorem 1 states as undecidable is whether the rewriting is confluent for all *hypergraphs with empty interface* $G \leftarrow 0$. Observe that the restriction to hypergraphs with empty interface would make the above proof fail: a non-joinable pre-critical pair $(S \leftarrow J)$, with J non empty, does not witness that rewriting is not confluent for all $G \leftarrow 0$.

A similar problem arises with term rewriting, when restricting to confluence of *ground terms* [30]. As an example consider the following term rewriting system defined on the signature with two unary symbols, f and g, and one constant c.

$$f(g(f(x))) \to x \qquad f(c) \to c \qquad g(c) \to c$$

The critical pair $f(g(x)) \leftarrow f(g(f(g(f(x))))) \to g(f(x))$ is not joinable, but the system is obviously ground confluent, as every ground term will eventually rewrite into c.

Our work therefore allows one to view Theorem 1 in a new light: as hypergraphs with empty interface are morally the graphical analogous of ground terms, we can say that ground confluence is not decidable for DPO rewriting with interfaces.

4 Confluence for PROP rewriting

As emphasised in the introduction, a major reason for interest in DPO rewriting with interfaces is that PROP rewriting (§2.2) may be interpreted therein. In this section we investigate how our confluence result behaves with respect to this interpretation. The outcome is that confluence is decidable for terminating PROP rewriting systems, where terms are taken modulo a chosen special Frobenius structure (Corollary 28). For arbitrary symmetric monoidal theories, confluence is also decidable, provided that certain additional conditions hold (Corollary 41).

4.1 From PROPs to Frobenius termgraphs

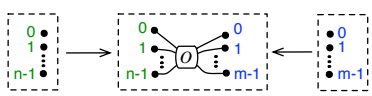
In this subsection we report a result from [4] that is crucial for the encoding of PROP rewriting into DPO rewriting with interfaces in $\mathbf{Hyp}_\Sigma$ (cf. Section 2.1).

First, we obtain our domain of interpretation by restricting the category $\mathsf{Csp}(\mathbf{Hyp}_\Sigma)$ with arrows the cospans $G_1 \leftarrow G_2 \rightarrow G_3$ of Σ-hypergraphs to those with G_1, G_3 discrete.

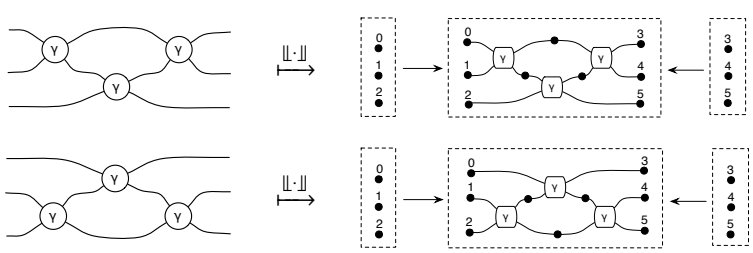
Definition 21 (Frobenius termgraphs). *Any $k \in \mathbb{N}$ can be seen as a discrete hypergraph with k vertices. The objects of the PROP* $\mathbf{FTerm}_\Sigma$ *of* Σ-Frobenius termgraphs *are natural numbers and arrows $n \to m$ are cospans $n \xrightarrow{f} G \xleftarrow{g} m$ in $\mathbf{Hyp}_\Sigma$ (where n, m are considered as hypergraphs).* $\mathbf{FTerm}_\Sigma$*, therefore, is a full subcategory of* $\mathsf{Csp}(\mathbf{Hyp}_\Sigma)$.

Explicitly, composition in $\mathbf{FTerm}_\Sigma$ is defined by pushout as in $\mathsf{Csp}(\mathbf{Hyp}_\Sigma)$ and the monoidal product $\oplus$ by coproduct in $\mathbf{Hyp}_\Sigma$. The idea behind the discreteness restriction is that f and g tell what are the "left and right dangling wires" in the string diagram encoded by G. In pictures, we shall represent n and m as actual discrete graphs— with n and m nodes respectively— and use number labels (and sometimes colours, whenever available to the reader) to help visualise how they get mapped to nodes of G.

Given a signature Σ, we define a PROP morphism $[\![\cdot]\!] \colon \mathbf{S}_\Sigma \to \mathbf{FTerm}_\Sigma$. Since $\mathbf{S}_\Sigma$ is the PROP freely generated by an SMT with no equations, it suffices to define $[\![\cdot]\!]$ on the generators: for each $o \colon n \to m$ in Σ, we let $[\![o]\!]$ be the following cospan of type $n \to m$.



Example 22. The two sides of the PROP rewriting rule (2) (Example 6) get interpreted as the following cospans in $\mathbf{FTerm}_\Sigma$.

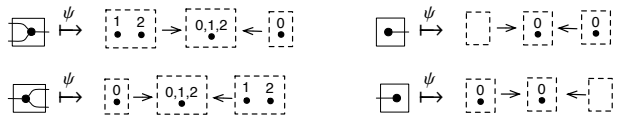


Proposition 23 ([4]). $\llbracket\cdot\rrbracket\colon \mathbf{S}_\Sigma \to \mathbf{FTerm}_\Sigma$ *is faithful.*

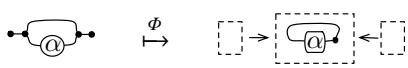
The encoding $\llbracket\cdot\rrbracket$ is an important part of Theorem 24 below. This is a pivotal result in our exposition, as it serves as a bridge between algebraic and combinatorial structures. Indeed, it provides a presentation, by means of generators and equations, for the PROP $\mathbf{FTerm}_\Sigma$: the disjoint union of the SMTs of $\mathbf{S}_\Sigma$ and **Frob**.

Theorem 24 ([4]). *There is an isomorphism of PROPs* $\Phi\colon \mathbf{S}_\Sigma + \mathbf{Frob} \xrightarrow{\cong} \mathbf{FTerm}_\Sigma$.

The isomorphism Φ is given as $[\llbracket\cdot\rrbracket, \psi]\colon \mathbf{S}_\Sigma + \mathbf{Frob} \to \mathbf{FTerm}_\Sigma$, where $\psi\colon \mathbf{Frob} \to \mathbf{FTerm}_\Sigma$ is the unique PROP morphism mapping the generators of **Frob** as follows



The special role played by **Frob** is what justifies the terminology *Frobenius termgraph*: it is used to model those features of the graph domain that are not part of the syntactic domain, e.g. the ability of building a "feedback loop" around some $\alpha\colon 1 \to 1$ in Σ.



4.2 Confluence for rewriting in $\mathbf{S}_\Sigma$ + Frob

We can use Theorem 24 to apply results for graphs with interfaces to $\mathbf{S}_\Sigma + \mathbf{Frob}$. First, one can turn the cospan $n \xrightarrow{i} G \xleftarrow{o} m = \Phi(d)$ interpreting a string diagram d into a graph with interface, which is defined as

$$\ulcorner n \xrightarrow{i} G \xleftarrow{o} m \urcorner \;:=\; G \xleftarrow{[i,o]} n + m.$$

For a system $\mathcal{R}$ we define the rewriting system $\ulcorner \Phi(\mathcal{R}) \urcorner$ in $\mathbf{FTerm}_\Sigma$ as

$$\{\langle \ulcorner \Phi(l) \urcorner, \ulcorner \Phi(r) \urcorner \rangle \mid \langle l, r \rangle \in \mathcal{R}\}.$$

Example 25. The PROP rewriting system $\mathcal{R}$ of Example 6 consists of just a single rule, let us call it $\langle d, e \rangle$. The resulting DPO rewriting system with interfaces $\ulcorner \Phi(\mathcal{R}) \urcorner$ is then presented in Example 7. Also, Example 22 is an intermediate step of this transformation, as it shows the cospans $\llbracket c \rrbracket = \Phi(c)$ and $\llbracket d \rrbracket = \Phi(d)$. One can obtain both graphs with interfaces $\ulcorner \Phi(c) \urcorner$ and $\ulcorner \Phi(d) \urcorner$ by "folding" the domain/codomain of the cospans into the interface of Example 7.

Observe that a rule in $\ulcorner \Phi(\mathcal{R}) \urcorner$ just consists of a pair of hypergraphs with a common interface, i.e., it is a DPO rule of the form $L \leftarrow n + m \to R$. Thus, PROP rewriting in $\mathbf{FTerm}_\Sigma$ coincides with DPO rewriting with interfaces: together with Theorem 24, this correspondence yields the following result.

Theorem 26 ([4]). *Let $\mathcal{R}$ be a rewriting system on* $\mathbf{S}_\Sigma + \mathbf{Frob}$.

1. *If $d \rightsquigarrow_{\mathcal{R}} e$, then $\ulcorner \Phi(d) \urcorner \Rightarrow_{\ulcorner \Phi(\mathcal{R}) \urcorner} \ulcorner \Phi(e) \urcorner$.*
2. *If $\ulcorner \Phi(d) \urcorner \Rightarrow_{\ulcorner \Phi(\mathcal{R}) \urcorner} (H \leftarrow J)$, then $\exists e$ such that $\ulcorner \Phi(e) \urcorner \cong (H \leftarrow J)$ and $d \rightsquigarrow_{\mathcal{R}} e$.*

One can read Theorem 26 as: DPO rewriting with interfaces is sound and complete for any symmetric monoidal theory with a chosen special Frobenius structure, i.e. one of shape $(\Sigma + \Sigma_F, E + E_F)$, with (Σ_F, E_F) the SMT of **Frob**. There are various relevant such theories in the literature, such as the ZX-calculus [15], the calculus of signal flow graphs [6], the calculus of stateless connectors [10] and monoidal computer [41].

The combination of the result above with Theorem 19 is however not sufficient for ensuring the decidability of the confluence for a terminating rewriting system $\mathcal{R}$ on $\mathbf{S}_\Sigma + \mathbf{Frob}$. Indeed, Theorem 19 and Theorem 26 ensure that if all the pre-critical pairs in $\ulcorner \Phi(\mathcal{R}) \urcorner$ are joinable, then the rewriting in $\mathcal{R}$ is confluent. However, for the decidability of confluence in $\mathcal{R}$ the reverse is also needed: if one pre-critical pair in $\ulcorner \Phi(\mathcal{R}) \urcorner$ is not joinable, then $\mathcal{R}$ should not be confluent. To conclude this fact, it is enough to check that all pre-critical pairs of $\ulcorner \Phi(\mathcal{R}) \urcorner$ lay in the image of $\ulcorner \Phi(\cdot) \urcorner$, i.e., that they all have discrete interfaces. The key observation is given by the lemma below.

Lemma 27 (Pre-critical pair with discrete interface). *Consider a pre-critical pair in* $\mathbf{Hyp}_\Sigma$ *as in* (3)*, Definition 8. If both K_1 and K_2 are discrete, so is the interface J.*

Proof. For $i = 1, 2$, since K_i is discrete, the hyperedges of C_i are exactly those of G_i that are not in $f_i(L_i)$. Since $[f_1, f_2] \colon L_1 + L_2 \to S$ is epi, all the hyperedges of G are either in $f_1(L_1)$ or $f_2(L_2)$. Therefore, J cannot contain any hyperedge. □

Since in every rule $L \leftarrow K \rightarrow R$ in $\ulcorner \Phi(\mathcal{R}) \urcorner$, K is discrete, from Lemma 27 and Theorem 19 we derive the following result.

Corollary 28. *Confluence is decidable for terminating rewriting systems on* $\mathbf{S}_\Sigma + \mathbf{Frob}$.

Proof. To decide confluence of a rewriting system $\mathcal{R}$ on $\mathbf{S}_\Sigma + \mathbf{Frob}$, it is enough to check whether all pre-critical pairs in $\ulcorner \Phi(\mathcal{R}) \urcorner$ are joinable. Indeed, if all pre-critical pairs are joinable, then $\rightsquigarrow_{\mathcal{R}}$ is confluent by Theorems 19 and 26. For the other direction, suppose that there exists a pre-critical pair $_{\ulcorner \Phi(\mathcal{R}) \urcorner} \Leftarrow (S \leftarrow J) \Rightarrow_{\ulcorner \Phi(\mathcal{R}) \urcorner}$ that is not joinable. By construction, in every rule $L \leftarrow K \rightarrow R$ in $\ulcorner \Phi(\mathcal{R}) \urcorner$, K is discrete. Therefore, by Lemma 27, also J is discrete. This is the key fact to entail that there exists d in $\mathbf{S}_\Sigma + \mathbf{Frob}$, such that $\ulcorner \Phi(d) \urcorner = (S \leftarrow J)$. By Theorem 26, d witnesses that $\rightsquigarrow_{\mathcal{R}}$ is not confluent.

Now, if $\mathcal{R}$ is terminating, then by Theorem 26, also $\ulcorner \Phi(\mathcal{R}) \urcorner$ is terminating. The latter is also computable and therefore joinability of pre-critical pairs of $\ulcorner \Phi(\mathcal{R}) \urcorner$ can easily be decided by following the steps in the second part of the proof of Corollary 20. □

4.3 Confluence for left-connected rewriting in $\mathbf{S}_\Sigma$

So far, we have shown a procedure to decide confluence for rewriting on $\mathbf{S}_\Sigma + \mathbf{Frob}$. In order to study PROP rewriting in absence of a chosen Frobenius structure, we focus on component $[\![\cdot]\!] \colon \mathbf{S}_\Sigma \to \mathbf{FTerm}_\Sigma$ of the isomorphism Φ.

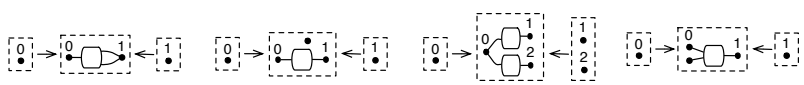
We first recall from [4] a combinatorial characterisation of the image of $\llbracket\cdot\rrbracket$. It is based on a few preliminary definitions. We call a sequence of hyperedges $e_1, e_2, \ldots, e_n$ a *directed path* if at least one target of e_k is a source for e_{k+1} and a *directed cycle* if additionally at least one target of e_n is a source for e_1. The *in-degree* of a node v in an hypergraph G is the number of pairs (h, i) where h is an hyperedge with v as its i-th target. Similarly, the *out-degree* of v is the number of pairs (h, j) where h is an hyperedge with v as its j-th source. We call *input* nodes those with in-degree 0, *output* nodes those with out-degree 0, and *internal* nodes the others. We write $\mathsf{in}(G)$ for the set of inputs and $\mathsf{out}(G)$ for the set of outputs.

Definition 29. *An hypergraph G is monogamous directed acyclic (mda) if*

1. *it contains no directed cycle (directed acyclicity) and*
2. *every node has at most in- and out-degree* 1 *(monogamy).*

A cospan $n \xrightarrow{f} G \xleftarrow{g} m$ in $\mathbf{FTerm}_\Sigma$ is monogamous directed acyclic when G is an mda-hypergraph, f is mono and its image is $\mathsf{in}(G)$, g is mono and its image is $\mathsf{out}(G)$.

Example 30. The following four cospans are *not* monogamous.

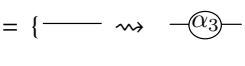
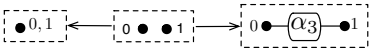


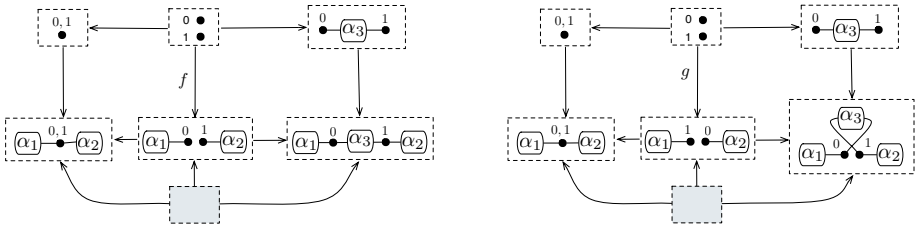
Theorem 31 ([4]). *$n \to G \leftarrow m$ in $\mathbf{FTerm}_\Sigma$ is in the image of $\llbracket\cdot\rrbracket$ iff it is mda.*

As for a graph with interface $G \xleftarrow{f} J$, we call it monogamous directed acyclic if so is G and the image of f coincides with $\mathsf{in}(G) + \mathsf{out}(G)$. This means that there exists a cospan $n \xrightarrow{i} G \xleftarrow{o} m$ such that $\ulcorner n \xrightarrow{i} G \xleftarrow{o} m \urcorner = G \xleftarrow{f} J$, i.e., $J = n + m$ and $f = [i, o]$.

We are now in position to interpret PROP rewriting for $\mathbf{S}_\Sigma$ in DPO-rewriting for mda-hypergraphs with interfaces, via the mapping $[\![\cdot]\!] \stackrel{\text{def}}{=} \ulcorner\llbracket\cdot\rrbracket\urcorner$ that takes string diagrams to mda hypergraphs with interfaces.

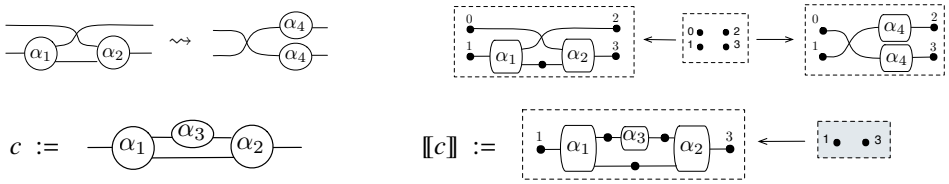
As shown in [4], this interpretation is generally *unsound*. There are two main reasons, which we illustrate in the next two examples. They motivate our restriction to PROP rewriting systems that make the interpretation sound, in Definition 34 below.

Example 32. Consider $\Sigma = \{\alpha_1 \colon 0 \to 1, \alpha_2 \colon 1 \to 0, \alpha_3 \colon 1 \to 1\}$ and the PROP rewriting system $\mathcal{R} = \{$  $\}$ on $\mathbf{S}_\Sigma$. In $\mathbf{FTerm}_\Sigma$, $[\![\mathcal{R}]\!]$ is given by the DPO rule of mda-hypergraphs with interface  . The rule is not left-linear and therefore pushout complements are not necessarily unique for the application of this rule, as witnessed by the following two DPO rewriting steps.

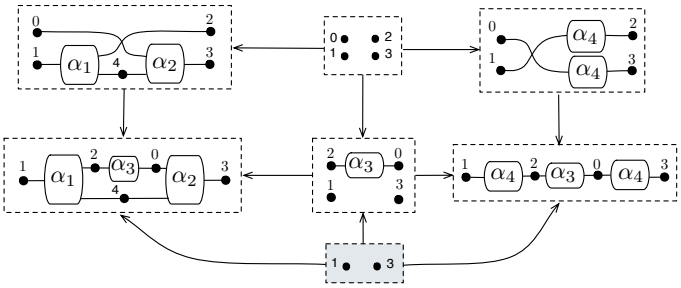


The different outcome is due to the fact that f maps 0 to the leftmost and 1 to the rightmost node, whereas g swaps the assignments. Even though both rewriting steps could be mimicked at the syntactic level in $\mathbf{S}_\Sigma$ + **Frob** (as guaranteed by Theorem 26, cf. [4, Ex. 4.8]), the rightmost one is illegal for $\mathcal{R}$ in $\mathbf{S}_\Sigma$.

Example 33. Take $\Sigma = \{\alpha_1 \colon 1 \to 2, \alpha_2 \colon 2 \to 1, \alpha_3 \colon 1 \to 1, \alpha_4 \colon 1 \to 1\}$ and a PROP rewriting system $\mathcal{R}$ on $\mathbf{S}_\Sigma$ given by the rewriting rule below left, interpreted in $[\![\mathcal{R}]\!]$ as below right. The next line introduces a diagram c of $\mathbf{S}_\Sigma$ and its interpretation.



Now, the left-hand side of the rule in $\mathcal{R}$ cannot be matched in c. However, their interpretation in $\mathbf{FTerm}_\Sigma$ yields a legal DPO rewriting step as below.



The two examples motivate the following definition.

Definition 34. *An mda-hypergraph G is* strongly connected *if for every input $x \in \mathsf{in}(G)$ and output $y \in \mathsf{out}(G)$, there exists a directed path from x to y. A DPO system with interface is called left-connected if it is left-linear and, for every rule $L \leftarrow K \rightarrow R$, $L \leftarrow K$ and $R \leftarrow K$ are mda-hypergraphs with interface and L is strongly connected. We call a PROP rewriting system $\mathcal{R}$ on $\mathbf{S}_\Sigma$ left-connected if $[\![\mathcal{R}]\!]$ is left-connected.*

Non-commutative bimonoids (Example 5(c), see also §5 below) and the Yang-Baxter rule of Example 6 are examples of left-connected rewriting systems.

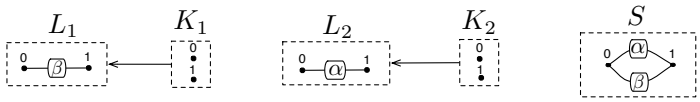
Intuitively, in Definition 34, strong connectedness prevents matches leaving "holes", as in Example 33, whereas left-linearity guarantees uniqueness of the pushout complements, and prevents the problem in Example 32. We are then able to prove the following.

Theorem 35. *Let $\mathcal{R}$ be a left-connected rewriting system on $\mathbf{S}_\Sigma$.*
1. *If $d \rightsquigarrow_{\mathcal{R}} e$, then $[\![d]\!] \Rightarrow_{[\![\mathcal{R}]\!]} [\![e]\!]$.*
2. *If $[\![d]\!] \Rightarrow_{[\![\mathcal{R}]\!]} (H \leftarrow J)$, then $\exists e$ such that $[\![e]\!] \cong (H \leftarrow J)$ and $d \rightsquigarrow_{\mathcal{R}} e$.*

Remark 36. For confluence, restricting to left-linearity is not particularly harmful. Indeed, an mda-hypergraph with interface $G \leftarrow J$ is not mono iff G has one node that is both input and output, i.e., an isolated node. A rule with a strongly connected $L \leftarrow K$ is not left-linear precisely when L is discrete, with a single node. Such a rule cannot be part of a terminating system, i.e. one where local confluence implies confluence.

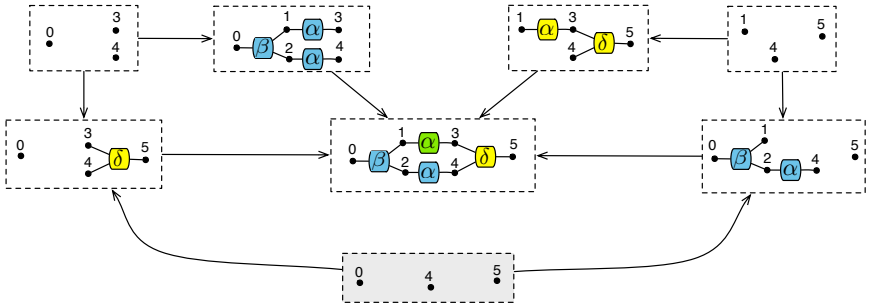
The above theorem allows us to use DPO rewriting with interfaces as a mechanism for rewriting $\mathbf{S}_\Sigma$. The last ingredient that we need for confluence is a suitable notion of pre-critical pair. One cannot simply reuse Definition 8. Indeed, we want to enforce that the common source $S \leftarrow J$ (cf. (3)) of the two derivations is an mda-hypergraph with interfaces, so that it is in the image of $[\![\cdot]\!]$ and we can reason about pre-critical pairs 'syntactically' in $\mathbf{S}_\Sigma$. However, while Lemma 27 guarantees that this is always the case for rewriting systems on $\mathbf{S}_\Sigma + \mathbf{Frob}$, with Definition 8 this is not guaranteed for $\mathbf{S}_\Sigma$ even in presence of left-connected rules, as shown by the two examples below.

Example 37. We concoct a pre-critical pair by instantiating (3) as shown below.



Although $L_1 \leftarrow K_1$ and $L_2 \leftarrow K_2$ are left-hand sides of left-connected rules, S is *not* monogamous, thus this pre-critical pair does not correspond to anything in the syntax.

Example 38. Even if we restrict to left-connected rules with an mda-hypergraph, defining the interface J by pullback as in Definition 34 may not yield an mda-hypergraph with interface. Here is an example, where two rules match in an mda-hypergraph G, but the interface contains one extra node 4 which is neither an input nor an output of G.



The previous two examples motivate the following definition.

Definition 39 (Mda pre-critical pair). *Let $\mathcal{R}$ be a left-connected DPO system containing the rules $L_1 \leftarrow K_1 \rightarrow R_1$ and $L_2 \leftarrow K_2 \rightarrow R_2$. Consider the following derivations with common source $S \leftarrow J$.*

$$\begin{tikzcd}
R_1 \arrow[d] & K_1 \arrow[l] \arrow[r] \arrow[d] & L_1 \arrow[dr, "f_1"] & & L_2 \arrow[dl, "f_2"'] & K_2 \arrow[l] \arrow[r] \arrow[d] & R_2 \arrow[d] \\
H_1 \arrow[ur, phantom, "\urcorner"] & C_1 \arrow[l] \arrow[rr] \arrow[urr, phantom, "\ulcorner"] & & S & & C_2 \arrow[ll] \arrow[r] \arrow[ull, phantom, "\urcorner"] & H_2 \arrow[ul, phantom, "\ulcorner"] \\
& & & J \arrow[u] \arrow[ull, bend left] \arrow[urr, bend right] & & &
\end{tikzcd}$$

We say that $(H_1 \leftarrow J) \Leftarrow (S \leftarrow J) \Rightarrow (H_2 \leftarrow J)$ is an mda pre-critical pair *if $[f_1, f_2]\colon L_1 + L_2 \rightarrow S$ is epi and $S \leftarrow J$ is an mda-hypergraph with interface; it is* joinable *if there exists an mda-hypergraph with interface $W \leftarrow J$ such that $(H_1 \leftarrow J) \Rightarrow^* (W \leftarrow J) \,{}^*\!\Leftarrow (H_2 \leftarrow J)$.*

We will drop the prefix mda, when there is no risk of confusion with Definition 8. We are now in position to state the confluence theorem for left-connected systems.

Theorem 40 (Local confluence for left-connected systems). *For a left-connected DPO system with interfaces, if all mda pre-critical pairs are joinable then rewriting is locally confluent: given an mda-hypergraph with interface $G_0 \leftarrow I$ and $(G_{1,1} \leftarrow I) \Leftarrow (G_0 \leftarrow I) \Rightarrow (G_{1,2} \leftarrow I)$, there exists an mda-hypergraph with interface $W \leftarrow I$ such that*

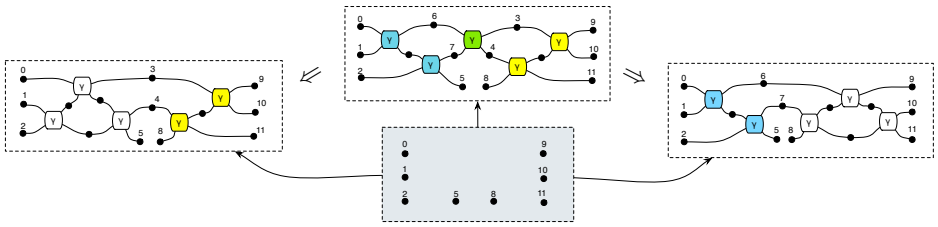
$$\begin{tikzcd}
& (G_0 \leftarrow I) \arrow[dl, Rightarrow] \arrow[dr, Rightarrow] & \\
(G_{1,1} \leftarrow I) \arrow[dr, Rightarrow, "*"'] & & (G_{1,2} \leftarrow I) \arrow[dl, Rightarrow, "*"] \\
& (W \leftarrow I). &
\end{tikzcd}$$

The proof of Theorem 40 follows steps analogous to the one of Theorem 19. The essential difference is that mda pre-critical pairs now have interfaces that are not necessarily pullbacks. The assumption of left-connectedness is, nevertheless, enough to ensure that the fundamental pieces, Constructions 14 and 16, can be reproduced.

Corollary 41. *Let $\mathcal{R}$ be a terminating left-connected rewriting system on $\mathbf{S}_\Sigma$. Then confluence of $\rightsquigarrow_{\mathcal{R}}$ is decidable.*

Proof. By Theorem 35 and 40, it is enough to check whether pre-critical pairs in $[\![\mathcal{R}]\!]$ are joinable. This is decidable since $\mathcal{R}$ is terminating and $[\![\mathcal{R}]\!]$ is computable. □

Example 42. The PROP rewriting system $\mathcal{R}$ of Example 6 is left-connected. Once interpreted as the DPO system with interfaces of Example 7, we can do critical pair analysis. The mda pre-critical pair below (where the middle grey graph acts as the interface for the rewriting steps) is not joinable, meaning that $\mathcal{R}$ is not confluent.

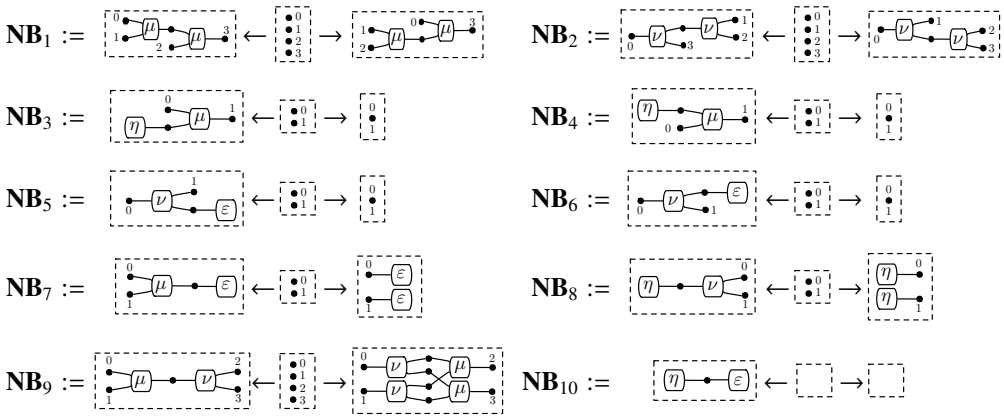


We emphasise that the decision procedure relies on the fact that there are only *finitely many* pre-critical pairs to consider — the above one being the only to feature a nontrivial overlap of rule applications. This is in contrast with a naive, 'syntactic' analysis, which as we observe in Example 6 yields infinitely many pre-critical pairs for $\mathcal{R}$.

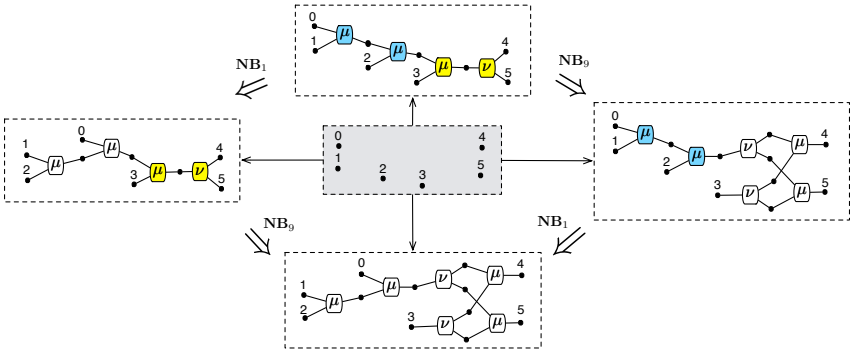
We will devote the next section to a positive example of our confluence result.

5 Case study: non-commutative bimonoids

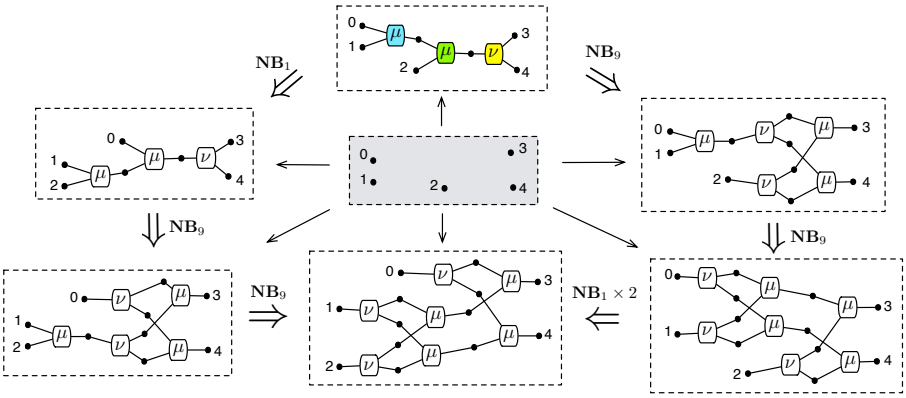
We conclude with an application of the left-connected case, showing confluence of the theory **NB** of non-commutative bimonoids (Example 5(c)). Below is the interpretation of the theory as a DPO system $[\![\mathcal{R}_{\mathbf{NB}}]\!]$, which was shown to be terminating in [4].



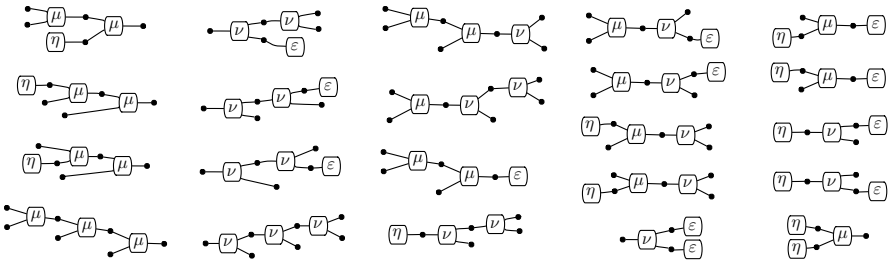
Given that the system is terminating, it suffices to show local confluence. Observe that $[\![\mathcal{R}_{\mathbf{NB}}]\!]$ is left-connected: monogamicity is ensured by the fact that it is in the image of $[\![\cdot]\!]$; strong connectedness and left-linearity hold by inspection of the set of rules. We can thus use Theorem 40 and local confluence follows from joinability of the pre-critical pairs. Among them, the pairs without overlap of rule applications pose no problem: they are trivially joinable in one step, by applying the other rule. One example is given below, with the middle grey graph acting as the interface for all depicted derivation steps.



Thus we confine ourselves to analysing actual critical pairs, with overlapping rule applications. One such pair is given below, also involving rules $\mathbf{NB}_1$ and $\mathbf{NB}_9$. Again, we show how it is joined, with the interface of each step drawn in the centre.



Overall there are 22 critical pairs to consider. For space reasons, for each of them we only show the graph exhibiting the overlap. It is straightforward to check that the corresponding pairs are all joinable.



We can thereby conclude that **NB** is a confluent rewriting system. Since it is also terminating, equivalence of terms in **NB** is decidable by means of rewriting. Note that, by virtue of Corollary 41, the above pre-critical pair analysis can be automated.

6 Conclusion

The starting observation of this paper (Theorem 19) is that the Knuth-Bendix property holds for DPO with interfaces; as an easy corollary (Corollary 20), for a terminating system, confluence is decidable. The relevance of this is two-fold. On the conceptual side, it puts graph rewriting in tight correspondence with term rewriting: when considering rewriting with interfaces, confluence is decidable both for graphs and terms [33], while the appropriate notion of ground confluence is undecidable in both cases [30,42].

On the side of applications, our result allows one to study confluence for SMTs. One simple consequence of Theorem 19 and of our previous work in [4] is that, for all those SMTs including a special Frobenius structure – which are already commonplace in computer science [10,11,47,15,16,6,7,1,23] – local confluence can be checked by means of critical pair analysis. Moreover, when termination is guaranteed, confluence can be decided automatically (Corollary 28). An analogous result (Corollary 41) holds for those SMTs that do not include a special Frobenius structure, but whose set of rules satisfies the left-connected conditions. Hence it applies to a variety of other non-Frobenius theories, such as those in [35,26,21]. In both cases, these decision procedures are amenable to implementation in string diagram rewriting tools like Quantomatic [32] (via an encoding of hypergraphs) or directly in hypergraph-based rewriting tools.

Acknowledgment. Aleks Kissinger and Fabio Zanasi acknowledge support from the ERC under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant n° 320571. The work of Filippo Bonchi has been partly supported by the project ANR-16-CE25-0011 REPAS and Labex MILYON/ANR-10-LABX-0070. The work of Fabio Gadducci has been partly supported by the project PRA_2016_64 "Through the fog" funded by the University of Pisa.

References

1. J. Baez and J. Erbele. Categories in control. *Theory and Application of Categories*, 30:836–881, 2015.
2. P. Baldan, F. Gadducci, and P. Sobocinski. Adhesivity is not enough: Local church-rosser revisited. In *MFCS 2011*, volume 6907 of *LNCS*, pages 48–59. Springer, 2011.
3. G. Bauer and F. Otto. Finite complete rewriting systems and the complexity of the word problem. *Acta Informatica*, 21(5):521–540, 1984.
4. F. Bonchi, F. Gadducci, A. Kissinger, P. Sobociński, and F. Zanasi. Rewriting modulo symmetric monoidal structure. In *LiCS 2016*, pages 710–719. ACM, 2016.
5. F. Bonchi, F. Gadducci, and B. König. Synthesising CCS bisimulation using graph rewriting. *Information and Computation*, 207(1):14–40, 2009.
6. F. Bonchi, P. Sobocinski, and F. Zanasi. A categorical semantics of signal flow graphs. In *CONCUR 2014*, volume 8704 of *LNCS*, pages 435–450. Springer, 2014.
7. F. Bonchi, P. Sobocinski, and F. Zanasi. Full abstraction for signal flow graphs. In *POPL 2015*, pages 515–526. ACM, 2015.
8. H. J. S. Bruggink, R. Cauderlier, M. Hülsbusch, and B. König. Conditional reactive systems. In *FSTTCS 2011*, volume 13 of *LIPIcs*, pages 191–203. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2011.
9. R. Bruni, F. Gadducci, and U. Montanari. Normal forms for algebras of connection. *Theoretical Computer Science*, 286(2):247–292, 2002.
10. R. Bruni, I. Lanese, and U. Montanari. A basic algebra of stateless connectors. *Theoretical Computer Science*, 366(1–2):98–120, 2006.
11. R. Bruni, H. C. Melgratti, and U. Montanari. A connector algebra for P/T nets interactions. In *CONCUR 2011*, volume 6901 of *LNCS*, pages 312–326. Springer, 2011.
12. A. Burroni. Higher dimensional word problems with applications to equational logic. *Theoretical Computer Science*, 115(1):43–62, 1993.
13. A. Carboni. Matrices, relations, and group representations. *Journal of Algebra*, 136(1):497–529, 1991.
14. A. Carboni and R. F. C. Walters. Cartesian bicategories I. *Journal of Pure and Applied Algebra*, 49(1-2):11–32, 1987.
15. B. Coecke and R. Duncan. Interacting quantum observables. In *ICALP 2008*, volume 5216 of *LNCS*, pages 298–310. Springer, 2008.
16. B. Coecke, R. Duncan, A. Kissinger, and Q. Wang. Strong complementarity and non-locality in categorical quantum mechanics. In *LiCS 2012*, pages 245–254. ACM, 2012.
17. A. Corradini. On the definition of parallel independence in the algebraic approaches to graph transformation. In *STAF 2016*, volume 9946 of *LNCS*. Springer, 2016.
18. H. Ehrig, A. Habel, J. Padberg, and U. Prange. Adhesive high-level replacement categories and systems. In *ICGT 2004*, volume 2987 of *LNCS*, pages 144–160. Springer, 2004.
19. H. Ehrig and B. König. Deriving bisimulation congruences in the DPO approach to graph rewriting. In *FoSSaCS 2004*, volume 2987 of *LNCS*, pages 151–166. Springer, 2004.
20. H. Ehrig and H.-J. Kreowski. Parallelism of manipulation in multidimensional information structures. In *MFCS 1976*, volume 45 of *LNCS*, pages 284–293. Springer, 1976.
21. M. P. Fiore and M. D. Campos. The algebra of directed acyclic graphs. In *Abramsky Festschrift*, volume 7860 of *LNCS*, pages 37–51. Springer, 2013.
22. B. Fong. Decorated cospans. *Theory and Applications of Categories*, 30(33):1096–1120, 2015.
23. B. Fong, P. Rapisarda, and P. Sobociński. A categorical approach to open & interconnected dynamical systems. In *LiCS 2016*, pages 495–504. ACM, 2016.

24. F. Gadducci. Graph rewriting for the π-calculus. *Mathematical Structures in Computer Science*, 17(3):407–437, 2007.
25. F. Gadducci and R. Heckel. An inductive view of graph transformation. In *WADT 1997*, volume 1376 of *LNCS*, pages 223–237. Springer, 1997.
26. D. R. Ghica. Diagrammatic reasoning for delay-insensitive asynchronous circuits. In *Abramsky Festschrift*, volume 7860 of *LNCS*, pages 52–68. Springer, 2013.
27. G. Huet and D. Lankford. On the uniform halting problem for term rewriting systems. Technical Report 283, IRIA, 1978.
28. M. Hyland and J. Power. Lawvere theories and monads. In *Plotkin Festschrift*, volume 172 of *ENTCS*, pages 437–458. Elsevier, 2007.
29. A. Joyal and R. Street. The geometry of tensor calculus, I. *Advances in Mathematics*, 88(1):55–112, 1991.
30. D. Kapur, P. Narendran, and F. Otto. On ground-confluence of term rewriting systems. *Information and Computation*, 86(1):14–31, 1990.
31. A. Kissinger. Finite matrices are complete for (dagger)-hypergraph categories. arXiv:1406.5942 [math.CT].
32. A. Kissinger and V. Zamdzhiev. Quantomatic: A proof assistant for diagrammatic reasoning. In *CADE-25*, volume 9195 of *LNCS*, pages 326–336. Springer, 2015.
33. D. E. Knuth and P. B. Bendix. Simple word problems in universal algebras. In *Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, 1970.
34. S. Lack and P. Sobociński. Adhesive and quasiadhesive categories. *Theoretical Informatics and Applications*, 39(3):511–546, 2005.
35. Y. Lafont. Towards an algebraic theory of Boolean circuits. *Journal of Pure and Applied Algebra*, 184(2–3):257–310, 2003.
36. S. Mac Lane. Categorical algebra. *Bulletin of the American Mathematical Society*, 71(1):40–106, 1965.
37. J. Meseguer and U. Montanari. Petri nets are monoids. *Information and Computation*, 88(2):105–155, 1990.
38. S. Mimram. Computing critical pairs in 2-dimensional rewriting systems. In *RTA 2010*, volume 6 of *LIPIcs*, pages 227–242. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2010.
39. S. Mimram. Towards 3-dimensional rewriting theory. *Logical Methods in Computer Science*, 10(2), 2014.
40. P. Padawitz. New results on completeness and consistency of abstract data types. In *MFCS 1980*, volume 88 of *LNCS*, pages 460–473. Springer, 1980.
41. D. Pavlovic. Monoidal computer I: Basic computability by string diagrams. *Information and Computation*, 226:94–116, 2013.
42. D. Plump. Hypergraph rewriting: Critical pairs and undecidability of confluence. In *Term Graph Rewriting: Theory and Practice*, pages 201–213. Wiley, 1993.
43. D. Plump. Checking graph-transformation systems for confluence. In *Manipulation of Graphs, Algebras and Pictures*, volume 26 of *ECEASST*. EASST, 2010.
44. V. Sassone and P. Sobociński. Reactive systems over cospans. In *LiCS 2005*, pages 311–320. ACM, 2005.
45. P. Selinger. A survey of graphical languages for monoidal categories. *Springer Lecture Notes in Physics*, 13(813):289–355, 2011.
46. P. Sobociński. *Deriving process congruences from reaction rules*. PhD thesis, BRICS, University of Aarhus, 2004.
47. P. Sobociński and O. Stephens. A programming language for spatial distribution of net systems. In *Petri Nets 2014*, volume 8489 of *LNCS*, pages 150–169. Springer, 2014.
48. R. Street. Limits indexed by category-valued 2-functors. *Journal of Pure and Applied Algebra*, 8(2):149 – 181, 1976.