# Left-Linear Rewriting in Adhesive Categories

**Paolo Baldan**
Department of Mathematics, University of Padua, Italy

**Davide Castelnovo**
Department of Mathematics, University of Padua, Italy

**Andrea Corradini**
Department of Computer Science, University of Pisa, Italy

**Fabio Gadducci**
Department of Computer Science, University of Pisa, Italy

## Abstract

When can two sequential steps performed by a computing device be considered (causally) independent? This is a relevant question for concurrent and distributed systems, since independence means that they could be executed in any order, and potentially in parallel. Equivalences identifying rewriting sequences which differ only for independent steps are at the core of the theory of concurrency of many formalisms. We investigate the issue in the context of the double pushout approach to rewriting in the general setting of adhesive categories. While a consolidated theory exists for linear rules, which can consume, preserve and generate entities, this paper focuses on left-linear rules which may also "merge" parts of the state. This is an apparently minimal, yet technically hard enhancement, since a standard characterisation of independence that – in the linear case – allows one to derive a number of properties, essential in the development of a theory of concurrency, no longer holds. The paper performs an in-depth study of the notion of independence for left-linear rules: it introduces a novel characterisation of independence, identifies well-behaved classes of left-linear rewriting systems, and provides some fundamental results including a Church-Rosser property and the existence of canonical equivalence proofs for concurrent computations. These results properly extends the class of formalisms that can be modelled in the adhesive framework.

**2012 ACM Subject Classification** Theory of computation → Models of computation; Theory of computation → Semantics and reasoning

**Keywords and phrases** Adhesive categories, double-pushout rewriting, left-linear rules, switch equivalence, local Church-Rosser property

**Digital Object Identifier** 10.4230/LIPIcs.CONCUR.2024.11

**Related Version** *Full Version*: `https://arxiv.org/abs/2407.06181`

**Funding** The research has been partially supported by the EuropeanUnion – NextGenerationEU under the National Recovery and Resilience Plan (NRRP) – Call PRIN 2022 PNRR – Project P2022HXNSC "Resource Awareness in Programming: Algebra, Rewriting, and Analysis", by the Italian MUR – Call PRIN 2022 – Project 20228KXFN2 "Spatio-Temporal Enhancement of Neural nets for Deeply Hierarchical Automatised Logic" and by the University of Pisa – Call PRA 2022 – Project 2022_99 "Formal Methods for the Healthcare Domain based on Spatial Information".

## 1 Introduction

One of the key pay-off of concurrency theory is the idea that the behaviour of a computational device can be modelled by abstracting its sequences of steps via a suitable equivalence. Intuitively, the equivalence captures when steps are causally unrelated and thus could be executed in any order, and possibly in parallel. Two seminal contributions to this line of research have been Mazurkiewicz's traces [37] and Winskel's event structures [39]. Concerning

35th International Conference on Concurrency Theory (CONCUR 2024).
Editors: Rupak Majumdar and Alexandra Silva; Article No. 11; pp. 11:1–11:24
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

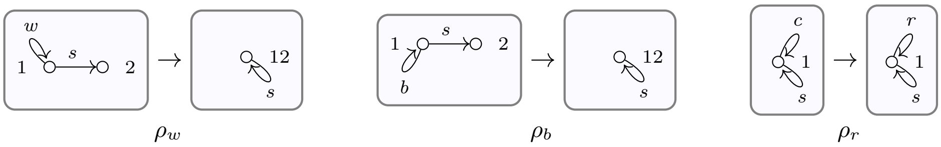


**Figure 1** Rewriting rules for the coffee example.

formalisms based on the rewriting paradigm, concurrency often boils down to having a notion of independence between two consecutive steps. Noteworthy examples are the interchange law [38] in term rewriting and permutation equivalence [35] in $\lambda$-calculi.

Among rewriting-based formalisms, those manipulating graph-like structures proved useful in many settings for representing the dynamics of distributed systems: the graph is used to represent the entities and their relations within states, while rewriting steps, modifying the graph, model computation steps that result in changes to the system state. The DPO (double-pushout) approach [25] is nowadays considered a standard one for these structures, thanks to its locality (rule application has a local effect on a state, differently from more recent approaches like SqPO [17] or PBPO$^+$ [40]) and to its flexibility: the rules allow to specify that, in a rewriting step, some entities in the current state are removed, some others are needed for the rewriting step to occur but remain unaltered, and some new entities can be generated. Concerning concurrency, the chosen notion is switch (also referred to as shift) equivalence, and independence is formally captured by what is called the Church-Rosser theorem of DPO rewriting [16, Chapter 3, Section 3.4].

A noteworthy advantage of DPO rewriting is that it can be formulated over adhesive categories and their variants [26, 32], which include, among others, toposes [31] and string diagrams [9]. Operating within the framework of adhesive categories allows one to devise the foundations of a rewriting theory that can be then instantiated to the context of interest, thus avoiding the need of repeatedly proving similar ad-hoc results for each specific setting.

So far, most of the theoretical results focused on *linear rules*: intuitively, a step is required only to consume, preserve and generate entities. However, in some situations it is also necessary to "merge" parts of the state. This happens in the context of string diagrams mentioned above, but also in graphical implementations of nominal calculi where, as a result of name passing, the received name is identified with a local one at the receiver [18, 27], or in the visual modelling of bonding in biological/chemical processes [41]. A similar mechanism is at the core of e-graphs (equality graphs), used for rewrite-driven compiler optimisations [42]: rather than merging nodes corresponding to equal expressions, the idea – conceptually and, as we will see, also mathematically similar – is to maintain an equivalence over the nodes.

Technically, to acquire the expressiveness needed for capturing fusions, i.e. the merge of some elements in the state, requires to move from linear to left-linear rewriting rules. While left-linear rules have been considered in various instances (mainly in categories of graphs) and some results concerning the corresponding rewriting theory have been put forward [6, 23, 24], the phenomena arising when devising a theory of independence and concurrency for rewriting with left-linear rules in the general setting of adhesive and quasi-adhesive categories have not been systematically explored, even if an investigation in the setting of quasi-topoi, considering non-linear rules yet restricting their applicability, has been presented in [7, 8].

Just to get some basic intuition about left-linear rewriting systems, their capabilities and the problems that can arise, assume we want to model a situation in which a coffee comes with a bag of white sugar and a bag of brown sugar. The situation is represented by the graph $G_0$ in Fig. 2: the loop $c$ is the coffee, the two loops $b$ and $w$ represent the white and

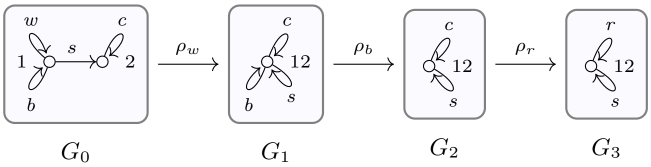


**Figure 2** A sequence of rewriting steps.

brown sugar. The corresponding nodes are connected to the coffee node by an $s$-labelled edge to indicate that they can be possibly added to the coffee. The available rules are in Fig. 1. Rules $\rho_w$ and $\rho_b$ model the addition of white and brown sugar, respectively, to the coffee. Note that adding one kind of sugar, say $x$, to the coffee is realised by deleting loop $x$ and merging the corresponding node with the coffee node so that a self-loop $s$ is created. Once some sugar has been added, the coffee is ready for drinking. This is represented by rule $\rho_r$, which checks for the presence of sugar, represented by the loop $s$, in the coffee, and brings it to the $r$ state, ready for drinking.

A rewriting sequence is depicted in Fig. 2, applying $\rho_w$, $\rho_b$, and $\rho_r$ in sequence, thus producing a coffee with both white and brown sugar, ready for drinking. A first interesting observation is that after applying rule $\rho_w$ to $G_0$, thus producing graph $G_1$, we can still apply rule $\rho_b$, with a non-injective match. Intuitively, once the coffee is ready for drinking because some sugar has been added, adding more sugar will not change its state. Indeed, the rules could also be applied in reverse order. Should they be considered independent? Moreover, since rule $\rho_r$ just requires the presence of sugar, it could be applied immediately after $\rho_w$. This might lead to think that $\rho_b$ and $\rho_r$ are independent. However, this is not completely clear, since in absence of $\rho_w$, the application of $\rho_b$ would be essential to enable $\rho_r$.

In this paper we perform a systematic exploration of these phenomena at the level of $\mathcal{M}$-adhesive categories, which encompass a large family of DPO-based formalisms.

As an initial step, we observe that for left-linear rules some fundamental properties that play a role in the concurrency theory of linear rewriting need to be deeply revised, or plainly do not hold. Firstly, the notion of sequential independence and the Church-Rosser theorem, ensuring that sequential independent steps can be performed in reverse order, does not hold out of the box. Even worse, given two independent steps it might be possible to switch them in several different ways. As we will observe, this prevents the development of a sensible theory of concurrency, since switches are used as a basis for equating sequences of rewriting steps that differ only in the order of independent steps.

We single out a class of left-linear rewriting systems, referred to as *well-switching rewriting systems*, where sequential independence ensures existence and uniqueness of the switch, and argue that this is the right setting for dealing with left-linear rewriting systems.

On the one hand, they capture most categories of interest for rewriting. Indeed, we identify general classes of left-linear rewriting systems on graph-like structures that are well-switching. As a sanity check, we prove that linear rewriting systems are well-switching.

On the other hand, we argue that a concurrency theory of rewriting can be developed for well-switching rewriting systems, recovering, sometimes in weakened form, most of the results that hold in the linear case. The development faces a main technical obstacle. In the literature on linear rewriting systems, a central result is the characterisation of switch equivalence of sequences of steps based on what are called processes or consistent permutations [2,4,15,30]. In turn this is the key to derive some fundamental properties for linear rewriting, namely

1. *globality of independence*: if two independent steps are moved forward or backward due to the application of switchings to a sequence, they remain independent;
2. *consistency of switching*: when transforming sequences of rewriting steps into equivalent ones by switching independent steps, the result does not depend on the order in which switchings are applied, but only on the associated permutation.

Properties (1) and (2) allow one to deduce a canonical form for equivalence proofs between sequences of rewriting steps seen as concurrent computations.

The fact that for left-linear rewriting systems the characterisation of switch equivalence via consistent permutations fails, and thus cannot be used for proving (1) and (2), leads to asking if these properties hold. We show that much of the theory can be retained: globality of independence holds in a weaker form, conceptually linked to the fact that fusions introduce a form of disjunctive causality, while consistency of switching holds unchanged. Finally, we prove that a canonical form for switching sequences can also be recovered. These results represent a solid basis for developing a satisfactory theory of concurrency for left-linear rules.

**Synopsis.** In §2 we recall the basic definitions of DPO rewriting for $\mathcal{M}$-adhesive categories. In §3 we present the key novelties of our proposal, introducing the standard notion of independence and a novel axiomatic notion of switchability, showing that the former fails to imply the latter for left-linear rewriting systems. We then define well-switching rewriting systems, a subclass of left-linear rewriting systems, and we prove that the classical results for linear rewriting systems are recovered for such a class. Finally, in §4 we outline directions for future work. Appendix A recalls basic results on adhesive categories and Appendix B presents *strong enforcing* rewriting systems, a class of systems where the local Church-Rosser Theorem holds. Appendix C of the extended version provides the proofs of our results.

## 2 $\mathcal{M}$-adhesive categories and rewriting systems

This section recalls the basic theory of *$\mathcal{M}$-adhesive categories* [1, 20, 21, 30, 32]. Given a category $\mathbf{X}$ we will not distinguish notationally between $\mathbf{X}$ and its class of objects, so "$X \in \mathbf{X}$" means that $X$ is an object of $\mathbf{X}$. We let $\mathsf{Mor}(\mathbf{X})$, $\mathsf{Mono}(\mathbf{X})$ and $\mathsf{Reg}(\mathbf{X})$ denote the class of all arrows, monos and regular monos of $\mathbf{X}$, respectively. Given an integer $n \in \mathbb{Z}$, $[0, n]$ denotes the set of natural numbers less than or equal to $n$; in particular, $[0, n] = \emptyset$ if $n < 0$.

### 2.1 $\mathcal{M}$-adhesivity

The key property of $\mathcal{M}$-adhesive categories is the *Van Kampen condition* [10, 31, 32]. Let $\mathbf{X}$ be a category. A subclass $\mathcal{A}$ of $\mathsf{Mor}(\mathbf{X})$ is called

- *stable under pushouts (pullbacks)* if for every pushout (pullback) square as the one below, if $m \in \mathcal{A}$ ($n \in \mathcal{A}$) then $n \in \mathcal{A}$ ($m \in \mathcal{A}$);
- *closed under composition* if $h, k \in \mathcal{A}$ implies $h \circ k \in \mathcal{A}$ whenever $h$ and $k$ are composable.

$$\begin{tikzcd}
A \arrow[r, "f"] \arrow[d, "m"'] & B \arrow[d, "n"] \\
C \arrow[r, "g"'] & D
\end{tikzcd}$$

▶ **Definition 2.1** (Van Kampen property). *Let $\mathbf{X}$ be a category and consider the diagram below. Given a class of arrows $\mathcal{A} \subseteq \mathsf{Mor}(\mathbf{X})$, we say that the bottom square is an* $\mathcal{A}$-Van Kampen square *if*

1. *it is a pushout square;*
2. *whenever the cube above has pullbacks as back and left faces and the vertical arrows belong to $\mathcal{A}$, then its top face is a pushout if and only if the front and right faces are pullbacks.*

*Pushout squares that enjoy only the "if" half of item (2) above are called* $\mathcal{A}$-stable.

*A* $\mathsf{Mor}(\mathbf{X})$*-Van Kampen square is called* Van Kampen *and a* $\mathsf{Mor}(\mathbf{X})$*-stable square* stable.

$$\begin{tikzcd}
 & A' \arrow[rr, "f'"] \arrow[dl, "m'"'] \arrow[dd, "a"' near end] & & B' \arrow[dl, "n'"'] \arrow[dd, "b"] \\
C' \arrow[rr, "g'" near end] \arrow[dd, "c"'] & & D' \arrow[dd, "d" near start] & \\
 & A \arrow[rr, "f" near end] \arrow[dl, "m"] & & B \arrow[dl, "n"] \\
C \arrow[rr, "g"'] & & D &
\end{tikzcd}$$

We can now define $\mathcal{M}$-adhesive categories.

▶ **Definition 2.2** ($\mathcal{M}$-adhesive category)**.** *Let* $\mathbf{X}$ *be a category and* $\mathcal{M}$ *a subclass of* $\mathsf{Mono}(\mathbf{X})$ *including all isomorphisms, closed under composition, and stable under pullbacks and pushouts. The category* $\mathbf{X}$ *is said to be* $\mathcal{M}$-adhesive *if*

1. *it has* $\mathcal{M}$-pullbacks*, i.e. pullbacks along arrows of* $\mathcal{M}$*;*
2. *it has* $\mathcal{M}$-pushouts*, i.e. pushouts along arrows of* $\mathcal{M}$*;*
3. $\mathcal{M}$*-pushouts are* $\mathcal{M}$*-Van Kampen squares.*

*A category* $\mathbf{X}$ *is said to be* strictly $\mathcal{M}$-adhesive *if* $\mathcal{M}$*-pushouts are Van Kampen squares.*

We write $m \colon X \rightarrowtail Y$ to denote that an arrow $m \colon X \to Y$ belongs to $\mathcal{M}$.

▶ Remark 2.3. *Adhesivity* and *quasiadhesivity* [28,32] coincide with strict $\mathsf{Mono}(\mathbf{X})$-adhesivity and strict $\mathsf{Reg}(\mathbf{X})$-adhesivity, respectively.

▶ **Example 2.4.** Every category $\mathbf{X}$ is $\mathsf{I}(\mathbf{X})$-adhesive, where $\mathsf{I}(\mathbf{X})$ is the class of isomorphisms.

$\mathcal{M}$-adhesivity is well-behaved with respect to the construction of slice and functor categories [36], as shown by the following theorems [19,32].

▶ **Theorem 2.5.** *Let* $\mathbf{X}$ *be an* $\mathcal{M}$*-adhesive category. Given an object* $X$ *the category* $\mathbf{X}/X$ *is* $\mathcal{M}/X$*-adhesive with* $\mathcal{M}/X := \{m \in \mathsf{Mor}(\mathbf{X}/X) \mid m \in \mathcal{M}\}$*. Similarly,* $X/\mathbf{X}$ *is* $X/\mathcal{M}$*-adhesive with* $X/\mathcal{M} := \{m \in \mathsf{Mor}(X/\mathbf{X}) \mid m \in \mathcal{M}\}$*.*

*Moreover for every small category* $\mathbf{Y}$*, the category* $\mathbf{X}^{\mathbf{Y}}$ *of functors* $\mathbf{Y} \to \mathbf{X}$ *is* $\mathcal{M}^{\mathbf{Y}}$*-adhesive, where* $\mathcal{M}^{\mathbf{Y}} := \{\eta \in \mathsf{Mor}(\mathbf{X}^{\mathbf{Y}}) \mid \eta_Y \in \mathcal{M} \text{ for every } Y \in \mathbf{Y}\}$*.*

We can list various examples of $\mathcal{M}$-adhesive categories (see [12,13,33]).

▶ **Example 2.6.** **Set** is adhesive, and, more generally, every topos is adhesive [33]. By the closure properties above, every presheaf $[\mathbf{X}, \mathbf{Set}]$ is adhesive, thus the category $\mathbf{Graph} = [E \rightrightarrows V, \mathbf{Set}]$ is adhesive where $E \rightrightarrows V$ is the two objects category with two morphisms $s, t \colon E \to V$. Similarly, various categories of hypergraphs can be shown to be adhesive, such as term graphs and hierarchical graphs [14]. Note that the category **sGraphs** of simple graphs, i.e. graphs without parallel edges, is $\mathsf{Reg}(\mathbf{sGraphs})$-adhesive [8] but not quasiadhesive.

A number of properties of adhesive categories that play a role in the theory of rewriting generalise to $\mathcal{M}$-adhesive categories. These include $\mathcal{M}$-pushout-pullback decomposition and uniqueness of pushouts complements (details and proofs are in Appendix A).

## 2.2 DPO rewriting systems derivations

$\mathcal{M}$-adhesive categories are the right context in which to perform abstract rewriting using what is called the Double-Pushout (DPO) approach.

▶ **Definition 2.7** ([29, 30]). *Let $\mathbf{X}$ be an $\mathcal{M}$-adhesive category. A* left $\mathcal{M}$-linear *rule $\rho$ is a pair of arrows $\rho = (l : K \to L, r : K \to R)$ such that $l$ belongs to $\mathcal{M}$. The rule $\rho$ is* $\mathcal{M}$-linear *if $r \in \mathcal{M}$ too. The object $L$ is the* left-hand side*, $R$ the* right-hand side*, and $K$ the* interface*.*

*A* left-linear rewriting system *is a pair $(\mathbf{X}, \mathsf{R})$ where $\mathbf{X}$ is an $\mathcal{M}$-adhesive category and $\mathsf{R}$ a set of left $\mathcal{M}$-linear rules. It is called* $\mathcal{M}$-linear *if every rule in $\mathsf{R}$ is $\mathcal{M}$-linear.*

*Given two objects $G$ and $H$ and a rule $\rho = (l, r)$ in $\mathsf{R}$, a* direct derivation $\mathscr{D}$ from $G$ to $H$ applying rule $\rho$ *is a diagram as the one below, in which both squares are pushouts. The arrow $m$ is called the* match *of the derivation, while $h$ is its* co-match*. We denote a direct derivation $\mathscr{D}$ from $G$ to $H$ as $\mathscr{D} \colon G \Mapsto H$.*

$$\begin{tikzcd}
L \arrow[d, "m"'] & K \arrow[l, tail, "l"'] \arrow[r, "r"] \arrow[d, "k"] & R \arrow[d, "h"] \\
G & D \arrow[l, tail, "f"] \arrow[r, "g"'] & H
\end{tikzcd}$$

A derivation can now be defined simply as a sequence of direct derivations.

▶ **Definition 2.8** (Derivation). *Let $\mathbf{X}$ be an $\mathcal{M}$-adhesive category and $(\mathbf{X}, \mathsf{R})$ a left-linear rewriting system. Given $G, H \in \mathbf{X}$, a* derivation *$\underline{\mathscr{D}}$ with source $G$ and target $H$, written $\underline{\mathscr{D}} : G \Mapsto H$, is defined as a sequence $\{\mathscr{D}_i\}_{i=0}^{n}$ of direct derivations such that $\mathscr{D}_i : G_i \Mapsto G_{i+1}$ is a direct derivation for every index $i$ and $G_0 = G$, $G_{n+1} = H$. Moreover, we have an* empty derivation *$G : G \Mapsto G$ for each $G \in \mathbf{X}$. We denote by $\mathsf{lg}(\underline{\mathscr{D}})$ the* length *of a derivation $\underline{\mathscr{D}}$.*

*Given a derivation $\underline{\mathscr{D}} = \{\mathscr{D}_i\}_{i=0}^{n}$, we define $r(\underline{\mathscr{D}}) = \{\rho_i\}_{i=0}^{n}$ as the sequence of rules such that every $\mathscr{D}_i$ applies rule $\rho_i \in \mathsf{R}$.*

Derivations naturally compose: given $\underline{\mathscr{D}} : G \Mapsto H$ and $\underline{\mathscr{E}} : H \Mapsto F$, we can consider their composition $\underline{\mathscr{D}} \cdot \underline{\mathscr{E}} : G \to F$, defined in the obvious way.

Often, we are interested in derivations only up to some notion of coherent isomorphism between them. This leads us to the following definition.

▶ **Definition 2.9** (Abstraction equivalence). *Let $\mathbf{X}$ be an $\mathcal{M}$-adhesive category and $(\mathbf{X}, \mathsf{R})$ a left-linear rewriting system. An* abstraction equivalence *between derivations $\underline{\mathscr{D}}$ and $\underline{\mathscr{D}'}$ with the same length and $r(\underline{\mathscr{D}}) = r(\underline{\mathscr{D}'})$ is a family of isomorphisms $\{\phi_X\}_{X \in \{G_i, D_i\}}$ such that the diagram below commutes for $i \in [0, \min(0, \mathsf{lg}(\underline{\mathscr{D}}) - 1)]$. We say that $\underline{\mathscr{D}}$ are $\underline{\mathscr{D}'}$ are* abstraction equivalent*, written $\underline{\mathscr{D}} \equiv_a \underline{\mathscr{D}'}$, if there exists an abstraction equivalence between them.*

$$\begin{tikzcd}
G'_i & D'_i \arrow[l, tail, "f'_i"'] \arrow[r, "g'_i"] & G'_{i+1} \\
L_i \arrow[u, "m'_i"] \arrow[d, "m_i"'] & K_i \arrow[l, tail, "l_i"'] \arrow[r, "r_i"] \arrow[u, "k'_i"] \arrow[d, "k_i"'] & R_i \arrow[u, "h'_i"] \arrow[d, "h_i"'] \\
G_i \arrow[uu, bend right, "\phi_{G_i}"'] & D_i \arrow[l, tail, "f_i"] \arrow[r, "g_i"'] \arrow[uu, bend right, "\phi_{D_i}"'] & G_{i+1} \arrow[uu, bend right, "\phi_{G_{i+1}}"']
\end{tikzcd}$$

$\mathcal{M}$-adhesivity ensures the uniqueness of the result of applying a rule to an object: two direct derivations using the same match are abstraction equivalent (see Proposition C.1 of the extended version).

# 3 Independence in rewriting

In this section we discuss the notion of independence between two consecutive rewriting steps, a fundamental ingredient of the rewriting theory, which comes into play when viewing a sequence of steps as a concurrent computation. We observe that moving from linear to left-linear rules leads to the failure of various basic properties of the classical notion of independence used in the linear setting and we single out a framework in which these can be re-established, possibly in a weakened form.

## 3.1 Sequentially independent and switchable derivations

Sequential independence is the canonical notion of independence in the DPO approach.

▶ **Definition 3.1** (Sequential independence)**.** *Let* $\mathbf{X}$ *be an* $\mathcal{M}$*-adhesive category and* $(\mathbf{X}, \mathsf{R})$ *a left-linear rewriting system. Let also* $\underline{\mathscr{D}} = \{\mathscr{D}_i\}_{i=0}^{1}$ *be a derivation of length* 2*, with* $\mathscr{D}_i$ *using rule* $\rho_i = (l_i, r_i)$*. We say that* $\mathscr{D}_0$ *and* $\mathscr{D}_1$ *are* sequentially independent *if there is a pair of arrows* $i_0 \colon R_0 \to D_1$ *and* $i_1 \colon L_1 \to D_0$ *such that the following diagram commutes. In this case* $(i_0, i_1)$ *is called an* independence pair*.*

$$\begin{tikzcd}
L_0 \arrow[d, "m_0"'] & K_0 \arrow[l, tail, "l_0"'] \arrow[r, "r_0"] \arrow[d, "k_0"'] & R_0 \arrow[dr, "h_0"'] \arrow[drrr, dotted, "i_0"] & & L_1 \arrow[dl, "m_1"] \arrow[dlll, dotted, "i_1"'] & K_1 \arrow[l, tail, "l_1"'] \arrow[r, "r_1"] \arrow[d, "k_1"] & R_1 \arrow[d, "h_1"] \\
G_0 & D_0 \arrow[l, tail, "f_0"] \arrow[rr, "g_0"'] & & G_1 & & D_1 \arrow[ll, tail, "f_1"] \arrow[r, "g_1"'] & G_2
\end{tikzcd}$$

Intuitively, the existence of the independence pair captures the possibility of switching the application of the two rules: $f_0 \circ i_1$ is a match for $\rho_1$ in $G_0$ and the existence of $i_0 : R_0 \to D_1$ means that $\rho_1$ does not delete items in the image of $K_0$, thus $\rho_0$ can be applied after $\rho_1$.

▶ Remark 3.2. It is worth mentioning that if $(\mathbf{X}, \mathsf{R})$ is a linear rewriting system then two derivations $\mathscr{D}_0$ and $\mathscr{D}_1$ can have at most one independence pair. Indeed if $(i_0, i_1)$ and $(i'_0, i'_1)$ are independence pairs, then

$$g_0 \circ i_1 = g_0 \circ i'_1 \qquad f_1 \circ i_0 = f_1 \circ i'_0$$

But in linear systems $g_0$ and $f_1$ are both monos, entailing $i_0 = i'_0$ and $i_1 = i'_1$.

We next formalise what it means for a derivation being the switch of another.

▶ **Definition 3.3** (Switch)**.** *Let* $\mathbf{X}$ *be an* $\mathcal{M}$*-adhesive category and* $(\mathbf{X}, \mathsf{R})$ *a left-linear rewriting system. Let also* $\underline{\mathscr{D}} = \{\mathscr{D}_i\}_{i=0}^{1}$ *be a derivation made of two sequentially independent derivations* $\mathscr{D}_0$*,* $\mathscr{D}_1$ *with independence pair* $(i_0, i_1)$*, as in the diagram on the left below. A* switch *of* $\underline{\mathscr{D}}$ *along* $(i_0, i_1)$ *is a derivation* $\underline{\mathscr{E}} = \{\mathscr{E}_i\}_{i=0}^{1}$*, between the same objects and using the same rules in reverse order, as in the diagram on the right below*

$$\begin{tikzcd}
L_0 \arrow[d, "m_0"'] & K_0 \arrow[l, tail, "l_0"'] \arrow[r, "r_0"] \arrow[d, "k_0"'] & R_0 \arrow[dr, "h_0"'] \arrow[drrr, "i_0"] & & L_1 \arrow[dl, "m_1"] \arrow[dlll, "i_1"'] & K_1 \arrow[l, tail, "l_1"'] \arrow[r, "r_1"] \arrow[d, "k_1"] & R_1 \arrow[d, "h_1"] \\
G_0 & D_0 \arrow[l, tail, "f_0"] \arrow[rr, "g_0"'] & & G_1 & & D_1 \arrow[ll, tail, "f_1"] \arrow[r, "g_1"'] & G_2
\end{tikzcd}
\qquad
\begin{tikzcd}
L_1 \arrow[d, "m'_0"'] & K_1 \arrow[l, tail, "l_1"'] \arrow[r, "r_1"] \arrow[d, "k'_0"'] & R_1 \arrow[dr, "h'_0"'] \arrow[drrr, dotted, "i'_0"] & & L_0 \arrow[dl, "m'_1"] \arrow[dlll, dotted, "i'_1"'] & K_0 \arrow[l, tail, "l_0"'] \arrow[r, "r_0"] \arrow[d, "k'_1"] & R_0 \arrow[d, "h'_1"] \\
G_0 & D'_0 \arrow[l, tail, "f'_0"] \arrow[rr, "g'_0"'] & & G'_1 & & D'_1 \arrow[ll, tail, "f'_1"] \arrow[r, "g'_1"'] & G_2
\end{tikzcd}$$

*such that there is an independence pair* $(i'_0, i'_1)$ *between* $\mathscr{E}_0$ *and* $\mathscr{E}_1$ *and*

$$m_0 = f'_0 \circ i'_1 \qquad h_1 = g'_1 \circ i'_0 \qquad m'_0 = f_0 \circ i_1 \qquad h'_1 = g_1 \circ i_0.$$

*If a switch of* $\mathscr{D}_0$ *and* $\mathscr{D}_1$ *exists we say that they are* switchable*.*

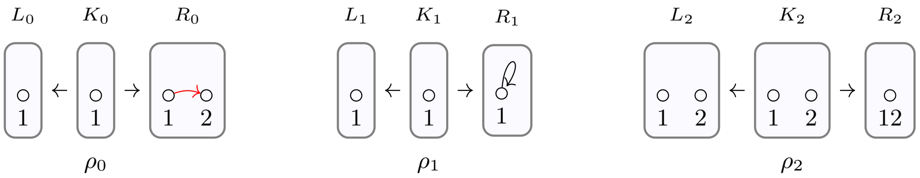


**Figure 3** A rewriting system in **Graph**.

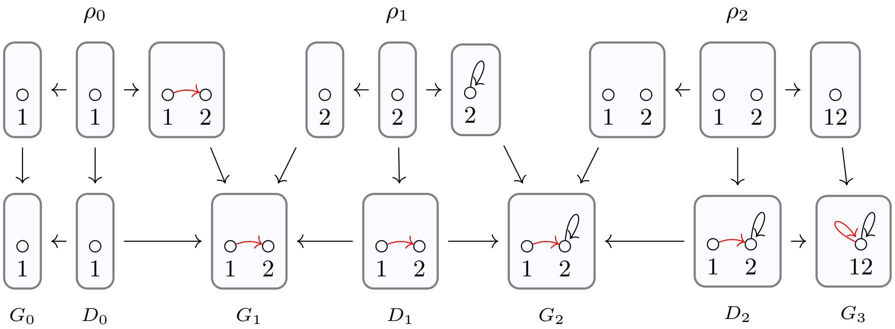


**Figure 4** The derivation $\mathscr{D}$.

It can be shown that switches along the same independence pair are unique up to abstraction equivalence (see Proposition C.3 of the extended version).

▶ **Example 3.4.** Consider a rewriting system in **Graph**, the category of directed graphs and graph morphisms, which is adhesive. The set of rules $\mathsf{R} = \{\rho_0, \rho_1, \rho_2\}$ is in Fig. 3, where numbers are used to represent the morphisms from the interface to the left- and right-hand sides. Rules $\rho_0$ and $\rho_1$ are linear: $\rho_0$ generates a new node and a new edge, while $\rho_1$ creates a self-loop edge. Instead, $\rho_2$ is left-linear but not linear, as it "merges" two nodes.

A derivation $\mathscr{D}$ consisting of three steps $\mathscr{D}_0$, $\mathscr{D}_1$, and $\mathscr{D}_2$, each applying the corresponding rule $\rho_i$, is depicted in Fig. 4. Note that the numbers in $\rho_1$ were changed consistently to represent the vertical morphisms. Steps $\mathscr{D}_1$ and $\mathscr{D}_2$ are clearly sequential independent via the independence pair $(R_1 \to D_2, L_2 \to D_1)$, mapping nodes according to their numbering. A switch of $\mathscr{D}$ along such independence pair is the derivation $\mathscr{E}$ in Fig. 5. Instead, in $\mathscr{D}$ the first two steps applying $\rho_0$ and $\rho_1$ are not sequential independent, intuitively because $\rho_1$ uses the node produced by $\rho_0$ for attaching a self-loop.

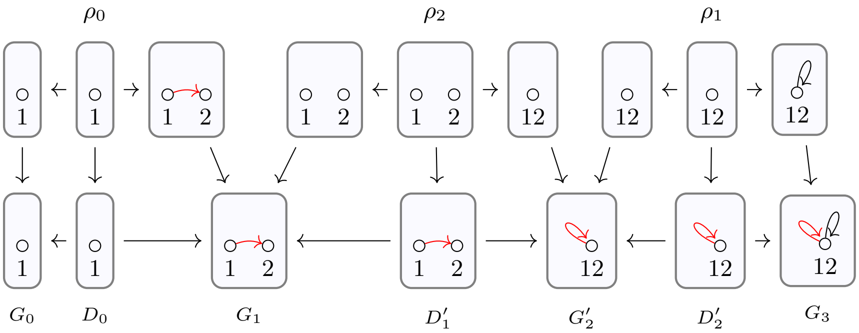
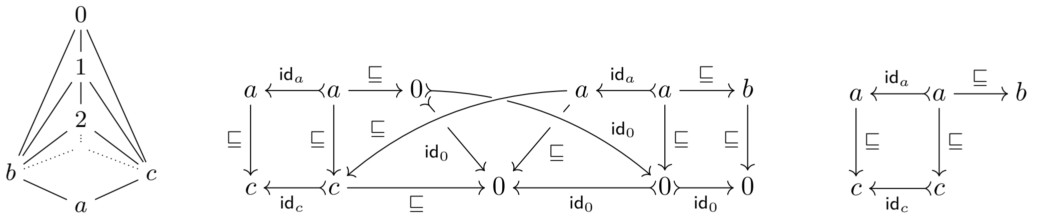


**Figure 5** The derivation $\mathscr{E}$.



**(a)** Poset $(P, \sqsubseteq)$. **(b)** Two rewriting steps. **(c)** No rewriting step.

**Figure 6** Sequential independence does not imply switchability.

As we mentioned, for linear rewriting systems it is canonical to identify derivations that are equal "up to switching", i.e. that differ only in the order of independent steps. The same notion can be given for left-linear rewriting systems by relying on the notion of switch.

We recall some notations on permutations. A *permutation* on $[0, n]$ is a bijection $\sigma : [0, n] \to [0, n]$. It is a *transposition* $\nu$ if there are $i, j \in [1, n]$, $i \neq j$ such that $\sigma(i) = j$, $\sigma(j) = i$, and $\sigma(k) = k$ otherwise; it is denoted as $(i, j)$. Given a permutation $\sigma$, an *inversion* for $\sigma$ is a pair $(i, j)$ such that $i < j$ and $\sigma(j) < \sigma(i)$; $\mathsf{inv}(\sigma)$ denotes the set of inversions of $\sigma$.

Switch equivalence is now defined as the equivalence relating derivations that can be obtained one from another by a sequence of switches. Moreover, intermediate graphs can be taken up to isomorphism according to abstraction equivalence.

▶ **Definition 3.5** (Switch equivalence). *Let $(\mathbf{X}, \mathsf{R})$ be a left-linear rewriting system. Let also $\underline{\mathscr{D}}, \underline{\mathscr{E}} : G \Mapsto H$ be derivations with the same length, $\underline{\mathscr{D}} = \{\mathscr{D}_i\}_{i=0}^{n}$ and $\underline{\mathscr{E}} = \{\mathscr{E}_i\}_{i=0}^{n}$. If $\mathscr{D}_i \cdot \mathscr{D}_{i+1}$ is a switch of $\mathscr{E}_i \cdot \mathscr{E}_{i+1}$ for some $i \in [0, n-1]$ and $\mathscr{D}_j = \mathscr{E}_j$ for each $j \notin \{i, i+1\}$ then we write $\underline{\mathscr{D}} \leftrightsquigarrow_{(i,i+1)} \underline{\mathscr{E}}$. A* switching sequence *is a sequence $\{\underline{\mathscr{D}}_k\}_{k=0}^{m}$ of derivations such that $\underline{\mathscr{D}}_0 \leftrightsquigarrow_{\nu_1} \underline{\mathscr{D}}_1 \leftrightsquigarrow_{\nu_2} \dots \leftrightsquigarrow_{\nu_m} \underline{\mathscr{D}}_m$ with $\nu_k = (i_k, i_k + 1)$.*

*Let us denote by $\nu_{k,h}$ the composition $\nu_h \circ \nu_{h-1} \circ \dots \nu_k$. We say that the switching sequence* consists of inversions *if for all $k \in [0, m]$ the transposition $\nu_k$ is an inversion for $\nu_{1,m}$.*

*Two derivations $\underline{\mathscr{D}}, \underline{\mathscr{E}} : G \Mapsto H$ are* switch equivalent, *written $\underline{\mathscr{D}} \equiv^{sh} \underline{\mathscr{E}}$, if there is a switching sequence $\{\underline{\mathscr{D}}_k\}_{k=0}^{m}$ such that $\underline{\mathscr{D}} \equiv_a \underline{\mathscr{D}}_0 \leftrightsquigarrow_{\nu_1} \underline{\mathscr{D}}_1 \leftrightsquigarrow_{\nu_2} \dots \leftrightsquigarrow_{\nu_m} \underline{\mathscr{D}}_m \equiv_a \underline{\mathscr{E}}$. To point out a chosen permutation of rewriting steps, we will also write $\underline{\mathscr{D}} \equiv^{sh}_{\sigma} \underline{\mathscr{E}}$, where $\sigma$ is the composition of the transposition appearing in a chosen switching sequence.*

▶ Remark 3.6. The possibility of an empty switching sequence assures that two abstraction equivalent derivations are switch equivalent.

## 3.2 Church-Rosser: Sequential independence and switchability

The fact that sequential independence implies switchability always holds for linear rules (see Proposition C.8 of the extended version). The result is so indispensable that typically, in the literature, it is not even stated, in the sense that switchability is not introduced axiomatically as in Definition 3.3, but it is based on the explicit construction of a switch.

For left-linear rewriting systems sequential independence does not imply switchability (while the converse implication clearly holds), as shown by the next example.

▶ **Example 3.7.** Consider the poset $(P, \sqsubseteq)$ in Fig. 6a where $P = \mathbb{N} \cup \{a, b, c\}$ and $\sqsubseteq$ is defined by $m \sqsubseteq n$ if $m \geq n$, $a \sqsubseteq x$ for all $x \in P$ and $b, c \sqsubseteq n$ for all $n \in \mathbb{N}$. Let $\mathbf{X}$ be the category associated with this order, which by Example 2.4 is $\mathsf{I}(\mathbf{X})$-adhesive. Consider a rewriting system whose set of rules contains the following

$$\begin{tikzcd} a & a \arrow[l, tail, "\mathsf{id}_a"'] \arrow[r, "\sqsubseteq"] & 0 \end{tikzcd} \qquad \begin{tikzcd} a & a \arrow[l, tail, "\mathsf{id}_a"'] \arrow[r, "\sqsubseteq"] & b \end{tikzcd}$$

We can then consider the derivation $\underline{\mathscr{D}} = \{\mathscr{D}_i\}_{i=0}^{1}$ in Fig. 6b. Note that the two direct derivations are sequential independent. However there is no switch since the rule applied by $\mathscr{D}_1$ cannot be applied to $c$. In fact, there is a unique morphism $a \to c$, yielding the diagram in Fig. 6c. But $b$ and $c$ do not have a supremum in the poset underlying $\mathbf{X}$, thus the arrows $a \to b$ and $a \to c$ do not have a pushout. Hence we do not get a direct derivation from $c$.

The conditions guaranteeing switchability are inspired by the notion of *canonical filler* [30].

▶ **Definition 3.8** (strong independence pair). *Let* $\mathbf{X}$ *be an* $\mathcal{M}$*-adhesive category and* $(\mathbf{X}, \mathsf{R})$ *a left-linear rewriting system. Let also* $(i_0, i_1)$ *be an independence pair between two direct derivations* $\mathscr{D}_0$*,* $\mathscr{D}_1$ *as in the solid part of the diagram below*

$$\begin{tikzcd}
L_0 \arrow[d, "m_0"'] & K_0 \arrow[l, tail, "l_0"'] \arrow[r, "r_0"] \arrow[d, "k_0"] \arrow[ddrr, dotted, "u_0"] & R_0 \arrow[dr, "h_0"'] \arrow[drrr, bend left=15, "i_0"] & & L_1 \arrow[dl, "m_1"] \arrow[dlll, bend right=15, "i_1"'] & K_1 \arrow[l, tail, "l_1"'] \arrow[r, "r_1"] \arrow[d, "k_1"] \arrow[ddll, dotted, "u_1"'] & R_1 \arrow[d, "h_1"] \\
G_0 & D_0 \arrow[l, tail, "f_0"] \arrow[rr, "g_0"'] & & G_1 & & D_1 \arrow[ll, tail, "f_1"] \arrow[r, "g_1"'] & G_2 \\
& & & P \arrow[ull, dotted, tail, bend left, "p_0"] \arrow[urr, dotted, tail, bend right, "p_1"'] & & &
\end{tikzcd}$$

*Consider the pullback of* $g_0$ *and* $f_1$*, which yields* $p_i : P \to D_i$ *for* $i \in \{0, 1\}$ *and the mediating arrows* $u_i \colon K_i \to P$ *for* $i \in \{0, 1\}$ *into the pullback object (see Proposition C.4 of the extended version for details). We say that* $(i_0, i_1)$ *is a* strong independence pair *if the first two squares depicted below are pushouts and if the pushout of* $r_1 : K_1 \to R_1$ *and* $u_1 : K_1 \to P$ *exists*

$$\begin{tikzcd}
K_0 \arrow[r, "r_0"] \arrow[d, "u_0"'] & R_0 \arrow[d, "i_0"] \\
P \arrow[r, "p_1"'] & D_1
\end{tikzcd}
\qquad
\begin{tikzcd}
K_1 \arrow[r, tail, "l_1"] \arrow[d, "u_1"'] & L_1 \arrow[d, "i_1"] \\
P \arrow[r, "p_0"'] & D_0
\end{tikzcd}
\qquad
\begin{tikzcd}
K_1 \arrow[r, "r_1"] \arrow[d, "u_1"'] & R_1 \arrow[d, dotted, "j_0"] \\
P \arrow[r, dotted, "q_1"'] & Q_1
\end{tikzcd}$$

We can now prove a Local Church-Rosser Theorem for strong independence pairs.

▶ **Proposition 3.9** (Local Church-Rosser Theorem). *Let* $(i_0, i_1)$ *be a strong independence pair between* $\mathscr{D}_0$ *and* $\mathscr{D}_1$*. Then* $\mathscr{D}_0$ *and* $\mathscr{D}_1$ *are switchable.*

The correspondence between sequential independence and switchability is fundamental. We name the class of rewriting systems where this property holds.

▶ **Definition 3.10** (Strong enforcing rewriting systems). *A left-linear rewriting system is* strong enforcing *if every independence pair between two direct derivations is strong.*

We can identify a large class of adhesive categories such that all left-linear rewriting systems over such categories are strong enforcing. This class includes **Set** and it is closed under comma and functor category constructions. As such, it includes essentially all categories (e.g., presheaves over set) that are typically considered for modelling purposes. Notably, it contains the category **Graph** of directed graphs. This is a natural generalisation from adhesive to $\mathcal{M}$-adhesive categories of a class studied in [6] (see Appendix B for details).

Still, there are $\mathcal{M}$-adhesive rewriting systems that are not strong enforcing.

▶ **Example 3.11** (Non-strong enforcing left-linear rewriting system). In light of Proposition 3.9, Example 3.7 provides an example of an independence pair that is not strong. This gives an example of a left-linear rewriting system in a $\mathcal{M}$-adhesive category that is quite pathological since $\mathcal{M} = \mathsf{I}(\mathbf{X})$. However, this is expected, since all natural examples seem to belong to the well-behaved class mentioned above.

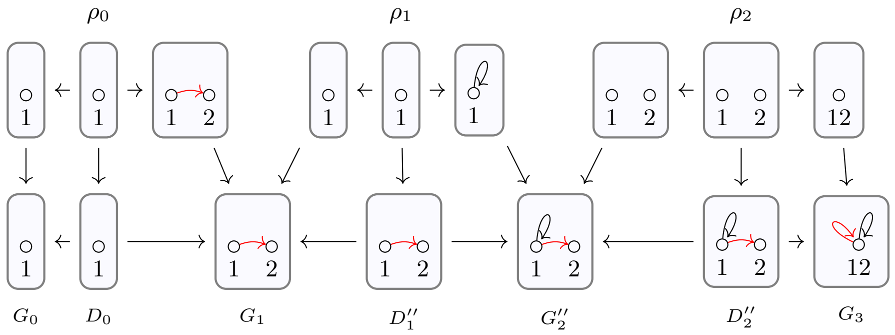


**Figure 7** The derivation $\underline{\mathscr{D}}'$.

▶ Remark 3.12. By Proposition 3.9 the existence of a strong independence pair entails switchability, which in turn entails sequential independence by construction. Strong enforcing rewriting systems are exactly those rewriting systems in which these three notions coincide.

Consistency with the theory of linear rewriting systems is ensured by the fact that all linear rewriting systems are strong enforcing (see Proposition C.8 of the extended version).

## 3.3 Well-switching rewriting systems

Even if we work in strong enforcing rewriting systems where sequential independence ensures switchability, when dealing with left-linear rules there is a further, possibly more serious issue, namely that there can be more than one independence pair between the same derivations (cfr. Remark 3.2). This hinders the very idea of using sequential independence as a basis of a theory of concurrency for rewriting systems, since exchanges performed using different independence pairs may lead to derivations that are not abstraction equivalent, thus equating computations that should definitively be taken apart, as shown in the example below.

▶ **Example 3.13.** Consider the derivation $\underline{\mathscr{E}}$ from Example 3.4 (see Fig. 5). The last two steps are sequential independent, but one easily sees that there are two distinct independence pairs, as the left-hand side of $\rho_1$ can be mapped either to node 1 or to node 2 in $D'_1$. Correspondingly, there are two switches of $\underline{\mathscr{E}}$: one is the derivation $\underline{\mathscr{D}}$ in Fig. 4 we started from, the other is the derivation $\underline{\mathscr{D}}'$ in Fig. 7.

As a consequence, $\underline{\mathscr{D}}$ and $\underline{\mathscr{D}}'$ would be switch equivalent, but this is not acceptable when viewing equivalence classes of derivations as concurrent computations: in $\underline{\mathscr{D}}$ the first two steps are not sequential independent, while in $\underline{\mathscr{D}}'$ they are, intuitively because in $\underline{\mathscr{D}}$ rule $\rho_1$ uses the node generated by $\rho_0$ (adding a self-loop to it), while in $\underline{\mathscr{D}}'$ rule $\rho_1$ uses the node that was in the initial graph. Also observe that the graphs $G_2$ and $G'_2$ produced after two steps in $\underline{\mathscr{D}}$ and $\underline{\mathscr{D}}'$ are not isomorphic. From the technical point of view, the property of being switch equivalent is not closed by prefix, and this prevents deriving a sensible concurrent semantics: In fact $\underline{\mathscr{D}} = \mathscr{D}_0 \cdot \mathscr{D}_1 \cdot \mathscr{D}_2$ and $\underline{\mathscr{D}}' = \mathscr{D}'_0 \cdot \mathscr{D}'_1 \cdot \mathscr{D}'_2$ are switch equivalent, while if we consider the first two steps, derivations $\mathscr{D}_0 \cdot \mathscr{D}_1$ and $\mathscr{D}'_0 \cdot \mathscr{D}'_1$ are not switch equivalent.

For these reasons we believe a theory of rewriting for left-linear rules in adhesive categories should be developed for systems where the uniqueness of the independence pair is ensured.

▶ **Definition 3.14** (Well-switching rewriting systems). *A left-linear rewriting system* $(\mathbf{X}, \mathsf{R})$ *is* well-switching *if it is strong enforcing and, for every derivation* $\underline{\mathscr{D}} := \{\mathscr{D}_i\}_{i=0}^{1}$*, there is at most one independence pair between* $\mathscr{D}_0$ *and* $\mathscr{D}_1$*.*

Clearly, linear rewriting systems are well-switching (see Proposition C.10 of the extended version). Moreover, we next observe that various classes of rewriting systems, comprising all the ones used in modelling the applications to the encoding of process calculi or of bio and chemical systems mentioned in the introduction, are actually well-switching.

A first class consists of those rewriting systems over possibly hierarchical graphical structures obtained as algebras of suitable signatures where rules are constrained not to merge elements of top level sorts in the hierarchy (for graphs, nodes can be merged while edges cannot). The idea here is to consider graph structures as presheaves on categories in which there are objects that play the role of *roots*, i.e. objects that are not the codomain of any arrow besides the identity.

▶ **Definition 3.15** (Root-preserving graphical rewriting systems). *Let* $\mathbf{X}$ *be a category, an object* $X \in \mathbf{X}$ *is a* root *if the only arrow with codomain* $X$ *is* $\mathsf{id}_X$*. The category* $\mathbf{X}$**-Graph** *of* $\mathbf{X}$-graphs *is the category* $\mathbf{Set}^{\mathbf{X}}$*. A* root-preserving graphical rewriting system *is a left-linear rewriting system* $(\mathbf{X\text{-}Graph}, \mathsf{R})$ *such that for each rule* $(l\colon K \to L, r\colon K \to R)$ *in* $\mathsf{R}$ *it holds*

1. *for every* $X \in \mathbf{X}$ *and* $x \in L(X)$*, there exists a root* $Y$ *and an arrow* $f\colon Y \to X$ *such that* $x$ *belongs to the image of* $L(f)\colon L(Y) \to L(X)$*;*
2. $r\colon K \to R$ *is mono on the roots, i.e. for every root* $X \in \mathbf{X}$ *the component* $r_X : K(X) \to R(X)$ *is injective.*

For instance, the category **Graph** can be obtained by taking as $\mathbf{X}$ the category generated by $E \rightrightarrows V$. In this case $E$ is the only root, hence, condition 1 asks that in the left-hand side of each rule there are no isolated nodes, while condition 2 asks that the morphism $r : K \to R$ is injective on edges, i.e. it can only merge nodes.

▶ **Lemma 3.16.** *All root-preserving graphical rewriting systems are well-switching.*

Another interesting class of well-switching rewriting systems is given by e-graphs.

▶ **Example 3.17** (E-graphs). Consider the category **EGraphs**, where objects are (directed) graphs endowed with an equivalence over nodes, and arrows are graph morphisms that preserve the equivalence, as considered in [5], closely related to e-graphs [42]. Formally, **EGraphs** can be seen as the subcategory of the presheaf $[E \rightrightarrows V \to Q, \mathbf{Set}]$ where objects are constrained to have the component $V \to Q$ surjective.

Explicitly, an e-graph $G$ is a triple $\langle s_G, t_G, q_G \rangle$ where $s_G, t_G : E_G \rightrightarrows V_G$ provides the graphical structure, while the surjective function $q_G : V_G \to Q_G$ maps each node to the corresponding equivalence class. Notice that the inclusion functor into $[E \rightrightarrows V \to Q, \mathbf{Set}]$ creates pullbacks and pushouts [36], so that they are computed component-wise.

A morphism in **EGraph** is mono if the components over $E$ and $V$ are mono, i.e. if it is mono as a morphism in **Graph**. It is regular mono if also the component on $Q$ is mono, i.e. if it reflects equivalence classes besides preserving them. This characterisation of regular monos and the fact that pullbacks and pushouts are computed component-wise allows us to prove quasi-adhesivity of **EGraphs** at once. Moreover, one can deduce that every rewriting system $(\mathbf{X}, \mathsf{R})$ that is left-linear with respect to $\mathsf{Reg}(\mathbf{EGraphs})$ is strong enforcing: this is done exploiting again the inclusion functor into $[E \rightrightarrows V \to Q, \mathbf{Set}]$.

Left-linear rewriting systems with respect to $\mathsf{Reg}(\mathbf{EGraphs})$ are well-switching. They have been used in [5] for the graphical implementation of nominal calculi, where, differently from [27], as a result of name passing the received name is not merged with a local one, but put in the same equivalence class, better tracing the causal dependencies among reductions.

## 3.4 A canonical form for switch equivalences

As discussed in the introduction, in the linear case a number of basic properties of switch equivalence, i.e. the globality of independence, the consistency of switching, and the existence of a canonical form for equivalence proofs, can be derived from a characterisation of switch equivalence in terms of consistent permutations or processes.

When dealing with left-linear rules, such a characterisation fails. Leaving aside the formal details (the interested reader can refer to Definition C.12 of the extended version), the intuition is quite simple. In the linear case one proves that two derivations using the same rules in different orders are switch equivalent when the colimits of the two derivations, seen as diagrams, are isomorphic and the isomorphism properly commutes with the embeddings of the rules. When considering left-linear derivations, this is no longer true. Intuitively this is due to the fact that a rewriting step can merge parts of an object thus making the colimit little informative. This effect can be seen for derivations $\underline{\mathscr{D}}$ in Fig. 4 and $\underline{\mathscr{D}}'$ in Fig. 7. They are not abstraction equivalent but the colimit of both derivations is a single node with two self-loops, in a way that all commutativity requirements are trivially satisfied.

In this section we show that, despite the failure of such characterisation, actually much of the theory can be retained, taking a different and sensibly more complex route for the proofs: globality of independence holds in a weak form, conceptually linked to the fact that fusions introduce a form of disjunctive causality, while consistency of switching holds unchanged. Finally, a canonical form for switching sequences can also be recovered.

We start by proving two lemmas dealing with derivations of length 3. Despite being technical, these lemmas are fundamental building blocks for obtaining our results.

The first lemma is at the core of globality for independence. It shows that if we have a derivation consisting of three rewriting steps, say $\mathscr{D}_0 \cdot \mathscr{D}_1 \cdot \mathscr{D}_2$ where $\mathscr{D}_0$ and $\mathscr{D}_1$ are independent, then if we switch $\mathscr{D}_2$ to the first position, obtaining a derivation $\mathscr{D}_2'' \cdot \mathscr{D}_0' \cdot \mathscr{D}_1'$, as in the diagram below where vertical lines represent permutations, the steps $\mathscr{D}_0'$ and $\mathscr{D}_1'$ are still independent.

$$\begin{tikzcd}
\mathscr{D}_0 \arrow[d, dash] & \mathscr{D}_1 \arrow[dr, dash] & \mathscr{D}_2 \arrow[dl, dash] \\
\mathscr{D}_0 \arrow[dr, dash] & \mathscr{D}_2' \arrow[dl, dash] & \mathscr{D}_1' \arrow[d, dash] \\
\mathscr{D}_2'' & \mathscr{D}_0' & \mathscr{D}_1'
\end{tikzcd}$$

If instead $\mathscr{D}_1$ and $\mathscr{D}_2$ are independent in $\mathscr{D}_0 \cdot \mathscr{D}_1 \cdot \mathscr{D}_2$ and we switch $\mathscr{D}_0$ to the last position, obtaining a derivation $\mathscr{D}_1' \cdot \mathscr{D}_2' \cdot \mathscr{D}_0''$, as in the diagram below, the steps $\mathscr{D}_1'$ and $\mathscr{D}_2'$ could be no longer independent. Intuitively, this is due to the fact that dealing with left-linear rewriting systems introduces disjunctive forms of causality. Hence, if in $\mathscr{D}_0 \cdot \mathscr{D}_1 \cdot \mathscr{D}_2$ we have that $\mathscr{D}_0$ and $\mathscr{D}_1$ are disjunctive causes of $\mathscr{D}_2$, at least one of the two is needed to enable the application of $\mathscr{D}_2$ and this explain why in $\mathscr{D}_1' \cdot \mathscr{D}_2' \cdot \mathscr{D}_0''$ we have that $\mathscr{D}_1'$ and $\mathscr{D}_2'$ are no longer independent. This is exemplified below.

$$\begin{tikzcd}
\mathscr{D}_0 \arrow[dr, dash] & \mathscr{D}_1 \arrow[dl, dash] & \mathscr{D}_2 \arrow[d, dash] \\
\mathscr{D}_1' \arrow[d, dash] & \mathscr{D}_0' \arrow[dr, dash] & \mathscr{D}_2 \arrow[dl, dash] \\
\mathscr{D}_1' & \mathscr{D}_2' & \mathscr{D}_0''
\end{tikzcd}$$

▶ **Example 3.18.** Consider the rewriting system in **Graph** consisting of the rules $\lambda_0$, $\lambda_1$, and $\lambda_2$ in Fig. 8. Consider the derivation $\underline{\mathscr{F}}$ in Fig. 9. Note that rule $\lambda_2$ needs the black self-loop on node 12 to be applied. This can be generated either by $\lambda_0$ or by $\lambda_1$ merging nodes 1 and 2,

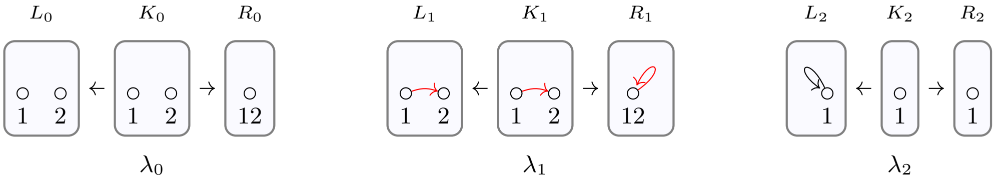


**Figure 8** A new rewriting system in **Graph**.

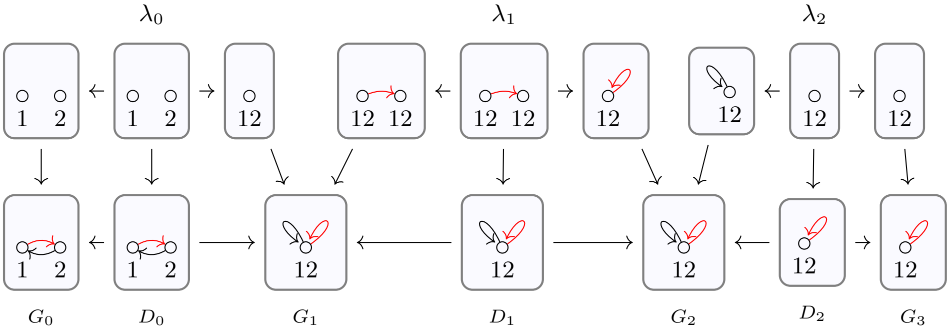


**Figure 9** The derivation $\underline{\mathscr{F}}$.

hence at least one of them must precede the application of $\lambda_2$. Indeed, in $\underline{\mathscr{F}}$ it is easily seen that the second and the third step applying $\lambda_1$ and $\lambda_2$ are independent. However, we could switch twice the application of $\lambda_0$, bringing it in the last position, obtaining a derivation $\underline{\mathscr{F}}'$ as depicted in Fig. 10, where the applications of $\lambda_1$ and $\lambda_2$ are no longer independent.

▶ **Lemma 3.19** (Globality of independence). *Let* $\mathbf{X}$ *be an* $\mathcal{M}$*-adhesive category and* $(\mathbf{X}, \mathsf{R})$ *a well-switching rewriting system. Let* $\mathscr{D}_0 \cdot \mathscr{D}_1 \cdot \mathscr{D}_2$ *be a three-steps derivation.*

1. *If* $\mathscr{D}_0$ *and* $\mathscr{D}_1$ *are switchable and there is a switching sequence* $\mathscr{D}_0 \cdot \mathscr{D}_1 \cdot \mathscr{D}_2 \leftrightsquigarrow_{(1,2)} \mathscr{D}_0 \cdot \mathscr{D}_2' \cdot \mathscr{D}_1' \leftrightsquigarrow_{(0,1)} \mathscr{D}_2'' \cdot \mathscr{D}_0' \cdot \mathscr{D}_1'$ *then in the last derivation* $\mathscr{D}_0'$ *and* $\mathscr{D}_1'$ *are switchable;*
2. *Suppose that* $\mathscr{D}_1$ *and* $\mathscr{D}_2$ *are switchable, hence* $\mathscr{D}_0 \cdot \mathscr{D}_1 \cdot \mathscr{D}_2 \leftrightsquigarrow_{(1,2)} \mathscr{D}_0 \cdot \mathscr{D}_2' \cdot \mathscr{D}_1'$ *and* $\mathscr{D}_0$ *and* $\mathscr{D}_2'$ *are switchable. If there is a switching sequence* $\mathscr{D}_0 \cdot \mathscr{D}_1 \cdot \mathscr{D}_2 \leftrightsquigarrow_{(0,1)} \mathscr{D}_1' \cdot \mathscr{D}_0' \cdot \mathscr{D}_2 \leftrightsquigarrow_{(1,2)} \mathscr{D}_1' \cdot \mathscr{D}_2' \cdot \mathscr{D}_0''$ *then in the last derivation* $\mathscr{D}_1'$ *and* $\mathscr{D}_2'$ *are switchable.*

The second lemma captures the essence of the consistency of switchings. If in a derivation of three rewriting steps, say $\mathscr{D}_0 \cdot \mathscr{D}_1 \cdot \mathscr{D}_2$, we invert them leading to a sequence reordered as $\mathscr{D}_2 \cdot \mathscr{D}_1 \cdot \mathscr{D}_0$, we obtain the same derivation, independently from the order of switchings.

▶ **Lemma 3.20** (Consistency of switchings). *Let* $\mathbf{X}$ *be an* $\mathcal{M}$*-adhesive category and* $(\mathbf{X}, \mathsf{R})$ *a well-switching rewriting system. Consider a derivation* $\underline{\mathscr{D}} = \{\mathscr{D}_i\}_{i=0}^{2}$ *and suppose that we have the following two switching sequences*

$$\underline{\mathscr{D}} \leftrightsquigarrow_{(0,1)} \underline{\mathscr{D}}' \leftrightsquigarrow_{(1,2)} \underline{\mathscr{D}}'' \leftrightsquigarrow_{(0,1)} \underline{\mathscr{D}}''' \qquad \underline{\mathscr{D}} \leftrightsquigarrow_{(1,2)} \underline{\mathscr{E}}' \leftrightsquigarrow_{(0,1)} \underline{\mathscr{E}}'' \leftrightsquigarrow_{(1,2)} \underline{\mathscr{E}}'''$$

*Then* $\underline{\mathscr{D}}'''$ *and* $\underline{\mathscr{E}}'''$ *are abstraction equivalent.*

The lemmas above open the way to the key result of this section, which establishes a canonical form for switching derivations.

The first result shows that when equating two derivation sequences by performing a series of switchings, we can limit ourselves to switchings that invert the order of steps that in the target derivation has to be reversed, i.e. we can limit ourselves to applying inversions.

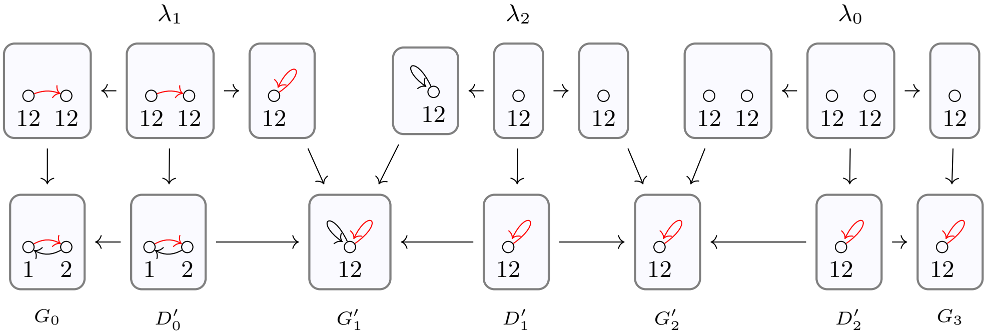


**Figure 10** The derivation $\mathscr{F}'$.

▶ **Theorem 3.21** (No need of useless switches). *Let $\mathscr{D}$ and $\mathscr{D}'$ be derivation sequences. If $\mathscr{D} \equiv^{sh} \mathscr{D}'$ then there is a switching sequence $\mathscr{D} \equiv_a \mathscr{D}_0 \leftrightsquigarrow_{\nu_1} \mathscr{D}_1 \leftrightsquigarrow_{\nu_2} \dots \leftrightsquigarrow_{\nu_n} \mathscr{D}_n \equiv_a \mathscr{D}'$ consisting only of inversions.*

While inversions can be performed in any order in the case of linear rewriting systems, this is no longer true for left-linear rewriting systems due to the fact that, as already observed, globality of independence holds in a weak form. For instance, if we get back to Example 3.18 and consider derivation $\mathscr{F}$ in Fig. 9, the last two steps applying rules $\lambda_1$ and $\lambda_2$ can be switched, thus leading to a derivation, call it $\mathscr{F}''$, applying $\lambda_0, \lambda_2, \lambda_1$. This $\mathscr{F}''$ is switch equivalent to $\mathscr{F}'$ in Fig. 10, hence starting from $\mathscr{F}''$ we can obtain $\mathscr{F}'$ via a switching sequence, but we cannot start by switching $\lambda_0$ and $\lambda_2$, even if this is an inversion.

Still, we can identify a canonical form for switching derivations: at each step one can apply a switch to the inversion with largest index.

▶ **Corollary 3.22** (Canonical form). *Let $\mathscr{D}$ and $\mathscr{D}'$ be derivation sequences. If $\mathscr{D} \equiv^{sh}_{\sigma} \mathscr{D}'$ then there is a switching sequence*

$$\mathscr{D} \equiv_a \mathscr{D}_0 \leftrightsquigarrow_{\nu_1} \mathscr{D}_1 \leftrightsquigarrow_{\nu_2} \dots \leftrightsquigarrow_{\nu_n} \mathscr{D}_n \equiv_a \mathscr{D}'$$

*where for $i \in [1, n]$ we have $\nu_i = (k, k+1)$ with $k = \max\{j \mid (j, j+1) \in \mathsf{inv}(\nu_{i,n})\}$.*

## 4 Conclusions and further work

We performed an in-depth investigation of the notion of independence between rewriting steps in the setting of left-linear rewriting systems over $\mathcal{M}$-adhesive categories, which encompasses most of the structures commonly used in analysing and modelling applications.

We showed that the canonical notion of independence adopted for linear systems does not enjoy some properties that are essential for developing a sensible theory of concurrency, notably a Church-Rosser theorem, and we identified a subclass of left-linear rewriting systems, which we call well-switching, as an appropriate setting where many key results can be re-established. Specifically, a Church-Rosser theorem, i.e. the possibility of performing independent steps in any order, is fully recovered. Moreover, the switch construction and correspondingly the switch equivalence, at the core of a theory of concurrency for rewriting systems, enjoy a number of fundamental properties: a weak form of globality of independence, consistency of switching and the existence of a canonical form for switch equivalence proofs.

The class of well-switching $\mathcal{M}$-adhesive rewriting systems is large. Generalising a result in [6], we showed that one of its defining properties (the fact that sequential independence implies switchability) holds for a general class of $\mathcal{M}$-adhesive categories that include all presheaves over **Set**. Moreover, the second property, uniqueness of switching, is ensured when rewriting possibly hierarchical graphical structures, as long as elements belonging to the sorts of the roots are not merged (e.g., one can consider rewriting systems over directed graphs where only nodes are merged, as it happens in most of the approaches to the modelling of calculi and biological systems that we cited in the introduction, as well as in string diagrams).

In the DPO approach to rewriting, the notion of sequential independence, identifying consecutive rewriting steps that can be performed in reverse order is typically closely linked to that of *parallel independence* [21]. Parallel independence relates two rewriting steps starting from the same object, which can be applied in any order producing, up to isomorphism, the same resulting object. Parallel independent steps, when applied in sequence, lead to sequential independent steps and, conversely, from two sequential independent steps, as a byproduct of the switch construction, one can get two parallel independent steps.

Under mild additional conditions, one can have a notion of *parallel rule* in a way that two parallel independent steps can be also combined in a single rewriting step. Even if this is not discussed explicitly in the paper, our theory can be easily accommodated to encompass a notion of parallel independence and parallel rule application, enjoying the properties outlined above. To ensure this, however, one must require that the underlying category $\mathbf{X}$ has binary coproducts and that the class $\mathcal{M}$ is closed under them.

The results in this paper open the way to the development of a concurrent semantics for left-linear rewriting systems over $\mathcal{M}$-adhesive categories. In [3] the authors argue that for computational systems that allow one to express merging or fusions, the right semantical models are weak prime domains, a generalisation of prime domains, a staple in concurrency theory, and connected event structures, their event-based counterpart. The mentioned paper discusses only the case of graph rewriting and does not focus into the problems induced by the occurring of "merges" to the theory of rewriting (the systems considered there are implicitly assumed to be well-switching). We plan to consolidate the statement that weak prime domains are "the" model for systems with fusions by showing that they allow to provide a semantics for left-linear rewriting systems in $\mathcal{M}$-adhesive categories.

On the practical side, the present paper just surveyed the possibility of modelling e-graphs. This appears to be quite interesting as concurrent rewriting on such structures is an active area of research [34]. We plan to make the correspondence precise and investigate how the analysis techniques enabled by a concurrent semantics of rewriting can impact on e-graphs.

## References

1 G. G. Azzi, A. Corradini, and L. Ribeiro. On the essence and initiality of conflicts in $\mathcal{M}$-adhesive transformation systems. *Journal of Logical and Algebraic Methods in Programming*, 109:100482, 2019.

2 P. Baldan, A. Corradini, H. Ehrig, M. Löwe, U. Montanari, and F. Rossi. Concurrent semantics of algebraic graph transformation systems. In G. Rozenberg, editor, *Handbook of Graph Grammars and Computing by Graph Transformation. Volume 3: Concurrency*, pages 107–187. World Scientific, 1999.

3 Paolo Baldan, Andrea Corradini, and Fabio Gadducci. Domains and event structures for fusions. In *LICS 2017*, pages 1–12. IEEE Computer Society, 2017.

4 Paolo Baldan, Andrea Corradini, Tobias Heindel, Barbara König, and Pawel Sobocinski. Processes for adhesive rewriting systems. In Luca Aceto and Anna Ingólfsdóttir, editors, *FOSSACS 2006*, volume 3921 of *LNCS*, pages 202–216. Springer, 2006.

5 Paolo Baldan, Fabio Gadducci, and Ugo Montanari. Concurrent rewriting for graphs with equivalences. In Christel Baier and Holger Hermanns, editors, *CONCUR 2006*, volume 4137 of *LNCS*, pages 279–294. Springer, 2006.

6 Paolo Baldan, Fabio Gadducci, and Pawel Sobocinski. Adhesivity is not enough: Local Church-Rosser revisited. In Filip Murlak and Piotr Sankowski, editors, *MFCS 2011*, volume 6907 of *LNCS*, pages 48–59. Springer, 2011.

7 Nicolas Behr, Russ Harmer, and Jean Krivine. Concurrency theorems for non-linear rewriting theories. In Fabio Gadducci and Timo Kehrer, editors, *ICGT 2021*, volume 12741 of *LNCS*, pages 3–21. Springer, 2021.

8 Nicolas Behr, Russ Harmer, and Jean Krivine. Fundamentals of compositional rewriting theory. *Journal of Logical and Algebraic Methods in Programming*, 135:100893, 2023.

9 F. Bonchi, F. Gadducci, A. Kissinger, P. Sobociński, and F. Zanasi. String diagram rewrite theory I: rewriting with Frobenius structure. *Journal of the Association for Computing Machinery*, 69(2):1–58, 2022.

10 R. Brown and G. Janelidze. Van Kampen theorems for categories of covering morphisms in lextensive categories. *Journal of Pure and Applied Algebra*, 119(3):255–263, 1997.

11 Aurelio Carboni, Stephen Lack, and Robert FC Walters. Introduction to extensive and distributive categories. *Journal of Pure and Applied Algebra*, 84(2):145–158, 1993.

12 D. Castelnovo. *Fuzzy algebraic theories and $\mathcal{M},\mathcal{N}$-adhesive categories.* PhD thesis, University of Udine, 2023.

13 D. Castelnovo, F. Gadducci, and M. Miculan. A new criterion for $\mathcal{M},\mathcal{N}$-adhesivity, with an application to hierarchical graphs. In P. Bouyer and L. Schröder, editors, *FOSSACS 2022*, volume 13242 of *LNCS*, pages 205–224. Springer, 2022.

14 Davide Castelnovo, Fabio Gadducci, and Marino Miculan. A simple criterion for $\mathcal{M},\mathcal{N}$-adhesivity. *Theoretical Computer Science*, 982:114280, 2024.

15 A. Corradini, U. Montanari, and F. Rossi. Graph processes. *Fundamenta Informaticae*, 26:241–265, 1996.

16 A. Corradini, U. Montanari, F. Rossi, H. Ehrig, R. Heckel, and M. Löwe. Algebraic approaches to graph transformation - Part I: Basic concepts and double pushout approach. In G. Rozenberg, editor, *Handbook of Graph Grammars and Computing by Graph Transformations. Volume 1: Foundations*, pages 163–246. World Scientific, 1997.

17 Andrea Corradini, Tobias Heindel, Frank Hermann, and Barbara König. Sesqui-pushout rewriting. In Andrea Corradini, Hartmut Ehrig, Ugo Montanari, Leila Ribeiro, and Grzegorz Rozenberg, editors, *ICGT 2006*, volume 4178 of *LNCS*, pages 30–45. Springer, 2006.

18 S. Crafa, D. Varacca, and N. Yoshida. Event structure semantics of parallel extrusion in the $\pi$-calculus. In L. Birkedal, editor, *FOSSACS 2012*, volume 7213 of *LNCS*, pages 225–239. Springer, 2012.

19 H. Ehrig, K. Ehrig, U. Prange, and G. Taentzer. *Fundamentals of Algebraic Graph Transformation.* Springer, 2006.

20 H. Ehrig, U. Golas, A. Habel, L. Lambers, and F. Orejas. $\mathcal{M}$-adhesive transformation systems with nested application conditions. Part 2: Embedding, critical pairs and local confluence. *Fundamenta Informaticae*, 118(1-2):35–63, 2012.

21 H. Ehrig, U. Golas, A. Habel, L. Lambers, and F. Orejas. $\mathcal{M}$-adhesive transformation systems with nested application conditions. Part 1: Parallelism, concurrency and amalgamation. *Mathematical Structures in Computer Science*, 24(4):240406, 2014.

22 H. Ehrig, A. Habel, J. Padberg, and U. Prange. Adhesive high-level replacement categories and systems. In H. Ehrig, G. Engels, F. Parisi-Presicce, and G Rozenberg, editors, *ICGT 2004*, LNCS, pages 144–160. Springe, 2004.

23 Hartmut Ehrig, Annegret Habel, and Francesco Parisi-Presicce. Basic results for two types of high-level replacement systems. In Michel Bauderon and Andrea Corradini, editors, *GETGRATS Closing Workshop 2001*, volume 51 of *ENTCS*, pages 127–138. Elsevier, 2001.

24 Hartmut Ehrig and Hans-Jörg Kreowski. Parallelism of manipulations in multidimensional information structures. In Antoni W. Mazurkiewicz, editor, *MFCS 1976*, volume 45 of *LNCS*, pages 284–293. Springer, 1976.

25 Hartmut Ehrig, Michael Pfender, and Hans Jürgen Schneider. Graph-grammars: An algebraic approach. In *SWAT 1973*, pages 167–180. IEEE Computer Society, 1973.

26 Hartmut Ehrig and Ulrike Prange. Weak adhesive high-level replacement categories and systems: A unifying framework for graph and Petri net transformations. In Kokichi Futatsugi, Jean-Pierre Jouannaud, and José Meseguer, editors, *Algebra, Meaning, and Computation, Essays Dedicated to Joseph A. Goguen*, volume 4060 of *LNCS*, pages 235–251. Springer, 2006.

27 F. Gadducci. Graph rewriting and the $\pi$-calculus. *Mathematical Structures in Computer Science*, 17(3):1–31, 2007.

28 R. Garner and S. Lack. On the axioms for adhesive and quasiadhesive categories. *Theory and Applications of Categories*, 27(3):27–46, 2012.

29 A. Habel and D. Plump. $\mathcal{M},\mathcal{N}$-adhesive transformation systems. In H. Ehrig, G. Engels, H.-J. Kreowski, and G. Rozenberg, editors, *ICGT 2012*, volume 7562 of *LNCS*, pages 218–233. Springer, 2012.

30 T. Heindel. *A category theoretical approach to the concurrent semantics of rewriting*. PhD thesis, Universität Duisburg–Essen, 2009.

31 P.T. Johnstone, S. Lack, and P. Sobociński. Quasitoposes, quasiadhesive categories and Artin glueing. In T. Mossakowski, U. Montanari, and M. Haveraaen, editors, *CALCO 2007*, volume 4624 of *LNCS*, pages 312–326. Springer, 2007.

32 S. Lack and P. Sobociński. Adhesive and quasiadhesive categories. *RAIRO-Theoretical Informatics and Applications*, 39(3):511–545, 2005.

33 S. Lack and P. Sobociński. Toposes are adhesive. In A. Corradini, H. Ehrig, U. Montanari, L. Ribeiro, and G. Rozenberg, editors, *ICGT 2006*, volume 4178 of *LNCS*, pages 184–198. Springer, 2006.

34 Henrich Lauko, Lukás Korencik, and Peter Goodman. On the optimization of equivalent concurrent computations. *CoRR*, abs/2208.06295, 2022.

35 J.J. Lévy. Optimal reductions in the lambda-calculus. In J.P. Seldin and J.R. Hindley, editors, *To H.B. Curry, Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 159–191. Academic Press, 1980.

36 S. MacLane. *Categories for the working mathematician.* Springer, 2013.

37 Antoni W. Mazurkiewicz. Trace theory. In Wilfried Brauer, Wolfgang Reisig, and Grzegorz Rozenberg, editors, *Advances in Petri Nets 1986*, volume 255 of *LNCS*, pages 279–324. Springer, 1986.

38 J. Meseguer. Conditional rewriting logic as a unified model of concurrency. *Theoretical Computer Science*, 96(1):73–155, 1992.

39 M. Nielsen, G. Plotkin, and G. Winskel. Petri Nets, Event Structures and Domains, Part 1. *Theoretical Computer Science*, 13:85–108, 1981.

40 Roy Overbeek, Jörg Endrullis, and Aloïs Rosset. Graph rewriting and relabeling with PBPO$^{+}$: A unifying theory for quasitoposes. *Journal of Logical and Algebraic Methods in Programming*, 133:100873, 2023.

41 I. Phillips, I. Ulidowski, and S. Yuen. Modelling of bonding with processes and events. In Gerhard W. Dueck and D. Michael Miller, editors, *RC 2013*, volume 7948 of *LNCS*, pages 141–154. Springer, 2013.

42 Max Willsey, Chandrakana Nandi, Yisu Remy Wang, Oliver Flatt, Zachary Tatlock, and Pavel Panchekha. egg: Fast and extensible equality saturation. *Proceedings of the ACM on Programming Languages*, 5(POPL):1–29, 2021.

## A Properties of $\mathcal{M}$-adhesive categories

This first appendix is devoted to the proofs of a few well-known results about $\mathcal{M}$-adhesive categories. Observe that our notion of $\mathcal{M}$-adhesivity follows [20, 21] and is different from the one of [1]. What is called $\mathcal{M}$-adhesivity in the latter paper corresponds to our strict $\mathcal{M}$-adhesivity. Moreover, in [1] the class $\mathcal{M}$ is assumed to be only stable under pullbacks. However, if $\mathcal{M}$ contains all split monos, then stability under pushouts can be deduced from the other axioms [12, Prop. 5.1.21].

## A.1 Some results on $\mathcal{A}$-stable and $\mathcal{A}$-Van Kampen squares

We start proving some general results regarding $\mathcal{A}$-Van Kampen and $\mathcal{A}$-stable squares. Let us begin recalling some classical results about about pullbacks and pushouts [36].

▶ **Lemma A.1.** *Let $\mathbf{X}$ be a category, and consider the diagram below, then the following hold*

1. *if the right square is a pullback, then the whole rectangle is a pullback if and only if the left square is one;*
2. *if the left square is a pushout, then the whole rectangle is a pushout if and only if the right square is one.*

$$\begin{tikzcd}
X \arrow[r, "f"] \arrow[d, "a"'] & Y \arrow[r, "g"] \arrow[d, "b"] & Z \arrow[d, "c"] \\
A \arrow[r, "h"'] & B \arrow[r, "k"'] & C
\end{tikzcd}$$

The following proposition establishes a key property of $\mathcal{A}$-Van Kampen squares with a mono as a side: they are not only pushouts, but also pullbacks [8, 22, 32].

▶ **Proposition A.2.** *Let $\mathcal{A}$ be a class of arrows stable under pushouts and containing all the isomorphisms. If the square below is $\mathcal{A}$-Van Kampen and $m \colon A \to C$ is mono and belongs to $\mathcal{A}$, then the square is a pullback and $n$ is a monomorphism.*

$$\begin{tikzcd}
A \arrow[r, "g"] \arrow[d, "m"'] & B \arrow[d, "n"] \\
C \arrow[r, "f"'] & D
\end{tikzcd}$$

The previous proposition allows us to establish the following result.

▶ **Lemma A.3.** *Let $\mathcal{A}$ be a class of arrows stable under pullbacks, pushouts and containing all isomorphisms. Suppose that the left square below is $\mathcal{A}$-Van Kampen, while the vertical faces in the right cube are pullbacks. Suppose moreover that $m \colon A \to C$ and $d \colon D' \to D$ are mono and that $d$ belongs to $\mathcal{A}$. Then $d \leq n$ if and only if $c \leq m$.*

$$\begin{tikzcd}
A \arrow[r, "g"] \arrow[d, "m"'] & B \arrow[d, "n"] \\
C \arrow[r, "f"'] & D
\end{tikzcd}$$

$$\begin{tikzcd}
& A' \arrow[rr, "g'"] \arrow[dl, "m'"'] \arrow[dd, "a"' near end] & & B' \arrow[dl, "n'"'] \arrow[dd, "b"] \\
C' \arrow[rr, "f'" near start, crossing over] \arrow[dd, "c"'] & & D' & \\
& A \arrow[rr, "g" near end] \arrow[dl, "m"] & & B \arrow[dl, "n"] \\
C \arrow[rr, "f"'] & & D \arrow[from=uu, "d" near start, crossing over] &
\end{tikzcd}$$

▶ Remark A.4. Recall that, given two monos $m : M \to X$ and $n : N \to X$ with the same codomain, $m \leq n$ means that there exists a, necessarily unique and necessarily mono, $k : M \to N$ fitting in the triangle below.

Notice, moreover, that if $m \leq n$ and $n \leq m$, then the arrow $k : M \to N$ is an isomorphism.

$$\begin{tikzcd}
M \arrow[rr, dotted, "k"] \arrow[dr, "m"'] & & N \arrow[dl, "n"] \\
& X &
\end{tikzcd}$$

Finally, we show that $\mathcal{A}$-stable pushouts enjoy a *pullback-pushout decomposition* property.

▶ **Proposition A.5.** *Let* $\mathbf{X}$ *be a category and* $\mathcal{A}$ *a class of arrows stable under pullbacks. Suppose that, in the diagram below, the whole rectangle is an* $\mathcal{A}$*-stable pushout and the right square a pullback. If the arrow* $k$ *is in* $\mathcal{A}$ *and it is a monomorphism, then both squares are pushouts.*

$$\begin{tikzcd}
X \arrow[r, "f"] \arrow[d, "a"'] & Y \arrow[r, "g"] \arrow[d, "b"] & Z \arrow[d, "c"] \\
A \arrow[r, "h"'] & B \arrow[r, "k"'] & C
\end{tikzcd}$$

## A.2 Useful properties of $\mathcal{M}$-adhesive categories

We are now going to apply the results of the previous section to $\mathcal{M}$-adhesive categories in order to establish some *high-level replacement properties* [19, 21, 22]. A first important result that can be immediately established, with the aid of Proposition A.2, is the following one.

▶ **Proposition A.6.** *Let* $\mathbf{X}$ *be an* $\mathcal{M}$*-adhesive category. Then* $\mathcal{M}$*-pushouts are pullbacks.*

From Proposition A.6, in turn, we can derive the following corollaries.

▶ **Corollary A.7.** *In a* $\mathcal{M}$*-adhesive category* $\mathbf{X}$*, every* $m \in \mathcal{M}$ *is a regular mono.*

The following result now follows at once noticing that a regular monomorphism which is also epic is automatically an isomorphism.

▶ **Corollary A.8.** *If* $\mathbf{X}$ *is an* $\mathcal{M}$*-adhesive categories, then every epimorphism in* $\mathcal{M}$ *is an isomorphism. In particular, every adhesive category* $\mathbf{X}$ *is* balanced*: if a morphism is monic and epic, then it is an isomorphism.*

▶ **Lemma A.9** ($\mathcal{M}$-pushout-pullback decomposition)**.** *Let* $\mathbf{X}$ *be an* $\mathcal{M}$*-adhesive category and suppose that, in the diagram below, the whole rectangle is a pushout and the right square a pullback. Then the following statements hold*

1. *if* $a$ *belongs to* $\mathcal{M}$ *and* $k$ *is a mono, then both squares are pushouts and pullbacks;*
2. *if* $f$ *and* $k$ *are in* $\mathcal{M}$*, then both squares are pushouts and pullbacks.*

$$\begin{tikzcd}
X \arrow[r, "f"] \arrow[d, "a"'] & Y \arrow[r, "g"] \arrow[d, "b"] & Z \arrow[d, "c"] \\
A \arrow[r, "h"'] & B \arrow[r, "k"'] & C
\end{tikzcd}$$

Let us turn our attention to pushout complements.

▶ **Definition A.10** (Pushout complement)**.** *Let* $f\colon X \to Y$ *and* $g\colon Y \to Z$ *be two composable arrows in a category* $\mathbf{X}$*. A* pushout complement *for the pair* $(f, g)$ *is a pair* $(h, k)$ *with* $h\colon X \to W$ *and* $k\colon W \to Z$ *such that the square below is a pushout.*

$$\begin{tikzcd}
X \arrow[r, "f"] \arrow[d, "h"'] & Y \arrow[d, "g"] \\
W \arrow[r, "k"'] & Z
\end{tikzcd}$$

Working in an $\mathcal{M}$-adhesive category guarantees that pushout complements are unique.

▶ **Lemma A.11.** *Let $f\colon X \to Y$ be an arrow in an $\mathcal{M}$-adhesive category $\mathbf{X}$ and suppose that the left square below is a pushout while the right one is a pullback, with $m\colon M \rightarrowtail X$ and $n\colon N \rightarrowtail Y$ in $\mathcal{M}$. Then $n \leq k$ if and only if $p_2 \leq m$.*

$$\begin{tikzcd}
M \arrow[r, "h"] \arrow[d, tail, "m"'] & Q \arrow[d, tail, "k"] & & P \arrow[r, "p_1"] \arrow[d, tail, "p_2"'] & N \arrow[d, tail, "n"] \\
X \arrow[r, "f"'] & Y & & X \arrow[r, "f"'] & Y
\end{tikzcd}$$

▶ **Corollary A.12** (Uniqueness of pushout complements)**.** *Let $\mathbf{X}$ be a $\mathcal{M}$-adhesive category. Given $m\colon X \rightarrowtail Y$ in $\mathcal{M}$ and $f\colon Y \to Z$, let $(h_1, k_1)$ and $(h_2, k_2)$ be pushout complements of $m$ and $f$ depicted below. Then there exists a unique isomorphism $\phi\colon W_1 \to W_2$ making the diagram below commutative.*

$$\begin{tikzcd}
& X \arrow[r, tail, "m"] \arrow[d, "h_1"'] \arrow[dl, bend right, "h_2"'] & Y \arrow[d, "f"] \\
W_2 \arrow[rr, tail, bend right, "k_2"'] & W_1 \arrow[l, dotted, "\phi"'] \arrow[r, tail, "k_1"] & Z
\end{tikzcd}$$

## B $\mathcal{M}$-adhesivity is not enough

In Section 3 we introduced the notion of strong enforcing left-linear rewriting system, a class that contains all the linear rewriting systems. This result can be further refined: in [6] a class $\mathbb{B}$ of (quasi-)adhesive category is defined for which the local Church-Rosser Theorem holds for left-linear rewriting system. In our language, this means that every left-linear rewriting system based on a category in $\mathbb{B}$ is strong enforcing. This section is devoted to repropose, and slightly generalise, the results of [6] to our context.

▶ **Definition B.1.** *Let $\mathcal{M}$ be a class of monos in a category $\mathbf{X}$, closed under composition, containing all isomorphisms and stable under pullbacks and pushouts. Suppose that the diagram below is given and the whole rectangle is a pushout. We say that $\mathbf{X}$ satisfies*

- *the $\mathcal{M}$-*mixed decomposition *property if, whenever $k$ belongs to $\mathcal{M}$ and the right half of the diagram below is a pullback, then the left one is a pushout;*
- *the $\mathcal{M}$-*pushout decomposition *property if whenever $a, b$ and $c$ belongs to $\mathcal{M}$ and the right half of the diagram below is a pushout, then its left half is a pushout too.*

$$\begin{tikzcd}
X \arrow[r, "f"] \arrow[d, "a"'] & Y \arrow[r, "g"] \arrow[d, "b"] & Z \arrow[d, "c"] \\
A \arrow[r, "h"'] & B \arrow[r, "k"'] & C
\end{tikzcd}$$

The class of categories of type $\mathbb{B}$ is closed under the same constructions of Theorem 2.5. This can be deduced at once from the fact that in such categories pullbacks and pushouts are computed component-wise.

▶ **Lemma B.2.** *Let $\mathbf{X}$ be a category satisfying the $\mathcal{M}$-mixed and $\mathcal{M}$-pushout decomposition properties, then the following hold*

1. *for every object $X$, $\mathbf{X}/X$ satisfies the $\mathcal{M}/X$-mixed and the $\mathcal{M}/X$-pushout decomposition properties, while $X/\mathcal{M}$-adhesive satisfies their $X/\mathcal{M}$ variants, where*

$$\mathcal{M}/X := \{m \in \mathcal{A}(\mathbf{X}/X) \mid m \in \mathcal{M}\} \qquad X/\mathcal{M} := \{m \in \mathcal{A}(X/\mathbf{X}) \mid m \in \mathcal{M}\}$$

2. *for every small category* $\mathbf{Y}$*, the category* $\mathbf{X}^{\mathbf{Y}}$ *satisfies the* $\mathcal{M}^{\mathbf{Y}}$*-mixed and the* $\mathcal{M}^{\mathbf{Y}}$*-pushout decomposition properties, where*

$$\mathcal{M}^{\mathbf{Y}} := \{\eta \in \mathcal{A}(\mathbf{X}^{\mathbf{Y}}) \mid \eta_Y \in \mathcal{M} \text{ for every } Y \in \mathbf{Y}\}$$

The following result shows that the mixed and pushout decomposition properties guarantee that every independence pair is strong.

▶ **Theorem B.3.** *Let* $\mathbf{X}$ *be an* $\mathcal{M}$*-adhesive category with all pushouts and satisfying the* $\mathcal{M}$*-mixed and the* $\mathcal{M}$*-pushout decomposition properties. Then every left-linear rewriting system on* $\mathbf{X}$ *is strong enforcing.*

**Proof.** Let $\underline{\mathscr{D}} = \{\mathscr{D}_i\}_{i=0}^{1}$ be the derivation made by two sequentially independent derivations

$$\begin{tikzcd}
L_0 \arrow[d, "m_0"'] & K_0 \arrow[l, tail, "l_0"'] \arrow[r, "r_0"] \arrow[d, "k_0"'] & R_0 \arrow[drrr, bend left=15, "i_0"] \arrow[dr, "h_0"'] & & L_1 \arrow[dlll, bend right=15, "i_1"'] \arrow[dl, "m_1"] & K_1 \arrow[l, tail, "l_1"'] \arrow[r, "r_1"] \arrow[d, "k_1"] & R_1 \arrow[d, "h_1"] \\
G_0 & D_0 \arrow[l, tail, "f_0"] \arrow[rr, "g_0"'] & & G_1 & & D_1 \arrow[ll, tail, "f_1"] \arrow[r, "g_1"'] & G_2
\end{tikzcd}$$

We have to show that $(i_0, i_1)$ is a strong independence pair. Now, by hypothesis $\mathbf{X}$ has all pushouts, thus the only thing to show is that the squares below are pushouts (the third one is the usual pullback of $f_1 : D_1 \rightarrowtail G_0$ along $g_0 : D_0 \to G_1$)

$$\begin{tikzcd}
K_0 \arrow[r, "r_0"] \arrow[d, "u_0"'] & R_0 \arrow[d, "i_0"] \\
P \arrow[r, "p_1"'] & D_1
\end{tikzcd}
\qquad
\begin{tikzcd}
K_1 \arrow[r, tail, "l_1"] \arrow[d, "u_1"'] & L_1 \arrow[d, "i_1"] \\
P \arrow[r, "p_0"'] & D_0
\end{tikzcd}
\qquad
\begin{tikzcd}
P \arrow[r, tail, "p_0"] \arrow[d, "p_1"'] & D_0 \arrow[d, "g_0"] \\
D_1 \arrow[r, tail, "f_1"'] & G_1
\end{tikzcd}$$

To see this, consider the following two diagrams

$$\begin{tikzcd}
K_0 \arrow[r, "u_0"'] \arrow[rr, bend left, "k_0"] \arrow[d, "r_0"'] & P \arrow[r, tail, "p_0"'] \arrow[d, "p_1"] & D_0 \arrow[d, "g_0"] \\
R_0 \arrow[r, "i_0"] \arrow[rr, bend right, "h_0"'] & D_1 \arrow[r, tail, "f_1"] & G_1
\end{tikzcd}
\qquad
\begin{tikzcd}
K_1 \arrow[r, "u_1"'] \arrow[rr, bend left, "k_1"] \arrow[d, tail, "l_1"'] & P \arrow[r, "p_1"'] \arrow[d, tail, "p_0"] & D_1 \arrow[d, tail, "f_1"] \\
L_1 \arrow[r, "i_1"] \arrow[rr, bend right, "m_1"'] & D_0 \arrow[r, "g_0"] & G_1
\end{tikzcd}$$

The thesis now follows from the $\mathcal{M}$-mixed and the $\mathcal{M}$-pushout decomposition property. ◀

Our next step is to identify sufficient conditions for a category $\mathbf{X}$ to satisfy the mixed and $\mathcal{M}$-pushout decomposition properties.

▶ **Definition B.4.** *Let* $\mathbf{X}$ *be an* $\mathcal{M}$*-adhesive category, the pair* $(\mathbf{X}, \mathcal{M})$ *is of* type $\mathbb{B}$ *if*

1. *every arrow in* $\mathcal{M}$ *is a coproduct coprojection;*
2. $\mathbf{X}$ *has all pushouts;*
3. $\mathbf{X}$ *has strict initial objects and for every* $X$ *the unique arrow* $?_X : 0 \to X$ *belongs to* $\mathcal{M}$*;*
4. *all pushouts are* $\mathcal{M}$*-stable.*

▶ Remark B.5. It is worth to examine more closely conditions 1 and 3 of the above definition.

- Let $m_0 : X_0 \rightarrowtail Y$ be in $\mathcal{M}$, the first condition means that there exists $m_1 : X_1 \to Y$ such that $(Y, \{m_i\}_{i=0}^{1})$ is a coproduct. This, together with property 3, entails that every coprojection in a coproduct is in $\mathcal{M}$. This follows since any coproduct $(X_1 + X_2, \{\iota_{X_i}\}_{i=0}^{1})$ fits in a pushout diagram as the one below.

- $\mathbf{X}$ has strict initial objects if it has initial objects and every arrow $f : X \to 0$ is an isomorphism. Notice that if initial objects are strict, then $?_X \colon 0 \to X$ is mono for every $X$: indeed for every pair $f, g \colon Y \rightrightarrows 0$ then, by strictness, $Y$ is initial and so $f = g$.

$$\begin{tikzcd}
0 \arrow[r, tail, "?_{X_1}"] \arrow[d, tail, "?_{X_2}"] & X_1 \arrow[d, tail, "\iota_{X_1}"] \\
X_2 \arrow[r, tail, "\iota_{X_2}"'] & X_1 + X_2
\end{tikzcd}$$

▶ **Example B.6.** The category **Set** of sets and functions is of type $\mathbb{B}$. Similarly, the category **Inj** of sets and injective functions is quasiadhesive and, with regular monos, of type $\mathbb{B}$.

Categories of type $\mathbb{B}$ satisfy a property resembling *extensivity* [11].

▶ **Proposition B.7.** *Let $(\mathbf{X}, \mathcal{M})$ be a pair of type $\mathbb{B}$. Then for every diagram as the one below, in which the bottom row is a coproduct cocone and the vertical arrows are in $\mathcal{M}$, the top row is a coproduct if and only if the two squares are pullbacks.*

$$\begin{tikzcd}
A \arrow[r, "f"] \arrow[d, tail, "r"'] & C \arrow[d, tail, "s"'] & B \arrow[l, "g"'] \arrow[d, tail, "t"] \\
X \arrow[r, "\iota_X"'] & X + Y & Y \arrow[l, "\iota_Y"]
\end{tikzcd}$$

**Proof.** Consider the cube below, in which the back faces are pullbacks. The bottom faces is an $\mathcal{M}$-Van Kampen pushout, thus the top face is a pushout if and only if the front faces are pullbacks. By strictness of $0$, $a \colon I \to 0$ is an isomorphism, so that $I$ is initial, therefore the $\mathcal{M}$-Van Kampen condition reduces to the request that $(C, \{f, g\})$ is a coproduct cocone if and only if the front faces are pullbacks, as claimed.

$$\begin{tikzcd}
& I \arrow[rr, "k"] \arrow[dl, "h"'] \arrow[dd, tail, "a"] & & B \arrow[dl, "g"'] \arrow[dd, tail, "t"] \\
A \arrow[rr, "f"] \arrow[dd, tail, "r"'] & & C \arrow[dd, tail, "s"] & \\
& 0 \arrow[rr, tail, "?_Y"] \arrow[dl, tail, "?_X"] & & Y \arrow[dl, tail, "\iota_Y"] \\
X \arrow[rr, tail, "\iota_X"'] & & X + Y &
\end{tikzcd}$$

◀

The previous result entails the following one, needed to show that in any pair $(\mathbf{X}, \mathcal{M})$ of type $\mathbb{B}$, the category $\mathbf{X}$ satisfies the $\mathcal{M}$-mixed and $\mathcal{M}$-pushout decomposition properties.

▶ **Proposition B.8.** *Let $(\mathbf{X}, \mathcal{M})$ be a pair of type $\mathbb{B}$, and suppose that the square below is an $\mathcal{M}$-pullback. Then there exists $E \in \mathbf{X}$, $e \colon E \rightarrowtail C$ in $\mathcal{M}$, $\phi : E \to B_1$ such that $(C, \{m, e\})$ is a coproduct and $g = f + \phi$. Moreover, such a square is a pushout if and only if $\phi$ is an isomorphism.*

$$\begin{tikzcd}
A \arrow[r, "f"] \arrow[d, tail, "m"'] & B_0 \arrow[d, tail, "\iota_{B_0}"] \\
C \arrow[r, "g"'] & B_0 + B_1
\end{tikzcd}$$

▶ **Lemma B.9.** *Let $(\mathbf{X}, \mathcal{M})$ be a pair of type $\mathbb{B}$, then $\mathbf{X}$ satisfies the $\mathcal{M}$-mixed and $\mathcal{M}$-pushout decomposition properties.*

**Proof.** $\mathcal{M}$-mixed decomposition property. This follows at once from Proposition A.5.

$\mathcal{M}$-pushout-decomposition property. Using Propositions B.7 and B.8 and the fact that arrows in $\mathcal{M}$ are coproduct coprojections, we can reduce to prove the property for diagrams as the one below. By hypothesis and Proposition B.8, $\varphi$ and $\varphi \circ \phi$ are both isomorphisms, therefore $\phi$ is an isomorphism too and we can conclude again using Proposition B.8.

$$\begin{tikzcd}
X \arrow[r, "f"] \arrow[d, tail, "\iota_X"'] & Y \arrow[r, "g"] \arrow[d, tail, "\iota_Y"] & Z \arrow[d, tail, "\iota_Z"] \\
X + A \arrow[r, "f+\phi"] \arrow[rr, bend right, "(g\circ f)+(\varphi\circ\phi)"'] & Y + B \arrow[r, "g+\varphi"] & Z + C
\end{tikzcd}$$

◀

▶ **Definition B.10.** *Let $\mathbf{X}$ be an $\mathcal{M}$-adhesive category, we say that the pair $(\mathbf{X}, \mathcal{M})$ is of* type $\mathbb{B}^+$ *if (at least) one of the following holds*

1. *$(\mathbf{X}, \mathcal{M})$ is of type $\mathbb{B}$;*
2. *$\mathbf{X}$ is $\mathbf{Y}/Y$ and $\mathcal{M} = \mathcal{N}/Y$for some $(\mathbf{Y}, \mathcal{N})$ of type $\mathbb{B}^+$ and $Y \in \mathbf{Y}$;*
3. *$\mathbf{X}$ is $Y/\mathbf{Y}$ and $\mathcal{M} = Y/\mathcal{N}$for some $(\mathbf{Y}, \mathcal{N})$ of type $\mathbb{B}^+$ and $Y \in \mathbf{Y}$:*
4. *$\mathbf{X}$ is $\mathbf{Y}^{\mathbf{A}}$ and $\mathcal{M} = \mathcal{N}^{\mathbf{A}}$ for some category category $\mathbf{A}$ and $(\mathbf{Y}, \mathcal{N})$ of type $\mathbb{B}^+$.*

▶ **Example B.11.** The pair $(\mathbf{Graph}, \mathsf{Mono}(\mathbf{Graph}))$ of graphs and their monos is of type $\mathbb{B}^+$ but not of type $\mathbb{B}$. Indeed, not every monomorphism of graphs is a coproduct coprojection.

From Lemmas B.2 and B.9 we can now deduce at once the following.

▶ **Corollary B.12.** *For every $(\mathbf{X}, \mathcal{M})$ of type $\mathbb{B}^+$, the category $\mathbf{X}$ has the $\mathcal{M}$-mixed and $\mathcal{M}$-pushout decomposition properties.*

Using Theorem B.3 we finally get the main result of our appendix.

▶ **Corollary B.13.** *Let $(\mathbf{X}, \mathcal{M})$ be a pair of type $\mathbb{B}^+$, then every left-linear rewriting system is strong enforcing.*