Electronic Communications of the EASST
Volume 73 (2016)

Graph Computation Models
Selected Revised Papers from GCM 2015

Composition of $\mathscr{M},\mathscr{N}$-adhesive Categories with Application to Attribution of Graphs

Christoph Peuser and Annegret Habel

21 pages

Guest Editors: Detlef Plump
Managing Editors: Tiziana Margaria, Julia Padberg, Gabriele Taentzer
ECEASST Home Page: http://www.easst.org/eceasst/ ISSN 1863-2122

Composition of $\mathscr{M},\mathscr{N}$-adhesive Categories with Application to Attribution of Graphs

Christoph Peuser[1*] **and Annegret Habel**[2]

[1]peuser@informatik.uni-oldenburg.de
[2]habel@informatik.uni-oldenburg.de
Carl von Ossietzky Universität Oldenburg, Germany

Abstract: This paper continues the work on $\mathscr{M},\mathscr{N}$-adhesive categories and shows some important composition properties for these categories. We present a new concept of attributed graphs and show that the corresponding category is $\mathscr{M},\mathscr{N}$-adhesive. As a consequence, we inherit all nice properties for $\mathscr{M},\mathscr{N}$-adhesive systems such as the Local Church-Rosser Theorem, the Parallelism Theorem, and the Concurrency Theorem for this type of attributed graphs.

Keywords: Graph transformation, attributed graphs, composition, adhesive categories, adhesive systems

1 Introduction

The double-pushout approach to graph transformation, which was invented in the early 1970's, is the best studied framework for graph transformation [Roz97, EEKR99, EKMR99, EEPT06, EEGH15]. As applications of graph transformation come with a large variety of graphs and graph-like structures, the double-pushout approach has been generalized to the abstract settings of high-level replacement systems, adhesive, $\mathscr{M}$-adhesive, $\mathscr{M},\mathscr{N}$-adhesive, and $\mathscr{W}$-adhesive categories. This paper continues the work of Habel and Plump [HP12] on $\mathscr{M},\mathscr{N}$-adhesive categories.

In the literature, there are several variants of attribution concepts, e.g. typed attributed graphs in the sense of Ehrig et al. [EEPT06], attributed graphs in the sense of Poskitt and Plump [PP12], attributed graphs as a graph with a marked sub-graph in the sense of Kastenberg and Rensink [KR12], separation of the graph structure and their attribution and data in the sense of Golas [Gol12], and attributed structures in the sense of Duval et al. [DEPR14]. The use of attributed graphs allows a more compact representation of a system. While graph transformation systems are Turing complete, representing numerical values as part of the graph can lead to very large representations of parts of a system that may be comparatively unimportant. Instead attributes allow us to concentrate on graphically representing those parts of a system we are primarily interested in.

Our main aim is to consider composition of categories to make it easier to define new models for transformation systems. To this end we prove closure results of $\mathscr{M},\mathscr{N}$-adhesive categories against product, functor, slice and coslice, comma, string and multiset category. We apply our

[*] This work is supported by the German Research Foundation through the Research Training Group (DFG GRK 1765) SCARE (www.scare.uni-oldenburg.de).

results to the construction of a category **AttGraphs** of attributed graphs from the well-known category **Graphs** of unlabelled graphs and a category **Att** of attribute collections. By closure results for $\mathscr{M},\mathscr{N}$-adhesive categories, we obtain that the category **AttGraphs** is $\mathscr{M},\mathscr{N}$-adhesive. By the results in [HP12], the Local Church-Rosser Theorem, the Parallelism Theorem and the Concurrency Theorem hold for the new type of attributed graphs provided that the HLR$^+$-properties [HP12] are satisfied.

The paper is structured as follows. In Section 2, we recall the definition of $\mathscr{M},\mathscr{N}$-adhesive categories. In Section 3, we prove some basic composition results and show that the string and multiset categories are $\mathscr{M},\mathscr{N}$-adhesive provided that an underlying category is. In Section 4, we introduce the category **AttGraphs** of attributed graphs and show that it is $\mathscr{M},\mathscr{N}$-adhesive. In Section 5 we present some related work and, in Section 6 some concluding remarks.

This paper is an extended version of the paper [PH15]. It contains all proofs and additional examples.

2 $\mathscr{M},\mathscr{N}$-adhesive Categories

In this section, we recall the definition of $\mathscr{M},\mathscr{N}$-adhesive categories, introduced in [HP12], generalizing the one of $\mathscr{M}$-adhesive categories [EGH10]. We assume that the reader is familiar with the basic concepts of category theory; standard references are [EEPT06, Awo10].

Definition 1 ($\mathscr{M},\mathscr{N}$-adhesive Categories) A category **C** is $\mathscr{M},\mathscr{N}$-adhesive, where $\mathscr{M}$ is a class of monomorphisms and $\mathscr{N}$ a class of morphisms, if the following properties are satisfied:

1. $\mathscr{M}$ and $\mathscr{N}$ contain all isomorphisms and are closed under composition and decomposition. Moreover, $\mathscr{N}$ is closed under $\mathscr{M}$-decomposition, that is, $f;g \in \mathscr{N}$, $g \in \mathscr{M}$ implies $f \in \mathscr{N}$.
2. **C** has $\mathscr{M},\mathscr{N}$-pushouts and $\mathscr{M}$-pullbacks. Also, $\mathscr{M}$ and $\mathscr{N}$ are stable under pushouts and pullbacks.
3. $\mathscr{M},\mathscr{N}$-pushouts are $\mathscr{M},\mathscr{N}$-van Kampen squares.

In the following we might sometimes use '$\mathscr{M},\mathscr{N}$-adhesive' without specifying $\mathscr{M}$ and $\mathscr{N}$ explicitly, especially in the sense of 'preserving $\mathscr{M},\mathscr{N}$-adhesiveness'.

Remark 1 **C** has $\mathscr{M},\mathscr{N}$-pushouts, *if there is a pushout whenever one of the given morphisms is in* $\mathscr{M}$ *and the other morphism is in* $\mathscr{N}$. **C** has $\mathscr{M}$-pullbacks, *if there exists a pullback whenever at least one of the given morphisms is in* $\mathscr{M}$. *A class* $\mathscr{X} \in \{\mathscr{M},\mathscr{N}\}$ is stable under $\mathscr{M},\mathscr{N}$-pushouts *if, given the* $\mathscr{M},\mathscr{N}$*-pushout (1) in Figure 1,* $m \in \mathscr{X}$ *implies* $n \in \mathscr{X}$ and stable under $\mathscr{M}$-pullbacks *if, given the* $\mathscr{M}$*-pullback (1) in Figure 1,* $n \in \mathscr{X}$ *implies* $m \in \mathscr{X}$. *An* $\mathscr{M},\mathscr{N}$*-pushout is an* $\mathscr{M},\mathscr{N}$-van Kampen square *if for the commutative cube (2) in Figure 1 with the pushout (1) as bottom square,* $b,c,d,m \in \mathscr{M}$, $f \in \mathscr{N}$, *and the back faces being pullbacks, we have that the top square is a pushout if and only if the front faces are pullbacks.*

In [HP12], it is shown that all $\mathscr{M}$-adhesive categories are $\mathscr{M},\mathscr{N}$-adhesive.

$$\begin{tikzcd}
A \arrow[r, "m"] \arrow[d, "f"'] \arrow[dr, phantom, "(1)"] & B \arrow[d, "g"] \\
C \arrow[r, "n"] & D
\end{tikzcd}$$

$$\begin{tikzcd}
 & & & A' \arrow[dll] \arrow[dr] \arrow[dd] & \\
 & C' \arrow[dr] \arrow[dd, "c"] & & & B' \arrow[dll] \arrow[dd, "b"] \\
 & & D' \arrow[dd, "d"] & A \arrow[dll, "f"'] \arrow[dr, "m"] & \\
 & C \arrow[dr, "n"'] & & & B \arrow[dll, "g"] \\
(2) & & D & &
\end{tikzcd}$$

Figure 1: $\mathscr{M},\mathscr{N}$-pushout and $\mathscr{M},\mathscr{N}$-van Kampen square

Lemma 1 ($\mathscr{M}$-adhesive $\Rightarrow$ $\mathscr{M},\mathscr{N}$-adhesive [HP12]) *Let* $\mathbf{C}$ *be any category and* $\mathscr{N}$ *be the class of all morphisms in* $\mathbf{C}$*. Then* $\mathbf{C}$ *is* $\mathscr{M},\mathscr{N}$*-adhesive if and only if* $\mathbf{C}$ *is* $\mathscr{M}$*-adhesive.*

In the following, we give some examples of categories that are $\mathscr{M},\mathscr{N}$-adhesive.

Lemma 2 (Basic $\mathscr{M},\mathscr{N}$-adhesive Categories [EEPT06]) *The following categories are* $\mathscr{M}$*-adhesive and, by Lemma 1,* $\mathscr{M},\mathscr{N}$*-adhesive where* $\mathscr{N}$ *is the class of all morphisms in* $\mathbf{C}$*: The category* **Sets** *of sets and functions, where* $\mathscr{M}$ *is the class of all injective functions. The category* **Graphs** *of graphs and graph morphisms, where* $\mathscr{M}$ *is the class of all injective graph morphisms. The category* **LGraphs** *of labelled graphs and graph morphisms, where* $\mathscr{M}$ *is the class of all injective graph morphisms. The discrete category*[1] *over some set L, referred to as* Disc(L)*, is* $\mathscr{M}$*-adhesive:* Disc(L) *only has identity morphisms, for which proving the required properties becomes trivial.*

The category **PLGraphs** *of partially labelled graphs and graph morphisms is* $\mathscr{M},\mathscr{N}$*-adhesive, but not* $\mathscr{M}$*-adhesive [HP12]:* $\mathscr{M}$ *and* $\mathscr{N}$ *are the classes of all injective and all (injective) undefinedness-preserving*[2] *graph morphisms, respectively.*

3 Composition of $\mathscr{M},\mathscr{N}$-Adhesive Categories

There are various ways to construct new categories from given ones. Beside the standard constructions (product, slice and coslice, functor and comma category as defined in [EEPT06]) we consider the constructions of a string category and a multiset category. For each of these constructions, we prove a composition result, saying more or less, whenever we start with $\mathscr{M}_i,\mathscr{N}_i$-adhesive categories, then the new constructed category is $\mathscr{M},\mathscr{N}$-adhesive for some $\mathscr{M},\mathscr{N}$.

We briefly recall the definition of the comma category, we refer to [EEPT06] for the definiton of product, slice and coslice and functor category.

Definition 2 (Comma Category[EEPT06]) Given two functors $F_1\colon \mathbf{C}_1 \to \mathbf{C}$ and $F_2\colon \mathbf{C_2} \to \mathbf{C}$ and an index set $\mathscr{I}$, the comma category $\mathbf{ComCat}(F_1,F_2,\mathscr{I})$ is defined as follows: The objects are all triples (C_1,C_2,op) with C_1 an object in $\mathbf{C}_1$, C_2 an object in $\mathbf{C}_2$, and a family of morphisms

[1] The discrete category over some collection S has S as objects and only identity morphisms.

[2] A morphism $f\colon G \to H$ *preserves undefinedness*, if it maps unlabelled items in G to unlabelled items in H.

$\mathrm{op} = [\mathrm{op}_i]_{i\in\mathscr{I}}$ where $\mathrm{op}_i\colon F_1(C_1) \to F_2(C_2)$ is a morphism in $\mathbf{C}$. The morphisms from (C_1, C_2, op) to $(C_1', C_2', \mathrm{op}')$ are all pairs (g,h) where $g\colon C_1 \to C_1'$ and $h\colon C_2 \to C_2'$ are morphisms in $\mathbf{C}_1$ and $\mathbf{C}_2$ respectively, such that the following diagram commutes for all $i \in \mathscr{I}$:

$$\begin{tikzcd}
F_1(C_1) \arrow[r, "F_1(g)"] \arrow[d, "\mathrm{op}_i"'] \arrow[dr, phantom, "="] & F_1(C_1') \arrow[d, "\mathrm{op}_i'"] \\
F_2(C_2) \arrow[r, "F_2(h)"'] & F_2(C_2')
\end{tikzcd}$$

The composition of morphisms in $\mathbf{ComCat}(F_1, F_2, \mathscr{I})$ is defined componentwise, and the identities are pairs of identities in the component categories $\mathbf{C}_1$ and $\mathbf{C}_2$.

First, we consider the standard constructions: product, slice and coslice, functor, and comma category ([EEPT06] A2 and A6).

Theorem 1 (Standard Constructions) *$\mathscr{M},\mathscr{N}$-adhesive categories can be constructed as follows:*

1. *If $\mathbf{C_i}$ is $\mathscr{M}_i,\mathscr{N}_i$-adhesive ($i = 1,2$), then the product category $\mathbf{C_1} \times \mathbf{C_2}$ is $\mathscr{M},\mathscr{N}$- adhesive where $\mathscr{M} = \mathscr{M}_1 \times \mathscr{M}_2$ and $\mathscr{N} = \mathscr{N}_1 \times \mathscr{N}_2$.*

2. *If $\mathbf{C}$ is $\mathscr{M},\mathscr{N}$-adhesive and X is an object of $\mathbf{C}$, then the slice category $\mathbf{C}\backslash X$ and the coslice category $X\backslash\mathbf{C}$ over X are $\mathscr{M}',\mathscr{N}'$-adhesive where the morphism classes $\mathscr{M}',\mathscr{N}'$ are restricted to the slice and coslice category, i.e., for $\mathscr{X} \in \{\mathscr{M},\mathscr{N}\}$, $\mathscr{X}' = \mathscr{X} \cap \mathbf{C}\backslash X$ and $\mathscr{X}' = \mathscr{X} \cap X\backslash\mathbf{C}$, respectively.*

3. *If $\mathbf{C}$ is $\mathscr{M},\mathscr{N}$-adhesive, then for every category $\mathbf{X}$, the functor category $[\mathbf{X},\mathbf{C}]$ is $\mathscr{M}_{\mathrm{ft}},\mathscr{N}_{\mathrm{ft}}$-adhesive with functor transformations $\mathscr{M}_{\mathrm{ft}}$ and $\mathscr{N}_{\mathrm{ft}}$.*[3]

4. *Let $\mathbf{C_i}$ be $\mathscr{M}_i,\mathscr{N}_i$-adhesive and $F_i\colon \mathbf{C_i} \to \mathbf{C}$ be functors ($i = 1,2$), where F_1 preserves $\mathscr{M}_1,\mathscr{N}_1$-pushouts and F_2 preserves $\mathscr{M}_2$-pullbacks. Then the comma category $\mathbf{ComCat}(F_1,F_2,\mathscr{I})$ is $\mathscr{M}^c,\mathscr{N}^c$-adhesive where $\mathscr{M}^c = (\mathscr{M}_1 \times \mathscr{M}_2) \cap \mathrm{Mor}$, $\mathscr{N}^c = (\mathscr{N}_1 \times \mathscr{N}_2) \cap \mathrm{Mor}$, and Mor is the set of all morphisms of the comma category.*

Remark 2 *Statement 4 in Theorem 1 above also holds for the case of a comma category* $\mathbf{ComCat}(F_1, F_2, \mathscr{I})$*, with $|\mathscr{I}| = 1$ and both functors F_1, F_2 pointing into* **Sets***, where the morphism* op *is always bijective. We will use $F_1 \downarrow_{\mathrm{bij}} F_2$ as a shorthand for these categories.*

Proof. The proof is a slight generalization of the corresponding one for $\mathscr{M}$-adhesive categories (see Theorem 4.15 in [EEPT06]).

1. The product category $\mathbf{C}_1 \times \mathbf{C}_2$ is $\mathscr{M},\mathscr{N}$-adhesive, because $\mathscr{M}$ and $\mathscr{N}$ inherit the required composition and decomposition properties from $\mathscr{M}_1$ and $\mathscr{M}_2$ and $\mathscr{N}_1$ and $\mathscr{N}_2$ respectively. Pushouts along $(\mathscr{M},\mathscr{N})$-pairs of morphisms can be constructed componentwise. So can

[3] For a class $\mathscr{X}$, $\mathscr{X}_{\mathrm{ft}}$ denotes the class of natural transformations $t\colon F \to G$, where all morphisms $t_X\colon F(X) \to G(X)$ are in $\mathscr{X}$.

the van Kampen square, since pullbacks can also be constructed componentwise. The stability of $\mathscr{M}$ and $\mathscr{N}$ is inherited from pushouts and pullbacks in $\mathbf{C}_i$.

2. Morphisms, pullbacks and pushouts can similarly be constructed componentwise for both slice categories $\mathbf{C}\backslash X$ and coslice categories $X\backslash\mathbf{C}$. This construction also ensures that $\mathscr{M}'$ and $\mathscr{N}'$ are stable under pushout and pullback.

3. The functor category $[\mathbf{X},\mathbf{C}]$, where $\mathscr{M}_{\text{ft}}$ and $\mathscr{N}_{\text{ft}}$ are functor transformations, is $\mathscr{M},\mathscr{N}$-adhesive, because the functor transformations $\mathscr{M}_{\text{ft}}$ (or $\mathscr{N}_{\text{ft}}$) are monomorphisms in $[\mathbf{X},\mathbf{C}]$ and the required composition and decomposition properties are inherited from $\mathscr{M}$ (or $\mathscr{N}$). Pushouts and pullbacks and the $\mathscr{M},\mathscr{N}$-van Kampen square are constructed pointwise, i.e. for each object $X \in [\mathbf{X},\mathbf{C}]$. The pointwise construction also ensures that $\mathscr{M}_{\text{ft}}$ and $\mathscr{N}_{\text{ft}}$ are stable under pushout and pullback.

4. The comma category $\mathbf{ComCat}(F_1,F_2,\mathscr{I})$ is $\mathscr{M}^c,\mathscr{N}^c$-adhesive, because $\mathscr{M}^c$ and $\mathscr{N}^c$ inherit the required composition and decompostion properties from $\mathscr{M}_i$ and $\mathscr{N}_i$. $\mathscr{M}^c,\mathscr{N}^c$-pushouts can be constructed componentwise, since F_1 preserves them. $\mathscr{M}^c$-pullbacks can be constructed componentwise, since F_2 preserves them. Consequently $\mathscr{M}^c,\mathscr{N}^c$-van Kampen squares can equally constructed componentwise. All of these constructions can also be done if we restrict ourselves to objects with a bijective morphism op. The componentwise construction ensures that $\mathscr{M}^c$ and $\mathscr{N}^c$ are stable under pushout and pullback.

□

Example 1 ([EEPT06]) *The category* **Graphs** *of unlabelled graphs is isomorphic to the functor category* $[\,E \rightrightarrows V\,, \mathbf{Sets}]$. *For the type graph TG, the category* $\mathbf{Graphs}_{TG}$ *of graphs typed over TG is isomorphic to the slice category* $\mathbf{Graphs}\backslash TG$.

Given a category $\mathbf{C}$, we construct a string category $\mathbf{C}^*$ and prove that $\mathscr{M},\mathscr{N}$-adhesive categories are closed under string construction.

Definition 3 (String Category) Given a category $\mathbf{C}$, the *string category* $\mathbf{C}^*$ is defined as follows: The objects are strings $a_1 \dots a_m$ of objects of $\mathbf{C}$, including the empty string λ. The morphisms between two strings $a_1 \dots a_m$ and $b_1 \dots b_n$ with $m \leq n$ are strings of morphisms such that $a_1 \dots a_m$ is *embedded* in $b_1 \dots b_n$, i.e. $f_1 \colon a_1 \to b_{i+1}, \dots, f_m \colon a_m \to b_{i+m}$ in $\mathbf{C}$. The empty string λ is an initial element for $\mathbf{C}^*$.

$$\begin{tikzcd}
 & & & a_1 \arrow[d, "f_1"'] & \dots & a_m \arrow[d, "f_m"'] & & & \\
b_1 & \dots & b_i & b_{i+1} & \dots & b_{i+m} & b_{i+m+1} & \dots & b_n
\end{tikzcd}$$

Remark 3 *Our definition for a string category is close to that of a free monoidal category, but is not a free monoidal category because we allow morphisms that do not preserve length. Such morphisms prevent us from defining a monoidal product on arrows. In the absence of these morphisms we cannot formulate rules that add elements to a string or remove them.*

Lemma 3 *For every morphism $f\colon A \to B$, the commuting square consisting of f and the identities id_A, id_B is a pushout and a pullback.*

Proof. Follows from standard category theory. □

Theorem 2 (**C** $\mathcal{M},\mathcal{N}$-adhesive $\Rightarrow$ $\mathbf{C}^*$ $\mathcal{M}^*,\mathcal{N}^*$-adhesive) *If* **C** *is $\mathcal{M},\mathcal{N}$-adhesive, then the string category $\mathbf{C}^*$ over* **C** *is $\mathcal{M}^*,\mathcal{N}^*$-adhesive where $\mathcal{M}^*$ and $\mathcal{N}^*$ contain those morphisms which are strings of morphisms in $\mathcal{M}$ and $\mathcal{N}$, respectively. $\mathcal{N}^*$ is further restricted to morphisms that preserve length, i.e. where domain and codomain are of equal length.*

We will use this decomposition frequently in the following proofs.

Lemma 4 (Decomposition of String Morphisms) *Any string morphism $f\colon a_1\ldots a_m \to b_1\ldots b_n$ can be decomposed into $f_=\colon a_1\ldots a_m \to b_{i+1}\ldots b_{i+m}$ that preserves length and $f_+\colon b_{i+1}\ldots b_{i+m} \to b_1\ldots b_n$ an embedding that consists of identity morphisms in* **C** *such that $f = f_=;f_+$. Both morphisms are in $\mathcal{M}^*$ or $\mathcal{N}^*$, if f is in $\mathcal{M}^*$ or $\mathcal{N}^*$ respectively.*

Proof. Straightforward. □

Proof of Theorem 2. By inspection of Definition 1.

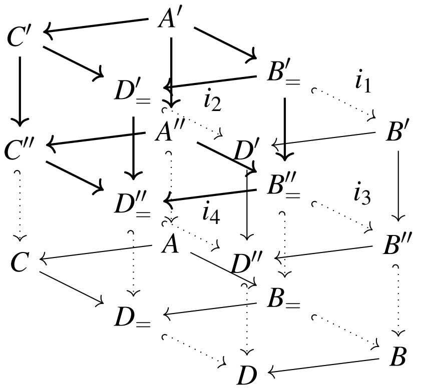
1. Closure properties: $\mathcal{M}^*$ and $\mathcal{N}^*$ contain all isomorphisms, since an isomorphism in $\mathbf{C}^*$ must be a list of isomorphisms in **C** and $\mathcal{M}$ and $\mathcal{N}$, respectively contain all isomorphisms in **C**. Isomorphisms must further be lists of equal length and $\mathcal{M}^*$ and $\mathcal{N}^*$ contain all morphisms with domain and codomain of equal length. Composition and decomposition properties can be inherited from **C**, e.g. the composition of morphisms in $\mathcal{M}^*$ and $\mathcal{N}^*$ implies the composition of a list of morphisms in $\mathcal{M}$ and $\mathcal{N}$, respectively, which is possible since **C** is $\mathcal{M},\mathcal{N}$-adhesive. This also applies to the closure of $\mathcal{N}^*$ under $\mathcal{M}^*$-decomposition, since $f;g \in \mathcal{N}^*$ implies a domain and codomain of equal length, which in turn implies $f \in \mathcal{N}^*$.

2a. Existence of $\mathcal{M}^*,\mathcal{N}^*$-pushouts and $\mathcal{M}^*$-pullbacks: Given the morphisms $f\colon a_1\ldots a_m \to b_1\ldots b_n$ and $g\colon a_1\ldots a_m \to c_1\ldots c_m$, where $f \in \mathcal{M}^*$ and $g \in \mathcal{N}^*$ we can construct $\mathcal{M}^*,\mathcal{N}^*$-pushouts as follows (see Figure 4b): We decompose f into $f_=$ and f_+ as in Lemma 4. We can construct (1) as a pushout componentwise from pushouts in the underlying category **C**. If we have a string of length zero, the resulting pushout will be the identity square of λ. The morphisms f_+, i_+ in (2) consists entirely of identity morphisms in **C**, making each component of (2) a pushout in **C** by Lemma 3. Then (2) is a pushout in $\mathbf{C}^*$. By composition of pushouts $(1)+(2)$ is a pushout.

Note that the remaining morphisms in $b_1\ldots b_n \to d_1\ldots d_n$ must be isomorphisms for (2) to be a pushout, since the pushout object is only unique up to isomorphism we can assume, without loss of generality, that the remaining morphisms are identities.

2b. $\mathbf{C}^*$ has pullbacks along $\mathcal{M}^*$-morphisms: Given two morphisms $f\colon a_1\ldots a_m \to c_1\ldots c_o \in \mathcal{M}^*$ and $g\colon b_1\ldots b_n \to c_1\ldots c_o$ the pullback object is constructed by identifying the largest substring $c_i\ldots c_{i+x}$ where all elements c_j have a coimage in both $a_1\ldots a_m$ and $b_1\ldots b_n$ and constructing their pullbacks in **C**. If there is no such common substring, the empty string λ is the pullback object instead.

3. $\mathcal{M}^*,\mathcal{N}^*$-pushouts are $\mathcal{M}^*,\mathcal{N}^*$-van Kampen squares:



$$\begin{tikzcd}
C' \arrow[r] \arrow[d] \arrow[dr, phantom, "(1)"] & D'_= \arrow[r, hook, dotted, "i_2"] \arrow[d] \arrow[dr, phantom, "(3)"] & D' \arrow[d] \arrow[dr, phantom, "(5)"] & B' \arrow[l] \arrow[d] \\
C'' \arrow[r] \arrow[d, hook, dotted] \arrow[dr, phantom, "(2)"] & D''_= \arrow[r, hook, dotted, "i_4"] \arrow[d, hook, dotted] \arrow[dr, phantom, "(4)"] & D'' \arrow[d, hook, dotted] \arrow[dr, phantom, "(6)"] & B'' \arrow[l] \arrow[d, hook, dotted] \\
C \arrow[r] & D_= \arrow[r, hook, dotted] & D & B \arrow[l]
\end{tikzcd}$$

Figure 2: Van Kampen square for the string category (I)

The top face is a pushout ⇒ the front faces are pullbacks:

1. We construct the cube in Figure 2 by decomposing the pushout at the top face as seen above into a pushout with morphisms that preserve length and another where the horizontal morphisms do not preserve length and decomposing all vertical morphisms in the commutative cube into pairs according to Lemma 4.

2.The morphisms in the middle layer of the cube are determined by those in the bottom layer, since the vertical morphisms in the lower half of the cube consist of identity morphisms only. As a result the middle layer consists of pushouts as well.

3. The core cube in the upper back is a van Kampen square, with the pushout of $A'' \to C''$ and $A'' \to B''_=$ as bottom face. This core cube can be constructed componentwise from cubes in $\mathbf{C}$, since all morphisms preserve length. If these strings are of length 0 this is an identity cube of empty strings λ.

4. Since the top face of the core cube is a pushout, (1) consists of pullbacks in $\mathbf{C}$ by the van Kampen property and is therefore a pullback in $\mathbf{C}^*$.

5. (2) and (3) are pullbacks, since their vertical and horizontal morphisms respectively consist of identity morphisms only and and the morphism shared with (1) preserve length. The pullback objects are then limited to the length of $D''_=$ and we can construct the pullbacks componentwise from pullbacks in $\mathbf{C}$ according to Lemma 3.

6. The commuting square (4) consists entirely of morphisms composed of identity morphisms in $\mathbf{C}$. Then $D''_=$ is the largest string for which each element has a pullback in $\mathbf{C}$ and therefore (4) is a pullback.

7. We examine the commutative square (5) componentwise: Some of its components have a coimage in the front right square of the core cube via the morphisms $i_1 \dots i_4$. Since these morphisms consists of identity morphisms in $\mathbf{C}$, these components are pullbacks. The remaining components of (5) have horizontal morphisms that coincide with those morphisms that are part of the top and bottom pushout that are identities, making these components pullbacks as well. Thus, (5) is a pullback.

8. (6) is a pullback, by Lemma 3 since $B'' \to B$, $D'' \to D$ consist of identity morphisms in $\mathbf{C}$.

$$\begin{tikzcd}
 & & A' \arrow[dll] \arrow[ddrr] \arrow[dddd] & & & \\
C' \arrow[ddr] \arrow[dddd] & & & & & \\
 & & & & B'_= \arrow[dlll] \arrow[dddd] \arrow[ddr, hook, dotted, "i_1"] & \\
 & D'_= \arrow[dddd] \arrow[ddrr, hook, dotted, "i_2"] & & & & \\
 & & A'' \arrow[dll] \arrow[ddrr] \arrow[dddd, hook, dotted] & & & B' \arrow[dll] \arrow[dddd] \\
C'' \arrow[ddr] \arrow[dddd, hook, dotted] & & & D' \arrow[dddd] & & \\
 & & & & B''_= \arrow[dlll] \arrow[dddd, hook, dotted] \arrow[ddr, hook, dotted, "i_3"] & \\
 & D''_= \arrow[dddd, hook, dotted] \arrow[ddrr, hook, dotted, "i_4"] & & & & \\
 & & A \arrow[dll] \arrow[ddrr] & & & B'' \arrow[dll] \arrow[dddd, hook, dotted] \\
C \arrow[ddr] & & & D'' \arrow[dddd, hook, dotted] & & \\
 & & & & B_= \arrow[dlll] \arrow[ddr, hook, dotted] & \\
 & D_= \arrow[ddrr, hook, dotted] & & & & \\
 & & & & & B \arrow[dll] \\
 & & & D & &
\end{tikzcd}$$

$$\begin{tikzcd}
C' \arrow[d] \arrow[dr, phantom, "(1)"] & A' \arrow[l] \arrow[d] \\
D'_= \arrow[d, hook, dotted, "i_2"'] \arrow[dr, phantom, "(2)"] & B'_= \arrow[l] \arrow[d, hook, dotted, "i_1"] \\
D' & B' \arrow[l]
\end{tikzcd}$$

Figure 3: Van Kampen square for the string category (II)

9. By composition of pullbacks $(1)+(2)+(3)+(4)$ and $(5)+(6)$ are pullbacks.
The front faces are pullbacks $\Rightarrow$ the top face is a pushout:
1. We decompose the vertical morphisms and the bottom pushout as in the cube in Figure 3. Additionally we construct the pushouts in the middle layer analogously.
2. The commutative cube in the upper back is a van Kampen square. We construct this cube componentwise from cubes in $\mathbf{C}$, since all morphisms preserve length.
3. Since the upper square at the right front face is a pullback and i_1,i_2,i_3,i_4 consist of idenities, $B'_=,B''_=,D'_=,D''_=$ is a pullback.
4. Then (1) is a pushout by the van Kampen property of $\mathbf{C}$, i.e. all component cubes have front faces as pullbacks, therefore all components of the top face are pushouts.
5. (2) is a pushout by Lemma 3, since i_1,i_2 consists of identities.
6. By composition of pushouts, $(1)+(2)$ is a pushout.
This concludes the proof. □

Example 2 *The category* **Strings** $=\mathrm{Disc}(L)^*$ *of strings, where* $\mathrm{Disc}(L)$ *is the discrete category over some alphabet L, is* $\mathscr{M}^*,\mathscr{N}^*$*-adhesive.*

Given a category $\mathbf{C}$, we construct a multiset category $\mathbf{C}^\oplus$ and prove that $\mathscr{M},\mathscr{N}$-adhesive categories are closed under multiset construction. In contrast to the above construction for a string category, we can construct a multiset category by ignoring the order of elements. We use $\{|\dots|\}$ to denote a multiset, e.g $\{|a,a,b|\}$ denotes the multiset with elements a,a and b.

Definition 4 (Multiset Category) Given a category $\mathbf{C}$, the *multiset category* $\mathbf{C}^\oplus$ is defined as follows: The objects are multisets $\{|A_1\dots A_m|\}$ of objects of $\mathbf{C}$, including the empty multiset $\emptyset$. The morphisms between two objects $\{|A_1\dots A_m|\}$ and $\{|B_1\dots B_n|\}$ (with $m\le n$) are strings of morphisms $f_i\colon A_i\to B_{j_i}$ in $\mathbf{C}$, where $j_i=j_k$ implies $i=k$, $i\in\{1,\dots,m\}$.

Remark 4 *A category of multisets* **MUL** *has previously been discussed for example in [SI13]. In contrast to those definitions we define a multiset category* $\mathbf{C}^\oplus$ *over a different category* $\mathbf{C}$.

$$\begin{tikzcd}
A \arrow[r] \arrow[d] \arrow[rr, bend left, "f"] \arrow[dd, bend right, "g"'] \arrow[dr, phantom, "(1)"] & B' \arrow[r, hook, dotted] \arrow[d] \arrow[dr, phantom, "(2)"] & B \arrow[d, hook, dotted] \\
C' \arrow[r] \arrow[d, hook, dotted] \arrow[dr, phantom, "(2')"] & D \arrow[r, hook, dotted] \arrow[d, hook, dotted] \arrow[dr, phantom, "(3)"] & E \arrow[d, hook, dotted] \\
C \arrow[r, hook, dotted] & F \arrow[r, hook, dotted] & G
\end{tikzcd}$$

(a) A pushout in $\mathbf{C}^{\oplus}$

$$\begin{tikzcd}
a_1\ldots a_m \arrow[r, "f_="'] \arrow[d, "g"'] \arrow[rr, bend left, "f"] \arrow[dr, phantom, "(1)"] & b_{i+1}\ldots b_{i+m} \arrow[r, hook, dotted, "f_+"'] \arrow[d] \arrow[dr, phantom, "(2)"] & b_1\ldots b_n \arrow[d] \\
c_1\ldots c_m \arrow[r] & d_{i+1}\ldots d_{i+m} \arrow[r, hook, dotted, "i_+"] & d_1\ldots d_n
\end{tikzcd}$$

(b) A pushout in $\mathbf{C}^*$

Figure 4: Pushouts in String and Multiset Categories

Theorem 3 (**C** $\mathscr{M},\mathscr{N}$-adhesive $\Rightarrow$ $\mathbf{C}^{\oplus}$ $\mathscr{M}^{\oplus},\mathscr{N}^{\oplus}$-adhesive) *If* **C** *is* $\mathscr{M},\mathscr{N}$*-adhesive, then the multiset category* $\mathbf{C}^{\oplus}$ *over* **C** *is* $\mathscr{M}^{\oplus},\mathscr{N}^{\oplus}$*- adhesive where* $\mathscr{M}^{\oplus}$ *and* $\mathscr{N}^{\oplus}$ *contain those morphisms which are strings of morphisms in* $\mathscr{M}$ *and* $\mathscr{N}$*, respectively.*

Lemma 4 applies to morphisms in $\mathscr{M}^{\oplus}$ or $\mathscr{N}^{\oplus}$.

Proof. By inspection of Definition 1.

1. Closure properties: $\mathscr{M}^{\oplus}$ and $\mathscr{N}^{\oplus}$ contain all isomorphisms, since an isomorphism in $\mathbf{C}^*$ must be a list of isomorphisms in **C** and $\mathscr{M}$ and $\mathscr{N}$, respectively contain all isomorphisms in **C**. Composition and decomposition properties can be inherited from **C**, e.g. the composition of morphisms in $\mathscr{M}^{\oplus}$ and $\mathscr{N}^{\oplus}$ implies the composition of a list of morphisms in $\mathscr{M}$ and $\mathscr{N}$, respectively, which is possible since **C** is $\mathscr{M},\mathscr{N}$-adhesive.

2a. Existence of $\mathscr{M}^{\oplus},\mathscr{N}^{\oplus}$-pushouts and $\mathscr{M}^{\oplus}$-pullbacks: Given the morphisms $f\colon A\to B$ and $g\colon A\to C$, where $f\in\mathscr{M}^{\oplus}$ and $g\in\mathscr{N}^{\oplus}$ and $\#A=l,\#B=m,\#C=n$ [4], we can construct $\mathscr{M}^{\oplus},\mathscr{N}^{\oplus}$-pushouts as follows (see Figure 4a): We decompose $f\colon A\to B$ as in Lemma 4, where $f_=$ preserves cardinality and f_+ consists of identities in **C**. We decompose g analogously. We can construct (1) as a pushout componentwise from pushouts in the underlying category **C**. If $\#A=0$ then A,B,C,D are the empty multiset $\emptyset$. (2) (and analogously (2')) is a pushout by Lemma 3. (3) consists entirely of identity morphisms in **C** and is therefore a pushout as well. By composition of pushouts $(1)+(2)+(2')+(3)$ is a pushout.

2b. $\mathbf{C}^{\oplus}$ has pullbacks along $\mathscr{M}^{\oplus}$-morphisms: Given two morphisms $f\colon A_1\ldots A_m\to C_1\ldots C_o\in\mathscr{M}^{\oplus}$ and $g\colon B_1\ldots B_n\to C_1\ldots C_o$ the pullback object is constructed by identifying the largest subset $C_i\ldots C_{i+x}$ where all elements C_j have a coimage in both $A_1\ldots A_m$ and $B_1\ldots B_n$ and constructing their pullbacks in **C**. If there is no such common subset, the empty multiset $\emptyset$ is the pullback object instead.

3. $\mathscr{M}^{\oplus},\mathscr{N}^{\oplus}$-pushouts are $\mathscr{M}^{\oplus},\mathscr{N}^{\oplus}$-van Kampen squares:

The top face is a pushout $\Rightarrow$ the front faces are pullbacks:

1. We decompose the top and bottom pushout as in Figure 5.

[4] # denotes the cardinality of a multiset, i.e. for a multiset $f\colon A\to\mathbb{N}$, $\#A=\sum_{a\in A}f(a)$.

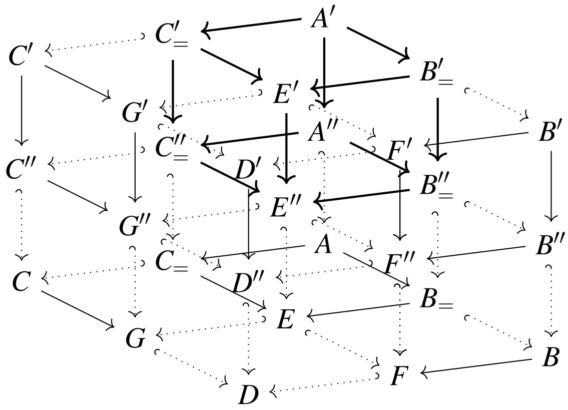


Figure 5: Van Kampen square for the multiset category

2. We decompose all vertical morphisms in the cube into pairs where the first morphism preserves cardinality and the second morphism is an inclusion consisting of identity morphisms in **C** (see Lemma 4).

3. We construct the interior morphisms shown in the cube above analogoulsy to the construction in the proof for the string category, once along $A \to B$ and once along $A \to C$.

4. The remaining pullbacks on the front sides can also be constructed analogously to the construction in the previous proof, using Lemma 3.

The front faces are pullbacks $\Rightarrow$ the top face is a pushout:

1. We decompose the bottom pushout as in the cube in Figure 5.

2. We decompose all vertical morphisms in the cube into pairs where the first morphism preserves cardinality and the second morphism is an inclusion consisting of identity morphisms in **C** (see Lemma 4).

3. We construct the interior morphisms shown in the cube above analogoulsy to the construction in the proof for the string category, once along $A \to B$ and once along $A \to C$.

4. The remaining pushouts on the top side can also be constructed analogously to the construction in the previous proof, using Lemma 3.

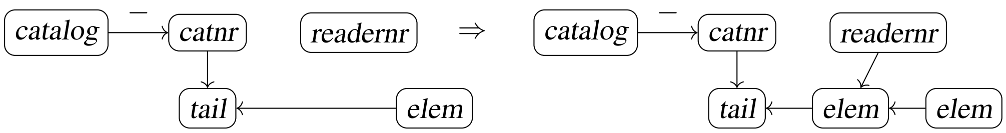
This concludes the proof. □

Example 3 *The categories* **MultiSets** $= \mathrm{Disc}(L)^{\oplus}$ *of multisets is* $\mathscr{M}^{\oplus},\mathscr{N}^{\oplus}$*-adhesive for suitable* $\mathscr{M},\mathscr{N}$*.*

Finally, we will look at the possibilities that these compositions give us.

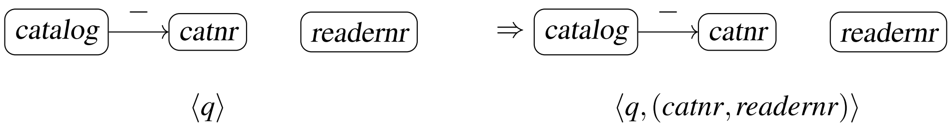
Example 4 (Library System) *Assume we have a library system modeled as a transformation system over labelled graphs, with books, authors, readers and a catalog as in [EK80]. We would like to extend this system with queues, such that a reader can reserve a book if it is currently unavailable. One straightforward way to implement this would be extending the rules to include the creation and management of such queues as additional nodes and edges in the graph. This* *may, however lead to a large part of the graph structure consisting of these queues, when they are in fact a minor element of the overall model. We sketch a rule in this setting below, adding a reader to the end of a queue that is associated with a catalog number.*

Reserve(catnr, readernr):



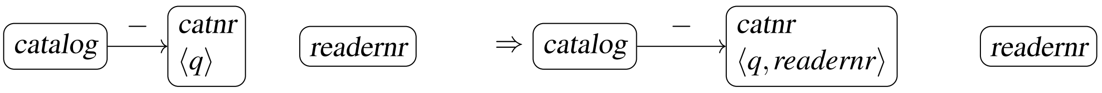
We could instead introduce queues as distinct components of the underlying model, using the compositions detailed in this chapter: We define the category $\mathbf{PLGraphs} \times \mathrm{Disc}(R \times C)^*$*, where* R *is a fixed set of reader numbers and* C *a set of catalog numbers and* $\mathrm{Disc}(A)$ *is the discrete category over some set* A*. An object of this category is a labelled graph together with a string of pairs of readers and catalog numbers. Instead of adding additional nodes like in the rule above, we append a pair of catalog number and reader number to the queue* $\langle q \rangle$*. Note that we would a notion of abstract rules here that allow us to refer to a queue of arbitrary length, such as e.g. the rule schemata in GP [PP12].*

Reserve(catnr, readernr):



If we want to separate queues for every title in the library we could instead have used a comma category: We define the category $\mathit{PLGraphs} \downarrow_{\mathrm{bij}} \mathrm{Q}$*, where* $\mathrm{Q}\colon (\mathrm{Disc}(R)^*)^{\oplus} \to \mathbf{Sets}$ *is some functor that preserves pullbacks and* $\mathit{PLGraphs}$ *is analogous to* Graphs*, mapping a graph to a union of its nodes and edges. The definition contains a multiset* ($(\mathrm{Disc}(R)^*)^{\oplus}$)*, since we may need more than one instance of a certain queue as transformations might otherwise have unwanted side effects. Now the rule can instead add the reader number directly to the queue associated with the catalog number. Again, some notion of abstract rules would be required.*

Reserve(catnr, readernr):



In all of the above cases, the resulting category is $\mathcal{M},\mathcal{N}$*-adhesive, since* **PLGraphs** *and discrete categories are* $\mathcal{M},\mathcal{N}$*-adhesive and the constructions we used preserve* $\mathcal{M},\mathcal{N}$*-adhesiveness.*

4 Attributed Graphs

In this section we define attributed graphs, where attribute values can be changed analogously to relabelling. The addition or removal of attributes to a node or edge should also be possible.

Defining a category of *attribute collections* which collect all the attributes of a node or an edge is our first step. These attribute collections consist of a set of names, each of which is associated with a value. Attributed graphs are defined, where each node or edge is associated with such an attribute collection.

We start by defining a category **PL** for representing the values. Since we can view labels as isolated attribute values, we will consider the simpler case of labels in the following definition. Let L be a label set including the symbol $\bot$ indicating undefinedness. As morphisms we use all identities as well as all morphisms from $\bot$ to a label in L.

Lemma 5 (**PL** is a Category) *For each alphabet L, the class of all elements in $L \cup \{\bot\}$*[5] *as objects and all morphisms of the form $\bot \to x$ and $x \to x$ ($x \in L \cup \{\bot\}$) forms the category* **PL** *where the composition of $x \to y$ and $y \to z$ is $x \to z$ and the identity on x is $x \to x$.*

$$\begin{tikzcd}
a \arrow[loop above] & b \arrow[loop above] & & \cdots & & z \arrow[loop above] \\
& & \bot \arrow[loop above] \arrow[ull] \arrow[ul] \arrow[urrr] & & &
\end{tikzcd}$$

Proof. Follows directly from the definiton. □

It can be shown that the category **PL** is $\mathscr{M},\mathscr{N}$-adhesive.

Lemma 6 (**PL** is $\mathscr{M},\mathscr{N}$-adhesive) *The category* **PL** *is $\mathscr{M},\mathscr{N}$-adhesive where $\mathscr{M}$ and $\mathscr{N}$ are the classes of all morphisms and all identities, respectively.*

Proof. By inspection of Definition 1:

1. Closure properties: $\mathscr{M}$ and $\mathscr{N}$ contain all identity morphisms, which are the only isomorphisms in **PL**. They are also closed under composition and decompostion. Since $f;g \in \mathscr{N} \Rightarrow f,g \in \mathscr{N}$, $\mathscr{N}$ is closed under $\mathscr{M}$-decomposition.

2. PL has $\mathscr{M},\mathscr{N}$-pushouts: Since $\mathscr{N}$ contains only identity morphisms, there are only two cases, with either $\bot \to l$ or another identity morphism as the horizontal morphism:

$$\begin{tikzcd}
l \arrow[r, "m"] \arrow[d, "f"'] \arrow[dr, phantom, "(3)"] & l \arrow[d, dashed, "g"] \\
l \arrow[r, dashed, "n"'] & l
\end{tikzcd}
\qquad
\begin{tikzcd}
\bot \arrow[r, "m"] \arrow[d, "f"'] \arrow[dr, phantom, "(4)"] & l \arrow[d, dashed, "g"] \\
\bot \arrow[r, dashed, "n"'] & l
\end{tikzcd}$$

The diagrams (3) and (4) are pushouts: the morphisms g,n are the only possible morphisms to obtain commutativity and the universal property holds. Since **PL** contains the initial element $\bot$, **PL** has all $\mathscr{M}$-pullbacks. $\mathscr{M}$ is trivially stable under pushouts and pullbacks, since $\mathscr{M}$ contains all morphisms. $\mathscr{N}$ is stable under pushouts and pullbacks because in the two cases above, the only possible morphisms f,g are identity morphisms and therefore in $\mathscr{N}$.

3. In PL, $\mathscr{M},\mathscr{N}$-pushouts are $\mathscr{M},\mathscr{N}$-van Kampen squares: Let (1) be a pushout, where $m \in \mathscr{M}$ and $f \in \mathscr{N}$. We have to show that, given a commutative cube (2) with (1) as bottom

[5] We assume that $\bot$ is not an element of L.

$$\begin{tikzcd}
 & & l \arrow[dll] \arrow[dd] \arrow[ddr] & \\
l \arrow[dd] \arrow[ddr] & & & \\
 & & l \arrow[dll] \arrow[ddr] & l \arrow[dll] \arrow[dd] \\
l \arrow[ddr] & l \arrow[dd] & & \\
 & & & l \arrow[dll] \\
 & l & & (a)
\end{tikzcd}
\qquad
\begin{tikzcd}
 & & \perp \arrow[dll] \arrow[dd] \arrow[ddr] & \\
\perp \arrow[dd] \arrow[ddr] & & & \\
 & & \perp \arrow[dll] \arrow[ddr] & l \arrow[dll] \arrow[dd] \\
\perp \arrow[ddr] & l \arrow[dd] & & \\
 & & & l \arrow[dll] \\
 & l & & (b)
\end{tikzcd}
\qquad
\begin{tikzcd}
 & & \perp \arrow[dll] \arrow[dd] \arrow[ddr] & \\
\perp \arrow[dd] \arrow[ddr] & & & \\
 & & l \arrow[dll] \arrow[ddr] & \perp \arrow[dll] \arrow[dd] \\
l \arrow[ddr] & \perp \arrow[dd] & & \\
 & & & l \arrow[dll] \\
 & l & & (c)
\end{tikzcd}
\qquad
\begin{tikzcd}
 & & \perp \arrow[dll] \arrow[dd] \arrow[ddr] & \\
\perp \arrow[dd] \arrow[ddr] & & & \\
 & & \perp \arrow[dll] \arrow[ddr] & \perp \arrow[dll] \arrow[dd] \\
\perp \arrow[ddr] & \perp \arrow[dd] & & \\
 & & & l \arrow[dll] \\
 & l & & (d)
\end{tikzcd}$$

Figure 6: Possible commutative cubes in **PL**

face, $b,c,d \in \mathscr{M}$ and pullbacks as back faces, the following holds:

$$\text{the top face is a pushout} \Leftrightarrow \text{the front faces are pullbacks}$$

We have already identified the possible pushouts (3) and (4) above. These pushouts lead to four possible cases for back faces that are pullbacks (see Figure 6):

For the pushout (3): The possible spans to construct the back faces of the cube with are:

- $l \leftarrow l \rightarrow l$, which lead to the identity cube (a). Then all faces in the cube are pushouts as well as pullbacks and, therefore, constitute an $\mathscr{M},\mathscr{N}$-van Kampen square.
- $\perp \leftarrow \perp \rightarrow \perp$, which leads to the cube (c) with pullbacks as back faces. Then the top face is the pushout (3) and the front faces are pullbacks.
- $l \leftarrow \perp \rightarrow l$, $\perp \leftarrow \perp \rightarrow l$, for both of which at least one of the back faces will not be a pullback

For the pushout (4): The possible spans to construct the back faces of the cube with are:

- $\perp \leftarrow \perp \rightarrow l$, which leads to the cube (b) with pullbacks as back faces. Then the top face is the pushout (4) and the front faces are pullbacks.
- $\perp \leftarrow \perp \rightarrow \perp$, which leads to the cube (d) with pullbacks as back faces. Then the top face is the pushout (3) and the front faces are pullbacks.

In both cases it is possible to contruct different commutative cubes, but all of these do not have pushouts as top faces nor pullbacks as front faces. □

Definition 5 ($Graphs\colon$ **Graphs** $\rightarrow$ **Sets**) The functor $Graphs\colon$ **Graphs** $\rightarrow$ **Sets** is defined to map graphs to their underlying set of nodes and edges and is given as follows: For a graph $G' = (V',E',s',t')$, let $Graphs(G') = V' + E'$ and for a graph morphism $f_{G'}\colon A \rightarrow B$, let $Graphs(f_{G'})$ be a functor, defined by $Graphs(f)(x) = f_{V'}(x)$ if $x \in V'$ and $f_{E'}(x)$ otherwise.

Lemma 7 *The functor $Graphs\colon$ **Graphs** $\rightarrow$ **Sets** preserves $\mathscr{M},\mathscr{N}$-pushouts, where $\mathscr{M}$ is the class of injective graph morphisms, $\mathscr{N}$ is the class of all morphisms.*

Proof. Given a pushout (P) in **Graphs**, we have to show that $Graphs((P))$ is a pushout in **Sets**, i.e. for every pair of commuting morphisms $g_1\colon Graphs(B) \rightarrow X$ and $g_2\colon Graphs(C) \rightarrow X$ there is a unique morphism $g\colon Graphs(D) \rightarrow X$.

$$\begin{tikzcd}
A \arrow[r, "f_1"] \arrow[d, "f_2"'] \arrow[dr, phantom, "(P)"] & B \arrow[d, "\pi_1"] \\
C \arrow[r, "\pi_2"] & D
\end{tikzcd}
\qquad
\begin{tikzcd}
Graphs(A) \arrow[r, "Graphs(f_1)"] \arrow[d, "Graphs(f_2)"'] \arrow[dr, phantom, "Graphs((P))"] & Graphs(B) \arrow[d, "Graphs(\pi_1)"] \arrow[ddr, dotted, bend left, "g_1"] & \\
Graphs(C) \arrow[r, "Graphs(\pi_2)"] \arrow[drr, dotted, bend right, "g_2"'] & Graphs(D) \arrow[dr, dashed, "g"] & \\
& & X
\end{tikzcd}$$

Let $x \in D$, then either $x \in B$ or $x \in C$. If $x \in B$, then $g_1(x) \in X$, if otherwise $x \in C$ then $g_2(x) \in X$. We can thus define $g(x) = \begin{cases} g_1(x) & \text{if } x \in B \\ g_2(x) & \text{otherwise} \end{cases}$

It remains to show that g is unique: The above construction follows directly from g_1, g_2 and $Graphs(\pi_1), Graphs(\pi_2)$. If we assume a different morphism $g' \colon Graphs(D) \to X$ it would have to follow these same constraints, hence $g' = g$. □

Definition 6 ($PL \colon \mathbf{PL}^{\oplus} \to \mathbf{Sets}$) The functor $PL \colon \mathbf{PL}^{\oplus} \to \mathbf{Sets}$ is defined to map a multiset of labels to a set with distinct elements and is given as follows: For a multiset of labels $m' : L' \to \mathbb{N}$ let $PL(m) = \bigcup_{l' \in L'} \bar{m}(l')$, where for $l' \in L'$, $\bar{m}(l') = \{l'_1, ..., l'_k\}$ iff $m(l') = k$. For a morphism $f \colon m_1 \to m_2$ let $PL(f) = PL(m_1) \to PL(m_2)$ be a morphism in **Sets**, such that $PL(f)(l'_1) = l'_2$ iff $PL(l_1) = l'_1, PL(l_2) = l'_2$ and $f(l_1) = l_2$ with $l_1, l_2 \in m_1, m_2$ respectively.

Example 5 *Given a morphism* $f \colon \{|a,a,b|\} \to \{|a,a,b,b|\}$ *in* $\mathrm{Disc}(L)^{\oplus}$, $PL(f)$ *is the morphism* $\{a_1, a_2, b_1\} \to \{a_1, a_2, b_1, b_2\}$. *PL 'flattens' a multiset into a set by making sure that there are distinct elements in the set for all elements in a multiset.*

Lemma 8 *The functor* $PL \colon \mathbf{PL}^{\oplus} \to \mathbf{Sets}$ *preserves pullbacks.*

Proof. Given a pullback (P) in $\mathbf{PL}^{\oplus}$, we have to show that $PL((P))$ is a pullback in **Sets**, i.e. for every pair of commuting morphisms $g_1 \colon X \to PL(B)$ and $g_2 \colon X \to PL(C)$ there is a unique morphism $g \colon X \to PL(A)$.

$$\begin{tikzcd}
A \arrow[r, "\pi_1"] \arrow[d, "\pi_2"'] \arrow[dr, phantom, "(P)"] & B \arrow[d, "f_1"] \\
C \arrow[r, "f_2"] & D
\end{tikzcd}
\qquad
\begin{tikzcd}
X \arrow[dr, dashed, "g"'] \arrow[drr, dotted, bend left, "g_1"] \arrow[ddr, dotted, bend right, "g_2"'] & & \\
& PL(A) \arrow[r, "PL(\pi_1)"] \arrow[d, "PL(\pi_2)"'] \arrow[dr, phantom, "PL((P))"] & PL(B) \arrow[d, "PL(f_1)"] \\
& PL(C) \arrow[r, "PL(f_2)"] & PL(D)
\end{tikzcd}$$

Let $x \in A$, then both $x \in B$ and $x \in C$ due to the morphisms π_1, π_2. We can thus define $g(x') = x$ if $g_1(x') = PL(\pi_1(x))$ with $x' \in X$.

It remains to show that g is unique: The above construction follows directly from g_1, g_2 and $PL(\pi_1), PL(\pi_2)$. If we assume a different morphism $g' \colon X \to PL(A)$ it would have to follow these same constraints, hence $g' = g$. □

Definition 7 (**Att** $= id_{\mathbf{Sets}} \downarrow_{\text{bij}} PL$) The category **Att** of *attribute collections* is the comma category $id_{\mathbf{Sets}} \downarrow_{\text{bij}} PL$ where $id_{\mathbf{Sets}}$ denotes the identity functor over **Sets**.

Lemma 9 (**Att** is $\mathcal{M}^c, \mathcal{N}^c$-adhesive) *The category* **Att** *of attribute collections is* $\mathcal{M}^c, \mathcal{N}^c$*-adhesive where* $\mathcal{M}^c, \mathcal{N}^c$ *are the classes of morphisms induced by the comma category construction.*

Proof. **Att** $= id_{\mathbf{Sets}} \downarrow_{\text{bij}} PL$ is $\mathcal{M}^c, \mathcal{N}^c$-adhesive, since **Sets** and **PL** are $\mathcal{M}, \mathcal{N}$-adhesive and a multiset and comma category construction preserve $\mathcal{M}, \mathcal{N}$-adhesiveness. □

To construct attributed graphs we define a functor from multisets of these attribute collections to sets for later use in the definition of a comma category. We also prove that this functor preserves pullbacks, since this is required for the comma category to preserve $\mathcal{M}, \mathcal{N}$-adhesiveness.

Definition 8 ($Att \colon \mathbf{Att}^{\oplus} \to \mathbf{Sets}$) The functor $Att \colon \mathbf{Att}^{\oplus} \to \mathbf{Sets}$ is defined to map attribute collections to sets with distinct values and is given by the following: A triple $(id_{\mathbf{Sets}}(S), PL(m), \text{op})^{\oplus}$ is mapped to the set $S^{\oplus} + PL(m)^{\oplus}$ where $\oplus$ is flattened analogously to the way PL does (see Definition 6). A morphism $f \colon A \to B$ in $\mathbf{Att}^{\oplus}$ is mapped to a morphism $Att(f) \colon Att(A) \to Att(B)$, where elements in $Att(A)$ are mapped to elements in $Att(B)$ based on the original mappings in $\mathbf{Att}^{\oplus}$.

Lemma 10 (Att preserves pullbacks) *The functor* $Att \colon \mathbf{Att}^{\oplus} \to \mathbf{Sets}$ *preserves pullbacks.*

Proof. In the same way that the defintion of Att is analogous to that of PL the preservation of pullbacks can be proven in a similar way. □

Definition 9 (**AttGraphs** $= Graphs \downarrow_{\text{bij}} Att$) The category **AttGraphs** of *attributed graphs* is the comma category $Graphs \downarrow_{\text{bij}} Att$.

Now we are able to show that the category **AttGraphs** of attributed graphs is $\mathcal{M}, \mathcal{N}$-adhesive. The categories **Graphs** of graphs and **Att** of attribute collections are $\mathcal{M}_G, \mathcal{N}_G$ and $\mathcal{M}_A, \mathcal{N}_A$-adhesive and the compositions of multiset category and comma category preserve $\mathcal{M}, \mathcal{N}$-adhesiveness.

Theorem 4 (**AttGraphs** is $\mathcal{M}^c, \mathcal{N}^c$-adhesive) *The category* **AttGraphs** *of attributed graphs is* $\mathcal{M}^c, \mathcal{N}^c$*-adhesive where* $\mathcal{M}^c, \mathcal{N}^c$ *are the classes of morphisms induced by the comma category construction.*

Proof. The proof is illustrated in Figure 7. By Lemmata 2 and 9, **Graphs** and **Att** are $\mathcal{M}_G, \mathcal{N}_G$ and $\mathcal{M}_A, \mathcal{N}_A$-adhesive, respectively. $\mathcal{M}_G, \mathcal{N}_G$ are monomorphisms in **Graphs** and $\mathcal{M}_A, \mathcal{N}_A$ are the classes of morphisms induced by the comma category construction of **Att**. By Theorem 3, $\mathbf{Att}^{\oplus}$ is $\mathcal{M}^{\oplus}, \mathcal{N}^{\oplus}$-adhesive. By Theorem 1 and Lemmata 7 and 10, $Graphs \downarrow_{\text{bij}} Att$ is $\mathcal{M}^c, \mathcal{N}^c$-adhesive. By Definition 9, **AttGraphs** is $\mathcal{M}^c, \mathcal{N}^c$-adhesive. □

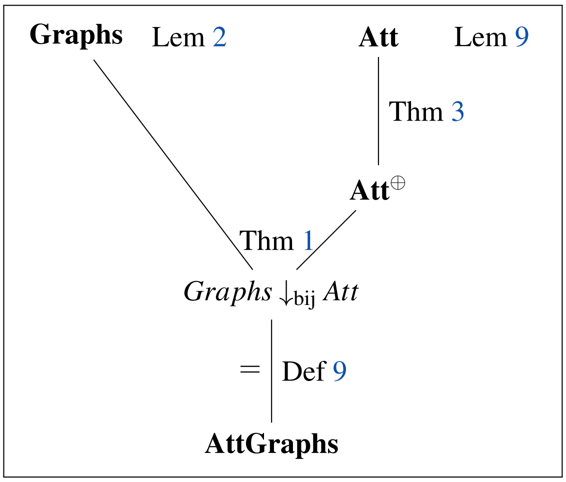


Figure 7: Proof of "**AttGraphs** is $\mathscr{M}^c,\mathscr{N}^c$-adhesive".

Example 6 *Figure 8 shows an attributed graph in our setting. The functor $Graphs$ maps the unlabelled graph U given below to a set $\{v_1,v_2,e_1\}$ of nodes and edges. The functor Att maps the multiset $\{|a=4,b=5,a=4|\}$ to the set $\{a_1,b_1,a_2\}$: whenever an element x occurs n times in the multiset, the symbols $x_1,\ldots,x_n$ occur in the corresponding set. In our case, there is a bijective morphism* $\text{op}\colon \mathbf{Sets} \to \mathbf{Sets}$. *Now we get an attributed graph $G=(U,l_G)$ from the unlabelled graph U and the multiset $\{|a=4,b=5,a=5|\}$ by defining $att_G(x) = Att^{-1}(op(Graphs(x)))$ for each item x in U. The resulting attributed graph is given in the picture below. To represent an attributed graph we write its attribute collections into their corresponding nodes or next to their corresponding edges.*

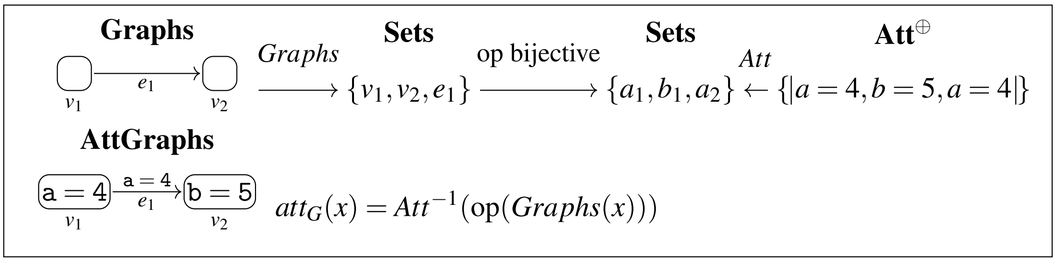

Graphs
e_1
v_1
v_2
Sets
Sets
Att$^\oplus$
$Graphs$
op bijective
Att
$\xrightarrow{Graphs} \{v_1,v_2,e_1\} \xrightarrow{\text{op bijective}} \{a_1,b_1,a_2\} \xleftarrow{Att} \{|a=4,b=5,a=4|\}$
AttGraphs
a = 4
a = 4
e_1
b = 5
v_1
v_2
$att_G(x) = Att^{-1}(\text{op}(Graphs(x)))$

Figure 8: Example of an object of **AttGraphs**

Figure 9 shows a rule containing the same graph. Note that, unlike the labelling function in partially labelled graphs, op *does not change, instead the attribute collections themselves are transformed. For this reason we may need multiple instances of these attribute collections, hence the use of a multiset. The elements of the multiset* **Att**$^\oplus$ *are in turn transformed in a way analogous to these attributed graphs.*

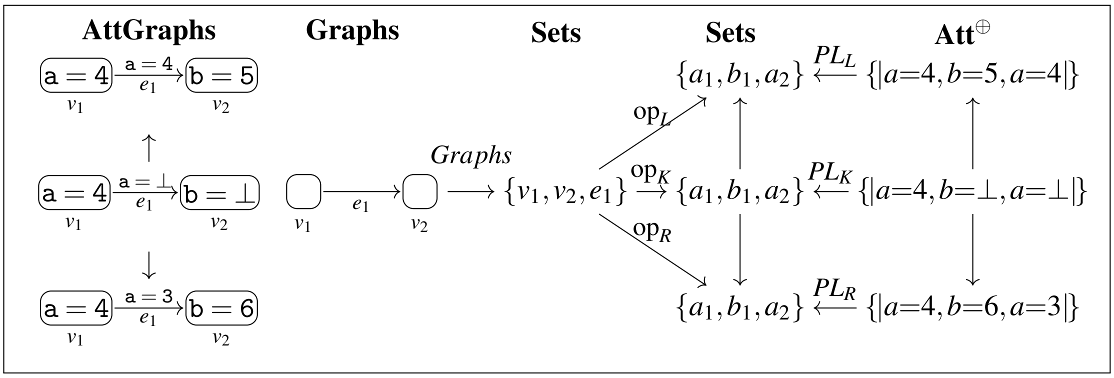


Figure 9: Example rule over **AttGraphs**

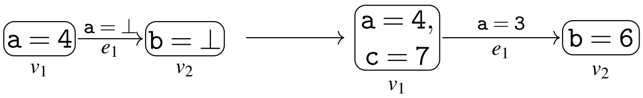


Figure 10: Matches need not specify *all* attributes

AttGraphs versus PLGraphs We based our approach on partially labelled graphs and we have to consider whether we have actually gained anything with this new category. We could, for example, define partially labelled graphs over an alphabet of sequences of $\langle name, value\rangle$-pairs which would also allow us to model graphs with attributes (this is close to the approach to attribution in [PP12]). In contrast to labels in **PLGraphs**, the attribute collections in **AttGraphs** have an internal structure. This allows us to write rules which specify only those attributes that we want to change, such as in Figure 10. Unlike in a sequence of attributes, named attributes can be deleted without implicitly changing the meaning of other, following attributes.

On Attributed Graph Transformation So far we have only considered rules that use concrete attribute values. In practice we would want to have more abstract rules with variables, such that a rule may e.g. compute the sum of two attributes and write the result to a third attribute. To allow computations in our rules we could adopt rule schemata as in e.g. [PP12]. A rule schema may contain variables or terms in place of attribute values. These variables are instantiated during matching and the terms evaluated, resulting in a concrete rule that we can handle with our approach. Alternatively we would have to prove that relevant algebra form an $\mathscr{M}, \mathscr{N}$-adhesive category to be able to handle computations directly, such as in [Gol12] or [DEPR14].

5 Related Concepts

Throughout the literature, various versions of adhesive and quasiadhesive, weak adhesive HLR, partial map adhesive, and $\mathscr{M}$-adhesive categories exist. In [EGH10], all these categories are shown to be also $\mathscr{M}$-adhesive ones. The categories of labelled graphs, typed graphs, and typed attributed graphs in [EEPT06], are known to be $\mathscr{M}$-adhesive categories if one chooses $\mathscr{M}$ to be the class of injective graph morphisms [EGH10]. Each such category induces a class of $\mathscr{M}$-adhesive systems for which several classical results of the double-pushout approach hold.

Unfortunately, the framework of $\mathscr{M}$-adhesive systems does not cover graph transformation with relabelling. In [HP12], the authors generalize $\mathscr{M}$-adhesive categories to $\mathscr{M},\mathscr{N}$-adhesive categories, where $\mathscr{N}$ is a class of morphisms containing the vertical morphisms in double-pushouts, and show that the category of partially labelled graphs is $\mathscr{M},\mathscr{N}$-adhesive, where $\mathscr{M}$ and $\mathscr{N}$ are the classes of injective and injective, undefinedness-preserving graph morphisms, respectively. Independently, Golas [Gol12] provided a general framework for attributed objects, so-called $\mathscr{W}$-adhesive systems which allows undefined attributes in the interface of a rule to change attributes, which is similar to relabelling. By Lemma 1 and Theorem 5 in [PH15], the hierarchy of adhesive categories in [EGH10] is extended in the following way:

$$\text{adhesive} \underset{\not\Leftarrow}{\overset{\Rightarrow}{\quad}} \text{adhesive HLR} \underset{\not\Leftarrow}{\overset{\Rightarrow}{\quad}} \mathscr{M}\text{-adhesive} \underset{\not\Leftarrow}{\overset{\Rightarrow}{\quad}} \mathscr{M},\mathscr{N}\text{-adhesive} \underset{\not\Leftarrow}{\overset{\Rightarrow}{\quad}} \mathscr{W}\text{-adhesive}$$

Composition of adhesive categories has previously been considered for $\mathscr{M}$-adhesive categories: the standard constructions of product, slice and coslice, functor, and comma categories are given in [EEPT06].

Parisi-Presicce et al. [PEM87] impose a simple structure on the sets of labels to allow variables both graphs and rules. As special case, they consider a preordering on the sets of labels to allow partially labelled graphs, i.e., partial labelling functions, and to allow the label-preserving matching morphisms. It is easy to see that this situation is a special case of [PEM87], 3.2 and 3.4 obtained by imposing a flat ordering on the labelling alphabet with a new "label" as top element. As far as we see, this is the first time where a special class of matching morphisms is considered.

In the literature, there are several variants of attribution concepts, e.g.:

Ehrig et al. Ehrig et al. [EEPT06] introduce typed attributed graphs, expanding the graph by including an algebra for attribute values. To facilitate attribution, typed attributed graphs extend graphs by attribution nodes and attribution edges. All possible data values of the algebra are assumed to be part of the graph. Nodes and edges are attributed by adding an attribution edge that leads to an attribution node. In contrast to typed attributed graphs our attributed graphs can have at most one value for an attribute. We constructed untyped graphs and even the attributes themselves have no types. Typed attributed graphs require that the graph is typed and thus do not allow e.g. the addition of attributes to a node or edge.

Poskitt and Plump. Poskitt and Plump [PP12] use a different approach to attribution. Here labels are replaced by sequences of attributes. Rules are complemented by rule schemata in which terms over the attributes are specified. These variables are substituted with attribute values and evaluated during rule application. Adding or removing attributes is possible, since the graphs are not typed. It is, however, not possible to find a match without fully specifying other, potentially uninteresting, attributes.

Golas. Attributed structures, as presented by Golas [Gol12], are likely the closest to our approach. Attributed structures can be defined over arbitrary $\mathscr{M}$-adhesive categories but are strictly limited to attribution. Only some of the results established for $\mathscr{M},\mathscr{N}$-adhesive transformation

Source	Typing	Add/Remove Attributes	Computation	Results	
[EEPT06]	mandatory	-	via algebra	$\mathcal{M}$	LCR, Par, Con, Amalg
[PP12]	none	yes	via rule schemata	$\mathcal{M},\mathcal{N}$	LCR, Par, Con
[Gol12]	mandatory	-	via algebra	$\mathcal{W}$	LCR
[DEPR14]	none	yes	via algebra	sesqui-PO	LCR
this paper	optional	yes	not considered	$\mathcal{M},\mathcal{N}$	LCR, Par, Con

Table 1: Comparison of Attribution Concepts

systems have been proven for these attributed structures. Since attributes are added to elements of the underlying structure based on a type, addition or removal of attributes is not possible.

Duval et al. Duval et al. [DEPR14] base their approach on sesqui-pushout rewriting to allow for cloning and merging of nodes. The addition or removal of attributes is also possible.

In most of these approaches attributes are based on an algebra that allows performing some computations on the attributes, in our setting this would require additional work to prove $\mathcal{M},\mathcal{N}$-adhesiveness for a suitable category. Fortunately we only need to provide this proof for such attributes once, enabling us to construct many different attributed structures without concerning ourselves with e.g. the underlying graphs. Alternatively we could follow the approach of [PP12] and introduce rule schemata to allow for computations over attributes.

The comparison is summarized in Table 1, based on the necessity of typing the graph, the ability to add or remove attributes via rules, whether computations over attribute values are supported and what properties have been shown for these approaches, where LCR is the Local-Church-Rosser Theorem, Par is the Parallelism Theorem, Con is the Concurrency Theorem and Amalg is the Amalgamation Theorem.

Further approaches are e.g. **Löwe et al.** [LKW93] which views graphs as a special case of algebras. These algebras can then additionally specify types for attributes. **Kastenberg and Rensink.** [KR12] take a similar approach to [EEPT06], but instead of only encoding the data values, operations and constants are also included in the graph.

6 Conclusion

In this paper, we have continued the work on $\mathcal{M},\mathcal{N}$-adhesive categories and have presented several examples (see Table 2).

The main contributions of the paper are the following:

(1) Closure results for $\mathcal{M},\mathcal{N}$-adhesive categories: product, slice/coslice, functor, comma, string, and multiset categories.

(2) A new concept of attributed graphs with a proof of $\mathcal{M},\mathcal{N}$-adhesiveness.

(3) An application to transformation systems saying that for these attributed graphs the Local Church-Rosser Theorem, the Parallelism Theorem and the Concurrency Theorem hold provided that the HLR^{+}-properties [HP12] are satisfied.

Category	Structures	Adhesiveness	Reference
Sets	sets	$\mathcal{M}$-adhesive	[EEPT06]
PL	sets of labels	$\mathcal{M},\mathcal{N}$-adhesive	Lemma 6
Att	attribute collections	$\mathcal{M},\mathcal{N}$-adhesive	Lemma 9
Graphs	unlabelled graphs	$\mathcal{M}$-adhesive	[EEPT06]
LGraphs	labelled graphs	$\mathcal{M}$-adhesive	[Ehr79]
PLGraphs	partially labelled graphs	$\mathcal{M},\mathcal{N}$-adhesive	[HP12, PH15]
AttGraphs	attributed graphs	$\mathcal{M},\mathcal{N}$-adhesive	Theorem 4

Table 2: Examples of $\mathcal{M},\mathcal{N}$-adhesive categories

Further topics might be:

(1) Proof of the HLR$^+$-properties for the category **AttGraphs** to obtain the Local Church-Rosser Theorem, the Parallelism Theorem and the Concurrency Theorem for this type of attributed graphs.

(2) Generalization of the approach to systems with so-called left-linear rules, i.e., rules where only the left morphism of the rule is required to be in $\mathcal{M}$ as, e.g., in [BGS11].

Acknowledgements. We would like to thank the anonymous referees as well as Wolfram Kahl and Berthold Hoffmann for helpful feedback on earlier versions of this paper.

Bibliography

[Awo10] S. Awodey. *Category Theory*. Oxford University Press, 2010.

[BGS11] P. Baldan, F. Gadducci, P. Sobociński. Adhesivity Is Not Enough: Local Church-Rosser Revisited. In *Mathematical Foundations of Computer Science (MFCS 2011)*. Lecture Notes in Computer Science 6907, pp. 48–59. 2011.

[DEPR14] D. Duval, R. Echahed, F. Prost, L. Ribeiro. Transformation of Attributed Structures with Cloning. In *Fundamental Approaches to Software Engineering*. Lecture Notes in Computer Science 8411, pp. 310–324. 2014.

[EEGH15] H. Ehrig, C. Ermel, U. Golas, F. Hermann. *Graph and Model Transformation - General Framework and Applications*. Monographs in Theoretical Computer Science. Springer, 2015.

[EEKR99] H. Ehrig, G. Engels, H.-J. Kreowski, G. Rozenberg (eds.). *Handbook of Graph Grammars and Computing by Graph Transformation*. Volume 2: Applications, Languages and Tools. World Scientific, 1999.

[EEPT06] H. Ehrig, K. Ehrig, U. Prange, G. Taentzer. *Fundamentals of Algebraic Graph Transformation*. EATCS Monographs of Theoretical Computer Science. Springer, 2006.

[EGH10] H. Ehrig, U. Golas, F. Hermann. Categorical Frameworks for Graph Transformation and HLR Systems based on the DPO Approach. *Bulletin of the EATCS* 112:111–121, 2010.

[Ehr79] H. Ehrig. Introduction to the Algebraic Theory of Graph Grammars. In *Graph-Grammars and Their Application to Computer Science and Biology*. Lecture Notes in Computer Science 73, pp. 1–69. 1979.

[EK80] H. Ehrig, H.-J. Kreowski. Applications of Graph Grammar Theory to Consistency, Synchronization and Scheduling in Data Base Systems. *Information Systems* 5:225–238, 1980.

[EKMR99] H. Ehrig, H.-J. Kreowski, U. Montanari, G. Rozenberg (eds.). *Handbook of Graph Grammars and Computing by Graph Transformation*. Volume 3: Concurrency, Parallelism, and Distribution. World Scientific, 1999.

[Gol12] U. Golas. A General Attribution Concept for Models in $\mathcal{M}$-adhesive Transformation Systems. In *Graph Transformations (ICGT'12)*. Lecture Notes in Computer Science 7562, pp. 187–202. 2012.

[HP12] A. Habel, D. Plump. $\mathcal{M},\mathcal{N}$-adhesive Transformation Systems. In *Graph Transformations (ICGT 2012)*. Lecture Notes in Computer Science 7562, pp. 218–233. 2012. Long version available at: http://formale-sprachen.informatik.uni-oldenburg.de/pub/index.html.

[KR12] H. Kastenberg, A. Rensink. Graph Attribution Through Sub-Graphs. Technical report TR-CTIT-12-27, University of Twente, 2012.

[LKW93] M. Löwe, M. Korff, A. Wagner. An Algebraic Framework for the Transformation of Attributed Graphs. In *Term Graph Rewriting: Theory and Practice*. Pp. 185–199. John Wiley, 1993.

[PEM87] F. Parisi-Presicce, H. Ehrig, U. Montanari. Graph Rewriting with Unification and Composition. In *Graph Grammars and Their Application to Computer Science*. Lecture Notes in Computer Science 291, pp. 496–514. 1987.

[PH15] C. Peuser, A. Habel. Attribution of Graphs by Composition of $\mathcal{M},\mathcal{N}$-adhesive Categories. In *Proceedings of the 6th International Workshop on Graph Computation (GCM 2015)*. CEUR Workshop Proceedings 1403, pp. 66–81. 2015.

[PP12] C. M. Poskitt, D. Plump. Hoare-Style Verification of Graph Programs. *Fundamenta Informaticae* 118(1-2):135–175, 2012.

[Roz97] G. Rozenberg (ed.). *Handbook of Graph Grammars and Computing by Graph Transformation*. Volume 1: Foundations. World Scientific, 1997.

[SI13] D. Singh, A. I. Isah. A Note on Category of Multisets (MUL). *International Journal of Algebra* 7(2):73–78, 2013.