Structural Decomposition of Reactions
of Graph-Like Objects

Tobias Heindel

Abteilung für Informatik und angewandte Kognitionswissenschaft
Universität Duisburg-Essen, Duisburg, Germany

Inspired by decomposition problems in rule-based formalisms in Computational Systems Biology and recent work on compositionality in graph transformation, this paper proposes to use *arbitrary* colimits to "deconstruct" models of reactions in which states are represented as objects of adhesive categories. The fundamental problem is the decomposition of complex reactions of large states into simpler reactions of smaller states.

The paper defines the *local decomposition problem* for transformations. To solve this problem means to "reconstruct" a given transformation as the colimit of "smaller" ones where the shape of the colimit and the decomposition of the source object of the transformation are fixed in advance. The first result is the soundness of colimit decomposition for arbitrary double pushout transformations in any category, which roughly means that several "local" transformations can be combined into a single "global" one. Moreover, a solution for a certain class of local decomposition problems is given, which generalizes and clarifies recent work on compositionality in graph transformation.

Introduction

Compositional methods for the synthesis and analysis of computational systems remain a fruitful research topic with potential applications in practice. Though compositionality is most clearly exhibited in semantics for process calculi where structural operational semantics (SOS) can be found in its "pure" form, a slightly broader perspective is appropriate to make use of the fundamental ideas of SOS in interdisciplinary research.

The first source of inspiration of the present paper is the κ-calculus [6], which is an influential modelling framework in Computational Systems Biology. The κ-calculus allows to give abstract, formal descriptions of biological systems that can be used to explain the reaction (rate) of complex systems, so-called *complexes*, in terms of the reaction (rate) of each of its subsystems, which are called *partial complexes*. Leaving quantitative aspects as a topic for future research, we concentrate on a specific sub-problem, namely the "purely structural" decomposition of reactions.

In the κ-calculus, system states are composed of partial complexes and they have an intuitive, *graphical* representation. Hence, it is natural to investigate the decomposition of (reactions of) system states using concepts from graph transformation. In its simplest form, the idea of composition of graph transformations is by means of coproducts. Intuitively, the coproduct of two graphs models the assembly of two states put side by side and the two (sub-)states react independently of each other. A well-known, related theorem about graph transformations is the so-called Parallelism Theorem (see e.g. [5, Theorem 17]).

A more general formalism of compositionality that is based on pushouts has been (re-)considered in [18]. In the latter work, system states are constructed as pushouts in categories of graphs, and more generally states are modelled as objects in adhesive categories [13] (for related work in graph transformation on amalgamation and distribution see [3] and also [5, Section 6.2]). Roughly, (de-)composition via pushouts amounts to "splitting" a state into two sub-states that are glued together along a common

interface. This composition mechanism behaves very much like the parallel composition in name passing calculi; in the composed process $P|Q$, the common interface of the two processes P and Q is essentially the intersection of their free names.

In this paper, we shall remove the restriction to pushouts as a composition mechanism and generalize the results of [18] from pushouts to (pullback stable) colimits of arbitrary shape. This considerably enlarges the set of available gluing patterns. As a simple example, we can now equip each sub-state with several interfaces; this would be appropriate for the model of a cell in an organism that is in direct contact with each of its neighbouring cells with some part of its membrane; each area of contact would be modelled by a different interface.

Content of the paper   After reviewing some basic category theoretical concepts and the definition of adhesive categories in Section 1, we begin Section 2 with the "deconstruction" of models of system states; more precisely, we explain in Section 2.1 how suitably finite objects in adhesive categories arise as the colimit of a diagram of "atomic" objects, namely irreducible objects in the sense of [1]. We further give a short introduction to transformations of graph-like structures in Section 2.2 and review double pushout transformations [8] as an transformation approach that – in principle – can be used in any category. How the "deconstruction" of objects can be lifted to transformations is described in Section 2.3, and we conclude Section 2 with a soundness result, which makes precise in what sense a family of "local" double pushout transformations can be composed to a "global" transformation.

The main problem, which is concerned with the decomposition of a "global" transformation into a family of "local" ones, is addressed in Section 3. We give a formal description of *local decomposition problems*, which consist of a *given* decomposition of a state (as a colimit of a certain shape) and a rule that describes a possible reaction of the state; to solve such a problem means to extend the decomposition of the state to a decomposition of the whole reaction (using colimits of the same shape). Section 3.1 presents a "global" solution, which first constructs the whole transformation "globally"; a "more local" solution of the problem is possible if we are given extra information that involve a generalization of the accommodations of [18]. At the end of Section 3, we present the *Accommodated Completeness Theorem*, which gives a partial solution to the local decomposition problem and generalizes the corresponding result of [18]. In Section 4, we conclude with a summary of our contributions, comment on related work, and point out directions for future research.

1   Preliminaries

After fixing notation for categories, functors, natural transformations and colimits, we recall the definition of adhesive categories and give definitions of further basic concepts; the standard reference for category theory is [16], [17] is a very accessible introduction.

Categories will be denoted by blackboard letters, $\mathbb{C}, \mathbb{D}$; we write '$X \in \mathbb{C}$' to express that X is an object of $\mathbb{C}$ and '$f : X \to Y$' in $\mathbb{C}$ if f is an arrow with domain X and codomain Y. The identity on an object X is $\mathrm{id}_X : X \to X$, and the composition of two morphisms $f : X \to Y$ and $g : Y \to Z$ is $g \circ f : X \to Z$.

We usually use calligraphic letters to denote functors, e.g. $\mathscr{F} : \mathbb{C} \to \mathbb{D}$ is a functor from the category $\mathbb{C}$ to the category $\mathbb{D}$. Finally, we use Greek letters to denote natural transformations; hence the expression '$\tau : \mathscr{F} \to \mathscr{G}$' is used to indicate that τ is a natural transformation with domain $\mathscr{F}$ and codomain $\mathscr{G}$, where $\mathscr{F}, \mathscr{G} : \mathbb{C} \to \mathbb{D}$ are functors. Given two categories $\mathbb{C}, \mathbb{D}$, the functor category that has functors from $\mathbb{C}$ to $\mathbb{D}$ as objects and natural transformations between them as morphisms is denoted by $[\mathbb{C}, \mathbb{D}]$.

A central notion of this paper are colimits; we follow the exposition in [16].

$$\mathscr{D}, \mathscr{E} : \mathbb{J} \to \mathbb{C} \qquad \begin{array}{c} \mathscr{D} \\ \delta \downarrow \\ \Delta D \end{array} \qquad \begin{array}{ccc} \mathscr{D} & & \\ \delta \downarrow & \searrow^{\gamma} & \\ \Delta \operatorname{col} \mathscr{D} & \dashrightarrow_{\Delta u} & \Delta C \end{array} \qquad \begin{array}{ccc} \mathscr{D} & \xrightarrow{\tau} & \mathscr{E} \\ \delta \downarrow & & \downarrow \eta \\ \Delta \operatorname{col} \mathscr{D} & \dashrightarrow_{\Delta \operatorname{col} \tau} & \Delta \operatorname{col} \mathscr{E} \end{array}$$

Figure 1: Functors, cocones and colimits

Definition 1 (Cocones and colimits). *Let $\mathscr{D}, \mathscr{E} : \mathbb{J} \to \mathbb{C}$ be functors. A* cocone *for $\mathscr{D}$ consists of an object $D \in \mathbb{C}$ and a natural transformation $\delta : \mathscr{D} \to \Delta D$ where $\Delta D : \mathbb{J} \to \mathbb{C}$ is the functor that maps each object $i \in \mathbb{J}$ to D and each morphism $f : i \to j$ in $\mathbb{J}$ to id_D, the identity on D.*

A cocone $\delta : \mathscr{D} \to \Delta D$ is a colimit *if for any other cocone $\gamma : \mathscr{D} \to \Delta C$, there is a unique morphism $u : D \to C$ in $\mathbb{C}$ such that $\gamma = \Delta u \circ \delta$ in $[\mathbb{J}, \mathbb{C}]$ where $(\Delta u)_i = u$ for each $i \in \mathbb{J}$. The object D of a colimit $\delta : \mathscr{D} \to \Delta D$ is the* colimit object *of $\mathscr{D}$ and is often referred to as* col $\mathscr{D}$.

If $\delta : \mathscr{D} \to \Delta D$ and $\eta : \mathscr{E} \to \Delta E$ are colimits and moreover $\tau : \mathscr{D} \to \mathscr{E}$ is a natural transformation, then the unique morphism $t : D \to E$ that satisfies $\eta \circ \tau = \Delta t \circ \delta$ is denoted col τ.

The notation for cocones and colimits is summarized in Figure 1. Colimits in the category of sets are particularly "well-behaved"; one relevant property is pullback stability, which we define together with cartesian transformations.

Definition 2 (Cartesian transformations and pullback stable colimits). *Let $\mathbb{C}, \mathbb{J}$ be categories, and let $\mathscr{K}, \mathscr{L} : \mathbb{J} \to \mathbb{C}$ be functors. Further let $\alpha : \mathscr{K} \to \mathscr{L}$ be a natural transformation; it is* cartesian, *if all naturality squares are pullbacks, i.e. for all morphisms $f : i \to j$ in $\mathbb{J}$, the span $\mathscr{K}_j \leftarrow \mathscr{K}_f - \mathscr{K}_i - \alpha_i \to \mathscr{L}_i$ is a pullback of $\mathscr{K}_j - \alpha_j \to \mathscr{L}_j \leftarrow \mathscr{L}_f - \mathscr{L}_i$.*

$$\begin{array}{ccc} \mathscr{K}_i & \xrightarrow{\alpha_i} & \mathscr{L}_i \\ \mathscr{K}_f \downarrow & & \downarrow \mathscr{L}_f \\ \mathscr{K}_j & \xrightarrow{\alpha_j} & \mathscr{L}_j \end{array} \qquad \begin{array}{ccc} \mathscr{H}_i & \xrightarrow{\tau_i} & \mathscr{L}_i \\ \varphi_i \downarrow & & \downarrow \lambda_i \\ H & \xrightarrow{t} & L \end{array} \qquad \begin{array}{ccc} \mathscr{H} & \xrightarrow{\tau} & \mathscr{L} \\ \varphi \downarrow & & \downarrow \lambda \\ \Delta H & \xrightarrow{\Delta t} & \Delta L \end{array}$$

Let $\lambda : \mathscr{L} \to \Delta L$ be a colimit of $\mathscr{L}$; it is pullback stable *if for any cartesian transformation $\tau : \mathscr{H} \to \mathscr{L}$ and every cocone $\varphi : \mathscr{H} \to \Delta H$ with a morphism $t : H \to L$ such that the middle diagram in the above display is a pullback for all $i \in \mathbb{J}$, the cocone $\varphi : \mathscr{H} \to \Delta H$ is actually a colimit of $\mathscr{H}$; in this situation, we say that φ is the pullback of λ along t (since the right one of the above diagrams is a pullback in $[\mathbb{J}, \mathbb{C}]$).*

The final definition on basic category theoretical notions concerns monos, which are usually denoted using a "tailed" arrow $\rightarrowtail$; very roughly, monos are a generalization of inclusions (of sets).

Definition 3 (Subobjects and proper monos). *Let $\mathbb{C}$ be a category, and let $m : M \rightarrowtail X$ and $n : N \rightarrowtail X$ be monos in $\mathbb{C}$. Then m is included in n, written $m \sqsubseteq n$, if there exists a (mono-)morphism $f : M \to N$ such that $m = n \circ f$. The* subobject *(represented by) m is the collection*

$$[m] = \Big\{ m' : M' \rightarrowtail X \;\Big|\; m' \text{ a mono such that } m \sqsubseteq m' \text{ and } m' \sqsubseteq m \Big\};$$

*moreover, $[m] \sqsubseteq [n]$ if $m \sqsubseteq n$. A* subobject *of X is a subobject $[m]$ such that the mono m has X as codomain, i.e. $m : M \rightarrowtail X$. The* subobject poset *of X, written $\mathrm{Sub}(X)$, consists of all subobjects of X and is ordered by the inclusion relation $\sqsubseteq$.*

A mono $m : M \rightarrowtail X$ in $\mathbb{C}$ is proper *if $\mathrm{id}_X \not\sqsubseteq m$, and an object $X \in \mathbb{C}$ has a proper subobject if there exist a proper mono $n : N \rightarrowtail X$ (which then represents a* proper subobject *of X).*

Systems states will be modelled as objects in adhesive categories, which have been introduced in [13] and are an established generalization of (categories of) structures that are intuitively "graph-like".

Definition 4 (Adhesive category). *A category is* adhesive *if*

- *it has pullbacks;*

- *it has pushouts along monos; i.e. a pushout of a span $B \leftarrow f - A \rightarrowtail m \rightarrow C$ exists if m is a mono; and*

- *pushouts along monos yield* Van Kampen (VK) squares, *i.e. if $B - n \rightarrow D \leftarrow g - C$ is the pushout of $B \leftarrow f - A \rightarrowtail m \rightarrow C$*
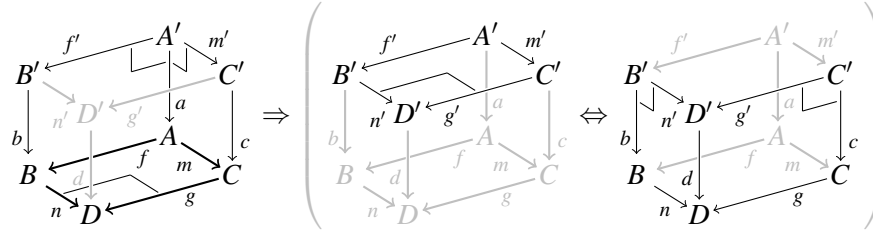
then in each commutative cube over this square as shown above on the left, which has pullbacks as back faces, the top face is a pushout if and only if the front faces are pullbacks (as illustrated above on the right).

The archetypal example of an adhesive category is the category of graphs, i.e. the functor category $[\cdot \rightrightarrows \cdot, \mathbb{SET}]$ where $\mathbb{SET}$ is the usual category of sets and functions. One common property of the category of graphs and adhesive categories is the fact that subobject posets are actually distributive lattices.

2   Colimits for composition and decomposition

Composition of General Systems by means of colimits has been proposed by Goguen [9]. Based on this idea, we shall "reconstruct" objects as colimits of their (primitive) constituents. We illustrate the idea for the simplest type of colimits, namely binary coproducts, which generalize disjoint unions of sets to objects in arbitrary categories. The question whether an object is the coproduct of two "components" amounts to asking whether the object is *connected*; in fact, a graph G is connected if and only if any "reconstruction" of G as a coproduct $G \cong H_1 + H_2$ is "trivial" in the sense that H_1 or H_2 is the empty graph.

The next more general type of colimit are fibred coproducts, i.e. pushouts. In fact, composition by pushouts of pairs of monos is the basic construction principle that is considered in recent work on compositionality of graph transformation [18]. Roughly, to decompose an object in this manner means to find an "interface subobject" such that the original object can be glued together from two (other) subobjects that overlap on the "interface subobject".

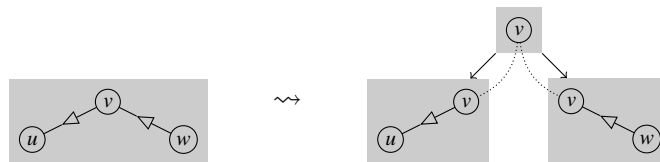A simple example of a pushout decomposition of a connected graph is given in Figure 2. The

Figure 2: Pushout decomposition of a connected graph

"interface" is just the node v, and the two edges are glued together at v; this is very much like the parallel

composition $u.v.0|v.w.0$ of the two CCS processes $u.v.0$ and $v.w.0$. However, we are not restricted to have (sets of) nodes as "interfaces".

Remark 1. *To avoid confusion, it is important to mention that the above notion of "interface subobject" seems not to be directly connected to the notion of interface that is used in techniques for the derivation of labelled transition systems for reactive systems [20] and graph transformation [7]. We hope that in the preset paper, this source of confusion is less problematic as we use arbitrary colimit shapes and not just assemblies of "interfaces". Moreover, we do not consider the possibility to change the shapes of diagrams as it is done for the case of graph transformation in [21], because we want to avoid the problem that colimits cannot be constructed componentwise.*

Next, based on results of [1], we introduce and illustrate a canonical decomposition procedure. The transformation framework that will be used later is introduced in Section 2.2 before we describe how the decomposition procedure can be lifted from objects to transformations (Section 2.3); a soundness theorem for decomposition of transformations forms the end of this section.

2.1   Decomposition of objects

To ensure that "concrete" decompositions of states and reactions do exist, we assume that each state can be modelled as an object with finitely many primitive constituents, which can be thought of as atoms, molecules or peptides – depending on whether we use metaphors from Physics, Chemistry, or Microbiology. If we use graphs as models, a canonical set of *irreducible objects* consists of nodes, loops, and edges (with distinct source and target nodes); these "atoms" of graphs are illustrated in Figure 3.

Figure 3: The three irreducible objects in the category of graphs

In UML models and informal graphical system specifications, a large family of different "graph-like" structures is used; typically, graph-like structures can either be encoded as graphs or they are instances of adhesive categories [13], which are an established framework that captures a large variety of structures that are intuitively "graph-like". Given an object (with finitely many subobjects) in an arbitrary adhesive category, its irreducible components can be characterized lattice theoretically as (the sources of) its irreducible subobjects; here, we shall use the following equivalent Definition [1].

Definition 5 (Irreducible object). *Let $\mathbb{C}$ be an adhesive category and let $X \in \mathbb{C}$ be an object; the object X is* (subobject) finite *if the lattice* $\mathrm{Sub}(X)$ *is finite.*

Let $Y \in \mathbb{C}$ be a finite object; it is an irreducible object *if*

1. *it has a proper subobject;*

2. *in every pushout $U \rightarrowtail^u Y \leftarrowtail^v V$ of a span of monos $U \leftarrowtail W \rightarrowtail V$, at most one of u and v represents a proper subobject of Y.*

A irreducible object Y is an irreducible object *or an* irreducible component *of X if there exists a mono $y \colon Y \rightarrowtail X$.*

The first requirement of this definition ensures that Y is "non-trivial"; in the case of graphs, it rules out the empty graph as an irreducible object.

Similar to the "deconstruction" of a given lattice element as a join of irreducible ones, finite objects in adhesive categories arise as the colimit of irreducible components [1, Corollary 1]. The relevant consequence in the context of the present paper is the following fact, which actually goes beyond Birkhoff's representation theorem for distributive lattices.

Fact 1 (Object decomposition). *Let $\mathbb{C}$ be an adhesive category, and let $X \in \mathbb{C}$ be a finite object with a proper subobject.*

There exists a diagram $\mathscr{X} : \mathbb{J} \to \mathbb{C}$ with a pullback stable colimit $\xi : \mathscr{X} \to \Delta X$ such that for each $j \in \mathbb{J}$, $\xi_j : \mathscr{X}_j \rightarrowtail X$ is a mono and $\mathscr{X}_j$ either is an irreducible object or it does not have a proper subobject.

Remark 2. *In fact, there are* canonical *decompositions of finite objects. In Fact 1, the role of the category $\mathbb{J}$ is played by the sub-lattice of $\mathrm{Sub}(X)$ that contains all irreducibles and the bottom element; the functor $\mathscr{X}$ chooses a representative $m \colon M \rightarrowtail X$ of each subobject $[m] \in \mathbb{J} \subseteq \mathrm{Sub}(X)$ and maps it to the domain M, which either is an irreducible object or it does not have a proper subobject.*

An example of a canonical decomposition is the graph on the left in Figure 4, which is the colimit object of the diagram of irreducible objects on the right in Figure 4 where all morphisms are inclusions.
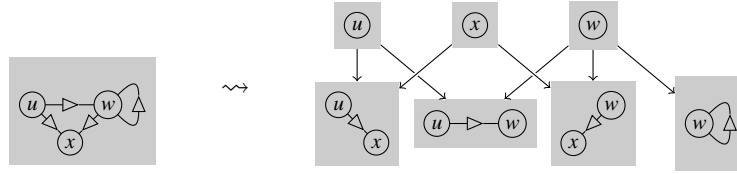

Figure 4: Canonical colimit decomposition of a graph

Summarizing, we assume objects to be finite to be able to "reconstruct" each object as a pullback stable colimit of irreducible objects. In Section 2.3, we shall describe an analogous notion of decomposition for reactions if they are modelled by one of the more common (graph) transformation approaches, which are covered by the modelling framework for reactions of systems that we introduce next.

2.2   Reactions as double square transformations

To describe reactions and their effects on complex system states, we shall use rule-based models. As a typical example of a rule, consider a pair of inclusions $L \supseteq K \subseteq R$ of sets, graphs or other graph-like objects. The reaction of a state that is represented by some object X corresponds to an *application* of this rule if the left hand side L is included in X, i.e. $L \subseteq X$, and the state after the reaction is modelled by an object Z that is obtained from X by the following construction: first, remove from X everything that is covered by L but not by K and call the intermediate result Y; second, adjoin to Y everything in R that is not covered by K. In $\mathbb{SET}$, we have the following double square diagram.

After a formalization of rules, we continue with examples, and define transformations by means of double square diagrams in any category, which will give us a fairly general modelling framework for reactions.

Definition 6 (Rule). *A rule is an arbitrary span of morphisms $L \xleftarrow{a} K \xrightarrow{b} R$, where L is the* left hand side *(LHS), R is the* right hand side *(RHS), and K is the* context *or* interface *of the rule.*

Rules are usually ranged over by letters p and q.[1] Intuitively, in any rule, the left morphism describes deletion (and cloning) actions and, dually, the right one specifies addition of new entities (and merging of
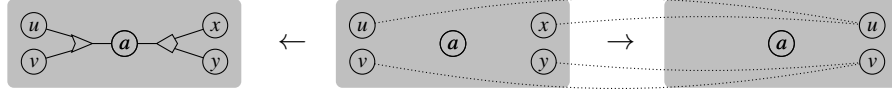
Figure 5: The single rule for reaction of solos

existing ones). As an illustrative example of a rule consider the graphical encoding of reaction in our *Mini Solos Calculus*.

Example 1 (Mini Solos Calculus). *We define a fragment of the calculus of explicit fusions [22], which in turn is closely related to the solos calculus [14].*

$$P ::= a!uv \mid a?xy \mid P|P \mid 0 \qquad\qquad P|a!uv|a?xy \to P[x/u, y/v]$$

A process is essentially a multiset of solos, *which come in two variants, namely* outputs *a!uv and* inputs *a?xy where a, u, v, x, y are elements of a countable set of names.*

　　To keep the example short, processes are considered as elements of a commutative monoid $(\mathbb{P}, |, 0)$, *all names are free, and the operational semantics is given "globally". In any process of the form P|a!uv|a?xy, the last two solos synchronize on a and "fuse" the pairs* (u, x) *and* (v, y), *which is achieved by simultaneous substitution of u for x and v for y in P (see [22] for an* SOS *style presentation, where* explicit fusions *and a sophisticated structural congruence are used).*
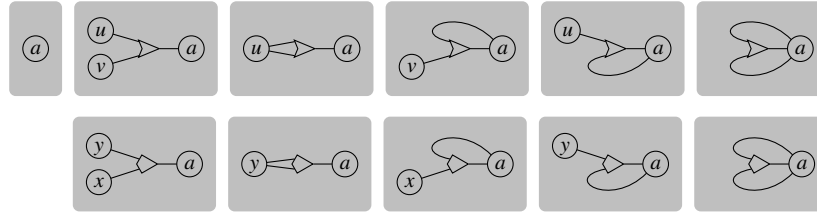

Figure 6: Irreducible objects in the graphical encoding of the Mini Solos Calculus

　　The Mini Solos Calculus has a simple, precise encoding as a graph transformation system. According to Definition 5, the irreducible objects are names and input and output solos as shown Figure 6 where darts represent outputs and kites represent inputs. The single reaction rule is given in Figure 5.

　　Independent of the choice of a particular transformation approach, a rule is applied to an object X by mapping (part of) its left hand side to X. In the more common approaches, namely double-, single- and sesqui-pushout rewriting [8, 15, 4], an application of a rule to an object X yields a transformed object Y provided that there exists a suitable double square diagram that has the same shape as the one in Figure 7.

$$
\begin{array}{ccc}
L & \xleftarrow{a} K \xrightarrow{b} & R \\
{\scriptstyle m}\downarrow & {\scriptstyle j}\downarrow & \downarrow{\scriptstyle n} \\
X & \xleftarrow{c} Y \xrightarrow{d} & Z
\end{array}
\qquad
\begin{array}{ll}
\text{rule:} & q = (L \leftarrow a- K -b\to R) \\
\text{match:} & L -m\to X \\
\text{transformation:} & X \vDash \langle q, m\rangle \Rightarrow Y
\end{array}
$$

Figure 7: Double square transformation

　　Such a diagram models the reaction of the state X, and different reaction types correspond to different rules. To avoid lengthy elaborations on the technical details of single-, double- and sesqui-pushout transformation, we use the following definition of transformation.

---

[1] The letter 'p' is taken from the word 'production'; in fact, rules are meant to generalize productions of Chomsky grammars.

Definition 7 (Double square transformation)**.** *Let $\mathbb{C}$ be any category. A* (double square) *transformation approach (in $\mathbb{C}$) is a collection $\mathbb{D}$ of commutative double square diagrams in $\mathbb{C}$, i.e. each element of $\mathbb{D}$ is a commutative diagram as shown in Figure 7.*[2]

Let $\mathbb{D}$ be a double square approach, and let $q = (L \leftarrow a\!- K -b\!\rightarrow R)$ be a rule. A morphism $m\colon L \to X$ is a match *if there exist a pair of composable morphisms $X \leftarrow c\!- Y \leftarrow j\!- K$ and a cospan $Y -d\!\rightarrow Z \leftarrow n\!- R$ such that the diagram in Figure 7 is a member of $\mathbb{D}$; in such a situation, q transforms X to Y (at the match m), and we express this by writing $X \vDash \langle q,m \rangle \Rightarrow Y$ or just $X \vDash q \Rightarrow Y$. Sometimes, we refer to '$X \vDash \langle q,m \rangle \Rightarrow Y$' as a* transformation *to implicitly refer to the double square diagram in Figure 7.*

In the case of graphs, different requirements on the exact nature of the squares correspond to different policies for the implicit deletion of dangling edges and the resolution of so-called conflicts of deletion and preservation (see [5]). The most well-known instance is the double pushout approach to rewriting [8], which is also the approach that is used in [18].

Definition 8 (Double pushout approach)**.** *Let $\mathbb{C}$ be any category.*

The double pushout approach*, denoted by $\mathbb{DPO}$, consists of all double square diagrams that comprise two pushout squares in $\mathbb{C}$, i.e. $\mathbb{DPO}$ contains all diagrams of the form that is illustrated to the right; the angles indicate that $L -m\!\rightarrow X \leftarrow c\!- Y$ is a pushout of $L \leftarrow a\!- K -j\!\rightarrow Y$ and similarly $Y -d\!\rightarrow Z \leftarrow n\!- R$ is the pushout of $Y \leftarrow j\!- K -b\!\rightarrow R$.*

$$
\begin{array}{ccccc}
L & \xleftarrow{a} & K & \xrightarrow{b} & R \\
{\scriptstyle m}\downarrow & & {\scriptstyle j}\downarrow & & \downarrow{\scriptstyle n} \\
X & \xleftarrow{c} & Y & \xrightarrow{d} & Z
\end{array}
$$

2.3 Decomposition of reactions

If we use a double square approach to model reactions, we can apply the notion of object decomposition from Section 2.1 to each object in a double square diagram and obtain a notion of decomposition for transformations. If we restrict to double pushout rewriting, we can generalize the soundness theorem of [18], which ensures that families of "local" transformations can be composed to obtain a "global" transformation; the complementary (partial) completeness theorem of [18] will be discussed in Section 3.

We start with a formal definition of decompositions of double square diagrams.

Definition 9 (Transformation decomposition)**.** *Let $\mathbb{D}$ be a double square transformation approach in $\mathbb{C}$ and let $\mathbb{J}$ be a category; moreover, let the left one of the following diagrams be a member of $\mathbb{D}$.*

$$
\begin{array}{ccccc}
L & \xleftarrow{a} & K & \xrightarrow{b} & R \\
{\scriptstyle m}\downarrow & & {\scriptstyle j}\downarrow & & \downarrow{\scriptstyle n} \\
X & \xleftarrow{c} & Y & \xrightarrow{d} & Z
\end{array}
\qquad \rightsquigarrow \qquad
\begin{array}{ccccc}
\mathscr{L} & \xleftarrow{\alpha} & \mathscr{K} & \xrightarrow{\beta} & \mathscr{R} \\
{\scriptstyle \mu}\downarrow & & {\scriptstyle \iota}\downarrow & & \downarrow{\scriptstyle \nu} \\
\mathscr{X} & \xleftarrow{\gamma} & \mathscr{Y} & \xrightarrow{\delta} & \mathscr{L}
\end{array}
\qquad
\begin{array}{ccccc}
\mathscr{L}_i & \xleftarrow{\alpha_i} & \mathscr{K}_i & \xrightarrow{\beta_i} & \mathscr{R}_i \\
{\scriptstyle \mu_i}\downarrow & & {\scriptstyle \iota_i}\downarrow & & \downarrow{\scriptstyle \nu_i} \\
\mathscr{X}_i & \xleftarrow{\gamma_i} & \mathscr{Y}_i & \xrightarrow{\delta_i} & \mathscr{L}_i
\end{array}
$$

A $\mathbb{J}$-decomposition of this diagram is a commutative double square diagram in $[\mathbb{J}, \mathbb{C}]$, as shown above in the middle, such that the right hand diagram in the above display belongs to $\mathbb{D}$ for each $i \in \mathbb{J}$, and moreover the diagram in the functor category $[\mathbb{J}, \mathbb{C}]$ that is shown in Figure 8 commutes where all cocones are colimits, i.e. $\lambda, \kappa, \rho, \xi, \upsilon$, and ζ are colimits.

Roughly, to decompose a double square diagram means to find a double square diagram in the relevant functor category such that the original diagram is induced by taking colimits of the six functors. The following example uses the category $\cdot \leftarrow \cdot \rightarrow \cdot$ for $\mathbb{J}$ and $\mathbb{C} = [\cdot \rightrightarrows \cdot, \mathbb{SET}]$ is the category of graphs.

---

[2] In other words, a transformation approach is essentially a full subcategory $\mathbb{D}$ of the functor category $[\square\square, \mathbb{C}]$ where $\square\square$ is the obvious category with six objects and fifteen morphisms – including identities.
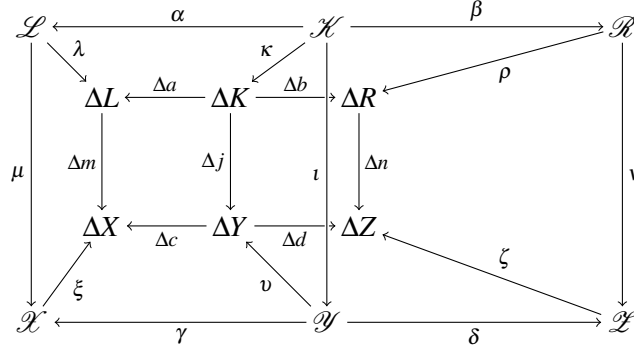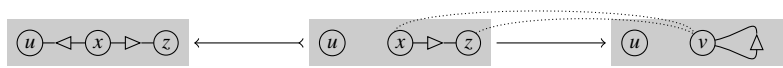
Figure 8: Colimit decomposition of a double square diagram

Example 2 (Decomposition of a double pushout diagram). *Consider the following small rule, which features the main actions that we have used in the Mini Solos Calculus; the rule deletes the edge from x to u and fuses the two nodes x and z, which become v.*

An application of this rule is shown at the top of Figure 9. To obtain a $(\cdot \leftarrow \cdot \rightarrow \cdot)$-decomposition, we chose
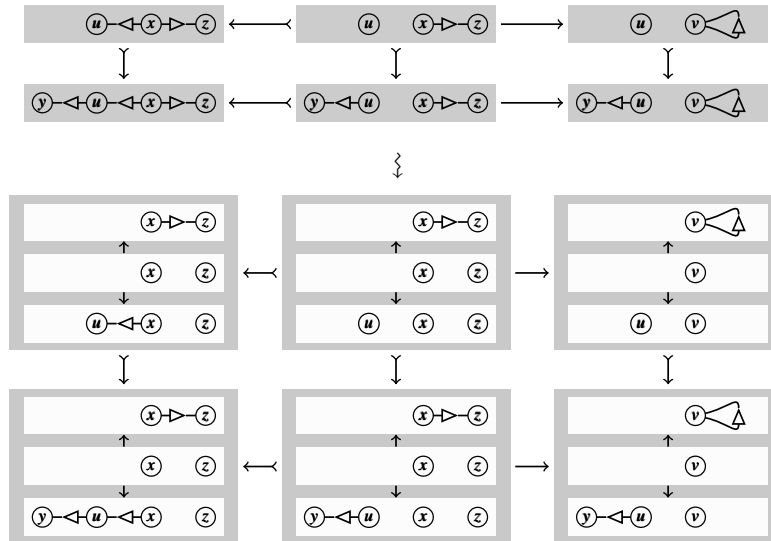

Figure 9: Pushout decomposition of a transformation

x and z in the left hand side and v – the image of x and z – in right hand side as "interfaces". The two components of each object are very roughly "everything to the left" of the "interface", and "everything to the right" of the "interface". Due to the particular choice of the objects in this example, the colimit construction, which yields the objects of the original diagram, is exactly the union of the components.

Our first theorem will be the soundness theorem for colimit decompositions of arbitrary double pushout transformations. The involved notion of soundness of a double square approach $\mathbb{D}$ can be described roughly as follows: whenever a double square diagram in the functor category $[\mathbb{J}, \mathbb{C}]$ consists of families

of diagrams in $\mathbb{D}$, we can combine the family of diagrams into one diagram in $\mathbb{D}$ by taking colimits. The formal details are as follows.

Definition 10 (Soundness). *Let $\mathbb{D}$ be a double square approach in $\mathbb{C}$; it is* sound w.r.t. colimit decomposition *if for each category $\mathbb{J}$ and each double square diagram in $[\mathbb{J}, \mathbb{C}]$ as shown on the left below*

$$
\begin{array}{ccc}
\mathscr{L} \xleftarrow{\alpha} \mathscr{K} \xrightarrow{\beta} \mathscr{R} & \qquad & \mathscr{L}_i \xleftarrow{\alpha_i} \mathscr{K}_i \xrightarrow{\beta_i} \mathscr{R}_i \\
\mu\downarrow \quad \iota\downarrow \quad \downarrow\nu & \qquad & \mu_i\downarrow \quad \iota_i\downarrow \quad \downarrow\nu_i \\
\mathscr{X} \xleftarrow{\gamma} \mathscr{Y} \xrightarrow{\delta} \mathscr{Z} & \qquad & \mathscr{X}_i \xleftarrow{\gamma_i} \mathscr{Y}_i \xrightarrow{\delta_i} \mathscr{Z}_i
\end{array}
$$

such that the right hand diagram in the above display belongs to $\mathbb{D}$ for each $i \in \mathbb{J}$ and colimits of $\mathscr{L}, \mathscr{K}, \mathscr{R}, \mathscr{X}, \mathscr{Y},$ and $\mathscr{Z}$ exist, the induced diagram, which is shown below, also belongs to $\mathbb{D}$.

$$
\begin{array}{ccc}
\operatorname{col}\mathscr{L} \xleftarrow{\operatorname{col}\alpha} \operatorname{col}\mathscr{K} \xrightarrow{\operatorname{col}\beta} \operatorname{col}\mathscr{R} \\
\operatorname{col}\mu\downarrow \qquad \operatorname{col}\iota\downarrow \qquad \downarrow\operatorname{col}\nu \\
\operatorname{col}\mathscr{X} \xleftarrow[\operatorname{col}\gamma]{} \operatorname{col}\mathscr{Y} \xrightarrow[\operatorname{col}\delta]{} \operatorname{col}\mathscr{Z}
\end{array}
$$

The next theorem differs from the soundness theorem of [18] in the following three ways: first we use arbitrary double pushout rewriting; second we use arbitrary colimits for decomposition; finally, we do not impose restrictions on the natural transformations α and β in Definition 9.

Theorem 1 (Soundness). *In categories that have (enough) pushouts, double pushout rewriting is sound w.r.t. colimit decomposition.*

Proof idea. If in the category of transformation, the two spans $(\operatorname{col}\mathscr{L}) \leftarrow\operatorname{col}\alpha- (\operatorname{col}\mathscr{K}) -\operatorname{col}\iota\rightarrow (\operatorname{col}\mathscr{Y})$ and $(\operatorname{col}\mathscr{Y}) \leftarrow\operatorname{col}\iota- (\operatorname{col}\mathscr{K}) -\operatorname{col}\beta\rightarrow (\operatorname{col}\mathscr{R})$ have pushouts, then the desired follows from the universal properties of pushouts an colimits. $\qquad\square$

Remark 3. *We can obtain a variation of this soundness theorem such that (one of the) resulting rule morphisms becomes a mono. (Such a result would be closer to the soundness theorem of [18].) For example, if the left morphism $\operatorname{col}\mathscr{L} \leftarrow\operatorname{col}\alpha- \operatorname{col}\mathscr{K}$ should be a mono, we would make the following additional assumptions: first, the colimit $\operatorname{col}\mathscr{L}$ must be a Van Kampen colimit [11], where Van Kampen colimits are the generalization of Van Kampen squares to colimits of arbitrary shape[3]; second, the natural transformation $\mathscr{L} \leftarrow\alpha- \mathscr{K}$ would be required to be cartesian (see Definition 2).*

This theorem shows that there are enough double square diagrams that are decomposable; the same proof idea can be used for single pushout transformations of graphs (and finite objects of locally cartesian closed adhesive categories). However, the main topic of this paper concerns the "converse" of this theorem and a more detailed analysis of situations in which a double square transformation can be decomposed.

3 The decomposition problem

To motivate the decomposition problem for transformations, assume that we have a "distributed representation" of a system state, namely an object X that is the colimit $\operatorname{col}\mathscr{X}$ of some functor $\mathscr{X}: \mathbb{J} \to \mathbb{C}$ and thus for each $j \in \mathbb{J}$ each "component" $\mathscr{X}_i$ could be stored on a different computer in a network. Moreover, assume that we know that there exists a transformation $X \vDash\langle q,m\rangle\Rightarrow Y$, e.g. since we have an

---

[3]There is also the corresponding variation of Fact 1 that yields Van Kampen colimit decompositions of objects [1, Corollary 1].

efficient (distributed) algorithm that determines that *m* is actually a match (while avoiding the construction of the whole double square diagram). Finally, we want to execute this transformation "locally" on each component $\mathscr{X}_i$ of X, e.g. because the construction of a "global" transformation $X \models \langle q,m \rangle \Rightarrow Y$ would be inefficient if each component $\mathscr{X}_i$ could instead be processed independently by one of the computers in the network. In the end, we want to find decompositions of *m* and *q* such that the "global" transformation corresponds to a family of "local" transformations. The formal description of the decomposition problem is as follows.

Definition 11 (Local decomposition problem). *Let $\mathbb{D}$ be a double square approach in $\mathbb{C}$ that is sound w.r.t. colimit decomposition; further let $\mathbb{J}$ be a category.*

A (local) $\mathbb{J}$-decomposition problem *is a triple $\langle \xi, q, m \rangle$ where $\xi \colon \mathscr{X} \to \Delta X$ is a colimit of a functor $\mathscr{X} \colon \mathbb{J} \to \mathbb{C}$, $q = (L \leftarrow\!a\!- K -\!b\!\to R)$ is a rule, and $m \colon L \to X$ is a match, i.e. $\mathbb{D}$ contains a diagram of the following form.*

$$\begin{array}{ccccc} L & \xleftarrow{\;a\;} & K & \xrightarrow{\;b\;} & R \\ \scriptstyle m \downarrow & & \scriptstyle j \downarrow & & \downarrow \scriptstyle n \\ X & \xleftarrow{\;c\;} & Y & \xrightarrow{\;d\;} & Z \end{array} \tag{1}$$

A solution *of a local $\mathbb{J}$-decomposition problem $\langle \xi, q, m \rangle$ is a pair $\langle \mu, \mathscr{Q} \rangle$ where $\mathscr{Q} = \mathscr{L} \leftarrow\!\alpha\!- \mathscr{K} -\!\beta\!\to \mathscr{R}$ is a rule in $[\mathbb{J}, \mathbb{C}]$ and $\mu \colon \mathscr{L} \to \mathscr{X}$ a natural transformation such that*

- *the three equations $m = \operatorname{col} \mu$, $a = \operatorname{col} \alpha$, and $b = \operatorname{col} \beta$ hold (for suitable choices of colimits of $\mathscr{L}, \mathscr{K}$, and $\mathscr{R}$);*

- *μ, α, and β can be completed to the double square diagram that is shown below on the left*

$$\begin{array}{ccccc} \mathscr{L} & \xleftarrow{\;\alpha\;} & \mathscr{K} & \xrightarrow{\;\beta\;} & \mathscr{R} \\ \scriptstyle \mu \downarrow & & \scriptstyle \iota \downarrow & & \downarrow \scriptstyle \nu \\ \mathscr{X} & \xleftarrow{\;\gamma\;} & \mathscr{Y} & \xrightarrow{\;\delta\;} & \mathscr{Z} \end{array} \qquad\qquad \begin{array}{ccccc} \mathscr{L}_i & \xleftarrow{\;\alpha_i\;} & \mathscr{K}_i & \xrightarrow{\;\beta_i\;} & \mathscr{R}_i \\ \scriptstyle \mu_i \downarrow & & \scriptstyle \iota_i \downarrow & & \downarrow \scriptstyle \nu_i \\ \mathscr{X}_i & \xleftarrow{\;\gamma_i\;} & \mathscr{Y}_i & \xrightarrow{\;\delta_i\;} & \mathscr{Z}_i \end{array}$$

where colimits of $\mathscr{Y}$ and $\mathscr{Z}$ exist and moreover, for each $i \in \mathbb{J}$, the right one of the above diagrams belongs to $\mathbb{D}$.

Following the terminology of [18], a rewriting approach is *complete* w.r.t. to colimit decomposition if each local decomposition problem has a (constructive) solution. Moreover, the latter work gives a partial completeness theorem: in any adhesive category, local $(\cdot \leftarrow \cdot \rightarrow \cdot)$-decomposition problems of a certain class have solutions in the double pushout approach.

In this section, two similar partial completeness results will be given. The first one applies to any sound double square approach, and in particular to single-, double- and sesqui-pushout graph transformation; its proof is based on pullback stability of colimit decompositions and the idea is to "lift" decomposition of objects by means of pullbacks. As it seems that the proof has rather the character of a brute force algorithm, we shall refine the basic proof idea to devise a "more local" method that provides solutions for a local decomposition problem with extra information. In fact, our second result will be a generalization of the completeness theorem of [18].

3.1   Global decomposition

We first present a procedure that can be used to split "global" reactions into
families of "local" ones that act on the constituents of states. The idea is to "lift"
the colimit decompositions of objects of Fact 1 to a double square transformation
by performing the following two constructions. Given a double square diagram as
shown to the right, we first assemble all involved entities of the reaction; this can be achieved by taking
the pushout of $X \leftarrow c - Y - d \rightarrow Z$, which yields an object U that represents all entities that may be involved
in the reaction that is modelled by the transformation. In a second step, we take successive pullbacks
to lift a decomposition of U to all other objects in the transformation diagram; in this way we obtain
a family of double square diagrams, namely one for each irreducible component of the object U. This
lifting procedure presupposes that the relevant double square approach is pullback stable.

Definition 12 (Pullback stability). *Let $\mathbb{C}$ be a category and let $\mathbb{D}$ be a double square approach; further
let $D \in \mathbb{D}$ be a diagram as illustrated in Figure 10. A pullback lifting of D is a double square diagram*
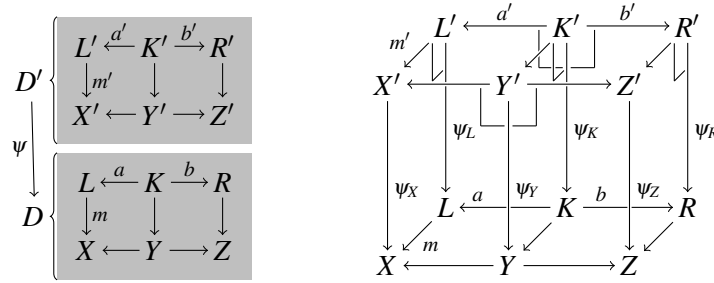

Figure 10: Pullback lifting D' of a double square diagram D

D' as shown in Figure 10 for which there exists a cartesian transformation $\psi \colon D \rightarrow D'$, *which means that
there are morphisms $\psi_L, \psi_K, \psi_R, \psi_X, \psi_Y,$ and ψ_Z that yield pullback squares in the right hand diagram in
Figure 10.*

The approach $\mathbb{D}$ is pullback stable *if any pullback lifting of any diagram in $\mathbb{D}$ is again a member of $\mathbb{D}$.*

The above mentioned common approaches to graph transformation are all pullback
stable and we can apply the following proposition to decompose transformations.

Proposition 1 (Global decomposition). *Let $\mathbb{C}$ be a category with pullbacks, let $\mathbb{D}$ be a pullback stable
double square approach that is sound w.r.t. colimit decomposition, and let the left one of the diagrams in* (2)
be a diagram in $\mathbb{D}$; moreover let $\mathbb{J}$ be a category and $\mathscr{X} \colon \mathbb{J} \rightarrow \mathbb{C}$ be a functor with colimit $\xi \colon \mathscr{X} \rightarrow \Delta X$.

$$\text{(2)}$$

*The local decomposition problem $\langle \xi, (L \leftarrow a - K - b \rightarrow R), m \rangle$ has a solution if there exists a cospan
$X - w \rightarrow U \leftarrow t - Z$ that satisfies $w \circ c = t \circ d$ and moreover there exists a diagram $\mathscr{U} \colon \mathbb{J} \rightarrow \mathbb{C}$ with a pullback
stable colimit $\varphi \colon \mathscr{U} \rightarrow \Delta U$ such that ξ is the pullback of φ along w, i.e. for each $i \in \mathbb{J}$, there exists a
morphism $\omega_i \colon \mathscr{X}_i \rightarrow \mathscr{U}_i$ that makes $\mathscr{U}_i \leftarrow \omega_i - \mathscr{X}_i - \xi_i \rightarrow X$ a pullback of $\mathscr{U}_i - \varphi_i \rightarrow U \leftarrow w - X$ (as illustrated in* (2)
on the right).

Proof sketch. Since the colimit $\mathscr{U}$ is pullback stable and the right hand diagram in (2) commutes, we can form successive pullbacks to obtain a colimit decomposition of the left hand double square diagram in (2). The desired then follows from the assumption that $\mathbb{D}$ is pullback stable. □

It is obvious, that (the proof of) this proposition is not of much use if we want to solve the local decomposition problem efficiently, e.g. in the category of graphs. Even if we construct the cospan $X - w \rightarrow U \leftarrow t - Z$ as a pushout, there are in general too many choices for suitable colimit decompositions of U. Also, this approach is fully "global" as we need to construct the whole double square diagram in $\mathbb{C}$ and do not only manipulate its components "locally".

Moreover, note that the proof of Proposition 1 only makes use of colimit decompositions of a particular type: all transformation decompositions in the functor category $[\mathbb{J}, \mathbb{C}]$ consists of cartesian transformations. Hence, if we consider only colimit decompositions of this kind, there might be a more elegant method for the solution of local decomposition problems. In fact, with some extra information on a local decomposition problem, we can devise a purely local decomposition method.

3.2   More local decompositions

We have seen that some decomposition problems can be solved by employing the notion of pullback lifting of double square diagrams. We shall now make use of the good behaviour of pushouts w.r.t. to pullbacks in adhesive categories to develop a "local" method for the solution of a simplified decomposition problem for double pushout transformations. Although the full details are somewhat technical, the central point is the following lemma, which follows directly from the definition of adhesive categories.

Lemma 1 (Double Pushout Lifting). *Let $\mathbb{C}$ be an adhesive category. Given a diagram as shown in Figure 11 on the left where a and m or a and b are mono, the bottom squares are pushouts and the vertical*
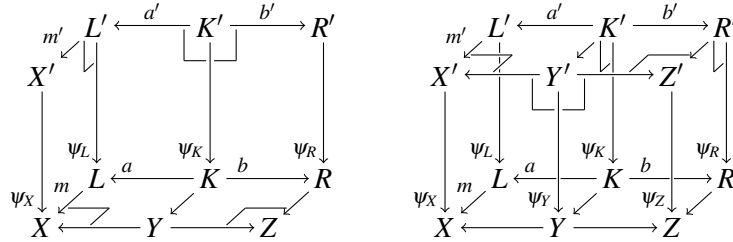

Figure 11: Double Pushout Lifting

faces are pullbacks, there exists an extension of this diagram as shown on the right in Figure 11, which has two pushout squares as top faces and all new lateral faces are pullbacks.

Proof sketch. The left hand "cube" can be constructed by taking pullbacks and thus the left square on top must be a pushout by the defining property of adhesive categories. The second pushout is obtained by taking a pushout of b' and the new morphism with domain K'. Using the definition of adhesive categories once more, we derive that the two remaining new faces are pullbacks as well. □

Note that this lemma is really directly related to the definition of adhesive categories. It is the main tool in the proof of the completeness theorem that we shall present below and it is also closely related to the accommodations that are used in [18] to obtain completeness of double pushout rewriting (w.r.t. pushout decompositions). Following the terminology of the latter work, we define a class of local decomposition problems that can be solved by means of Double Pushout Lifting.

Definition 13 (Accommodations for decomposition problems). *Let* $\langle \xi, (L \leftarrow a - K - b \rightarrow R), m \rangle$ *be a local* $\mathbb{J}$-*decomposition problem where* $\xi \colon \mathscr{X} \rightarrow \Delta X$ *and* $m \colon L \rightarrow X$.

An accommodation (scheme) *for* $\langle \xi, (L \leftarrow a - K - b \rightarrow R), m \rangle$ *is a pullback stable colimit* $\rho \colon \mathscr{R} \rightarrow \Delta R$ *such that pulling back* ρ *along* b *and pulling back* ξ *along* $m \circ a$ *yield isomorphic results, i.e. for each* $i \in \mathbb{J}$, *there is a diagram of the following form.*

$$
\begin{array}{ccccc}
 & & \mathscr{K}_i' & \longrightarrow & \mathscr{R}_i \\
 & \nearrow & & & \downarrow \rho_i \\
\mathscr{X}_i & & & & \\
\downarrow \xi_i & & \downarrow & & \downarrow \\
\downarrow & {}_m \swarrow L \xleftarrow{\ a\ } & K & \xrightarrow{\ b\ } & R \\
X \swarrow & {}_{m \circ a} & & &
\end{array}
\tag{3}
$$

In [18], attention is restricted to local $\mathbb{J}$-decomposition problems where $\mathbb{J}$ is the shape of spans, i.e. $\mathbb{J}$ is fixed to be the category $(\cdot \leftarrow \cdot \rightarrow \cdot)$; in fact, the object X in Definition 13 is assumed to arise as the pushout of a pair of monos and thus is a Van Kampen square. A reader that is familiar with the latter work, will notice that accommodation schemes correspond to suitable pairs of accommodations in the sense of [18, Definition 9] and will be able to verify that the following theorem is a generalization of the Completeness Theorem of [18]; the details are omitted due to space constraints (see also Remark 3).

Theorem 2 (Accommodated Completeness). *Let* $\mathbb{C}$ *be an adhesive category, let* $\langle \xi, (L \leftarrow a - K - b \rightarrow R), m \rangle$ *be a local* $\mathbb{J}$-*decomposition problem for the* $\mathbb{DPO}$ *approach where at least one of* m *and* b *is a mono; then* $\langle \xi, (L \leftarrow a - K - b \rightarrow R), m \rangle$ *has as solution.*

Proof sketch. First we construct successive pullbacks of ξ along m and a, which yields cartesian morphisms $\mu \colon \mathscr{L} \rightarrow \mathscr{X}$ and $\alpha \colon \mathscr{K} \rightarrow \mathscr{L}$, respectively. Since ρ is an accommodation, there is a cartesian transformation $\beta \colon \mathscr{K} \rightarrow \mathscr{R}$. The candidate for a solution is $\langle \mu, \mathscr{L} \leftarrow \alpha - \mathscr{K} - \beta \rightarrow \mathscr{R} \rangle$. Finally, we apply the Double Pushout Lifting Lemma and soundness of $\mathbb{DPO}$ to complete the proof. $\square$

If we assume that in the situation of Diagram (3) in Definition 13, the right hand side R is relatively small in comparison with the state X, the decomposition procedure that is implicitly given by the proof of Theorem 2 is more local than the global one of Section 3.1 insofar as we "only" need to find a suitable accommodation of the right hand side R of the rule $L \leftarrow a - K - b \rightarrow R$. More concretely, if we work in the category of graphs and use the canonical decompositions of Section 2.1, the object $\mathscr{K}_i'$ will be the empty graph for many $i \in \mathbb{J}$ and then the typical first candidate for $\mathscr{R}_i$ is also the empty graph; it remains to be explored how good this heuristics is in practice.

4 Conclusion

We have shown how arbitrary colimits can be used to decompose transformations of graph-like objects and have generalized the results of [18] in this setting: first, we have given a soundness theorem for double pushout rewriting that allows to combine families of "local" transformations into a single "global" one; moreover, our completeness theorem provides solutions for *accommodated* decomposition problems in adhesive categories; finally, we have given a precise formulation of a general local decomposition problem that can serve as a base for future research for double square transformation approaches that are used to model reactions of graph-like objects.

The first, immediate direction for future work is the description of the soundness (and completeness) theorem in a more standard SOS style using (an extension of) the de Simone format with terms over a

suitable signature. If we take the "closed system" perspective of the present paper and further restrict to (de-)composition by means of (binary) coproducts then the solution seems rather straightforward: if we have two "local" transitions $X_1 \vDash \langle q_1, m_1 \rangle \Rightarrow Y_i$ and $X_2 \vDash \langle q_2, m_2 \rangle \Rightarrow Y_2$ then $X_1 + X_2 \vDash \langle q_1 + q_2, m_1 + m_2 \rangle \Rightarrow Y_1 + Y_2$ is the corresponding global transformation; however, even in the simple case of coproducts it is not quite clear how the fact that the "constants" of the signature are actually interpreted as (irreducible) objects of some adhesive category features.

Besides the problem of a "classical" SOS account of the results of this paper, we are confronted with further issues. For example, it is very desirable to make the step from the closed systems perspective of double pushout rewriting to borrowed context rewriting [7, 2], which captures essential aspects of the interaction of open systems with their environment in the form of labelled transition systems that are automatically derived from rules of reaction.

Moreover, we want to reconsider the problem of reaction *rates* (which is already non-trivial in closed systems). The relevant problem in stochastic graph transformation [10] and the κ-calculus[6] are the numerous symmetries of objects that have to be considered to obtain adequate quantitative system models. To "deconstruct" these symmetries, we plan to use the canonical colimit decomposition of finite objects in adhesive categories. The goal is a distributed method for the computation of all isomorphisms of a given object and we also hope to gain fundamental insights into the "internal" structure of finite objects in adhesive categories.

References

[1] Paolo Baldan, Filippo Bonchi, Andrea Corradini, Tobias Heindel & Barbara König (To appear): *A Lattice-Theoretical Perspective on Adhesive Categories*. Journal of Symbolic Computation .

[2] Paolo Baldan, Hartmut Ehrig & Barbara König (2006): *Composition and Decomposition of DPO Transformations with Borrowed Context*. In: *Proc. of ICGT '06 (International Conference on Graph Transformation)*, Springer, pp. 153–167. LNCS 4178.

[3] Paul Boehm, Harald-Reto Fonio & Annegret Habel (1987): *Amalgamation of graph transformations: a synchronization mechanism*. Journal of Computer and System Sciences 34(2-3), pp. 377–408.

[4] Andrea Corradini, Tobias Heindel, Frank Hermann & Barbara König (2006): *Sesqui-Pushout Rewriting*. In: *Graph Transformations, Third International Conference, ICGT 2006, Natal, Rio Grande do Norte, Brazil, September 17-23, 2006, Proceedings*, pp. 30–45.

[5] Andrea Corradini, Ugo Montanari, Francesca Rossi, Hartmut Ehrig, Reiko Heckel & Michael Löwe (1997): *Algebraic Approaches to Graph Transformation – Part* I: *Basic Concepts and Double Pushout Approach*. In Rozenberg [19], pp. 163–246.

[6] Vincent Danos & Cosimo Laneve (2004): *Formal Molecular Biology*. Theoretical Computer Science 325(1), pp. 69–110.

[7] Hartmut Ehrig & Barbara König (2006): *Deriving Bisimulation Congruences in the DPO Approach to Graph Rewriting with Borrowed Contexts*. Mathematical Structures in Computer Science 16(6), pp. 1133–1163.

[8] Hartmut Ehrig, Michael Pfender & Hans Jürgen Schneider (1973): *Graph-Grammars: An Algebraic Approach*. In IEEE [12], pp. 167–180.

[9] Joseph Goguen (1973): *Categorical foundations for general systems theory*. Advances in Cybernetics and Systems Research 1.

[10] Reiko Heckel, Georgios Lajios & Sebastian Menge (2006): *Stochastic Graph Transformation Systems*. Fundamenta Informaticae 74(1), pp. 63–84.

[11] Tobias Heindel & Paweł Sobociński (2009): *Van Kampen colimits as bicolimits in Span*. In: Algebra and Coalgebra in Computer Science: Third International Conference, Calco 2009, Udine, Italy, September 7–10, 2009, Proceedings, number 5728 in LNCS, Springer, pp. 335–349.

[12] IEEE, editor (1973): *14th Annual Symposium on Foundations of Computer Science, 15-17 October 1973, The University of Iowa, USA*. IEEE Computer Society Press.

[13] Stephen Lack & Paweł Sobociński (2005): *Adhesive and Quasiadhesive Categories*. Theoretical Informatics and Applications 39(2), pp. 511–546.

[14] Cosimo Laneve, Joachim Parrow & Björn Victor (2001): *Solo Diagrams*. In: Proceedings of the 4th International Symposium on Theoretical Aspects of Computer Software, Springer-Verlag, pp. 127–144.

[15] Michael Löwe (1993): *Algebraic Approach to Single-Pushout Graph Transformation*. Theoretical Computer Science 109(1&2), pp. 181–224.

[16] Saunders Mac Lane (1971): *Categories for the Working Mathematician*. Number 5 in Graduate Texts in Mathematics. Springer.

[17] Benjamin C. Pierce (1991): *Basic Category Theory for Computer Scientists*. MIT Press.

[18] Arend Rensink (2010): *Compositionality in Graph Transformation*. In: Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide & Paul G. Spirakis, editors: ICALP (2), Lecture Notes in Computer Science 6199, Springer, pp. 309–320. Available at `http://dx.doi.org/10.1007/978-3-642-14162-1_26`.

[19] Grzegorz Rozenberg, editor (1997): *Handbook of Graph Grammars and Computing by Graph Transformations, Volume 1: Foundations*. World Scientific.

[20] Vladimiro Sassone & Paweł Sobociński (2003): *Deriving Bisimulation Congruences using 2-categories*. Nordic Journal of Computing 10(2), pp. 163–183.

[21] Gabriele Taentzer (1996): *Parallel and distributed graph transformation: Formal description and application to communication-based systems*. Ph.D. thesis, Technische Universität Berlin.

[22] Lucian Wischik & Philippa Gardner (2005): *Explicit fusions*. Theoretical Computer Science 340(3), pp. 606–630.