Transformation Systems with Incremental Negative Application Conditions

Andrea Corradini[1], Reiko Heckel[2], Frank Hermann[3], Susann Gottmann[3,⋆], and Nico Nachtigall[3,⋆]

[1] Dipartimento di Informatica, Università di Pisa, Italy
andrea@di.unipi.it
[2] University of Leicester, UK
reiko@mcs.le.ac.uk
[3] Interdisciplinary Center for Security, Reliability and Trust, Université du Luxembourg, Luxembourg
{frank.hermann,susann.gottmann,nico.nachtigall}@uni.lu

Abstract. In several application areas, Graph Transformation Systems (GTSs) are equipped with Negative Application Conditions (NACs) that specify "forbidden contexts", in which the rules shall not be applied. The extension to NACs, however, introduces inhibiting effects among transformation steps that are not local in general, causing a severe problem for a concurrent semantics. In fact, the relation of *sequential independence* among derivation steps is not invariant under switching, as we illustrate with an example. We first show that this problem disappears if the NACs are restricted to be *incremental*. Next we present an algorithm that transforms a GTS with arbitrary NACs into one with incremental NACs only, able to simulate the original GTS. We also show that the two systems are actually equivalent, under certain assumptions on NACs.

Keywords: graph transformation, concurrent semantics, negative application conditions, switch equivalence.

1 Introduction

Graph Transformation Systems (GTSs) are an integrated formal specification framework for modelling and analysing structural and behavioural aspects of systems. The evolution of a system is modelled by the application of rules to the graphs representing its states and, since typically such rules have local effects, GTSs are particularly suitable for modelling concurrent and distributed systems where several rules can be applied in parallel. Thus, it is no surprise that a large body of literature is dedicated to the study of the concurrent semantics of graph transformation systems [6,1,2].

The classical results include – among others – the definitions of parallel production and shift equivalence [15], exploited in the Church-Rosser and Parallelism theorems [7]: briefly, derivations that differ only in the order in which independent steps are applied are considered to be equivalent. Several years

⋆ Supported by the Fonds National de la Recherche, Luxembourg (3968135, 4895603).

N. Martí-Oliet and M. Palomino (Eds.): WADT 2012, LNCS 7841, pp. 127–142, 2013.

later, taking inspiration from the theory of Petri nets, deterministic processes were introduced [6], which are a special kind of GTSs, endowed with a partial order, and can be considered as canonical representatives of shift-equivalence classes of derivations. Next, the unfolding of a GTS was defined as a typically infinite non-deterministic process which summarises all the possible derivations of a GTS [4]. Recently, all these concepts have been generalised to transformation systems based on ($\mathcal{M}$-)adhesive categories [8,5,3].

In this paper, we consider the concurrent semantics of GTSs that use the concept of Negative Application Conditions (NACs) for rules [11], which is widely used in applied scenarios. A NAC allows one to describe a sort of "forbidden context", whose presence around a match inhibits the application of the rule. These inhibiting effects introduce several dependencies among transformation steps that require a shift of perspective from a purely local to a more global point of view when analysing such systems.

Existing contributions that generalise the concurrent semantics of GTSs to the case with NACs [17,10] are not always satisfactory. While the lifted Parallelism and Concurrency Theorems provide adequate constructions for composed rules specifying the effect of concurrent steps, a detailed analysis of possible interleavings of a transformation sequence leads to problematic effects caused by the NACs. As shown in [12], unlike the case without NACs, the notion of *sequential independence* among derivation steps is not stable under switching. More precisely, it is possible to find a derivation made of three direct transformations $s = (s_1; s_2; s_3)$ where s_2 and s_3 are sequentially independent and to find a derivation $s' = (s'_2; s'_3; s'_1)$ that is shift equivalent to s (obtained with the switchings $(1 \leftrightarrow 2; 2 \leftrightarrow 3)$), but where s'_2 and s'_3 are sequentially dependent on each other. This is a serious problem from the concurrent semantics point of view, because for example the standard colimit technique [6] used to generate the process associated with a derivation does not work properly, since the causalities between steps do not form a partial order in general.

In order to address this problem, we introduce a restricted kind of NACs, based on incremental morphisms [12]. We first show that sequential independence is invariant under shift equivalence if all NACs are incremental. Next we analyse to which extent systems with general NACs can be transformed into systems with incremental NACs. For this purpose, we provide an algorithmic construction *INC* that takes as input a GTS and yields a corresponding GTS with incremental NACs only. We show that the transformation system obtained via *INC* simulates the original one, i.e., each original transformation sequence induces one in the derived system. Thus, this construction provides an over-approximation of the original system. We also show that this simulation is even a bisimulation, if the NACs of the original system are obtained as colimits of incremental NACs.

In the next section we review main concepts for graph transformation systems. Sect. 3 discusses shift equivalence and the problem that sequential independence with NACs is not stable in general. Thereafter, Sect. 4 presents incremental NACs and shows the main result on preservation of independence. Sect. 5 presents the algorithm for transforming systems with general NACs into

those with incremental ones and shows under which conditions the resulting system is equivalent. Finally, Sect. 6 provides a conclusion and sketches future developments. The proofs of the main theorems are included in the paper.

2 Basic Definitions

In this paper, we use the double-pushout approach [9] to (typed) graph transformation, occasionally with negative application conditions [11]. However, we will state all definitions and results at the level of adhesive categories [16]. A category is *adhesive* if it is closed under pushouts along monomorphisms (hereafter *monos*) as well as under pullbacks, and if all pushouts along a mono enjoy the van Kampen property. That means, when such a pushout is the bottom face of a commutative cube such as in the left of Fig. 1, whose rear faces are pullbacks, the top face is a pushout if and only if the front faces are pullbacks. In any adhesive category we have uniqueness of pushout complements along monos, monos are preserved by pushouts and pushouts along monos are also pullbacks. As an example, the category of typed graphs for a fixed type graph TG is adhesive [8].

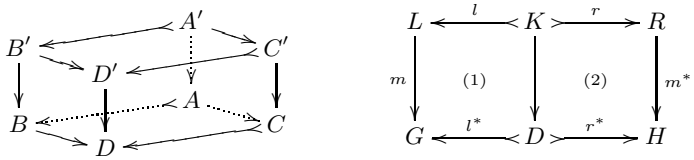


Fig. 1. van Kampen condition (left) and DPO diagram (right)

In the rest of the paper, unless differently stated, all objects and arrows live in an arbitrary but fixed adhesive category $\mathbf{C}$.

A *rule* $p = (L \overset{l}{\hookleftarrow} K \overset{r}{\rightarrowtail} R)$ consists of a span of two monos l and r. Given a morphism $m : L \to G$ called the *match*, a *direct transformation* (or *step*) $G \overset{p,m}{\Longrightarrow} H$ from G to a H exists if a double-pushout (DPO) diagram can be constructed as in the right of Fig. 1, where (1) and (2) are pushouts.

The applicability of rules can be restricted by specifying negative conditions requiring the non-existence of certain structures in the context of the match. A *(negative) constraint* on an object L is a morphism $n : L \to \hat{L}$. A morphism $m : L \to G$ satisfies n (written $m \models n$) iff there is no mono $q : \hat{L} \rightarrowtail G$ such that $n; q = m$. A negative application condition (NAC) on L is a set of constraints N. A morphism $m : L \to G$ *satisfies* N (written $m \models N$) if and only if m satisfies every constraint in N, i.e., $\forall n \in N : m \models n$.

All along the paper we shall consider only monic matches and monic constraints: possible generalisations are discussed in the concluding section.

A graph transformation system (GTS) $\mathcal{G}$ consists of a set of rules, possibly with NACs. A derivation in $\mathcal{G}$ is a sequence of direct transformations $s = (G_0 \overset{p_1,m_1}{\Longrightarrow} G_1 \overset{p_2,m_2}{\Longrightarrow} \cdots \overset{p_n,m_n}{\Longrightarrow} G_n)$ such that all p_i are in $\mathcal{G}$; we denote it also as $s = s_1; s_2; \ldots; s_n$, where $s_k = (G_{k-1} \overset{p_k,m_k}{\Longrightarrow} G_k)$ for $k \in \{1, \ldots, n\}$.

3 Independence and Shift Equivalence

Based on the general framework of adhesive categories, this section recalls the relevant notions for sequential independence and shift equivalence and illustrates the problem that independence is not stable under switching in presence of NACs. In the DPO approach, two consecutive direct transformations $s_1 = G_0 \overset{p_1,m_1}{\Longrightarrow} G_1$ and $s_2 = G_1 \overset{p_2,m_2}{\Longrightarrow} G_2$ as in Fig. 2 are *sequentially independent* if there exist morphisms $i : R_1 \to D_2$ and $j : L_2 \to D_1$ such that $j; r_1^* = m_2$ and $i; l_2^* = m_1^*$. In this case, using the local Church-Rosser theorem [8] it is possible to construct a derivation $s' = G_0 \overset{p_2,m_2'}{\Longrightarrow} G_1' \overset{p_1,m_1'}{\Longrightarrow} G_2$ where the two rules are applied in the opposite order. We write $s_1; s_2 \sim_{sh} s'$ to denote this relation.

Given a derivation $s = s_1; s_2; \ldots s_i; s_{i+1}; \ldots; s_n$ containing sequentially independent steps s_i and s_{i+1}, we denote by $s' = switch(s, i, i+1)$ the equivalent derivation $s' = s_1; s_2; \ldots s_i'; s_{i+1}'; \ldots; s_n$, where $s_i; s_{i+1} \sim_{sh} s_i'; s_{i+1}'$. Shift equivalence $\equiv_{sh}$ over derivations of $\mathcal{G}$ is defined as the transitive and "context" closure of $\sim_{sh}$, i.e., the least equivalence relation containing $\sim_{sh}$ and such that if $s \equiv_{sh} s'$ then $s_1; s; s_2 \equiv_{sh} s_1; s'; s_2$ for all derivations s_1 and s_2.

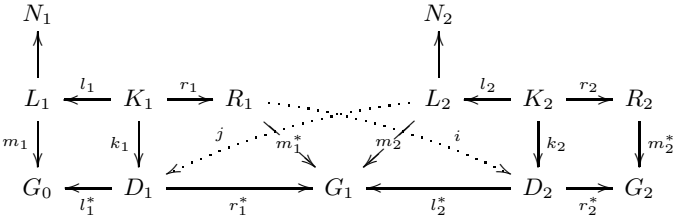


Fig. 2. Sequential independence

The definitions of independence and shift equivalence carry over to transformations with NACs [18] by requiring that the match for p_2 in G_0 given by $m_2' = j; l_1^*$ satisfies the NAC of p_2 and the induced match of p_1 into graph G_1' obtained by $G_0 \overset{p_2,m_2'}{\Longrightarrow} G_1'$ satisfies the NAC of p_1.

Throughout the paper, we use a short notation for transformation rules in our examples as for instance in Ex. 1 below. A rule p is depicted as $(L \overset{p}{\Longrightarrow} R)$ showing the left and right hand sides of p. The intermediate interface graph K containing all preserved elements can be obtained as intersection of L and R. Numbers and positions of elements indicate the mappings. If a rule has a NAC, then we depict it inside the left hand side and indicate the NAC-only elements by dotted line style. If a NAC contains more than one constraint, then they are marked by different numbers. However, this situation does not appear in any figure of this paper.

Example 1 (context-dependency of independence with NACs). Fig. 3 presents three transformation sequences starting with graph G_0 via rules $p1, p2$ and $p3$. Rule $p3$ has a NAC, which is indicated by dotted lines (one node and two edges).

In the first sequence $s = G_0 \overset{p_1,m_1}{\Longrightarrow} G_1 \overset{p_2,m_2}{\Longrightarrow} G_2 \overset{p_3,m_3}{\Longrightarrow} G_3 = (s_1; s_2; s_3)$ shown in the top of Fig. 3, steps s_1 and s_2 are sequentially independent, and so are s_2 and s_3. After switching the first and the second step we derive $s' = switch(s,\ 1, 2) = (s'_2; s'_1; s_3)$ (middle of Fig. 3) so that both sequences are shift equivalent ($s \equiv_{sh} s'$). Since s'_1 and s_3 are independent, we can perform a further switch $s'' = switch(s', 2, 3) = (s'_2; s'_3; s''_1)$ shown in the bottom sequence in Fig. 3. However, steps s'_2 and s'_3 are dependent from each other in s'', because the match for rule p_3 will not satisfy the corresponding NAC for a match into G_0. Hence, independence can change depending on the derivation providing the context, even if derivations are shift equivalent.

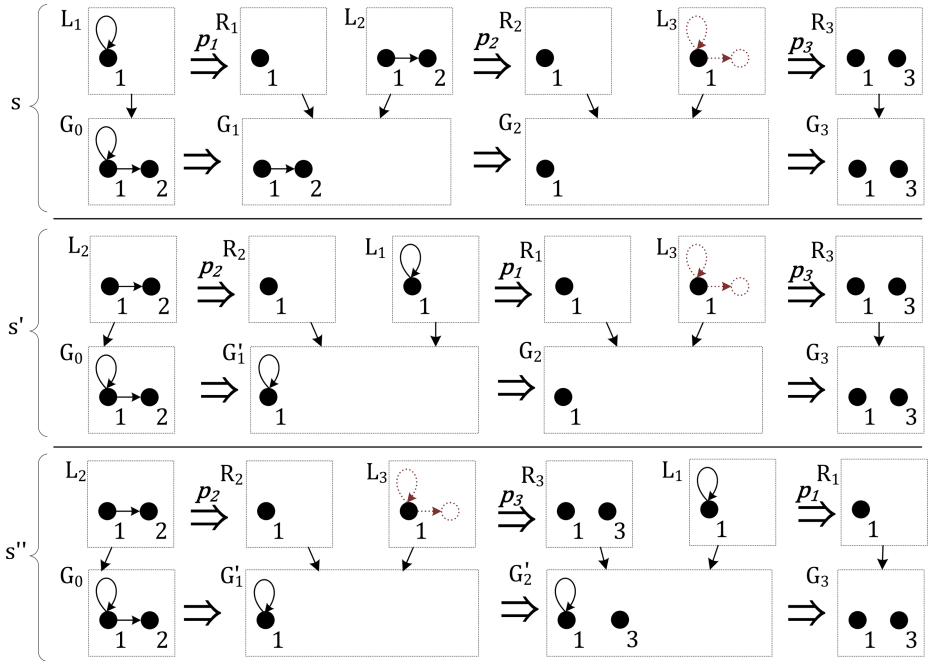


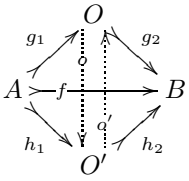
Fig. 3. Independence of p_2 and p_3 is not preserved by switching with p_1

4 Restricting to Incremental NACs

In this section we show that under certain assumptions on the NACs of the rules, the problem identified in Ex. 1 does not occur. Intuitively, for each constraint $n : L \to \hat{L}$ in a NAC we will require that it is *incremental*, i.e., that $\hat{L}$ does not

extend L in two (or more) independent ways. Therefore, if there are two different ways to decompose n, one has to be an extension of the other. Incremental arrows have been considered in [12] for a related problem: here we present the definition for monic arrows only, because along the paper we stick to monic NACs.

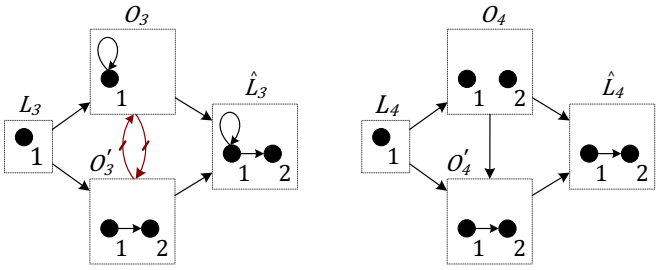
Definition 1 (incremental monos and NACs). *A mono $f\colon A \rightarrowtail B$ is called* incremental*, if for any pair of decompositions $g_1; g_2 = f = h_1; h_2$ as in the diagram below where all morphisms are monos, there is either a mediating morphism $o : O \rightarrowtail O'$ or $o' : O' \rightarrowtail O$, such that the resulting triangles commute.*



A monic NAC N over L is incremental *if each constraint $n : L \rightarrowtail \hat{L} \in N$ is incremental.*

Example 2 (Incremental NACs). The left diagram below shows that the negative constraint $n_3 : L_3 \to \hat{L}_3 \in N_3$ of rule p_3 of Ex. 1 is not incremental, because $\hat{L}_3$ extends L_3 in two independent ways: by the loop on 1 in O_3, and by the outgoing edge with one additional node 2 in O'_3. Indeed, there is no mediating arrow from O_3 to O'_3 or vice versa relating these two decompositions.

Instead the constraint $n_4 : L_4 \to \hat{L}_4 \in N_4$ of rule p_4 of Fig. 5 is incremental: it can be decomposed in only one non-trivial way, as shown in the top of the right diagram, and for any other possible decomposition one can find a mediating morphism (as shown for one specific case).

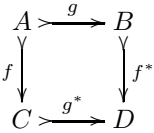


Intuitively, the problem stressed in Ex. 1 is due to the fact that rules p_1 and p_2 delete from G_0 two independent parts of the forbidden context for p_3. Therefore p_3 depends on the firing of p_1 *or* on the firing of p_2, while p_1 and p_2 are independent. This form of *or-causality* from sets of *independent* events is known to be a source of ambiguities in the identification of a reasonable causal ordering among the involved events, as discussed in [19]. The restriction to incremental NACs that we consider here is sufficient to avoid such problematic situations (as proved in the main result of this section) essentially because if both p_1 and p_2 delete from G_0 part of an incremental NAC, then they cannot be independent, since the NAC cannot be factorized in two independent ways.

Incrementality of monos enjoys some nice properties: it is preserved by decomposition of arrows, and it is both preserved and reflected by pushouts along monos, as stated in the next propositions.

Proposition 1 (decomposition of monos preserve incrementality). *Let $f\colon A \rightarrowtail B$ be an incremental arrow and $f = g;h$ with monos $g\colon A \rightarrowtail C$ and $h\colon C \rightarrowtail B$. Then both h and g are incremental.*

Proposition 2 (preservation and reflection of incrementality by POs). *In the diagram to the right, let $B \overset{f^*}{\rightarrowtail} D \overset{g^*}{\leftarrowtail} C$ be the pushout of the monic arrows $B \overset{g}{\leftarrowtail} A \overset{f}{\rightarrowtail} C$. Then f is incremental if and only if f^* is incremental.*



We come now to the main result of this section: if all NACs are incremental, then sequential independence of direct transformations is invariant with respect to the switch of independent steps.

Theorem 1 (invariance of independence under shift equivalence). *Assume transformation sequences $s = G_0 \overset{p_1,m_1}{\Longrightarrow} G_1 \overset{p_2,m_2}{\Longrightarrow} G_2 \overset{p_3,m_3}{\Longrightarrow} G_3$ and $s' = G_0 \overset{p_2,m_2'}{\Longrightarrow} G_1' \overset{p_3,m_3'}{\Longrightarrow} G_2' \overset{p_1,m_1''}{\Longrightarrow} G_3$ using rules p_1, p_2, p_3 with incremental NACs only as in the diagram below, such that $s \equiv_{sh} s'$ with $s' = switch(switch(s,1,2),2,3)$.*

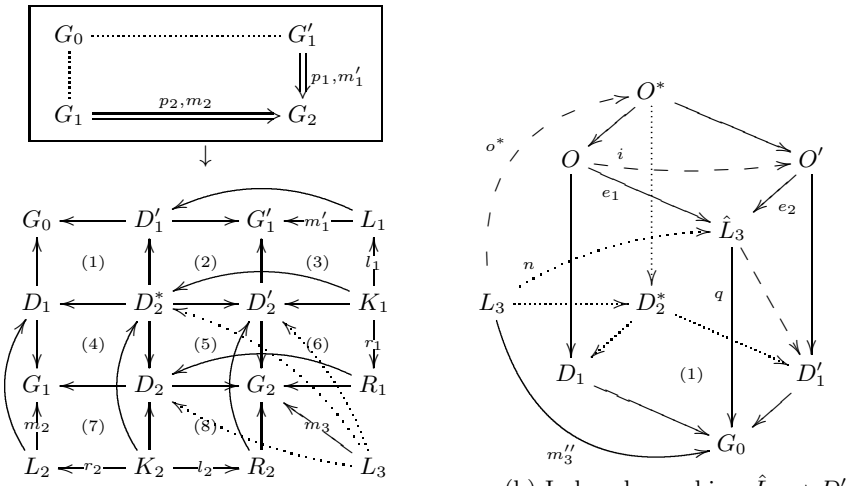
$$\begin{array}{ccccc} G_0 & \overset{p_2,m_2'}{\Longrightarrow} & G_1' & \overset{p_3,m_3'}{\Longrightarrow} & G_2' \\ \Big\Downarrow{\scriptstyle p_1,m_1} & & \Big\Downarrow{\scriptstyle p_1,m_1'} & & \Big\Downarrow{\scriptstyle p_1,m_1''} \\ G_1 & \overset{p_2,m_2}{\Longrightarrow} & G_2 & \overset{p_3,m_3}{\Longrightarrow} & G_3 \end{array}$$

Then, $G_1 \overset{p_2,m_2}{\Longrightarrow} G_2$ and $G_2 \overset{p_3,m_3}{\Longrightarrow} G_3$ are sequentially independent if and only if $G_0 \overset{p_2,m_2'}{\Longrightarrow} G_1'$ and $G_1' \overset{p_3,m_3'}{\Longrightarrow} G_2'$ are.

Proof. Let N_1, N_2 and N_3 be the NACs of p_1, p_2 and p_3, respectively. Due to sequential independence of $G_1 \overset{p_2,m_2}{\Longrightarrow} G_2$ and $G_2 \overset{p_3,m_3}{\Longrightarrow} G_3$, match $m_3 : L_3 \to G_2$ extends to a match $m_3^* : L_3 \to G_1$ satisfying N_3. Using that both m_3 and m_3^* satisfy N_3, we show below that the match $m_3'' : L_3 \to G_0$, that exists by the classical local Church-Rosser, satisfies N_3, too. This provides one half of the independence of $G_0 \overset{p_2,m_2'}{\Longrightarrow} G_1'$ and $G_1' \overset{p_3,m_3'}{\Longrightarrow} G_2'$.

By reversing the two horizontal sequences in the diagram above with the same argument we obtain the proof for the other half, i.e., that the comatch of p_2 into G_2' satisfies the equivalent right-sided NAC of N_2, which is still incremental thanks to Prop. 2. Finally reversing the vertical steps yields the reverse implication, that independence of the upper sequence implies independence of the lower.

The diagram in Fig. 4(a) shows a decomposition of the transformations $G_0 \overset{p_1,m_1}{\Longrightarrow} G_1 \overset{p_2,m_2}{\Longrightarrow} G_2$ and $G_0 \overset{p_2,m_2'}{\Longrightarrow} G_1' \overset{p_1,m_1'}{\Longrightarrow} G_2$ according to the proof of the



(a) Match $m_3'' \colon L_3 \to G_0$ (b) Induced morphism $\hat{L}_3 \to D_1'$

Fig. 4. Constructions for proof of Thm. 1

local Church-Rosser theorem ([8], Thm. 3.20). Hence $D_2' G_2 D_2 D_2^*$ is a pullback while all other squares are pushouts, and all morphisms are monos.

The match m_3 is also shown. Let us assume that $G_1 \overset{p_2,m_2}{\Longrightarrow} G_2$ and $G_2 \overset{p_3,m_3}{\Longrightarrow} G_3$ are independent. Then, there exists $L_3 \to D_2$ commuting with m_3 such that $m_3^* = L_3 \to D_2 \to G_1$ satisfies N_3. Also, $G_1' \overset{p_1,m_1'}{\Longrightarrow} G_2$ and $G_2 \overset{p_3,m_3}{\Longrightarrow} G_3$ are independent because equivalence of s and s' requires to switch them, so there exists $L_3 \to D_2'$ commuting with m_3 such that $m_3' = L_3 \to D_2' \to G_1'$ satisfies N_3.

There exists a morphism $L_3 \to D_2^*$ commuting the resulting triangles induced by pullback (5). Matches $L_3 \to D_2^* \to D_1$ and $L_3 \to D_2^* \to D_1'$ satisfy N_3 because they are prefixes of matches m_3^* and m_3', respectively; indeed, it is easy to show that $m; m' \models n \Rightarrow m \models n$ for injective matches m, m' and constraint n.

To show that $m_3'' = L_3 \to D_2^* \to D_1 \to G_0 = L_3 \to D_2^* \to D_1' \to G_0$ satisfies N_3, by way of contradiction, assume $n : L_3 \to \hat{L}_3 \in N_3$ with morphism $q : \hat{L}_3 \to G_0$ commuting with m_3''. We can construct the cube in Fig. 4(b) as follows. The bottom face is pushout (1), faces front left (FL), front right (FR) and top (TOP) are constructed as pullbacks. The commutativity induces unique morphism $O^* \to D_2^*$ making the back faces commuting and thus, all faces in the cube commute. Back left face (BL) is a pullback by pullback decomposition of pullback (TOP+FR) via (BL+(1)) and back right face (BR) is a pullback by pullback decomposition of pullback (TOP+FL) via (BR+(1)). We obtain $o^* \colon L_3 \to O^*$ as induced morphism from pullback (BL+FL) and using the assumption $m_3'' = n; q$. Further, by the van Kampen property, the top face is a pushout. Since the constraint is incremental and $L_3 \to O \to \hat{L}_3 = L_3 \to O' \to \hat{L}_3$, without loss of generality we have a morphism $i : O \to O'$ commuting the triangles.

We show that $e_2 \colon O' \leftrightarrow \hat{L}_3$ is an isomorphism. First of all, e_2 is a mono by pullback (FR) and mono $D'_1 \to G_0$. Pushout (TOP) implies that the morphism pair (e_1, e_2) with $e_1 \colon O \to \hat{L}_3$ and $e_2 \colon O' \to \hat{L}_3$ is jointly epimorphic. By commutativity of $i; e_2 = e_1$, we derive that also $(i; e_2, e_2)$ is jointly epi. By definition of jointly epi, we have that for arbitrary (f, g) it holds that $i; e_2; f = i; e_2; g$ and $e_2; f = e_2; g$ implies $f = g$. This is equivalent to $e_2; f = e_2; g$ implies $f = g$. Thus, e_2 is an epimorphism. Together with e_2 being a mono (see above) we conclude that e_2 is an isomorphism, because adhesive categories are balanced [16]. This means, there exists a mediating morphism $\hat{L}_3 \to O' \to D'_1$ which contradicts the earlier assumption that $L_3 \to D^*_2 \to D'_1$ satisfies N_3. □

Example 3. If in Fig. 3 we replace rule p_3 by rule p_4 of Fig. 5 that has an incremental NAC, so that $s = G_0 \overset{p_1,m_1}{\Longrightarrow} G_1 \overset{p_2,m_2}{\Longrightarrow} G_2 \overset{p_4,m_4}{\Longrightarrow} G_3 = (s_1; s_2; s_4)$, then the problem described in Ex. 1 does not hold anymore, because s_2 and s_4 are not sequentially independent, and they remain dependent in the sequence $s'' = s'_2; s'_4; s''_1$.

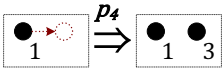


Fig. 5. Rule p_4 with incremental NAC

5 Transforming General NACs into Incremental NACs

In this section we show how to compile a set of rules P with arbitrary NACs into a (usually much larger) set of rules $INC(P)$ having incremental NACs only. The construction guarantees that every derivation using rules in P can be transformed into a derivation over $INC(P)$. Additionally, we show that P and $INC(P)$ are actually equivalent if all constraints in P are obtained as colimits of incremental constraints.

The example shown in Fig. 6 can help getting an intuition about the transformation. It shows one possible outcome (indeed, the algorithm we shall present is non-deterministic) of the application of the transformation to rule p_3, namely the set $INC(\{p_3\}) = \{p_{31}, p_{32}\}$ containing rules with incremental NACs only. It is not difficult to see that p_3 can be applied to a match if and only if either p_{31} or p_{32} can be applied to the same match (determined by the image of node 1), and the effect of the rules is the same (adding a new node). In fact, if either p_{31} or p_{32} can be applied, then also p_3 can be applied to the same match, because at least one part of its NAC is missing (the loop if p_{31} was applied, otherwise the edge). Viceversa, if p_3 can be applied, then either the loop on 1 is missing, and p_{31} is applicable, or the loop is present but there is no non-looping edge from 1, and thus p_{32} can be applied. As a side remark, notice that the NACs p_{31} or p_{32} "cover" the non-incremental NAC of p_3, which is possible because the left-hand side of p_{32} is larger.

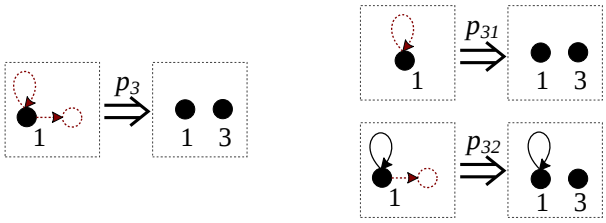


Fig. 6. Rule p_3 (left) and the set $INC(\{p_3\}) = \{p_{31}, p_{32}\}$ (right)

Let us start with some auxiliary technical facts that hold in adhesive categories and that will be exploited to show that the compilation algorithm terminates, which requires some ingenuity because sometimes a single constraint can be compiled into several ones.

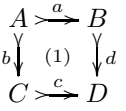
Definition 2 (finitely decomposable monos). *A mono $A \overset{f}{\rightarrowtail} B$ is called* at most k-decomposable*, with $k \geq 0$, if for any sequence of arrows $f_1; f_2; \cdots; f_h = f$ where for all $1 \leq i \leq h$ arrow f_i is a mono and it is not an iso, it holds $h \leq k$. Mono f is called* k-decomposable *if it is at most k-decomposable and either $k = 0$ and f is an iso, or there is a mono-decomposition like the above with $h = k$. A mono is* finitely decomposable *if it is k-decomposable for some $k \in \mathbb{N}$. A 1-decomposable mono is called* atomic*.*

From the definition it follows that all and only the isos are 0-decomposable. Furthermore, any atomic (1-decomposable) mono is incremental, but the converse is false in general. For example, in **Graph** the mono $\{\bullet\} \rightarrowtail \{\bullet \rightarrow \bullet\}$ is incremental but not atomic. Actually, it can be shown that in **Graph** all incremental monos are at most 2-decomposable, but there exist adhesive categories with k-decomposable incremental monos for any $k \in \mathbb{N}$.

Furthermore, every finitely decomposable mono $f : A \rightarrowtail B$ can be factorized as $A \rightarrowtail K \overset{g}{\rightarrowtail} B$ where g is incremental and "maximal" in a suitable sense.

Proposition 3 (decomposition and incrementality). *Let $f : A \rightarrowtail B$ be finitely decomposable. Then there is a factorization $A \rightarrowtail K \overset{g}{\rightarrowtail} B$ of f such that g is incremental and there is no K' such that $f = A \rightarrowtail K' \rightarrowtail K \overset{g}{\rightarrowtail} B$, where $K' \rightarrowtail K$ is not an iso and $K' \rightarrowtail K \overset{g}{\rightarrowtail} B$ is incremental. In this case we call g* maximally incremental w.r.t. *f.*

Proposition 4 (preservation and reflection of k-decomposability). *Let the square to the right be a pushout and a be a mono. Then b is a k-decomposable mono if and only if d is a k-decomposable mono.*



In the following construction of incremental NACs starting from general ones, we will need to consider objects that are obtained starting from a span of monos, like pushout objects, but that are characterised by weaker properties.

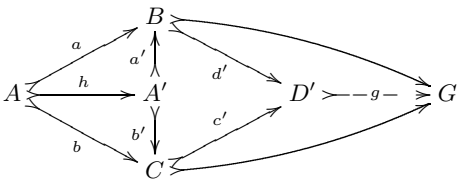


Fig. 7. Quasi-pushout of monos in an adhesive category

Definition 3 (quasi-pushouts of monos). *Let $B \hookleftarrow A \rightarrowtail C$ be a span of monos as in Fig. 7, and let $A \rightarrowtail A'$ be a mono such that there are monos $A' \rightarrowtail B$ and $A' \rightarrowtail C$ making the triangles commute. Let $B \rightarrowtail D' \hookleftarrow C$ be the pushout of $B \hookleftarrow A' \rightarrowtail C$. Then $B \rightarrowtail D' \hookleftarrow C$ is a* quasi-pushout (based on A') *of $B \hookleftarrow A \rightarrowtail C$, and D' is the* quasi-pushout object. *If $A \rightarrowtail A'$ is not an iso, the quasi-pushout is called* proper.

Proposition 5 (properties of quasi-pushouts)

1. *Let $B \hookleftarrow A \rightarrowtail C$ be a span of monos, and let $B \rightarrowtail G \hookleftarrow C$ be a cospan of monos such that the square $B \hookleftarrow A \rightarrowtail C \rightarrowtail G \hookleftarrow B$ commutes. Then there is a quasi-pushout $B \rightarrowtail D' \hookleftarrow C$ of $B \hookleftarrow A \rightarrowtail C$ such that the mediating morphism $g : D' \to G$ is mono.*
2. *Let $B \hookleftarrow A \rightarrowtail C$ be a span of monos. If objects B and C are finite (i.e., they have a finite number of subobjects), then the number of non-isomorphic distinct quasi-pushouts of the span is finite.*
3. *In span $B \hookleftarrow A \overset{b}{\rightarrowtail} C$, suppose that mono b is k-decomposable, and that $B \overset{d'}{\rightarrowtail} D' \hookleftarrow C$ is a quasi-pushout based on A', where $h : A \rightarrowtail A'$ is not an iso. Then mono $d' : B \rightarrowtail D'$ is at most $(k-1)$-decomposable.*
4. *Quasi-pushouts preserve incrementality: if $B \overset{d'}{\rightarrowtail} D' \hookleftarrow C$ is a quasi-pushout of $B \hookleftarrow A \overset{b}{\rightarrowtail} C$ and b is incremental, then also $d' : B \rightarrowtail D'$ is incremental.*

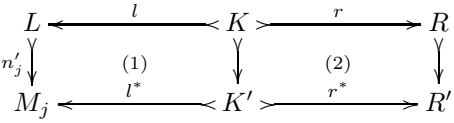
We describe now how to transform a rule p with arbitrary finitely decomposable constraints into a set of rules with simpler constraints: this will be the basic step of the algorithm that will compile a set of rules with finitely decomposable NACs into a set of rules with incremental NACs only.

Definition 4 (compiling a rule with NAC). *Let $p = \langle L \hookleftarrow K \rightarrowtail R, N \rangle$ be a rule with NAC, where the NAC $N = \{n_i : L \rightarrowtail L_i \mid i \in [1, s]\}$ is a finite set of finitely decomposable monic constraints and at least one constraint, say n_j, is not incremental. Then we define the set of rules with NACs $INC(p, n_j)$ in the following way.*

(a) *If* $K \rightarrowtail L \overset{n_j}{\rightarrowtail} L_j$ *has no pushout complement, then* $INC(p, n_j) = \{p'\}$*, where* p' *if obtained from* p *by dropping constraint* n_j*.*

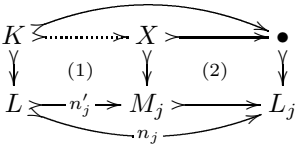
(b) *Otherwise, let* $L \overset{n'_j}{\rightarrowtail} M_j \overset{k}{\rightarrowtail} L_j$ *be a decomposition of* n_j *such that* k *is maximally incremental w.r.t.* n_j *(see Prop. 3). Then* $INC(p, n_j) = \{p', p_j\}$*, where:*

1. p' *is obtained from* p *by replacing constraint* $n_j : L \rightarrowtail L_j$ *with constraint* $n'_j : L \rightarrowtail M_j$*.*
2. $p_j = \langle M_j \leftarrowtail K' \rightarrowtail R', N' \rangle$*, where* $M_j \leftarrowtail K' \rightarrowtail R'$ *is obtained by applying rule* $\langle L \leftarrowtail K \rightarrowtail R \rangle$ *to match* $n'_j : L \rightarrowtail M_j$*, as in the next diagram.*



Furthermore, N' *is a set of constraints* $N' = N'_1 \cup \cdots \cup N'_s$ *obtained as follows. (1)* $N'_j = \{k : M_j \rightarrowtail L_j\}$*. (2) For all* $i \in [1, s] \setminus \{j\}$*,* $N'_i = \{n_{ih} : M_j \rightarrowtail L_{ih} \mid L_i \rightarrowtail L_{ih} \leftarrowtail M_j$ *is a quasi-pushout of* $L_i \leftarrowtail L \rightarrowtail M_j\}$*.*

Before exploring the relationship between p and $INC(p, n_j)$ let us show that the definition is well given, i.e., that in Def. 4(b).2 the applicability of $\langle L \leftarrowtail K \rightarrowtail R \rangle$ to match $n'_j : L \rightarrowtail M_j$ is guaranteed.



In fact, by the existence of a pushout complement of $K \rightarrowtail L \overset{n_j}{\rightarrowtail} L_j$ we can build a pushout that is the external square of the diagram on the right; next we build the pullback (2) and obtain $K \rightarrowtail X$ as mediating morphism. Since (1) + (2) is a pushout, (2) is a pullback and all arrows are mono, from Lemma 4.6 of [16] we have that (1) is a pushout, showing that $K \rightarrowtail L \rightarrowtail M_j$ has a pushout complement.

The following result shows that $INC(p, n_j)$ can simulate p, and that if the decomposition of constraint n_j has a pushout complement, then also the converse is true.

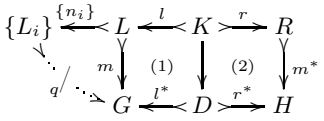


Fig. 8. DPO diagram with NAC

Proposition 6 (relationship between p **and** $INC(p, n_j)$**).** *In the hypotheses of Def. 4, if* $G \overset{p}{\Longrightarrow} H$ *then* $G \overset{INC(p,n_j)}{\Longrightarrow} H$*. Furthermore, if the decomposition of* n_j *(*$L \overset{n'_j}{\rightarrowtail} M_j \overset{k}{\rightarrowtail} L_j$*) has a pushout complement, then* $G \overset{INC(p,n_j)}{\Longrightarrow} H$ *implies* $G \overset{p}{\Longrightarrow} H$*.*

Example 4. Fig. 6 shows one possible outcome of $INC(\{p_3\})$, as discussed at the beginning of this section. As the NAC of p_3 is a colimit of incremental arrows, $\{p_3\}$ and $INC(\{p_3\})$ are equivalent.

Instead, let $p = \langle id_{\{\bullet^1\}}, n : \{\bullet^1\} \rightarrowtail \{\bullet^1 \rightarrow \bullet^2 \rightarrow \bullet^3\}\rangle$ be a rule (the identity rule on graph $\{\bullet^1\}$) with a single negative constraint n, which is not incremental. Then according to Def. 4 we obtain $INC(p, n) = \{p', p_1\}$ where $p' = \langle id_{\{\bullet^1\}}, n' : \{\bullet^1\} \rightarrowtail \{\bullet^1 \rightarrow \bullet^2\}\rangle$ and $p_1 = \langle id_{\{\bullet^1 \rightarrow \bullet^2\}}, n'' : \{\bullet^1 \rightarrow \bullet^2\} \rightarrowtail \{\bullet^1 \rightarrow \bullet^2 \rightarrow \bullet^3\}\rangle$. Note that all constraints in $INC(p, n)$ are incremental, but the splitting of n as $n'; n''$ does not have a pushout complement. Indeed, we can find a graph to which p_1 is applicable but p is not, showing that the condition we imposed on NACs to prove that p and $INC(p, n_j)$ are equivalent is necessary. In fact, let $G = \{\bullet^2 \leftarrow \bullet^1 \rightarrow \bullet^{2'} \rightarrow \bullet^{3'}\}$, and let x be the inclusion morphism from $\{\bullet^1 \rightarrow \bullet^2\}$ to G. Then $G \overset{p_1,x}{\Longrightarrow} G$, but the induced inclusion match $m : \{\bullet^1\} \rightarrow G$ does not satisfy constraint n.

Starting with a set of rules with arbitrary (but finitely decomposable) NACs, the construction of Def. 4 can be iterated in order to get a set of rules with incremental NACs only, that we shall denote $INC(P)$. As expected, $INC(P)$ simulates P, and they are equivalent if all NACs are obtained as colimits of incremental constraints.

Definition 5 (compiling a set of rules). *Let P be a finite set of rules with NACs, such that all constraints in all NACs are finitely decomposable. Then the set $INC(P)$ is obtained by the following procedure.*

$INC(P) := P$
while *(there is a rule in $INC(P)$ with a non-incremental constraint)* **do**
  let $\hat{k} = max\{k \mid$ *there is a k-decomposable non-incremental constraint in $INC(P)$*$\}$
  let $\hat{n}$ be a $\hat{k}$-decomposable non-incremental constraint of $\hat{p} \in INC(P)$
  Set $INC(P) := (INC(P) \setminus \{\hat{p}\}) \cup INC(\hat{p}, \hat{n})$
endwhile
return $INC(P)$

Theorem 2 (correctness and conditional completeness of compilation)

1. *The algorithm of Def. 5 terminates.*
2. *$INC(P)$ contains rules with incremental NACs only.*
3. *$INC(P)$ simulates P, i.e., $G \overset{P}{\Longrightarrow} H$ implies $G \overset{INC(P)}{\Longrightarrow} H$ for all G.*
4. *Suppose that each constraint of each rule in P is the colimit of incremental monos, i.e., for each constraint $L \rightarrowtail L'$, L' is the colimit object of a finite diagram $\{L \rightarrowtail L_i\}_{i \in I}$ of incremental monos. Then P and $INC(P)$ are equivalent, i.e., we also have that $G \overset{INC(P)}{\Longrightarrow} H$ implies $G \overset{P}{\Longrightarrow} H$.*

Proof. Point 2 is obvious, given the guard of the **while** loop, provided that it terminates. Also the proofs of points 3 and 4 are pretty straightforward, as they follow by repeated applications of Prop. 6. The only non-trivial proof is that of termination.

To this aim, let us use the following lexicographic ordering, denoted $\langle \mathbb{N}^k, \sqsubseteq \rangle$ for a fixed $k \in \mathbb{N}$, that is obviously well-founded. The elements of $\mathbb{N}^k$ are sequences of natural numbers of length k, like $\sigma = \sigma_1\sigma_2 \ldots \sigma_k$. The ordering is defined as $\sigma \sqsubset \sigma'$ iff $\sigma_h < \sigma'_h$, where $h \in [1, k]$ is the *highest* position at which σ and σ' differ.

Now, let k be the minimal number such that all non-incremental constraints in P are at most k-decomposable, and define the *degree* of a rule p, $deg(p)$, as the sequence $\sigma \in \mathbb{N}^k$ given by

$$\sigma_i = |\{n \mid n \text{ is an } i\text{-decomposable non-incremental constraint of } p\}|$$

Define $deg(Q)$ for a finite set of rules as the componentwise sum of the degrees of all the rules in Q.

Next we conclude by showing that at each iteration of the loop of Def. 5 the degree $deg(INC(P))$ decreases strictly. Let $\hat{p}$ be a rule and $\hat{n}$ be a non-incremental constraint, $\hat{k}$-decomposable for a maximal $\hat{k}$. The statement follows by showing that $INC(\hat{p}, \hat{n})$ has at least one $\hat{k}$-decomposable non-incremental constraint less than $\hat{p}$, while all other constraints are at most $(\hat{k} - 1)$-decomposable.

This is obvious if $INC(\hat{p}, \hat{n})$ is obtained according to point (a) of Def. 4. Otherwise, let $INC(\hat{p}, \hat{n}) = \{p', p_j\}$ using the notation of point (b). In this case rule p' is obtained from $\hat{p}$ by replacing the selected constraint with one that is at most $(\hat{k}-1)$-decomposable. Furthermore, each other constraint n_i is replaced by a set of constraints, obtained as quasi-pushouts of n_i and n'_j. If n_i is incremental, so are all the new constraints obtained as quasi-pushouts, by Prop. 5(4), and thus they don't contribute to the degree. If instead n_i is non-incremental, then it is h-decomposable for $h \leq \hat{k}$, by definition of $\hat{k}$. Then by Prop. 5(3) all constraints obtained as proper quasi-pushouts are at most $(h-1)$-decomposable, and only one (obtained as a pushout) will be h-decomposable. □

6 Discussion and Conclusion

In our quest for a stable notion of independence for conditional transformations, we have defined a restriction to *incremental NACs* that guarantees this property (Thm. 1). Incremental NACs turn out to be quite powerful, as they are sufficient for several case studies of GTSs. In particular, the well studied model transformation from class diagrams to relational data base models [14] uses incremental NACs only. In an industrial application for translating satellite software (pages 14-15 in [20]), we used a GTS with more than 400 rules, where only 2 of them have non-incremental NACs. Moreover, the non-incremental NACs could also have been avoided by some modifications of the GTS. Incremental morphisms have been considered recently in [12], in a framework different but related to ours,

where requiring that matches are *open maps* one can restrict the applicability of transformation rules without using NACs.

We have also presented a construction that compiles as set of rules with general (finitely-decomposable) NACs into a set of rules with incremental NACs only. For NACs that are obtained as colimits of incremental ones, this compilation yields an equivalent system, i.e., for every transformation in the original GTS there exists one compatible step in the compiled one and vice versa (Thm. 2), and therefore the rewrite relation on graphs is still the same. In the general case, the compiled system provides an overapproximation of the original GTS, which nevertheless can still be used to analyse the original system.

In fact our intention is to define a stable notion of independence on transformations with general NACs. Using the compilation, we can declare a two-step sequence independent if this is the case for all of its compilations, or more liberally, for at least one of them. Both relations should lead to notions of equivalence that are finer than the standard shift equivalence, but that behave well thanks to Thm. 1. Moreover, independence should be expressed directly on the original system, rather than via compilation. Such a revised relation will be the starting point for developing a more advanced theory of concurrency for conditional graph transformations, including processes and unfoldings of GTSs.

The main results in this paper can be applied for arbitrary adhesive transformation systems with monic matches. However, in some cases (like for attributed graph transformation system) the restriction to injective matches is too strict (rules contain terms that may be mapped by the match to equal values). As shown in [13], the concept of NAC-schema provides a sound and intuitive basis for the handling of non-injective matches for systems with NACs. We are confident that an extension of our results to general matches is possible based on the concept of NAC-schema.

Another intersting topic that we intend to study is the complexity of the algorithm of Def. 5, and the size of the set of rules with incremental constraints, $INC(P)$, that it generates. Furthermore, we plan to extend the presented results for shift equivalence to the notion of permutation equivalence, which is coarser and still sound according to [13]. Finally, we also intend to address the problem identified in Ex. 1 at a more abstract level, by exploiting the event structures with or-causality of events that are discussed in depth in [19].

References

1. Baldan, P., Corradini, A., Montanari, U., Rossi, F., Ehrig, H., Löwe, M.: Concurrent Semantics of Algebraic Graph Transformations. In: Ehrig, H., Kreowski, H.J., Montanari, U., Rozenberg, G. (eds.) The Handbook of Graph Grammars and Computing by Graph Transformations. Concurrency, Parallelism and Distribution, vol. 3, pp. 107–188. World Scientific (1999)
2. Baldan, P., Corradini, A., Heindel, T., König, B., Sobociński, P.: Processes for adhesive rewriting systems. In: Aceto, L., Ingólfsdóttir, A. (eds.) FOSSACS 2006. LNCS, vol. 3921, pp. 202–216. Springer, Heidelberg (2006)

3. Baldan, P., Corradini, A., Heindel, T., König, B., Sobociński, P.: Unfolding grammars in adhesive categories. In: Kurz, A., Lenisa, M., Tarlecki, A. (eds.) CALCO 2009. LNCS, vol. 5728, pp. 350–366. Springer, Heidelberg (2009)
4. Baldan, P., Corradini, A., Montanari, U., Ribeiro, L.: Unfolding semantics of graph transformation. Inf. Comput. 205(5), 733–782 (2007)
5. Corradini, A., Hermann, F., Sobociński, P.: Subobject Transformation Systems. Applied Categorical Structures 16(3), 389–419 (2008)
6. Corradini, A., Montanari, U., Rossi, F.: Graph processes. Fundamenta Informaticae 26(3/4), 241–265 (1996)
7. Ehrig, H.: Introduction to the Algebraic Theory of Graph Grammars (A Survey). In: Claus, V., Ehrig, H., Rozenberg, G. (eds.) Graph Grammars 1978. LNCS, vol. 73, pp. 1–69. Springer, Heidelberg (1979)
8. Ehrig, H., Ehrig, K., Prange, U., Taentzer, G.: Fundamentals of Algebraic Graph Transformation. EATCS Monographs in Theor. Comp. Science. Springer (2006)
9. Ehrig, H., Pfender, M., Schneider, H.: Graph grammars: an algebraic approach. In: 14th Annual IEEE Symposium on Switching and Automata Theory, pp. 167–180. IEEE (1973)
10. Ehrig, H., Habel, A., Lambers, L.: Parallelism and Concurrency Theorems for Rules with Nested Application Conditions. EC-EASST 26, 1–24 (2010)
11. Habel, A., Heckel, R., Taentzer, G.: Graph Grammars with Negative Application Conditions. Fundamenta Informaticae 26(3,4), 287–313 (1996)
12. Heckel, R.: DPO Transformation with Open Maps. In: Ehrig, H., Engels, G., Kreowski, H.-J., Rozenberg, G. (eds.) ICGT 2012. LNCS, vol. 7562, pp. 203–217. Springer, Heidelberg (2012)
13. Hermann, F., Corradini, A., Ehrig, H.: Analysis of Permutation Equivalence in M-adhesive Transformation Systems with Negative Application Conditions. In: MSCS (to appear, 2013)
14. Hermann, F., Ehrig, H., Golas, U., Orejas, F.: Efficient Analysis and Execution of Correct and Complete Model Transformations Based on Triple Graph Grammars. In: Bézivin, J., Soley, R., Vallecillo, A. (eds.) Proc. Int. Workshop on Model Driven Interoperability (MDI 2010), pp. 22–31. ACM (2010)
15. Kreowski, H.J.: Is parallelism already concurrency? Part 1: Derivations in graph grammars. In: Ehrig, H., Nagl, M., Rozenberg, G., Rosenfeld, A. (eds.) Graph Grammars 1986. LNCS, vol. 291, pp. 343–360. Springer, Heidelberg (1987)
16. Lack, S., Sobocinski, P.: Adhesive and quasiadhesive categories. ITA 39(3), 511–545 (2005)
17. Lambers, L., Ehrig, H., Orejas, F., Prange, U.: Parallelism and Concurrency in Adhesive High-Level Replacement Systems with Negative Application Conditions. In: Ehrig, H., Pfalzgraf, J., Prange, U. (eds.) Proceedings of the ACCAT Workshop at ETAPS 2007, vol. 203/6, pp. 43–66. Elsevier (2008)
18. Lambers, L.: Certifying Rule-Based Models using Graph Transformation. Ph.D. thesis, Technische Universität Berlin (2009)
19. Langerak, R., Brinksma, E., Katoen, J.P.: Causal ambiguity and partial orders in event structures. In: Mazurkiewicz, A., Winkowski, J. (eds.) CONCUR 1997. LNCS, vol. 1243, pp. 317–331. Springer, Heidelberg (1997)
20. Ottersten, B., Engel, T.: Interdisciplinary Centre for Security, Reliability and Trust - Annual Report 2011. University of Luxembourg, Interdisciplinary Centre for Security, Reliability and Trust (SnT), http://www.uni.lu/content/download/52106/624943/version/1/file/SnT_AR2011_final_web.pdf