The Pullback-Pushout Approach to Algebraic Graph Transformation*

Andrea Corradini[1](✉), Dominque Duval[2](✉), Rachid Echahed[2](✉), Frédéric Prost[2](✉) and Leila Ribeiro[3](✉)

[1] Dipartimento di Informatica, Università di Pisa, Pisa, Italy
andrea@di.unipi.it
[2] CNRS and Université Grenoble Alpes, Grenoble, France
Dominique.Duval@imag.fr, Rachid.Echahed@imag.fr, Frederic.Prost@imag.fr
[3] INF - Universidade Federal do Rio Grande do Sul, Porto Alegre, Brazil
leila@inf.ufrgs.br

Abstract. Some recent algebraic approaches to graph transformation include a pullback construction involving the match, that allows one to specify the cloning of items of the host graph. We pursue further this trend by proposing the Pullback-Pushout (PB-PO) Approach, where we combine smoothly the classical modifications to a host graph specified by a rule (a span of graph morphisms) with the cloning of structures specified by another rule. The approach is shown to be a conservative extension of AGREE (and thus of the SQPO approach), and we show that it can be extended with standard techniques to attributed graphs. We discuss conditions to ensure a form of locality of transformations, and conditions to ensure that the attribution of transformed graphs is total.

1 Introduction

Algebraic graph transformations have been dominated by two main approaches, namely the Double Pushout (DPO) [9] and the Single Pushout (SPO) [14]. These two approaches offer a very simple and abstract definition of a large class of graph transformation systems [5, 8]. However, they are not suited for modeling transformations where certain items of the host graph should be copied (cloned), possibly together with the connections to the surrounding context. This feature is instead naturally available in approaches to graph transformation based on node replacement, like Node-Label-Controlled (NLC) grammars [10], and is needed in several application domains. The NLC approach is typically presented in set-theoretical terms, but a categorical formulation was proposed in [1]. The key points there are that a rule is represented as a morphism from the right-hand side (rhs) to the left-hand side (lhs) (both enriched to represent abstractly the possible embedding context), and a match is a morphism *from* the host graph to

* This work has been partially supported by the LabEx PERSYVAL-Lab (ANR-11-LABX-0025-01) funded by the French program Investissement d'avenir and by the Brazilian agency CNPq.

the lhs. Then rewriting is modeled by a pullback: the cloning of edges due to node replacement is obtained by the multiplicative effect of the limit construction.

Independently, some approaches were proposed to enrich DPO with cloning, including Adaptive Star Grammars [6], Sesqui-Pushout (SQPO) [4] and AGREE [3]. Even if the presentations differ, all are based on the idea of introducing a limit construction in the first phase of rewriting, to model cloning. Coherently with the DPO, in these approaches a match is a morphism from the lhs of the rule *to* the host graph but, at least for SQPO and AGREE, the match determines implicitly a morphism *from* the host graph to an enriched version of the lhs, which is pulled back along a suitable morphism to model deletion and cloning of items. Other approaches to structure transformations where the match goes *from* the host graph to the rule include [19] for refactoring object-oriented systems, and [20] for ontologies: in both cases some form of cloning can be modeled easily.

The analysis of these approaches led us to define (yet) an(other) algebraic approach to graph transformation, called PB-PO, that we introduce in this paper. The PB-PO approach conservatively extends AGREE [3], and thus SQPO [4] with injective matches, by streamlining the definition of transformation and making explicit the fact that when cloning is a concern, it is natural to include in the transformation a pullback construction based on (part of) the match, that has to go *from* the host graph to (part of) the lhs of the rule. A rule in PB-PO is made of two spans, the *top* and the *bottom* ones, forming two commutative squares. A match consists of a pair of morphisms, the *instantiation* from the lhs of the top span to the host graph (like a standard match in DPO and similar approaches), and the *typing* from the host graph to the lhs of the bottom span, that is used to clone items with the first phase of a transformation, which is a pullback. As the name of the approach suggests, the second phase is a standard pushout which glues the pullback object with the rhs of the top span. Thus a PB-PO transformation can be seen as a combination of a standard transformation of structures, modeled by the top span, with a sort of retyping modeled by the bottom span.

Like other categorical approaches supporting cloning (e.g. [1, 3]) also PB-PO may specify transformations that are not local, in the sense that they affect part of the host graph that is not in the image of the instantiation. After showing in which sense the new approach extends AGREE, we propose a formal notion of locality for PB-PO rules and a sufficient condition to ensure it.

Next we consider the enrichment of the PB-PO with attributes, following the ideas developed in [7] for the SQPO approach. A key feature of this approach is to allow attributes of items of the host graph to be changed through the application of a rule, a feature that is possible thanks to the use of *partially* attributed structures. As a consequence, in general the result of transforming a completely attributed graph via a PB-PO rule could be a partially attributed graph. We present some sufficient syntactic conditions over rules in order to ensure that the result of a transformation is totally attributed.

The paper is organized as follows: In Section 2, we define PB-PO rewriting and in Section 3, we show its relation with the AGREE and SQPO approaches. Then, we discuss issues regarding the locality of PB-PO rewriting in Section 4. In Section 5,

we show how the PB-PO approach extends to deal with attributed structures. Finally, we conclude in Section 6.

2 The PB-PO transformation of structures

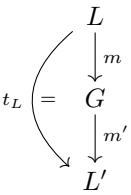
In this section we introduce the PB-PO approach to structure transformation. The main differences with respect to other algebraic approaches is the shape of a rule and, as a consequence, the definition of a match. To make the presentation lighter, we start assuming that all objects and diagrams belong to a category of "structures" **G** with "enough" pullbacks and pushouts so that the required constructions exist. We will introduce any additional requirement on **G** when needed. Typical examples of categories of interest are that of graphs, of hypergraphs, or of typed graphs (i.e., the slice category **Graph** $\downarrow T$ for a given type graph T). Such categories have all limits and colimits.

Definition 1 (Rule, Match and Rewrite Step). *A* PB-PO rule ρ *is a commutative diagram as follows:*

$$\begin{array}{ccccc} L & \xleftarrow{l} & K & \xrightarrow{r} & R \\ \downarrow{\scriptstyle t_L} & = & \downarrow{\scriptstyle t_K} & = & \downarrow{\scriptstyle t_R} \\ L' & \xleftarrow[l']{} & K' & \xrightarrow[r']{} & R' \end{array} \tag{1}$$

We call $L \xleftarrow{l} K \xrightarrow{r} R$ *the* top span *of* ρ *and* $L' \xleftarrow{l'} K' \xrightarrow{r'} R'$ *its* bottom span. *The three vertical arrows are called the* left-hand (lhs) side*, the* interface *and the* right-hand (rhs) side *of* ρ*. We say that* ρ *is in* canonical form *if the left square is a pullback and the right square is a pushout.*

A (PB-PO) match *of* ρ *in an object* G *is a factorization of its left-hand side through* G*, i.e. a pair* (m, m') *such that* $m' \circ m = t_L$*, as shown on the right. Arrow* $m : L \to G$ *is called the* instantiation (part) *and arrow* $m' : G \to L'$ *the* typing (part) *of the match.*

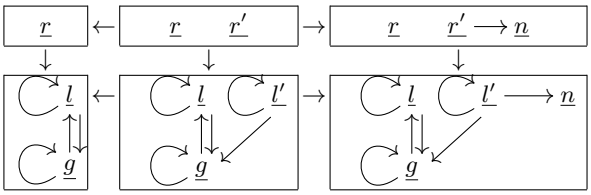


A PB-PO rewrite step *from* G *to* H *via rule* ρ*, denoted* $G \Rightarrow_\rho H$*, is defined by the following diagram, where square* (a) *is a pullback, arrow* $n : K \to D$ *(making square* (a') *commuting) is uniquely determined by the universal property of pullbacks, square* (b) *is a pushout, and arrow* $p' : H \to R$ *makes square* (b') *commuting and is uniquely determined by the properties of pushouts.*

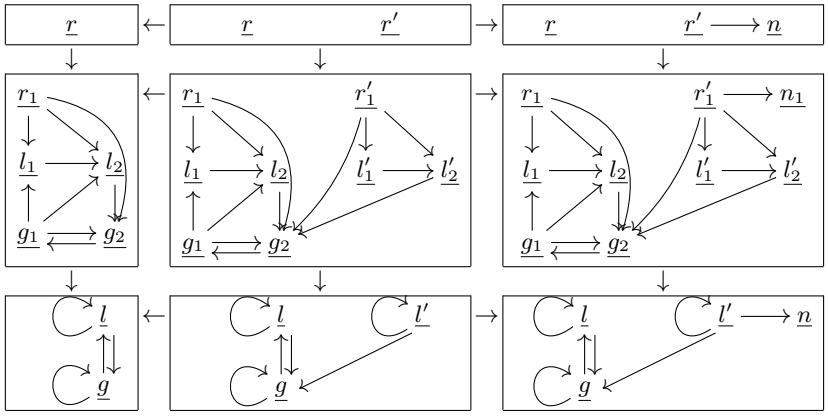
$$\begin{array}{ccccccc} L & \xleftarrow{l} & K & & \xrightarrow{r} & R \\ \downarrow{\scriptstyle m} & = (a') & \downarrow{\scriptstyle n} & PO\ (b) & & \downarrow{\scriptstyle p} \\ G & \xleftarrow{g} & D & & \xrightarrow{h} & H \\ {\scriptstyle t_L}= \downarrow{\scriptstyle m'} & PB\ (a) & {\scriptstyle t_K}= \downarrow{\scriptstyle n'} & = (b') & & {\scriptstyle t_R}= \downarrow{\scriptstyle p'} \\ L' & \xleftarrow[l']{} & K' & & \xrightarrow[r']{} & R' \end{array} \tag{2}$$

Note that if rule ρ is in canonical form and it is applied to a match (m, m'), in the resulting Diagram (2) we have that (a') is a pullback and (b') is a pushout by obvious properties of these universal constructions.
It is worth observing that object R' of a rule is not involved directly in a rewrite step, but it determines a default typing for the result of rewriting. Thus a PB-PO rewrite step maps a PB-PO match (m, m') to another PB-PO match (p, p').

Example 2. This example is related to the copy of some "local" web pages, as discussed in [3]. Nodes represent web pages, and edges represent hyperlinks among them. Then it is reasonable to expect that creating a copy of a set of local pages will only copy the hyperlinks contained in such pages, and not those in remote pages pointing to them. This is modeled by the following PB-PO rule ρ in the category of graphs, where the vertical morphisms map $\underline{r}, \underline{r'}, \underline{n}$ respectively to $\underline{l}, \underline{l'}, \underline{n}$. Note that in order to avoid confusion between morphism names and graph node names, the latter will be underlined in the rest of the paper.



A match $L \overset{m}{\to} G \overset{m'}{\to} L'$ classifies the nodes of G as either *local* ($\underline{l}$) or *global* ($\underline{g}$) thanks to the typing $m' : G \to L'$ and it distinguishes one *root* node ($\underline{r}$) of G thanks to the instantiation $m : L \to G$. In addition, $\underline{r}$ is local since $m' \circ m = t_L$. The *local subgraph* of G is defined as the subgraph of G generated by the local nodes. By applying the rule ρ to the match (m, m') we get a graph H which contains G together with a copy of its local subgraph with all its outgoing edges, and with an additional edge from the copy of the root to a new node $\underline{n}$. Here is an instance of such a rewrite step, where the root of G is $\underline{r_1}$, its local nodes are $\underline{r_1}, \underline{l_1}, \underline{l_2}$ and its global nodes are $\underline{g_1}, \underline{g_2}$:

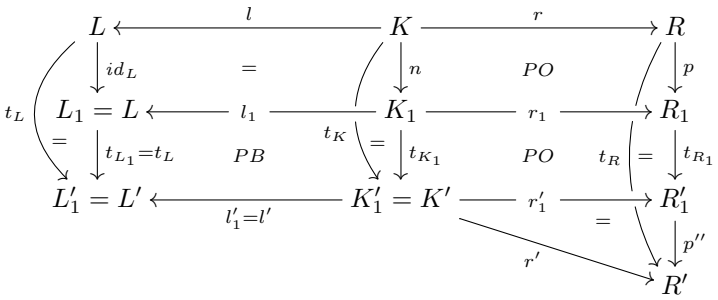


A natural question is whether rules in canonical forms are as expressive as general rules. The following result answers positively to this question.

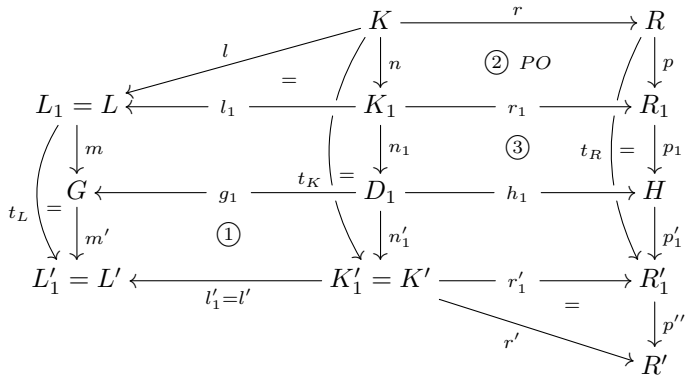
Proposition 3 (canonical forms). *For each* PB-PO *rule* ρ *there exists a rule* ρ_1 *in canonical form which is equivalent, that is*

- ρ *and* ρ_1 *have the same lhs* $t_L : L \to L'$
- *for each match* (m, m') *with* $L \overset{m}{\to} G \overset{m'}{\to} L' = t_L$, $G \Rightarrow_\rho H$ *if and only if* $G \Rightarrow_{\rho_1} H$.

Proof. The following diagram shows how one can build from rule ρ (whose components are named as in Diagram (1)) a corresponding rule ρ_1 (where corresponding components have subscript 1).

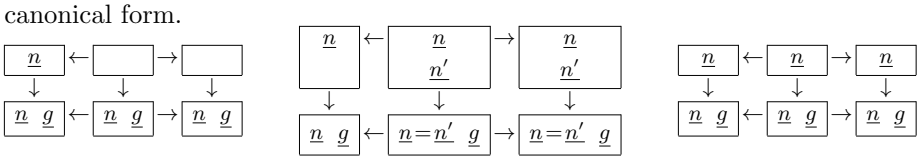


First rule ρ is applied using the PB-PO approach to match (id_L, t_L), generating the pullback object K_1 and the pushout object R_1. Next we build the pushout of r_1 and t_{K_1}, obtaining object R'_1. It is obvious by construction that rule ρ_1, made of top span $L_1 \overset{l_1}{\leftarrow} K_1 \overset{r_1}{\to} R_1$ and of bottom span $L'_1 \overset{l'_1}{\leftarrow} K'_1 \overset{r'_1}{\to} R'_1$, is canonical, and also that the lhs of ρ and ρ_1 coincide.
Now let G be an object and (m, m') be a PB-PO match of ρ (and ρ_1) in G, and consider the following diagram. We argue that $G \Rightarrow_\rho H$ if and only if $G \Rightarrow_{\rho_1} H$.



In fact, $G \Rightarrow_\rho H$ if and only if ① is a pullback and ② + ③ is a pushout, while $G \Rightarrow_{\rho_1} H$ if and only if ① is a pullback and ③ is a pushout. Thus we can conclude by observing that since ② is a pushout by construction, ③ is a pushout if and only if ② + ③ is a pushout, by well known properties of composition and decomposition of pushouts. □

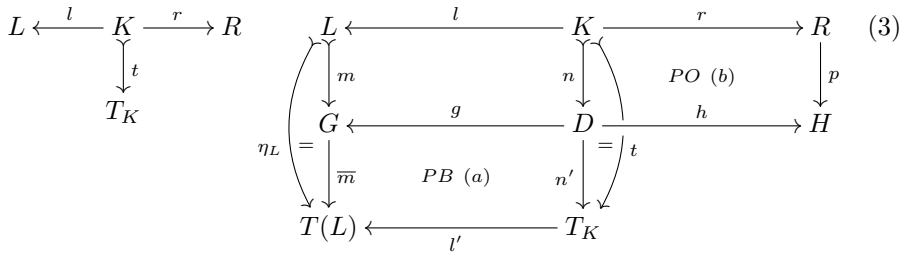
Example 4. It is easy to check that the following three PB-PO rules are equivalent and act as identities on any graph with at least one node. The last one is in canonical form.



3 Relating PB-PO with AGREE and SQPO rewriting

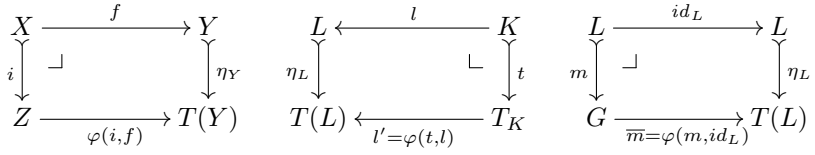
In this section, we first show that PB-PO extends both AGREE and SQPO with monic matches (in categories where AGREE rewriting is defined), and then discuss informally how the greater expressive power can be exploited in designing transformation rules.

An AGREE rule α [3] is a triple of arrows with the same source, as in the left part of (3), and its application to an AGREE match $m : L \rightarrowtail G$ is shown in the right part of (3), defining a rewrite step $G \Rightarrow_{\alpha}^{\text{AGREE}} H$ as explained below.



(3)

Thus an AGREE rule is made of the usual top span enriched with a *mono* $t : K \rightarrowtail T_K$ having a role similar to arrow $t_K : K \to K'$ in a PB-PO rule. The definition of rewriting requires the existence in the underlying category of a *partial map classifier* [2], i.e. for each object Y, there exists an arrow $\eta_Y : Y \rightarrowtail T(Y)$ such that for each pair of arrows $Z \xleftarrow{i} X \xrightarrow{f} Y$ (a *partial map* from Z to Y) there is a unique arrow $\varphi(i, f)$ such that the left diagram of (4) is a pullback.



(4)

The application of the AGREE rule α to a match $m : L \to G$ is obtained by first taking the pullback (a) of $\overline{m}$ and l', and then the pushout of the resulting mediating arrow n and of r. Both $\overline{m}$ and l' are uniquely determined by $T(L)$ as shown in the mid and right diagrams of (4). By comparing Diagrams (2) and (3) we easily obtain the following result.

Proposition 5 (relating AGREE and PB-PO). *Let α be an* AGREE *rule in a category with a partial map classifier. Then there is a* PB-PO *rule ρ_α such that for each mono $m : L \rightarrowtail G$ we have $G \Rightarrow_\alpha^{\text{AGREE}} H$ if and only if $G \Rightarrow_{\rho_\alpha} H$ using match $(m, \overline{m})$ with $\overline{m} : G \to T(L)$.*

Proof. Let $\alpha = (L \xleftarrow{l} K \xrightarrow{r} R, K \xrightarrow{t} T_K)$ be an AGREE rule, and $T_K \xrightarrow{r'} R' \xleftarrow{t_R} R$ be the pushout of $T_K \xleftarrow{t} K \xrightarrow{r} R$. Let ρ_α be the PB-PO rule having $L \xleftarrow{l} K \xrightarrow{r} R$ as top span, $T(L) \xleftarrow{l'} T_K \xrightarrow{r'} R'$ as bottom span, and η_L, t and t_R as the three vertical arrows relating them. Then the statement immediately follows by comparing Diagrams (2) and (3) defining $G \Rightarrow_{\rho_\alpha} H$ and $G \Rightarrow_\alpha^{\text{AGREE}} H$, respectively. □

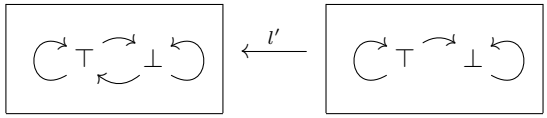
We easily obtain a similar result for SQPO rewriting with monic matches. An SQPO rule has the shape $\sigma = (L \xleftarrow{l} K \xrightarrow{r} R)$ and its application to a match $L \to G$ is defined in [4] via a double-square diagram where the right square is a pushout (as in DPO), but the left square is a *final pullback complement*. In [3] it was shown that for a monic match $L \rightarrowtail G$, $G \Rightarrow_\sigma^{\text{SQPO}} H$ if and only if $G \Rightarrow_{\alpha_\sigma}^{\text{AGREE}} H$, where $\alpha_\sigma = (\sigma, \eta_K : K \rightarrowtail T(K))$ is obtained by enriching σ with the partial map classifier applied to K. The following result is then obvious.

Corollary 6 (relating SQPO and PB-PO). *Let σ be a* SQPO *rule in a category with a partial map classifier. Then there is a* PB-PO *rule ρ_σ such that for each mono $m : L \rightarrowtail G$ we have $G \Rightarrow_\sigma^{\text{SQPO}} H$ if and only if $G \Rightarrow_{\rho_\sigma} H$ using match $(m, \overline{m})$ with $\overline{m} : G \to T(L)$.*

There is therefore a progressively increasing expressive power moving from DPO to SQPO to AGREE to PB-PO, at least for injective matches. A detailed analysis of the expressive power of PB-PO is a topic of future work, but we make a few considerations with respect to the kind of cloning typical of approaches based on node replacement. Standard DPO with left injective rules cannot model cloning of nodes at all. Instead SQPO can, with a non-injective lhs $l : K \to L$. Referring to the right diagram in (3), if a node $n \in L$ is cloned (i.e., it has more than one inverse image in K via l), then its image in G will be cloned as well in D. Furthermore, for any *embedding edge* e, i.e. an edge incident to $m(n)$ in G but not in $m(L)$, there will be one copy of e in D for each counter-image of $m(n)$. With AGREE the same kind of node cloning can be specified, but thanks to the additional arrow $t : K \rightarrowtail T_K$ in the rule, one can specify explicitly which embedding edges have to be copied for each cloned node of G. Moving to PB-PO, note that arrows $t_L : L \to L'$ and $l' : K' \to L'$ are explicitly provided by a PB-PO rule, while the corresponding arrows $\eta_L : L \rightarrowtail T(L)$ and $l' : T_K \to T(L)$ in (3) are uniquely determined by $l : K \to L$ and $t : K \to T_K$ in AGREE. With suitable definitions of object L' and arrow $l' : K' \to L'$, and using the m' part of a match, in PB-PO one can

- classify in a fine way the *context items* of the host graph G, i.e. those not in the image of m;
- for each group of such items, specify if it is deleted, preserved or copied;
- specify additional application conditions.

Example 7. Suppose that in an information system there are two security levels: $\top$, for private information, and $\bot$ for public information. We can model the transformation of a graph containing both private and public information nodes so that in the resulting graph there is no access (arrow) from public to private ones. This can be done with a rule having the empty graph for L, K and R, the following inclusion $K' \subseteq L'$ for l', and the identity for r':



Given a morphism $m' : G \to L'$, mapping all private information nodes to $\top$ and public nodes to $\bot$, the application of this rule to the match $(\emptyset \to G, m')$ would erase all arrows from public nodes to private nodes.

4 Constraining the effects of PB-PO rewriting

As just discussed (and evident from Example 7), a PB-PO rewrite step can affect any item of the host graph, that is, changes are not limited to the image of L and its incident edges (as in other approaches like DPO, SPO and SQPO). This holds for AGREE as well, as discussed in [3] where a notion of *local rule* was introduced. Informally, let us denote with $A \setminus B$ the largest subobject of A disjoint from B. Then an AGREE rule is local if for all matches $m : L \to G$ we have that $G \setminus m(L)$ is preserved after the transformation, i.e. referring to Diagram (3), if $D \setminus n(K) \to G \setminus m(L)$ is an isomorphism. Also, in [3] a sufficient condition for an AGREE rule to be local was identified.
In the case of PB-PO, the greater flexibility in the definition of rules and of matches on the one hand allows us to introduce a more general notion of locality, called *Γ-preservation*, parametrized by a subobject $\Gamma \rightarrowtail L'$ of the lhs of the bottom span. On the other hand, however, whether a rewrite step is Γ-preserving or not depends not only on the rule but also on the match. After introducing the notion of Γ-preservation and characterizing a sufficient condition to ensure it, we relate it to locality of AGREE transformations.

Definition 8 (Γ-preserving rewrite steps). *Let ρ be a* PB-PO *rule as in Diagram (1), $inc : \Gamma \rightarrowtail L'$ be a mono, and (m, m') be a match of ρ in G. Let G_Γ be defined by the pullback on the left of Diagram (5).*
Then we say that the rewrite step $G \Rightarrow_\rho H$ is Γ-preserving *if, referring to Diagram (2), the two squares on the right of Diagram (5) are pullbacks.*

$$\begin{array}{ccc} G_\Gamma & \rightarrowtail & G \\ \downarrow & \lrcorner & \downarrow{\scriptstyle m'} \\ \Gamma & \rightarrowtail & L' \end{array} \qquad \begin{array}{ccccc} G_\Gamma & \leftarrow & G_\Gamma & \rightarrowtail & G_\Gamma \\ \downarrow{\scriptstyle i} & & \downarrow{\scriptstyle j} & \lrcorner & \downarrow{\scriptstyle h \circ j} \\ G & \leftarrow & D & \rightarrow & H \end{array} \tag{5}$$

Intuitively, G_Γ represents the subobject of G typed by Γ. The right diagram of (5) says that subobject G_Γ remains unchanged in D (left square) and in the resulting structure H (right square). Note that in Diagram (5) it follows from the three squares being pullbacks that i, j and also $h \circ j$ are mono.
The next result presents sufficient conditions for a rewrite step to be Γ-preserving, under suitable assumptions on the underlying category and on the rule.

Proposition 9 (conditions for Γ-preservation). *Let us assume that the underlying category of structures* **G** *is adhesive [13] and has a* strict *initial object* 0 *(i.e., each arrow with target* 0 *must have* 0 *as source). Let ρ be a* PB-PO *rule in canonical form or right-linear, i.e., where r is a mono, and let (m, m') be a match of ρ in G with $m : L \rightarrowtail G$ mono. Then we have that if the two squares of Diagram (6) are pullbacks, then $G \Rightarrow_\rho H$ is a Γ-preserving rewrite step.*

$$\begin{array}{ccc} 0 & \longrightarrow & L \\ \downarrow & \lrcorner & \downarrow{\scriptstyle m} \\ G_\Gamma & \overset{i}{\rightarrowtail} & G \end{array} \qquad\qquad \begin{array}{ccc} \Gamma & \overset{id}{\rightarrowtail} & \Gamma \\ \downarrow & \lrcorner & \downarrow{\scriptstyle inc} \\ K' & \overset{l'}{\rightarrow} & L' \end{array} \tag{6}$$

Informally, the left pullback ensures that the subobject of G typed over Γ is disjoint from the image of L in G, thus it is not affected by the top rule; the right pullback guarantees that the subobject Γ of L' (and thus the items of G typed on it) is preserved identically when pulled back along l', thus it is not affected by the bottom rule. It is still open if the previous result also holds for rules that are neither in canonical form nor right linear. To conclude this section, we relate the PB-PO notion of Γ-preservation with locality of AGREE rules.

Proposition 10 (Γ-preservation and AGREE locality). *Let α be an* AGREE *rule as in Diagram (3) and ρ_α be the associated* PB-PO *rule as in Proposition 5. Let Γ_α be the subobject $T(0)$ of $T(L)$. If α is local in the sense of [3], then for each* AGREE *match $m : L \rightarrowtail G$ the rewrite step $G \Rightarrow_{\rho_\alpha} H$ is Γ_α-preserving.*

5 The PB-PO transformation of attributed structures

For attributed structures we follow the same approach as in [7]. Given a category **G** called the category of structures, with pullbacks and pushouts, a category **A** called the category of attributes, and two functors $S : \mathbf{G} \to \mathbf{Set}$ and $T : \mathbf{A} \to \mathbf{Set}$, the category of attributed structures **AttG** and the category of partially attributed structures **PAttG** are defined as in [7]. The issue is that there are not enough pushouts in the category **PAttG**. Let **Pfn** denote the category of sets with partial maps. A partial map f from X to Y is denoted $f : X \rightharpoonup Y$ and its domain of definition is denoted $\mathcal{D}(f)$. The partial order $\leq$ between partial maps is defined as usual: let $f, g : X \rightharpoonup Y$, then $f \leq g$ means that $\mathcal{D}(f) \subseteq \mathcal{D}(g)$ and $f(x) = g(x)$ for all $x \in \mathcal{D}(f)$. Then **Pfn** with this partial order is a 2-category. By composing S and T with the inclusion of **Set** in **Pfn** we get two functors $S_p : \mathbf{G} \to \mathbf{Pfn}$ and $T_p : \mathbf{A} \to \mathbf{Pfn}$. Let $|...|$ denote any of the four functors S, T, S_p, T_p (sometimes, $|...|$ is omitted).

Definition 11 (attributed structures). *The category of* attributed structures **AttG** *(with respect to the functors S and T) is the comma category $(S \downarrow T)$. This means that an* attributed structure *is a triple $\widehat{G} = (G, A, \alpha)$ made of an object G in $\mathbf{G}$, an object A in $\mathbf{A}$ and a map $\alpha : |G| \to |A|$; and a* morphism of attributed structures $\widehat{g} : \widehat{G}_1 \to \widehat{G}_2$, *where $\widehat{G}_1 = (G_1, A_1, \alpha_1)$ and $\widehat{G}_2 = (G_2, A_2, \alpha_2)$, is a pair $\widehat{g} = (g, a)$ made of a morphism $g : G_1 \to G_2$ in $\mathbf{G}$ and a morphism $a : A_1 \to A_2$ in $\mathbf{A}$ such that $\alpha_2 \circ |g| = |a| \circ \alpha_1$. The category of* partially attributed structures **PAttG** *is defined similarly: a* partially attributed structure *is a triple $\widehat{G} = (G, A, \alpha)$ made of an object G in $\mathbf{G}$, an object A in $\mathbf{A}$ and a partial map $\alpha : |G| \rightharpoonup |A|$; and a* morphism of partially attributed structures $\widehat{g} : \widehat{G}_1 \to \widehat{G}_2$, *where $\widehat{G}_1 = (G_1, A_1, \alpha_1)$ and $\widehat{G}_2 = (G_2, A_2, \alpha_2)$, is a pair $\widehat{g} = (g, a)$ made of a morphism $g : G_1 \to G_2$ in $\mathbf{G}$ and a morphism $a : A_1 \to A_2$ in $\mathbf{A}$ such that $\alpha_2 \circ |g| \geq |a| \circ \alpha_1$. A morphism of partially attributed structures (g, a) is called* strict *when $\alpha_2 \circ |g| = |a| \circ \alpha_1$.*

$$\begin{array}{ccc} \widehat{G}_1 & & G_1 \\ {\scriptstyle \widehat{g}}\downarrow & = & {\scriptstyle g}\downarrow \\ \widehat{G}_2 & & G_2 \end{array} \qquad \begin{array}{ccc} |G_1| & \xrightarrow{\alpha_1} & |A_1| \\ {\scriptstyle |g|}\downarrow & \geq & \downarrow{\scriptstyle |a|} \\ |G_2| & \xrightarrow{\alpha_2} & |A_2| \end{array} \qquad \begin{array}{c} A \\ \downarrow{\scriptstyle a} \\ A_2 \end{array}$$

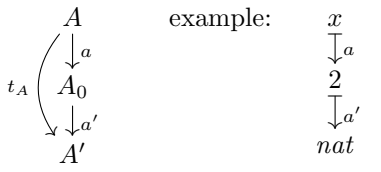
Given an attributed structure $\widehat{G} = (G, A, \alpha)$, we write $\underline{n} : x$ when $\underline{n}$ has attribute x (i.e. $\alpha(\underline{n}) = x$) and $\underline{n} : \bot$ when $\underline{n}$ is not attributed (i.e. $\underline{n} \notin \mathcal{D}(\alpha)$), $|G|_\bot$ denotes the set of elements of $|G|$ which are not attributed. An attributed structure $\widehat{G} = (G, A, \alpha)$ is said attributed *over* A, an attributed morphism $\widehat{g} = (g, a)$ is said attributed *over* a, and when $a = id_A$ then $\widehat{g}$ can be said attributed *over* A.

Remark 12. A morphism of partially attributed structures $\widehat{g} = (g, a) : \widehat{G}_1 \to \widehat{G}_2$ is such that $\widehat{g}(\underline{n}_1 : x_1) = g(\underline{n}_1) : a(x_1)$ and $\widehat{g}(\underline{n}_1 : \bot) = g(\underline{n}_1) : \bot$ or $g(\underline{n}_1) : x_2$ for some x_2. When $\widehat{g}$ is strict, the last case is forbidden, so that the restriction of $\widehat{g}$ determines a map $|g|_\bot : |G_1|_\bot \to |G_2|_\bot$.

Definition 13. *Let $\widehat{g} = (g, a) : \widehat{G}_1 \to \widehat{G}_2$ be a morphism of partially attributed structures. Then $\widehat{g}$ (or g) is* injective *if $|g| : |G_1| \to |G_2|$ is injective. Assume that $\widehat{g}$ is strict, then $\widehat{g}$ is* surjective on non-attributed elements *if $|g|_\bot : |G_1|_\bot \to |G_2|_\bot$ is surjective. Besides, $\widehat{g}$* preserves attributes *if $\widehat{G}_1$ and $\widehat{G}_2$ are attributed over the same A and $a = id_A$.*

As in [7], we assume that all horizontal arrows in the rules preserve attributes, and we will see that this implies that all horizontal arrows in the rewrite steps also preserve attributes. This implies that there are objects A, A_0 and A' and arrows $a : A \to A_0$ and $a' : A_0 \to A'$ in $\mathbf{A}$ such that, in each rewrite step diagram, the vertical arrows in the top squares are over a and the vertical arrows in the bottom squares are over a'. Let $t_A = a' \circ a : A \to A'$. Typically, elements of A are terms with variables, A' describes types (for example, it could be the final algebra in which carrier sets are singletons) and the morphism t_A gives a type

to each variable, and the morphism a denotes an instantiation of the variables (and terms) in A which respects their types:



The definitions of PB-PO attributed rewrite rules, matches and steps must ensure that the result of a step is indeed a well-formed attributed structure. Therefore we have to impose some restrictions on rules and matches with respect to re-attribution (i.e. change of attribute value): (i) only items that are explicitly preserved by the rule can be re-attributed; (ii) items being re-attributed can not be identified with anything neither by the match nor by t_L; (iii) the bottom span of the rule must agree with the upper span with respect to re-attribution (for example, it is not possible that the attribute of the bottom span of an item – its type – is changed and the value of the item in the upper span remains unchanged); and (iv) the left- and right-hand sides of the spans of a rule must be fully-attributed. Some of these conditions are defined for the rule and some for the match. Examples 17 and 18 motivate these conditions and illustrate re-attribution issue.

Definition 14 (PB-PO attributed rewrite rules). *Given a morphism $t_A : A \to A'$ of* $\mathbf{A}$*, a* PB-PO attributed rewrite rule over t_A *is a* PB-PO *rewrite rule ρ in the category* **PAttG** *of partially attributed structures, i.e., a diagram:*

$$\begin{array}{ccccc} \widehat{L} & \xleftarrow{\widehat{l}} & \widehat{K} & \xrightarrow{\widehat{r}} & \widehat{R} \\ \downarrow{\scriptstyle \widehat{t}_L} & = & \downarrow{\scriptstyle \widehat{t}_K} & = & \downarrow{\scriptstyle \widehat{t}_R} \\ \widehat{L}' & \xleftarrow[\widehat{l'}]{} & \widehat{K}' & \xrightarrow[\widehat{r'}]{} & \widehat{R}' \end{array}$$

with the following restrictions: the top line is attributed over A, the bottom line is attributed over A', $\widehat{l}$, $\widehat{l'}$, $\widehat{r}$, and $\widehat{r'}$ are attribute preserving, the vertical morphisms are attributed over t_A, the objects $\widehat{L}$, $\widehat{R}$, $\widehat{L'}$ and $\widehat{R}'$ are totally attributed and the morphism $\widehat{t}_K : \widehat{K} \to \widehat{K}'$ is strict and injective on non-attributed items.

The following condition ensures that whenever an item will be re-attributed, it is (the image of) an item that is preserved by the rule.

Definition 15 (re-attribution condition). *Given a* PB-PO *attributed rewrite rule ρ (with notations as above), a totally attributed structure $\widehat{G}$ and a* PB-PO *match $(\widehat{m}, \widehat{m}')$ of ρ in $\widehat{G}$, the match satisfies the* re-attribution condition *with respect to ρ if:*

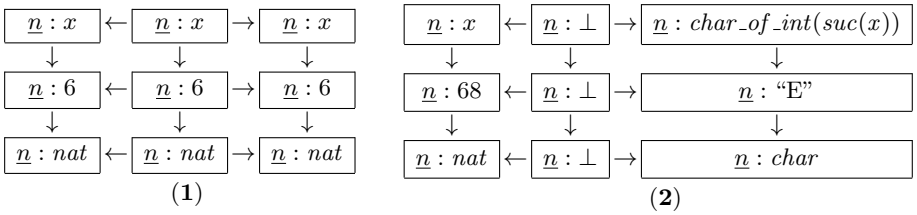
> *for each $\underline{n}_G$ in $|G|$, if there is some $\underline{n}_{K'}$ in $|K'|_\perp$ with $m'(\underline{n}_G) = l'(\underline{n}_{K'})$ then there is an $\underline{n}_K$ in $|K|$ with $\underline{n}_G = m(l(\underline{n}_K))$ and $\underline{n}_{K'} = t_K(\underline{n}_K)$.*

Definition 16 (PB-PO attributed rewrite system). *Given a* PB-PO *attributed rewrite rule* ρ *and a totally attributed structure* $\widehat{G}$*, a* PB-PO attributed match *of* ρ *in* $\widehat{G}$ *is a* PB-PO *match of* ρ *in* $\widehat{G}$ *in the category* **PAttG***, i.e., a pair* $(\widehat{m}, \widehat{m}') = ((m, a), (m', a'))$ *such that* $\widehat{m} \circ \widehat{m}' = \widehat{t}_L$*, with the following restrictions:* $\widehat{m}$ *is injective and* $(\widehat{m}, \widehat{m}')$ *satisfies the re-attribution condition with respect to* ρ*. The* PB-PO attributed rewrite step *applying a* PB-PO *attributed rewrite rule* ρ *to a* PB-PO *attributed match* $(\widehat{m}, \widehat{m}')$ *is the* PB-PO *rewrite step applying* ρ *to* $(\widehat{m}, \widehat{m}')$ *in the category* **PAttG**.

Example 17. All examples are diagrams having the shape of Diagram (2). We start with two basic examples of re-attribution.

(1) Identity: $\underline{n}$ is preserved, with its attribute.

(2) Identity of structure only: $\underline{n}$ is preserved, but its type and attribute are changed.

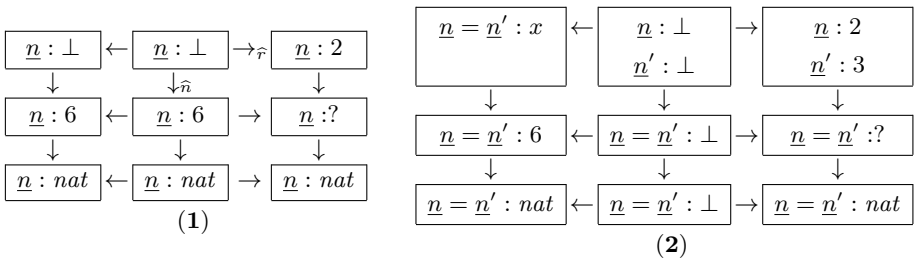


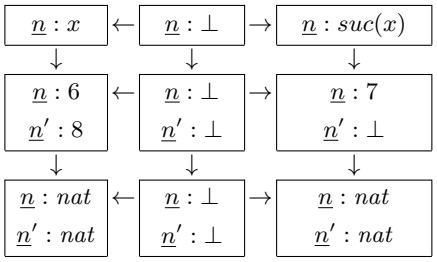
Example 18. We now give three examples to motivate the restrictions about rules and matches made in Definitions 14 and 16.

(1) Here $\widehat{t}_K$ is not strict (and thus the rule is not well-formed). The pushout of $(\widehat{r}, \widehat{n})$ does not exist, so that the rewrite step cannot be constructed.

(2) Here $\widehat{t}_K$ is not injective on non-attributed items: again, the rule is not well-formed, and the pushout of $(\widehat{r}, \widehat{n})$ does not exist.

(3) Here the issue is that $(\widehat{m}, \widehat{m}')$ does not satisfy the re-attribution condition, since $\underline{n}'$ (in G) should be re-attributed but it is not an item preserved by the rule (not an image of an element in K). The pushout of $(\widehat{r}, \widehat{n})$ exists, but the resulting H is not totally attributed.





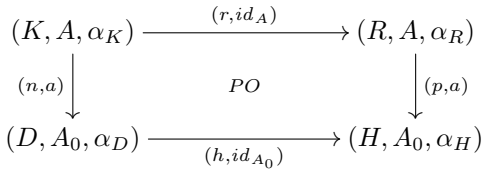
(**3**)

We will use "PB-PO-A" for "PB-PO attributed". In the rest of this section we show that the restrictions imposed on attributed rules and matches in Definitions 14, 15 and 16 are sufficient to guarantee that a rewriting step can be completed, and that the resulting structure is totally attributed. This result is preceded by two technical lemmas concerning pullbacks and pushouts in **PAttG**, respectively.

Lemma 19 (on pullbacks in PAttG). *Let $\widehat{G} \overset{\widehat{m}'}{\to} \widehat{L}' \overset{\widehat{l}'}{\leftarrow} \widehat{K}'$ be a cospan in* **PAttG***, with $\widehat{G}$ and $\widehat{L}'$ totally attributed and $\widehat{l}'$ attribute-preserving. Let us denote $\widehat{G} = (G, A_0, \alpha_G)$, $\widehat{L}' = (L', A', \alpha_{L'})$, $\widehat{K}' = (K', A', \alpha_{K'})$, $\widehat{m}' = (m', a')$ and $\widehat{l}' = (l', id_{A'})$, as in the diagram below. Let $G \overset{g}{\leftarrow} D \overset{n'}{\to} K'$ be the pullback of $G \overset{m'}{\to} L' \overset{l'}{\leftarrow} K'$ in* **G***. Let $\alpha_D : |D| \rightharpoonup |A_0|$ be the partial map such that, for each $\underline{n}_D \in |D|$: if $|n'|(\underline{n}_D) : \bot$ then $\underline{n}_D : \bot$, otherwise $\underline{n}_D : x_0$ where x_0 is the attribute of $|g|(\underline{n}_D)$. Let $\widehat{D} = (D, A_0, \alpha_D)$, $\widehat{g} = (g, id_{A_0})$ and $\widehat{n}' = (n', a')$. Then $\widehat{g} : \widehat{D} \to \widehat{G}$ and $\widehat{n}' : \widehat{D} \to \widehat{K}'$ are morphisms in* **PAttG***, $\widehat{n}'$ is strict, and $\widehat{G} \overset{\widehat{g}}{\leftarrow} \widehat{D} \overset{\widehat{n}'}{\to} \widehat{K}'$ is the pullback of $\widehat{G} \overset{\widehat{m}'}{\to} \widehat{L}' \overset{\widehat{l}'}{\leftarrow} \widehat{K}'$ in* **PAttG***.*

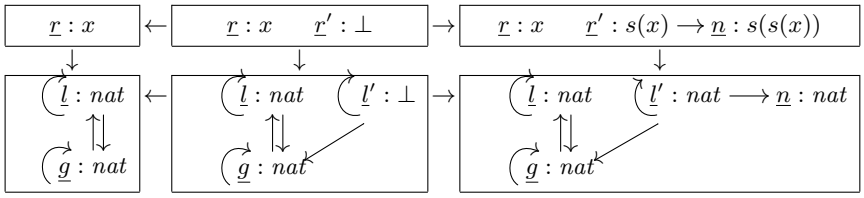
$$\begin{array}{ccc} (G, A_0, \alpha_G) & \xleftarrow{(g, id_{A_0})} & (D, A_0, \alpha_D) \\ {\scriptstyle (m', a')}\downarrow & PB & \downarrow{\scriptstyle (n', a')} \\ (L', A', \alpha_L) & \xleftarrow[(l', id_{A'})]{} & (K', A', \alpha_K) \end{array}$$

Lemma 20 (on pushouts in PAttG). *Assume that the functor $S : \mathbf{G} \to \mathbf{Set}$ preserves pushouts. Let $\widehat{D} \overset{\widehat{n}}{\leftarrow} \widehat{K} \overset{\widehat{r}}{\to} \widehat{R}$ be a span in* **PAttG***, with $\widehat{R}$ totally attributed, $\widehat{r}$ attribute-preserving and $\widehat{n}$ injective, strict, and surjective on non-attributed elements. Let us denote $\widehat{D} = (D, A_0, \alpha_D)$, $\widehat{K} = (K, A, \alpha_K)$, $\widehat{R} = (R, A, \alpha_R)$, $\widehat{r} = (r, id_A)$ and $\widehat{n} = (n, a)$, as in the diagram below. Let $D \overset{h}{\to} H \overset{p}{\leftarrow} R$ be the pushout of $D \overset{n}{\leftarrow} K \overset{r}{\to} R$ in* **G***. Then there is a unique total map $\alpha_H : |H| \to |A_0|$ such that, for each $\underline{n}_H \in |H|$: if $|p|(\underline{n}_R) = \underline{n}_H$ for some $\underline{n}_R : x$ in $|R|$ then $\underline{n}_H : a(x)$, and if $|h|(\underline{n}_D) = \underline{n}_H$ for some $\underline{n}_D : x_0$ in $|D|$ then $\underline{n}_H : x_0$. Let $\widehat{H} = (H, A_0, \alpha_H)$, $\widehat{h} = (h, id_{A_0})$ and $\widehat{p} = (p, a)$. Then $\widehat{H}$ is totally attributed, $\widehat{h} : \widehat{D} \to \widehat{H}$ and $\widehat{p} : \widehat{R} \to \widehat{H}$ are morphisms in* **PAttG***, and $\widehat{D} \overset{\widehat{h}}{\to} \widehat{H} \overset{\widehat{p}}{\leftarrow} \widehat{R}$ is the pushout of $\widehat{D} \overset{\widehat{n}}{\leftarrow} \widehat{K} \overset{\widehat{r}}{\to} \widehat{R}$ in* **PAttG***.*

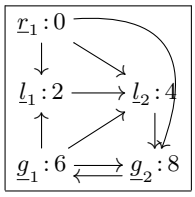


Theorem 21 (rewriting totally attributed structures). *Assume that the functor $S : \mathbf{AttG} \to \mathbf{Set}$ preserves pullbacks and pushouts. Then for every* PB-PO-A *rule and every* PB-PO-A *match of this rule, the* PB-PO-A *rewrite step exists, and in addition the resulting $\widehat{H}$ is totally attributed.*

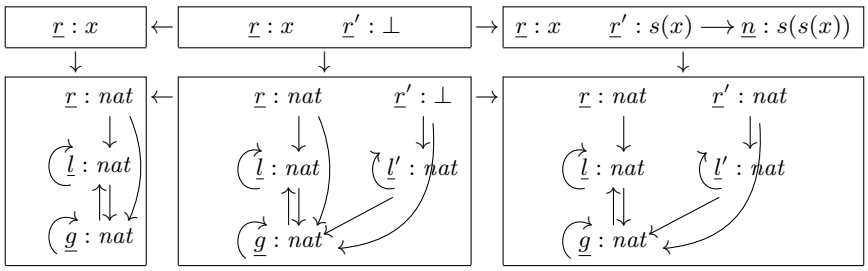
Example 22. Let us recall that the rule of Example 2 specifies that the local web pages of the host graph G (i.e., those mapped by m' to node $\underline{l}$ of L') are cloned with all outgoing edges, while edges from the global pages to cloned ones are not copied. Additionally, a selected local node, "root", is linked to a new page.
The following rule intends to enrich the one of Example 2 by specifying that the copy of the local root page should get as attribute the successor of the attribute of the original page ($s : nat \to nat$ is the successor function), and the new page should get in turn its successor.



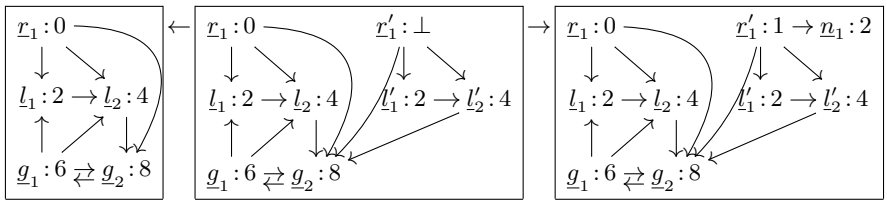
However, if we consider the attributed graph $\widehat{G}$ on the right and the same match as in Example 2 (mapping $\underline{r}_1$, $\underline{l}_1$ and $\underline{l}_2$ to $\underline{l}$ and $\underline{g}_1$ and $\underline{g}_2$ to $\underline{g}$), this match does not satisfy the re-attribution condition, because only $\underline{r}_1$ has a pre-image in K.

$\widehat{G} =$ 

The next rule is instead the "right" extension of the rule in Example 2:



The obvious match satisfies the re-attribution condition, and the resulting $G \xleftarrow{g} D \xrightarrow{h} H$ is:



6 Conclusions and Related Works

We presented a new categorical approach to graph transformation, the PB-PO approach, that combines the standard transformation of structures of the DPO approach with a retyping of the host graph, that allows to model both deletion and cloning of items. PB-PO is shown to be a conservative extension of the AGREE approach, and thus of the SQPO approach with monic matches. The more general framework allows to define a notion of locality parametric with respect to a subgraph of the type graph, and we presented sufficient conditions for a match and a rule to ensure such locality. Finally we extended the approach to attributed structures, presenting sufficient conditions to ensure that the result of transforming a totally attributed structure is still totally attributed.

We discussed in Section 3 the relationships with SQPO and AGREE, of which the PB-PO approach can be considered as an evolution. Adaptive Star Grammars [6] were also proposed to model cloning. They include star replacement rules, which can be seen as a restricted kind of DPO rules, and adaptive star rules, which can be applied to arbitrarily large matches via an adaptation mechanism that creates the needed number of copies of items of the lhs. It should be possible to describe this adaptation mechanism with a limit construction, from which one could explore the feasibility of encoding this approach in PB-PO.

Rewriting in the category of spans [15] has been proposed as a framework that generalizes DPO, SPO and SQPO rewriting, thanks to a powerful gluing construction able to model cloning. Transformations based on both pushouts and pullbacks are used in the quite different framework of collagories [12] or that of model migration [16, section 4.5]. The analysis of the relationships of PB-PO with these contributions will be a topic of future work. Let us also mention that pullbacks are also used in [11] to model the effect of a rule on a graph while the same graph can be subject of other changes caused by the environment, but because of the restriction to injective rules no cloning effect is modeled.

Finally, since the PB-PO rules are defined as two connected spans, one may expect from this approach to model situations where one span is used for transforming data graphs while the other span can be used for transforming the typing information, just like in [17] where rules, defined as two connected co-spans, are used to model co-evolutions of meta-models and models.

References

1. Bauderon, M., Jacquet, H.: Node rewriting in graphs and hypergraphs: a categorical framework. Theor. Comput. Sci. 266(1-2), 463–487 (2001)
2. Cockett, J., Lack, S.: Restriction categories II: partial map classification. Theor. Comput. Sci. 294(1–2), 61–102 (2003)
3. Corradini, A., Duval, D., Echahed, R., Prost, F., Ribeiro, L.: AGREE - Algebraic Graph Rewriting with Controlled Embedding. In: ICGT 2015. LNCS, vol. 9151, pp. 35–51. Springer (2015)
4. Corradini, A., Heindel, T., Hermann, F., König, B.: Sesqui-pushout rewriting. In: ICGT 2006. LNCS, vol. 4178, pp. 30–45. Springer (2006)
5. Corradini, A., Montanari, U., Rossi, F., Ehrig, H., Heckel, R., Löwe, M.: Algebraic approaches to graph transformation - part I: basic concepts and double pushout approach. In: Rozenberg [18], pp. 163–246
6. Drewes, F., Hoffmann, B., Janssens, D., Minas, M.: Adaptive star grammars and their languages. Theor. Comput. Sci. 411(34-36), 3090–3109 (2010)
7. Duval, D., Echahed, R., Prost, F., Ribeiro, L.: Transformation of attributed structures with cloning. In: FASE 2014. LNCS, vol. 8411, pp. 310–324. Springer (2014)
8. Ehrig, H., Heckel, R., Korff, M., Löwe, M., Ribeiro, L., Wagner, A., Corradini, A.: Algebraic approaches to graph transformation - part II: single pushout approach and comparison with double pushout approach. In: Rozenberg [18], pp. 247–312
9. Ehrig, H., Pfender, M., Schneider, H.J.: Graph-grammars: An algebraic approach. In: 14th Annual Symposium on Switching and Automata Theory, Iowa City, Iowa, USA, October 15-17, 1973. pp. 167–180. IEEE Computer Society (1973)
10. Engelfriet, J., Rozenberg, G.: Node replacement graph grammars. In: Rozenberg [18], pp. 1–94
11. Heckel, R., Ehrig, H., Wolter, U., Corradini, A.: Double-pullback transitions and coalgebraic loose semantics for graph transformation systems. Applied Categorical Structures 9(1), 83–110 (2001)
12. Kahl, W.: Amalgamating pushout and pullback graph transformation in collagories. In: ICGT 2010. LNCS, vol. 6372, pp. 362–378. Springer (2010)
13. Lack, S., Sobociński, P.: Adhesive Categories. In: FOSSACS'04. LNCS, vol. 2987, pp. 273–288. Springer (2004)
14. Löwe, M.: Algebraic approach to single-pushout graph transformation. Theor. Comput. Sci. 109(1&2), 181–224 (1993)
15. Löwe, M.: Refined graph rewriting in span-categories - A framework for algebraic graph transformation. In: ICGT 2012. LNCS, vol. 7562, pp. 111–125. Springer (2012)
16. Mantz, F.: Coupled Transformations of Graph Structures applied to Model Migration. Ph.D. thesis, University of Marburg (2014)
17. Mantz, F., Taentzer, G., Lamo, Y., Wolter, U.: Co-evolving meta-models and their instance models: A formal approach based on graph transformation. Science of Computer Programming 104, 2–43 (2015)
18. Rozenberg, G. (ed.): Handbook of Graph Grammars and Computing by Graph Transformations, Volume 1: Foundations. World Scientific (1997)
19. Schulz, C., Löwe, M., König, H.: A categorical framework for the transformation of object-oriented systems: Models and data. J. Symb. Comput. 46(3), 316–337 (2011)
20. Wouters, L., Gervais, M.P.: Ontology Transformations. In: IEEE International Enterprise Distributed Object Computing Conference. pp. 71–80 (2012)