Single-Pushout Rewriting of Partial Algebras

Michael Löwe * and Marius Tempelmeier †

* Fachhochschule für die Wirtschaft (FHDW) Hannover
† Sennheiser electronic GmbH & Co. KG

Abstract We introduce Single-Pushout Rewriting for *arbitrary* partial algebras. Thus, we give up the usual restriction to graph structures, which are algebraic categories with unary operators only. By this generalisation, we obtain an integrated and straightforward treatment of graphical structures (objects) and attributes (data). We lose co-completeness of the underlying category. Therefore, a rule is no longer applicable at any match. We characterise the new application condition and make constructive use of it in some practical examples.

1 Introduction

The current frameworks for the (algebraic) transformation of typed graphs are not completely satisfactory from the software engineering perspective. For example, it is hardly possible to specify and handle associations with "at-most-one"-multiplicity, since most frameworks are based on some (adhesive) categories of graphs which allow multiple edges between the same pair of vertices.[1]

Another example is the handling of attributes. The standard approaches to the transformation of attributed graphs, compare for example [5,13], explicitly distinguish two parts, i. e. the *structure part* (objects and links) which can be changed by transformations and the base-type and -operation part (data) which is immutable. Typically, objects can be attributed with data via some special edges the source of which is in the structure part and the target of which is data. This set-up either leads to set-valued or immutable attribute structures. Both is not satisfactory from the software engineering point of view.[2]

Another problem in current frameworks for attributed graphs is the infiniteness of rules stipulated by the infiniteness of the term algebra which is typically used in the rules. Even if the algebra for the objects which are subject to transformation is finite (for example integers modulo some maximum), the term algebra tends to contain infinitely many terms.

All these problems are more or less caused by the usage of total algebras. In this paper, we use partial algebras instead as the underlying category for single-pushout rewriting. In partial algebras, operation definitions can be

[1] Typically, some negative application conditions [8] are employed to handle these requirements making the framework more complicated.

[2] In object-oriented programming languages, for example, attributes have the standard multiplicity `0..1`.

changed without deleting and adding an object (edge). Thus, we get a straightforward model for "at-most-one" associations. We also give up the distinction between structure and data, i. e. we allow arbitrary signatures which are able to integrate both parts. We lose co-completeness of the base category and import some application conditions into single-pushout rewriting. But we gain a seamless integration of structure and data. Finally, partial term algebras in the rules help to keep rules finite.

The paper is a short version of [15], where many more examples and details can be found. Section 2 introduces our concept of partial algebra. We show explicitly the similarities between partial algebras and hypergraphs. Section 3 provides sufficient and necessary conditions for the existence of pushouts in categories of partial algebras and partial morphisms. It contains our main results. Section 4 defines the new single-pushout approach and shows similarities and differences to the sesqui-pushout approach [3]. Section 5 demonstrates by some example that the new application conditions are useful in many situations. Finally, Section 6 discusses related work and provides some conclusions.

2 Graphs and Partial Algebras

In this section, we introduce the basic notions and constructions for partial algebras. We use a rather unusual approach in order to emphasise the close connection of categories of partial algebras to categories of hypergraphs. We employ a kind of objectification for partial mappings. A partial map $f : A \to B$ is not just a subset of $A \times B$ satisfying the uniqueness condition $(*)$ $(a, b_1), (a, b_2) \in f$ implies $b_1 = b_2$. Instead, a partial map $f : A \to B$ is a triple $(f, d_f : f \to A, c_f : f \to B)$ consisting of a set f of *map entries* together with two *total* maps $d_f : f \to A$ and $c_f : f \to B$ which provide the argument and the return value for every entry respectively. The uniqueness condition $(*)$ above translates to $\forall e_1, e_2 \in f : d_f(e_1) = d_f(e_2) \Longrightarrow e_1 = e_2$ in this set-up.

A *signature* $\Sigma = (S, O)$ consists of a set of sort names S and a domain- and co-domain-indexed family of operation names $O = (O_{w,v})_{w,v \in S^*}$.[3] A *graph* G wrt. a signature consists of a carrier set G_s (of vertices) for every sort name $s \in S$ and a set of hyperedges $\left(f^G, d_f^G : f^G \to G^w, c_f^G : f^G \to G^v\right)$ for every operation name $f \in O_{w,v}$ and $w, v \in S^*$ where the *total* mappings d_f^G and c_f^G provide the "arguments" and "return values".[4] A *graph morphism* $h : G \to H$ between to graphs G and H wrt. the same signature $\Sigma = (S, O)$ consists of a family of vertex mappings $h = (h_s : G_s \to H_s)_{s \in S}$ and a family of edge mappings $h^O = \left(h_f^O : f^G \to f^H\right)_{f \in O}$ such that for all operation names $f \in O_{w,v}$ and for

[3] Note that we generalise the usual notion of signature which allows single sorts as co-domain specification for operation names only. Operation names in $O_{w,\epsilon}$ will be interpreted as predicates, operation names in $O_{w,v}$ with $|v| > 1$ will be interpreted as operations which deliver several results simultaneously.

[4] For $w \in S^*$ and a family $(G_s)_{s \in S}$ of sets, G^w is recursively defined by (i) $G^\epsilon = \{*\}$, (ii) $G^w = G_s$ if $w = s \in S$ and (iii) $G^w = G^v \times G^u$ if $w = vu$.

all edges $e \in f^G$ the following homomorphism condition holds:[5]

$$\text{(h)}\ d_f^H\left(h_f^O(e)\right) = h^w\left(d_f^G(e)\right) \text{ and } c_f^H\left(h_f^O(e)\right) = h^v\left(c_f^G(e)\right).$$

The category of all graphs and graph morphisms wrt. a signature Σ is denoted by $\mathcal{G}_\Sigma$ in the following.[6] $\mathcal{G}_\Sigma$ is complete and co-complete. All limits and co-limits can be constructed component-wise on the underlying sets. The pullback for a co-span $B \overset{m}{\to} A \overset{n}{\leftarrow} C$ is given by the *partial product* $B \times_{(m,n)} C$ with the two *projection morphisms* $\pi_B^{B\times C} : B \times_{(m,n)} C \to B$ and $\pi_C^{B\times C} : B \times_{(m,n)} C \to C$:

$$\begin{aligned}
\forall s \in S : \left(B \times_{(m,n)} C\right)_s &= \{(x,y) :: m_s(x) = n_s(y)\} \\
\forall f \in O_{w,v} : f^{\left(B\times_{(m,n)}C\right)} &= \left\{(x,y) :: m_f^O(x) = n_f^O(y)\right\} \\
\forall f \in O_{w,v} : d_f^{\left(B\times_{(m,n)}C\right)} &::= (x,y) \mapsto d_f^B(x) \,||^w\, d_f^C(y) \\
\forall f \in O_{w,v} : c_f^{\left(B\times_{(m,n)}C\right)} &::= (x,y) \mapsto c_f^B(x) \,||^v\, c_f^C(y),
\end{aligned}$$

where the operator $||^w : B^w \times C^w \to (B \times C)^w$ transforms pairs of w-tuples into w-tuples of pairs: $\left(\left(x_1, \ldots, x_{|w|}\right), \left(y_1, \ldots, y_{|w|}\right)\right) \mapsto \left(\left(x_1, y_1\right), \ldots \left(x_{|w|}, y_{|w|}\right)\right)$.

A graph $G \in \mathcal{G}_{\Sigma=(S,O)}$ is a *partial algebra*, if it satisfies the following condition for all $f \in O$:

$$\text{(u)}\ \forall e_1, e_2 \in f^G : d_f^G(e_1) = d_f^G(e_2) \implies e_1 = e_2.$$

The full sub-category of $\mathcal{G}_\Sigma$ consisting of all partial algebras[7] is denoted by $\mathcal{A}_\Sigma$ in the following. In a partial algebra A, operation names $f \in O_{\epsilon,v}$ with $|v| > 0$ are interpreted as (partial) *constants*, i. e. $f^A : A^\epsilon \to A^v$ is a partial map from the standard one-element set $A^\epsilon = \{*\}$ into A^v. Symmetrically, operation names $p \in O_{w,\epsilon}$ with $|w| > 0$ are interpreted as *predicates*, since $p^A : A^w \to \{*\}$ is a partial map into the standard one-element set, i. e. it determines a subset on A^w only, namely the part of A^w where it is defined. Finally for operation names $f \in O_{\epsilon,\epsilon}$, there is only two possible interpretations in A, namely $f^A = \emptyset$ (false) or $f^A = \{(*,*)\}$ (true). Thus, f^A is just a boolean flag in this case.

Due to (u) being a set of Horn-axioms, $\mathcal{A}_\Sigma$ is an epireflective sub-category of $\mathcal{G}_\Sigma$, i.e. there is a reflection $\eta : \mathcal{G}_\Sigma \to \mathcal{A}_\Sigma$ that maps a graph $G \in \mathcal{G}_\Sigma$ to

[5] Given a sort indexed family of mappings $\left(f_s : G_s \to H_s\right)_{s\in S}$, $f^w : G^w \to H^w$ is recursively defined for every $w \in S^*$ by (i) $f^\epsilon = \{(*,*)\}$, (ii) $f^w = f_s$ if $w = s \in S$, and (iii) $f^w = f^v \times f^u$, if $w = vu$.

[6] The identity morphisms in $\mathcal{G}_\Sigma$ are given by families of identity mappings and composition of morphisms is provided by component-wise composition of the underlying mappings.

[7] Note that the interpretation of an operation name $f \in O_{w,v}$ in a partial algebra A is indeed a partial mapping: due to the uniqueness condition (u), the assignment $\left(f^A, d_f^A : f^A \to A^w, c_f^A : f^A \to A^v\right) \mapsto \left\{\left(d_f^A(e), c_f^A(e)\right) :: e \in f^A\right\}$ provides a partial map from A^w to A^v. And, for a partial map $f : A^w \to A^v$, there is the inverse mapping $f \mapsto \left(f, d_f^A ::= (d,c) \mapsto d, c_f^A ::= (d,c) \mapsto c\right)$ up to renaming of the elements in f^A.

a pair $\left(G^{\mathcal{A}} \in \mathcal{A}_\Sigma, \eta_G : G \to G^{\mathcal{A}}\right)$ such that any graph morphism $h : G \to A$ with $A \in \mathcal{A}_\Sigma$ has a unique extension $h^* : G^{\mathcal{A}} \to A$ with $h^* \circ \eta_G = h$. Since epireflective subcategories are closed wrt. products and sub-objects defined by regular monomorphisms (equalisers), the limits in $\mathcal{A}_\Sigma$ coincide with the limits constructed in $\mathcal{G}_\Sigma$. $\mathcal{A}_\Sigma$ has also all co-limits, since epireflections map co-limits to co-limits. In general, however, the co-limits in $\mathcal{A}_\Sigma$ do not coincide with the co-limits constructed in $\mathcal{G}_\Sigma$. The reflection provides the necessary correction. If, for example, $(b : A \to B, c : A \to C)$ is a span in $\mathcal{A}_\Sigma$ and $(c^* : B \to D, b^* : C \to D)$ is its pushout constructed in $\mathcal{G}_\Sigma$, $\left(\eta_D \circ c^* : B \to D^{\mathcal{A}}, \eta_D \circ b^* : C \to D^{\mathcal{A}}\right)$ is the pushout in $\mathcal{A}_\Sigma$.

Besides being complete and co-complete, the most important property of $\mathcal{A}_\Sigma$ for the rest of the paper is the existence of right adjoints to all inverse image functors. If we fix an algebra $A \in \mathcal{A}_\Sigma$, $\mathcal{A}_\Sigma \downarrow^{\mathrm{M}} A$ denotes the *category of all sub-algebras* of A. The objects in $\mathcal{A}_\Sigma \downarrow^{\mathrm{M}} A$ are all monomorphisms $m : M \rightarrowtail A$ and a morphism in $\mathcal{A}_\Sigma \downarrow^{\mathrm{M}} A$ from $m : M \rightarrowtail A$ to $n : N \rightarrowtail A$ is a (mono)morphism $h : M \rightarrowtail N$ in $\mathcal{A}_\Sigma$ such that $n \circ h = m$. For every $\mathcal{A}_\Sigma$-morphism $g : A \to B$, the *inverse image functor* $g^* : \mathcal{A}_\Sigma \downarrow^{\mathrm{M}} B \to \mathcal{A}_\Sigma \downarrow^{\mathrm{M}} A$ maps an object $m : M \to B \in \mathcal{A}_\Sigma \downarrow^{\mathrm{M}} B$ to $\pi_A^{A \times M} : A \times_{(g,m)} M \to A$ and a morphism $h : (m : M \to B) \to (n : N \to B)$ to the uniquely determined morphism $g^*(h) : A \times_{(g,m)} M \to A \times_{(g,n)} N$ such that $\pi_A^{A \times N} \circ g^*(h) = \pi_A^{A \times M}$ and $\pi_N^{A \times N} \circ g^*(h) = h \circ \pi_M^{A \times M}$.

Fact 1. *In a category $\mathcal{A}_\Sigma$ of partial algebras, every inverse image functor $g^* : \mathcal{A}_\Sigma {\downarrow^{\mathrm{M}}} B \to \mathcal{A}_\Sigma {\downarrow^{\mathrm{M}}} A$ has a right adjoint called $g_* : \mathcal{A}_\Sigma {\downarrow^{\mathrm{M}}} A \to \mathcal{A}_\Sigma {\downarrow^{\mathrm{M}}} B$.*

Proof. Given a sub-algebra $m : M \rightarrowtail A$, we construct the sub-algebra $g_*(M) \subseteq B$ and the inclusion morphism $g_*(m) : g_*(M) \hookrightarrow B$ as follows:

$$\forall s \in S : g_*(M)_s = \left\{b \in B_s :: \forall\, a \in g_s^{-1}(b)\ \exists\, x \in M : m_s(x) = a \quad \text{and}\right.$$
$$\left.\forall f \in O_{w,v} : f^{g_*(M)} = \left\{e \in f^B :: \forall\, e_a \in {g_f^O}^{-1}(e)\ \exists\, e_x \in M : m_f^O(e_x) = e_a\right\}\right.,$$

such that $d_f^{g_*(M)} = d^B_{f\ |f^{g_*(M)}}$ and $c_f^{g_*(M)} = c^B_{f\ |f^{g_*(M)}}$ for every operation symbol.

The co-unit $\varepsilon : g^*\left(g_*\left(m : M \rightarrowtail A\right)\right) \to (m : M \rightarrowtail A)$ can be defined on every element $(a,b) \in A \times_{(g,g_*(m))} g_*(M)$ by $\varepsilon(a,b) = c$ such that $m(c) = a$. Note that ε is completely defined, since, by definition of $g_*(m)$, a must have a pre-image wrt. m for every pair $(a,b) \in A \times_{(g,g_*(m))} g_*(M)$. It is uniquely defined, since m is monic. By definition of ε, $m \circ \varepsilon = g^*(g_*(m)) = \pi_A^{A \times_{(g,g_*(m))} g_*(M)}$ which means that ε is a morphism in $\mathcal{A}_\Sigma {\downarrow^{\mathrm{M}}} A$.

Now, let an object $x : X \rightarrowtail B \in \mathcal{A}_\Sigma \downarrow^{\mathrm{M}} B$ and a morphism $k : g^*(x) \to m \in \mathcal{A}_\Sigma \downarrow^{\mathrm{M}} A$, i.e. $m \circ k = \pi_A^{A \times_{(g,x)} X}$ be given. We construct $k^* : x \to g_*(m)$ by $e \mapsto x(e)$ for every $e \in X$. The mappings of k^* are completely defined: (i) if $x(e) \notin g(A)$, $x(e) \in g_*(M)$ because $|g^{-1}(x(e))| = \left|(g \circ m)^{-1}(x(e))\right| = 0$, and, otherwise, the existence of k with $m \circ k = \pi_A^{A \times_{(g,x)} X}$ enforces that every g-pre-image of $x(e)$ has a pre-image under m. By definition, $g_*(m) \circ k^* = x$. By definition of the inverse image functor, $g^*(k^*) : \left(A \times_{(g,x)} X\right) \to \left(A \times_{(g,g_*(m))} g_*(M)\right)$

maps (a, e) to $(a, k^*(e))$. Thus, $\varepsilon\left(g_*(k^*)(a, e)\right) = \varepsilon(a, k^*(e)) = c$ with $m(c) = a$ and $k(a, e) = c'$ with $m(c') = \pi_A^{A \times_{(g,x)} X}(a, e) = a$. Since m is monic, $c = c'$. The morphism k^* is uniquely determined, since $g_*(M) \subseteq B$ and $g_*(m)$ is monic. □

3 Partial Morphisms on Partial Algebras

In order to obtain a framework for single-pushout rewriting, we proceed from the category $\mathcal{A}_\Sigma$ of partial algebras with *total* morphisms to the category $\mathcal{A}_\Sigma^{\mathrm{P}}$ of partial algebras and *partial* morphisms. In this section, we investigate the conditions under which pushouts can be constructed in $\mathcal{A}_\Sigma^{\mathrm{P}}$.

A concrete partial morphism over an arbitrary complete category $\mathcal{C}$ is a span of $\mathcal{C}$-morphisms $(p : K \rightarrowtail P, q : K \to Q)$ such that p is monic. Two concrete partial morphisms (p_1, q_1) and (p_2, q_2) are equivalent and denote the same *abstract partial morphism* if there is an isomorphism i such that $p_1 \circ i = p_2$ and $q_1 \circ i = q_2$; in this case we write $(p_1, q_1) \equiv (p_2, q_2)$ and $[(p, q)]_\equiv$ for the class of spans that are equivalent to (p, q). The *category of partial morphisms* $\mathcal{C}^{\mathrm{P}}$ over $\mathcal{C}$ has the same objects as $\mathcal{C}$ and abstract partial morphisms as arrows. The identities are defined by $\mathrm{id}_A^{\mathcal{C}^{\mathrm{P}}} = [(\mathrm{id}_A, \mathrm{id}_A)]_\equiv$ and composition of partial morphisms $[(p : K \rightarrowtail P, q : K \to Q)]_\equiv$ and $[(r : J \rightarrowtail Q, s : J \to R)]_\equiv$ is given by

$$[(r, s)]_\equiv \circ_{\mathcal{C}^{\mathrm{P}}} [(p, q)]_\equiv = [(p \circ r' : M \rightarrowtail P, s \circ q' : M \to R)]_\equiv$$

where $(M, r' : M \rightarrowtail K, q' : M \to J)$ is an arbitrarily chosen but fixed pullback of q and r. Note that there is the faithful embedding functor $\iota : \mathcal{C} \to \mathcal{C}^{\mathrm{P}}$ defined by identity on objects and $(f : A \to B) \mapsto [\mathrm{id}_A : A \rightarrowtail A, f : A \to B]$ on morphisms. We call $[d : A' \rightarrowtail A, f : A' \to B]$ a *total* morphism and, by a slight abuse of notation, write $[d, f] \in \mathcal{C}$, if d is an isomorphism. From now on, we mean the abstract partial morphism $[f, g]_\equiv$ if we write $(f : B \rightarrowtail A, g : B \to C)$.

The single-pushout approach defines direct derivations by a single pushout in a category of partial morphisms. There is a general result for the existence of pushouts in a category $\mathcal{C}^{\mathrm{P}}$ of partial morphisms based on the notions *final triple* and *hereditary pushout* in the underlying category $\mathcal{C}$ of total morphisms.

Definition 2. *(Final triple) A* triple *for a pair* $((l, r), (p, q))$ *of* $\mathcal{C}^{\mathrm{P}}$*-morphisms with common domain is given by a collection* $\left(\overline{p}, p^*, \overline{r}, \overline{l}, l^*, \overline{q}\right)$ *of* $\mathcal{C}$*-morphisms such that* p^*, $\overline{p}$, l^*, *and* $\overline{l}$ *are monic and* (i) $(\overline{r}, \overline{p})$ *is pullback of* (r, p^*), (ii) $(\overline{q}, \overline{l})$ *is pullback of* (q, l^*), *and* (iii) $l \circ \overline{p} = p \circ \overline{l}$. *A triple* $\left(\overline{p}, p^*, \overline{r}, \overline{l}, l^*, \overline{q}\right)$ *for* $((l, r), (p, q))$ *is* final, *if, for any other triple* $\left(p', p'^*, r', l', l'^*, q'\right)$, *there is a unique collection* (u_1, u_2, u_3) *of* $\mathcal{C}$*-morphisms such that* (iv) $\overline{p} \circ u_1 = p'$, (v) $\overline{l} \circ u_1 = l'$, (vi) $p^* \circ u_2 = p'^*$, (vii) $u_2 \circ r' = \overline{r} \circ u_1$, (viii) $l^* \circ u_3 = l'^*$, *and* (ix) $u_3 \circ q' = \overline{q} \circ u_1$, *compare left part of Fig. 1.*

Definition 3. *(Hereditary pushout) A pushout* (q', p') *of* (p, q) *in* $\mathcal{C}$ *is* hereditary *if for each commutative cube as in the right part of Fig. 1, which has pullback squares* (p_i, i_0) *and* (q_i, i_0) *of* (i_2, p) *and* (i_1, q) *resp. as back faces such that* i_1 *and* i_2 *are monomorphisms, in the top square,* (q_i', p_i') *is pushout of* (p_i, q_i), *if*

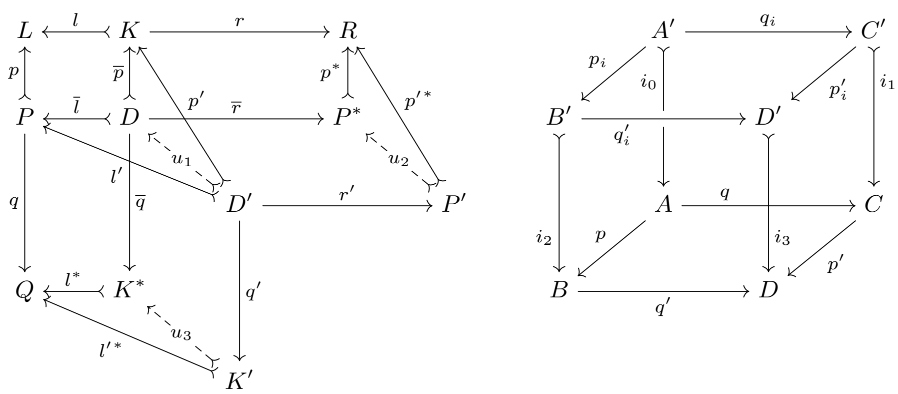

Figure 1. Final Triple and Hereditary Pushout

and only if, in the front faces, (p_i', i_1) *and* (q_i', i_2) *are pullbacks of* (i_3, p') *and* (i_3, q') *resp. and* i_3 *is monic.*[8]

Fact 4. *(Pushout in* $\mathcal{C}^{\mathrm{P}}$*) Two partial morphisms* $(l : K \rightarrowtail L, r : K \to R)$ *and* $(p : P \rightarrowtail L, q : P \to Q)$ *have a pushout* $((l^*, r^*), (p^*, q^*))$ *in* $\mathcal{C}^{\mathrm{P}}$*, if and only if there is* (i) *a* final triple $\bar{l} : D \to P$, $\overline{p} : D \to K$, $\overline{r} : D \to P^*$, $\overline{q} : D \to K^*$, $p^* : P^* \to R$, $l^* : K^* \to Q$ *for* $((l, r), (p, q))$ *and* (ii) *a* hereditary pushout $(r^* : K^* \to H, q^* : P^* \to H)$ *for* $(\overline{q}, \overline{r})$ *in* $\mathcal{C}$*, compare sub-diagrams (1) – (3) and (4) resp. in Figure 2.*

$$
\begin{tikzcd}
L & K \arrow[l, tail, "l"'] \arrow[r, "r"] & R \\
P \arrow[u, tail, "p"] \arrow[d, "q"'] \arrow[ur, phantom, "(1)"] & D \arrow[l, tail, "\bar{l}"'] \arrow[r, "\overline{r}"] \arrow[u, tail, "\overline{p}"'] \arrow[d, "\overline{q}"] \arrow[ur, phantom, "(2)"] \arrow[dl, phantom, "(3)"] \arrow[dr, phantom, "(4)"] & P^* \arrow[u, tail, "p^*"'] \arrow[d, "q^*"] \\
Q & K^* \arrow[l, tail, "l^*"] \arrow[r, "r^*"'] & H
\end{tikzcd}
$$

Figure 2. *Pushout in* $\mathcal{G}^{\mathrm{P}}$

The proof can be found in [14]. A version of the proof that does not presuppose a choice of pullbacks that is compatible with pullback composition and decomposition is contained in [15].

Since $\mathcal{A}_\Sigma$ is complete, we can construct the category $\mathcal{A}_\Sigma^{\mathrm{P}}$ of partial algebras and partial morphisms. We use the partial product as the chosen pullback for morphism composition, compare above. The general results about pushouts of partial morphisms carry over to $\mathcal{A}_\Sigma^{\mathrm{P}}$ as follows:

[8] For details on hereditary pushouts see [10,11]

Proposition 5. *(Final triple) Every pair* $((l,r),(p,q))$ *of* $\mathcal{A}^{\mathrm{P}}_{\Sigma}$*-morphisms with common domain has a final triple.*

Proof. *(Sketch)* The existence of final triples follows from $\mathcal{A}^{\mathrm{P}}_{\Sigma}$ being co-complete and having right adjoints for all inverse image functors, compare Fact 1. A detailed proof can be found in [15].

Corollary 6. *(Pushout in* $\mathcal{A}^{\mathrm{P}}_{\Sigma}$*) A pair of morphisms* $(l : K \rightarrowtail L, r : K \to R)$ *and* $(p : P \rightarrowtail L, q : P \to Q)$ *has a pushout in* $\mathcal{A}^{\mathrm{P}}_{\Sigma}$*, if and only if the* $\mathcal{A}_{\Sigma}$*-pushout of* $(\overline{q}, \overline{r})$ *is hereditary, where* $\overline{l} : D \to P$*,* $\overline{p} : D \to K$*,* $\overline{r} : D \to P^*$*,* $\overline{q} : D \to K^*$*,* $p^* : P^* \to R$*,* $l^* : K^* \to Q$ *is final triple of* $((l,r),(p,q))$*, see Figure 2.*

Proof. Direct consequence of Fact 4 and Proposition 5.

It is well-known that all pushouts in the category of sets and mappings and in arbitrary categories $\mathcal{G}_{\Sigma}$ of graphs over a given signature are hereditary. This provides the following sufficient criterion for hereditary pushouts in $\mathcal{A}_{\Sigma}$.

Proposition 7. *(Sufficient condition) If a pushout in* $\mathcal{A}_{\Sigma}$ *is also pushout in the larger category* $\mathcal{G}_{\Sigma}$ *of graphs, then it is hereditary in* $\mathcal{A}_{\Sigma}$*.*

Proof. Let an arbitrary commutative cube as in the right part of Fig. 1 in $\mathcal{A}_{\Sigma}$ be given such that the back faces are pullbacks. Then this is also a situation in $\mathcal{G}_{\Sigma}$ and the back faces are also pullbacks in $\mathcal{G}_{\Sigma}$, due to $\mathcal{A}_{\Sigma}$ being an epireflection of $\mathcal{G}_{\Sigma}$.

Let the front faces be pullbacks in $\mathcal{A}_{\Sigma}$ and i_3 be a monomorphism. Then the front faces are also pullbacks in $\mathcal{G}_{\Sigma}$. Since all pushouts in $\mathcal{G}_{\Sigma}$ are hereditary, D' together with p'_i and q'_i is pushout in $\mathcal{G}_{\Sigma}$. Since (i) $\mathcal{A}_{\Sigma}$ is closed wrt. sub-algebras, (ii) D is in $\mathcal{A}_{\Sigma}$, and (iii) i_3 is monic, D' is also in $\mathcal{A}_{\Sigma}$ and its reflector $\eta_{D'}$ is an isomorphism. Thus, D' together with p'_i and q'_i is pushout in $\mathcal{A}_{\Sigma}$.

Let (D', q'_i, p'_i) be pushout of (p_i, q_i) in $\mathcal{A}_{\Sigma}$. Construct (D'', q''_i, p''_i) as pushout of (p_i, q_i) in $\mathcal{G}_{\Sigma}$. We obtain the epic reflector $\eta_{D''} : D'' \twoheadrightarrow D'$ with $p'_i = \eta_{D''} \circ p''_i$ and $q'_i = \eta_{D''} \circ q''_i$. Since D'' is pushout, we also get $i'_3 : D'' \rightarrowtail D$ with $i'_3 \circ p''_i = p' \circ i_1$ and $i'_3 \circ q''_i = q' \circ i_2$. Since $i_3 \circ \eta_{D''} \circ p''_i = i_3 \circ p'_i = p' \circ i_1 = i'_3 \circ p''_i$ and $i_3 \circ \eta_{D''} \circ q''_i = i_3 \circ q'_i = q' \circ i_2 = i'_3 \circ q''_i$, we can conclude $i_3 \circ \eta_{D''} = i'_3$. Since all pushouts in $\mathcal{G}_{\Sigma}$ are hereditary, i'_3 is monic implying that $\eta_{D''}$ is monic as well. Thus, $\eta_{D''}$ is an isomorphism and D' is also the pushout in $\mathcal{G}_{\Sigma}$. This immediately provides monic i_3 and pullbacks in the front faces of the cube in the right part ofFig. 1. □

But not all pushouts in $\mathcal{A}_{\Sigma}$ are hereditary. Here is a typical example:

Example 8. Consider the signature $\Sigma^c = (S_c, O^c)$ with

$$S_c = \{s\}$$
$$O^c_{w,v} = \begin{cases} \{f\} & w = \epsilon, v = s \\ \emptyset & \text{otherwise,} \end{cases}$$

$$
\begin{tikzcd}
& a \arrow[rr, "q_i = q"] \arrow[dl, "p_i"'] \arrow[dd, tail, "i_0 = \mathrm{id}_A"] & & c \dashleftarrow f^C \arrow[dl, "p'_i = p'"] \arrow[dd, tail, "i_1 = \mathrm{id}_C"] \\
b \arrow[rr, "q'_{i\,s} = q'_s"'] \arrow[dd, tail, "i_{2\,s} = \mathrm{id}_{B_s}"'] & & d \dashleftarrow f^D \arrow[dd, tail, "i_3 = \mathrm{id}_D"] & \\
& a \arrow[rr, "q"] \arrow[dl, "p"'] & & c \dashleftarrow f^C \arrow[dl, "p'"] \\
b \dashleftarrow f^B \arrow[rr, "q'"'] & & d \dashleftarrow f^D &
\end{tikzcd}
$$

Figure 3. Simple Non-Hereditary Pushout in $\mathcal{A}_\Sigma$

the three algebras

$$
\begin{aligned}
&A ::= A_s = \{a\},\ f^A = \emptyset, \\
&B ::= B_s = \{b\},\ f^B = \left(\{f^B\},\ d_f^B(f^B) = *,\ c_f^B(f^B) = b\right), \\
&C ::= C_s = \{c\},\ f^C = \left(\{f^C\},\ d_f^C(f^C) = *,\ c_f^C(f^C) = c\right),
\end{aligned}
$$

and the two morphisms $p : A \to B ::= a \mapsto b$ and $q : A \to C ::= a \mapsto c$.

The pushout of (p, q) in $\mathcal{A}^{\mathrm{P}}_{\Sigma^c}$ consists of the algebra

$$D\ ::=\ D_s = \{d\},\ f^D = \left(\{f^D\},\ d_f^D(f^D) = *,\ c_f^D(f^D) = d\right)$$

and the two morphisms

$$
\begin{aligned}
p' &: C \to D ::= c \mapsto d, f^C \mapsto f^D \\
q' &: B \to D ::= b \mapsto d, f^B \mapsto f^D.
\end{aligned}
$$

This pushout is depicted at the bottom of Fig. 3 and is not hereditary. We construct the following cube of morphisms, compare Fig. 3: $A' = A$, $i_0 = \mathrm{id}_A$, B' is defined by $B'_s = B_s$ and $f^{B'} = \emptyset$, i_2 maps b in B'_s to b in B_s, $C' = C$, $i_1 = \mathrm{id}_C$, $q_i = q$, and p_i maps a to b. Note that (i_0, q_i) is pullback of (q, i_1) and (i_0, p_i) is pullback of (p, i_2). Constructing $(D' = D, p'_i = p', q'_i ::= b \mapsto d)$ as the pushout of (p', q'), we obtain $i_3 = \mathrm{id}_D$. But (i_2, q'_i) is not pullback of (q', i_3): $B \times_{(q', i_3)} D'$ contains a defined constant for f, since $i_3(f^D) = q'(f^B)$, and B' does not. $\square$

Note that the $\mathcal{A}_\Sigma$-pushout of the morphisms p and q in Example 8 does not coincide with the pushout of p and q constructed in the larger category $\mathcal{G}_\Sigma$ of graphs. The pushout in $\mathcal{G}_\Sigma$ is the graph

$$G ::= G_s = \{g\}, f^G = \left(\{f_C^G, f_B^G\}, d_f^G(f_C^G) = d_f^G(f_B^G) = *, c_f^G(f_C^G) = c_f^G(f_B^G) = g\right)$$

together with the morphisms

$$
\begin{aligned}
p'' &: C \to G ::= c \mapsto g, f^C \mapsto f_C^G \\
q'' &: B \to G ::= b \mapsto g, f^B \mapsto f_B^G.
\end{aligned}
$$

The partial algebra D is the epireflection of the graph G and the reflector $\eta_G : G \to D$ maps as follows: $g \mapsto d$, $f_C^G \mapsto f^D$, and $f_B^G \mapsto f^D$. The identification $\eta_G(f_C^G) = \eta_G(f_B^G) = f^D$ of the reflector provided the possibility to construct the cube in Example 8 that disproves hereditariness of the pushout of (p, q). The following proposition shows that this construction of a counterexample is always possible if the pushouts in $\mathcal{A}_\Sigma$ and $\mathcal{G}_\Sigma$ are different.

Proposition 9. *(Necessary condition) If a pushout in $\mathcal{A}_\Sigma$ is hereditary, it is also pushout in the larger category $\mathcal{G}_\Sigma$ of graphs.*

Proof. Let $(p : A \to B, q : A \to C)$ be a span of morphisms in $\mathcal{A}_\Sigma$, let $(E, q'' : B \to E, p'' : C \to E)$ be its pushout in $\mathcal{G}_\Sigma$, and let $(q' : B \to D, p' : C \to D)$ be its pushout in $\mathcal{A}_\Sigma$. Since $\mathcal{A}_\Sigma$ is epireflective sub-category of $\mathcal{G}_\Sigma$, we know that $D = E^{\mathcal{A}}$, $q' = \eta_E \circ q''$ and $p' = \eta_E \circ p''$ where $\eta_E : E \to E^{\mathcal{A}}$ is the reflector for the graph E. Suppose D and E are not isomorphic, then there are $e_1 \neq e_2$ with $\eta_E(e_1) = \eta_E(e_2)$. We distinguish two cases: $e_1, e_2 \in E_s$ for some sort $s \in S$ and $e_1, e_2 \in f^E$ for some operation name $f \in O_{w,v}$ and $w, v \in S^*$.

In the first case, construct the following commutative cube, compare right part of Fig. 1: A', B', and C' have the same carrier sets as A, B, and C respectively, their operations, however, are completely undefined. The embeddings i_0, i_1, and i_2 are identities on the carriers and empty mappings on the operations. The morphisms q_i and p_i coincide with q and p respectively on the carriers and are empty for all operations. Note that (q_i, i_0) and (p_i, i_0) are pullbacks of (q, i_1) and (p, i_2) respectively. Construct $(q'_i : B' \to D', p'_i : C' \to D')$ as the pushout of (p_i, q_i) in $\mathcal{G}_\Sigma$. Since all operations are undefined, it is also pushout in $\mathcal{A}_\Sigma$. And we know, that $D'_s = E_s$ for all sorts $s \in S$. Thus, $e_1 \neq e_2$ in D' and i_3 is not monic.

In the second case, we can, without loss of generality, suppose $E_s = D_s$ for all sorts $s \in S$. Since p' and q' are jointly epic, both e_1 and e_2 have pre-images under p' and/or q'. Let $e'_1, e'_2 \in f^B \uplus f^C$ be those pre-images and suppose, without loss of generality, $e'_1 \in f^B$. Since $e_1 \neq e_2$, we conclude $[e'_1]_{\equiv^f} \neq [e'_2]_{\equiv^f}$, where the equivalence $\equiv^f \subseteq \left(f^B \uplus f^C\right) \times \left(f^B \uplus f^C\right)$ is generated by $\left\{\left(p_f^O(e), q_f^O(e)\right) :: e \in f^A\right\}$. Construct the following cube à la Fig. 1(right part): The algebras A', B', and C' coincide in all carriers and operations except f with A, B, and C respectively. For f, we let

$$\begin{aligned}
f^{B'} &= f^B - \{e \in [e'_1]_{\equiv^f} :: e \in f^B\} \\
f^{C'} &= f^C - \{e \in [e'_1]_{\equiv^f} :: e \in f^C\} \\
f^{A'} &= f^A - \{e \in f^A :: q_f^O(e) \in [e'_1]_{\equiv^f} \vee p_f^O(e) \in [e'_1]_{\equiv^f}\}.
\end{aligned}$$

By this construction, we erase the whole structure that generated $[e'_1]_{\equiv^f}$ from A, B, and C. Note that, due to $[e'_1]_{\equiv^f} \neq [e'_2]_{\equiv^f}$, e'_2 is kept in f^B or f^C. Let i_0, i_1, and i_2 be the natural inclusions. And let q_i and p_i be the restrictions of q and p to A'. Since we erased the whole equivalence class $[e'_1]_{\equiv^f}$, (q_i, i_0) and (p_i, i_0) are pullbacks of (q, i_1) and (p, i_2) respectively. Let (D', q'_i, p'_i) be the pushout of (q_i, p_i). Then, (i_2, q'_i) is not pullback of (i_3, q'): By assumption,

$q'(e'_1) = e_1 = i_3(x)$ where $x = q'_i(e'_2)$ or $x = p'_i(e'_2)$. The function entry e'_1, however, does not possess a pre-image under i_2. □

Theorem 10. *A pushout in $\mathcal{A}_\Sigma$ is hereditary, if and only if it is pushout in $\mathcal{G}_\Sigma$.*

Proof. Direct consequence of Propositions 7 and 9.

Corollary 11. *Morphisms $(l : K \rightarrowtail L, r : K \to R)$ and $(p : P \rightarrowtail L, q : P \to Q)$ have a pushout in $\mathcal{A}^{\mathrm{P}}_\Sigma$, if and only if the $\mathcal{A}_\Sigma$-pushout of $(\overline{q}, \overline{r})$ is pushout in $\mathcal{G}_\Sigma$, where $\left(\overline{l}, \overline{p}, \overline{r}, \overline{q}, p^*, l^*\right)$ is final triple for $((l, r), (p, q))$, compare Figure 2.*

4 Single-Pushout Rewriting of Partial Algebras

In this section, we introduce single-pushout rewriting of partial algebras. We restrict rules to partial morphisms $(l : K \rightarrowtail L, r : K \rightarrowtail R)$ that do not identify items, i. e. the right-hand side of which are injective. Furthermore, we only allow matches that produce total co-matches. For this set-up, we can characterise the application conditions stipulated by the absence of some pushouts in categories of partial algebras with partial morphisms. And we can show a close connection of single-pushout and sesqui-pushout rewriting. In the following, let $\mathcal{A}^{\mathrm{P}}_\Sigma$ be a category of partial algebras and partial morphisms with respect to a given signature $\Sigma = \left(S, (O_{w,v})_{w,v \in S^*}\right)$.

Definition 12. *(Rule, match, and transformation) A* transformation rule *t is a* partial *morphism $t = (l : K \rightarrowtail L, r : K \rightarrowtail R)$ the right-hand side r of which is injective. A* match *for a rule $t : L \to R$ in a host algebra G is a* total *morphism $m : L \to G$. A* direct transformation *with a rule $t : L \to R$ at a match $m : L \to G$ from algebra G to algebra $t@m$ exists if there is a* total co-match *$m\langle t\rangle : R \to t@m$ and a partial* trace *$t\langle m\rangle : G \to t@m$, such that $(t\langle m\rangle, m\langle t\rangle)$ is pushout of t and m in $\mathcal{A}^{\mathrm{P}}_\Sigma$.*

There are two reasons why a transformation with a rule r at a match m cannot be performed: (i) There is no pushout of t and m in $\mathcal{A}^{\mathrm{P}}_\Sigma$ and (ii) the co-match in the pushout of t and m is not total. Therefore, we have some application conditions as in the double-pushout approach [5]. Since we restricted the rules to right-hand sides which do not identify any items, the application conditions can easily be characterised.[9]

Proposition 13. *(Application conditions) A transformation with a rule $t : L \to R$ at a match $m : L \to G$ exists, if and only if*

1. *the match does not identify items that are preserved with items that are deleted by the rule, i. e. for all $x \neq y \in L : m(x) = m(y)$ and t defined for x implies that t is also defined for y,*

[9] Note that Definition 12 can be generalised to arbitrary right-hand sides in rules. In the general case, however, the application condition introduced by the requirement that participating pushouts are hereditary is more complex.

$$
\begin{tikzcd}
 & & K' \arrow[dll, tail, "x"'] \arrow[dl, tail, "w"] \arrow[ddd, "y"] \\
L \arrow[d, "m"] & K \arrow[l, tail, "s^*"] \arrow[d, "m^*"] & \\
G & D \arrow[l, tail, "s"] & \\
 & & D' \arrow[ull, tail, "z"] \arrow[ul, tail, dashed, "w^*"']
\end{tikzcd}
\qquad
\begin{tikzcd}
L \arrow[d, "m"'] \arrow[dr, phantom, "(3)"] & K \arrow[l, tail, "l"'] \arrow[r, tail, "r"] \arrow[d, "m\langle l\rangle"] \arrow[dr, phantom, "(4)"] & R \arrow[d, "m\langle t\rangle"] \\
G & K^* \arrow[l, tail, "l^*"] \arrow[r, tail, "r^*"'] & t@m
\end{tikzcd}
$$

Figure 4. Single- versus Sesqui-Pushout Transformation

2. *the rule does not add operation definitions that are already present in the host graph G, i. e. for all $w, v \in S^*, f \in O_{w,v}, x \in L^w, e_R \in f^R, e_G \in f^G$:*

$$d_f^R(e_R) = t^w(x) \wedge d_f^G(e_G) = m^w(x) \implies \exists e_L \in f^L : m(e_L) = e_G,$$

3. *and the match does not identify domains of different added operation definitions, i. e. for all $w, v \in S^*, f \in O_{w,v}, e_1 \neq e_2 \in f^R$:*

$$m\,\langle t\rangle^w \left(d_f^R(e_1)\right) = m\,\langle t\rangle^w \left(d_f^R(e_2)\right) \implies \exists e_1' \in f^L : e_1 = t_f^O(e_1').$$

Note that the second clause above also implies $t(e_L) = e_R$ and the third clause also implies $\exists e_2' : e_2 = t_f^O(e_2')$.

Proof. The first condition is the well-known condition which is called *conflict-free* in [12] that characterises matches that produce pushouts in $\mathcal{G}_\Sigma$ with total co-match. Conditions 2 and 3 translate the result of Corollary 11 to the concrete situation where r is monic and p, $\overline{p}$, and p^* are isomorphisms. □

Since we restricted transformations to total co-matches, we obtain a close connection of our transformations to Sesqui-Pushout Rewritings in the sense of [3], which are composed of final pullback complements and pushouts.

Definition 14. *(Final Pullback Complement) In a pullback (s^*, m^*) of (m, s), compare left part of Fig. 4, the pair (s, m^*) constitutes a final pullback complement of (m, s^*), if for any other pullback (x, y) of (m, z) and morphism w such that $s^* \circ w = x$ there is a unique morphism w^* with $s \circ w^* = z$ and $w^* \circ y = m^* \circ w$.*

Theorem 15. *(Single- and Sesqui-Pushout Transformation) Given a rule $t = (l : K \rightarrowtail L, r : K \rightarrowtail R)$, a match $m : L \to G$, and a direct transformation $(m\,\langle t\rangle, t\,\langle m\rangle = (l^* : K^* \rightarrowtail G, r^* : K^* \rightarrowtail t@m))$, then there is a total morphism $m\,\langle l\rangle : K \to K^*$ such that* $(l^*, m\,\langle l\rangle)$ and $(r^*, m\,\langle l\rangle)$ are final pullback complements of (m, l) and $(m\,\langle t\rangle, r)$ resp., compare (3) and (4) in Fig. 4.

Proof. That $(l^*, m\,\langle l\rangle)$ is final pullback complement of (m, l) is a direct consequence of the construction of final triples in [15] and the fact that the co-match

$$
\begin{tikzcd}
& A \arrow[rr, tail, "\mathrm{id}_A"] \arrow[dl, tail, "\mathrm{id}_A"'] \arrow[dd, "n"'] & & A \arrow[dl, tail, "k"] \arrow[dd, "n"] \\
A \arrow[rr, tail, "k"'] \arrow[dd, "n"'] & & B \arrow[dd, "n^*"] & \\
& C \arrow[rr, tail, "\mathrm{id}_C"] \arrow[dl, tail, "\mathrm{id}_C"'] & & C \arrow[dl, tail, "k^*"] \\
C \arrow[rr, tail, "k^*"'] & & D &
\end{tikzcd}
$$

Figure 5. Hereditary Pushout and Final Pullback Complement

is total. It remains to show that hereditary pushouts along monomorphisms are final pullback complements as well: Let (k^*, n^*) be hereditary pushout of n and monic k. We construct the commutative cube in Fig. 5, in which the morphisms k, id_C, and id_A are monic, the top and left face are pullbacks and the back face is pushout. By (k^*, n^*) being hereditary, the bottom and the right face are pullbacks and k^* is monic. Moreover in the left face, (id_C, n) is final pullback complement of (n, id_A) and the back face is hereditary. Thus, the left and the back face constitute a pushout in the category of partial morphisms. The front face is hereditary and, therefore, pushout in the category of partial morphisms. Since $(\mathrm{id}_A, \mathrm{id}_A) = (k, \mathrm{id}_A) \circ (\mathrm{id}_A, k)$ and $(\mathrm{id}_C, \mathrm{id}_C) = (k^*, \mathrm{id}_C) \circ (\mathrm{id}_C, k^*)$, the right face must be pushout in the category of partial morphisms and (k^*, n) must be final pullback complement of (n^*, k). □

Thus, single-pushout rewriting with right-linear rules and total co-matches is almost Sesqui-Pushout Rewriting. This analysis provides good chances to reestablish most of the theory known for the single- and the sesqui-pushout approach, for example with respect to parallel and sequential independence, concurrency, and amalgamation. And it shows that our approach is closely connected to some other current research lines, for example [4]. But the application conditions for transformations in $\mathcal{A}^{\mathrm{P}}_{\Sigma}$ in Proposition 13 also produce some *new* and *unfamiliar* behaviour, for example if decomposition of rules is concerned.

Example 16. (Transformation Decomposition) In the standard single-pushout approach at injective matches, rule decomposition carries over to transformations: If a rule t can be decomposed into two rules t_1 and t_2, i. e. $t = t_2 \circ t_1$, every transformation with rule t at an injective match m can be decomposed into a transformation with t_1 followed by a transformation with t_2, such that $t\langle m\rangle = t_2 \langle m \langle t_1\rangle\rangle \circ t_1 \langle m\rangle$ and $m\langle t\rangle = (m\langle t_1\rangle)\langle t_2\rangle$. This is no longer true in the new set-up. Consider again the signature of Example 8, the partial algebras

$$
\begin{aligned}
&L ::= L_s = \{l\},\ f^L = \emptyset,\\
&E ::= E_s = \{e\},\ f^E = \left(\{f^E\},\ d^E_f(f^E) = *,\ c^E_f(f^E) = e\right),\\
&R ::= R_s = \{r\},\ f^R = \emptyset, \text{ and}\\
&G ::= G_s = \{g\},\ f^G = \left(\{f^G\},\ d^G_f(f^G) = *,\ c^G_f(f^G) = g\right),
\end{aligned}
$$

the rules $t_1 : L \to E ::= l \mapsto e$ and $t_2 : E \to R ::= e \mapsto r$, and the match $m : L \to G ::= l \mapsto g$. Note that t_2 is partial, since it does not map the operation definition in E. Since $t_2 \circ t_1 : L \to R$ is a rule without new operation definitions in R, there is the transformation $(t_2 \circ t_1)\,@m$. The rule t_1, however, cannot be applied at m due to a violation of the second condition in Proposition 13. $\square$

The careful analysis of these new features is left to future research.

5 Examples

The new behaviour discovered in Example 16, can be usefully exploited in many practical applications as a condition that prevents rule application. Our first

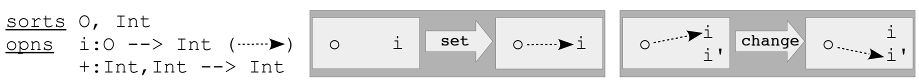

Figure 6. Setting and Changing an Attribute

example is a simple integer attribute `i` that can be set or changed for objects of type `O`. Figure 6 shows the underlying signature[10] and the two rules. Note that the `set`-rule can only be applied in a situation where the `i`-attribute of `o` has not been set yet, compare the second condition in Proposition 13. If there is an old value, the `change`-rule must be applied.

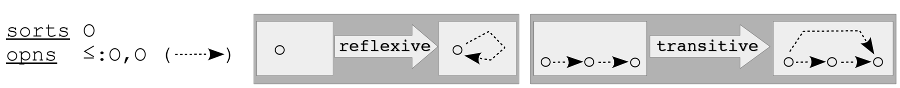

Figure 7. Reflexive/Transitive Closure

The next example handles the reflexive and transitive closure of a relation on the set `O`. We just apply the two rules `reflexive` and `transitive` as long as there are matches. Note that the algorithm terminates, since the rule `reflexive` cannot add loops to objects that possess a loop already, compare the second condition in Proposition 13. If all abbreviations are added, also the rule `transitive` is not applicable any more.

The last example shows a typical copying process, here for a tree structure, compare Figure 8. The partial dyadic operation `t` builds up trees, the unary predicate `r` marks the root for the copy process, the `start` and the three `copy` rules perform the copy process, and the operation `c` keeps track of already built copies. Again, the application conditions of single pushout rewriting for partial

[10] In the signature, we declare the visualisations for the operations in brackets.

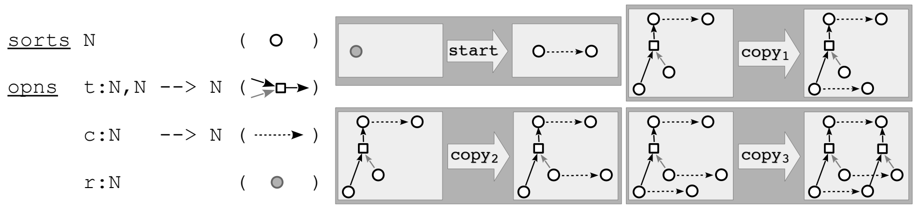

Figure 8. Copying

algebras guarantee that *exactly one* copy is made. Note that this copy mechanism also works if `t` builds up arbitrary hierarchical or even cyclic structures.

This last example describes a typical software engineering situation in the area of model transformation: some structured model, for example a class model, has to be transformed into another structured system, for example a relational database. In this context, keeping track of already finished transformations is essential for the control flow of the transformation process and to avoid that some parts are performed twice.

More detailed examples in this area can be found in [15].

6 Related Work and Conclusions

We have introduced single-pushout rewriting of arbitrary partial algebras. As usual, transformations are defined by a single pushout of partial morphisms. Thus, general composition and decomposition properties of pushouts can be exploited for a rich theory. The new approach is built on a category of partial morphisms that *does not have all* pushouts. We provided a good characterisation of the situations which admit pushouts by hereditariness of underlying pushouts of total morphisms, compare Theorem 10. Informally, pushouts can be built if the applied rule does not try to define operations where they are defined already. This application condition can easily be checked in every concrete situation. By some examples, we showed the practical relevance of the application conditions for system design and the termination of derivation sequences. (More examples can be found in [15].) Within our approach, we do not have to distinguish between graph structures (objects and links) and data structures (base-types and -operations). We can easily model associations and attributes with at-most-one-multiplicity.

There are only a few articles in the literature that address rewriting of partial algebras, for example [2] and [1] for the double- and single-pushout approach resp. But both papers stay in the framework of signatures with *unary* operation symbols only and aim at an underlying category that is co-complete.

Aspects of partial algebras occur in all papers that are concerned with relabelling of nodes and edges, for example [9], or that invent mechanisms for exchanging the attribute value without deleting and adding an object, for example [7]. Most of these approaches avoid "real" partial algebras by completing them to total ones by some undefined-values.

Thus, our approach is new, shows some application potentials, and seems promising wrt. theoretical results.

References

1. Peter Burmeister, Miquel Monserrat, Francesc Rosselló, and Gabriel Valiente. Algebraic transformation of unary partial algebras II: single-pushout approach. *Theor. Comput. Sci.*, 216(1-2):311–362, 1999.
2. Peter Burmeister, Francesc Rosselló, Joan Torrens, and Gabriel Valiente. Algebraic transformation of unary partial algebras I: double-pushout approach. *Theor. Comput. Sci.*, 184(1-2):145–193, 1997.
3. Andrea Corradini, Tobias Heindel, Frank Hermann, and Barbara König. Sesqui-pushout rewriting. In Andrea Corradini, Hartmut Ehrig, Ugo Montanari, Leila Ribeiro, and Grzegorz Rozenberg, editors, *ICGT*, volume 4178 of *Lecture Notes in Computer Science*, pages 30–45. Springer, 2006.
4. Vincent Danos, Tobias Heindel, Ricardo Honorato-Zimmer, and Sandro Stucki. Reversible sesqui-pushout rewriting. In *Graph Transformation - 7th International Conference, ICGT 2014, Held as Part of STAF 2014, York, UK, July 22-24, 2014. Proceedings*, pages 161–176, 2014.
5. Hartmut Ehrig, Karsten Ehrig, Ulrike Prange, and Gabriele Taentzer. *Fundamentals of Algebraic Graph Transformation*. Springer, 2006.
6. Hartmut Ehrig, Gregor Engels, Hans-Jörg Kreowski, and Grzegorz Rozenberg, editors. *Graph Transformations - 6th International Conference, ICGT 2012, Bremen, Germany, September 24-29, 2012. Proceedings*, volume 7562 of *Lecture Notes in Computer Science*. Springer, 2012.
7. Ulrike Golas. A general attribution concept for models in M-adhesive transformation systems. In Ehrig et al. [6], pages 187–202.
8. Annegret Habel, Reiko Heckel, and Gabriele Taentzer. Graph grammars with negative application conditions. *Fundam. Inform.*, 26(3/4):287–313, 1996.
9. Annegret Habel and Detlef Plump. M,n-adhesive transformation systems. In Ehrig et al. [6], pages 218–233.
10. Tobias Heindel. Hereditary pushouts reconsidered. In Hartmut Ehrig, Arend Rensink, Grzegorz Rozenberg, and Andy Schürr, editors, *ICGT*, volume 6372 of *Lecture Notes in Computer Science*, pages 250–265. Springer, 2010.
11. Richard Kennaway. Graph rewriting in some categories of partial morphisms. In Hartmut Ehrig, Hans-Jörg Kreowski, and Grzegorz Rozenberg, editors, *Graph-Grammars and Their Application to Computer Science*, volume 532 of *Lecture Notes in Computer Science*, pages 490–504. Springer, 1990.
12. Michael Löwe. Algebraic approach to single-pushout graph transformation. *Theor. Comput. Sci.*, 109(1&2):181–224, 1993.
13. Michael Löwe, Martin Korff, and Annika Wagner. An algebraic framework for the transformation of attributed graphs. In M. R. Sleep et al, editor, *Term Graph Rewriting: Theory and Practice*, pages 185 – 199. Wiley, 1993.
14. Miquel Monserrat, Francesc Rossello, Joan Torrens, and Gabriel Valiente. Single pushout rewriting in categories of spans I: The general setting. Technical Report LSI-97-23-R, Department de Llenguatges i Sistemes Informtics, Universitat Politcnica de Catalunya, 1997.
15. Marius Tempelmeier and Michael Löwe. Single-Pushout Transformation partieller Algebren. Technical Report 2015/1 (in German), FHDW-Hannover, 2015.