Graph Transformation with Focus on Incident Edges*

Dominique Duval, Rachid Echahed, and Frédéric Prost

University of Grenoble
B. P. 53, F-38041 Grenoble, France
{Dominique.Duval, Rachid.Echahed, Frederic.Prost}@imag.fr

Abstract. We tackle the problem of graph transformation with particular focus on node cloning. We propose a new approach to graph rewriting, called *polarized node cloning*, where a node may be cloned together with either all its incident edges or with only its outgoing edges or with only its incoming edges or with none of its incident edges. We thus subsume previous works such as the sesqui-pushout, the heterogeneous pushout and the adaptive star grammars approaches. We first define polarized node cloning algorithmically, then we propose an algebraic definition. We use polarization annotations to declare how a node must be cloned. For this purpose, we introduce the notion of polarized graphs as graphs endowed with some annotations on nodes and we define graph transformations with polarized node cloning by means of sesqui-pushouts in the category of polarized graphs.

1 Introduction

Graph transformation [22, 11, 13] extends string rewriting [3] and term rewriting [1] in several respects. In the literature, there are many ways to define graphs and graph rewriting. The proposed approaches can be gathered in two main streams: (i) the algorithmic approaches, which define a graph rewrite step by means of the algorithms involved in the implementation of graph transformation (see e.g. [2, 10]); (ii) the second stream consists of the algebraic approaches, first proposed in the seminal paper [14], and which use categorical constructs to define graph transformation in an abstract way. The most popular algebraic approaches are the double pushout (DPO) [14, 5] and the single pushout (SPO) [21, 16, 17, 12].

In this paper we are interested in graph transformation with particular focus on *node cloning*. Indeed, making copies of values is a very useful feature shared by most popular programming languages (see for instance the so-called shallow cloning [15] or deep cloning [20] operations). Informally, by cloning a node n, we mean making zero, one or more copies of n with "some" of its incident edges. The classical DPO and SPO approaches of graph transformation are clearly not well suited to perform cloning of nodes. As far as we are aware of, there are two algebraic attempts to deal with node cloning : the sesqui-pushout approach

* This work has been funded by the project CLIMT of the French *Agence Nationale de la Recherche* (ANR-11-BS02-016).

(SqPO) [4] and the heterogeneous pushout approach (HPO) [7]. The sesqui-pushout approach has the ability to clone nodes with all their incident edges whereas the HPO clones a node only with its outgoing edges. Our aim in this paper is to investigate a new flexible way to perfom node cloning, so that every copy of a node n can be made either with all the incident edges (denoted hereafter $n^{\pm}$), with only its outgoing edges (n^{+}), with only its incoming edges (n^{-}), or without any of its incident edges (denoted simply as n). We call this kind of graph transformation *polarized node cloning*. To achieve this task, we introduce the notion of *polarized graphs*. Informally, we define a polarized graph $\mathbb{X}$ as a graph X where each node n is annotated as $n^{\pm}$, n^{+}, n^{-} or just n. The rules in our approach are made of a polarized graph $\mathbb{K}$, consisting of a graph K with annotated nodes, and a span of graphs $L \xleftarrow{l} K \xrightarrow{r} R$. The annotations of $\mathbb{K}$ indicate the cloning strategy of incident edges. We prove that the polarized node cloning can be described as a SqPO rewriting of polarized graphs, preceded by the polarization of every node n in the left hand side as $n^{\pm}$ and followed by forgetting all polarizations in the right hand side. This is called the *polarized sesqui-pushout* rewriting (PSqPO for short).

The paper is organized as follows. The notion of polarized node cloning of graphs is defined in an elementary algorithmic way in Section 2. In Section 3 we define polarized graphs and the corresponding sesqui-pushout rewriting, from which we get the polarized sesqui-pushout rewriting for the polarized node cloning of graphs. Our approach is adapted to labeled graphs and illustrated through some examples in Section 4. A comparison with related work is made in Section 5 and concluding remarks are given in Section 6. An Appendix is added in order to ease the verification of the accuracy of our results. Detailed proofs can be found in [8]. We use categorical notions which may be found for instance in [19].

2 Polarized Node Cloning of Graphs

In this section we introduce some notations involving graphs and define the notion of *polarized node cloning*.

2.1 Graphs

Definition 1. *A* graph X *is made of a set of* nodes $|X|$*, a set of* edges $X_{\to}$ *and two functions* source *and* target *from* $X_{\to}$ *to* $|X|$*. An edge e with source n and target p is denoted* $n \xrightarrow{e} p$*. The set of edges from n to p in X is denoted* $X_{n\to p}$*. A* morphism *of graphs* $f : X \to Y$ *is made of two functions (both denoted f)* $f : |X| \to |Y|$ *and* $f : X_{\to} \to Y_{\to}$*, such that* $f(n) \xrightarrow{f(e)} f(p)$ *for each edge* $n \xrightarrow{e} p$*. This provides the category* **Gr** *of graphs.*

In order to build large graphs from smaller ones, we will use the *sum* of graphs and the *edge-sum* for adding edges to a graph, as defined below using the symbol + for the coproduct in the category of sets, i.e., the disjoint union of sets.

Definition 2. *Given two graphs X_1 and X_2, the* sum X_1+X_2 *is the coproduct of X_1 and X_2 in the categry of graphs, which means that $|X_1+X_2| = |X_1|+|X_2|$ and $(X_1+X_2)_{\to} = X_{1\to} + X_{2\to}$ and the source and target functions for X_1+X_2 are induced by the source and target functions for X_1 and for X_2. Given two graphs X and E such that $|E| \subseteq |X|$, the* edge-sum $X +_e E$ *is the pushout, in the category of graphs, of X and E over their common subgraph made of the nodes of E and no edge. This means that $|X +_e E| = |X|$ and $(X +_e E)_{\to} = X_{\to} + E_{\to}$ and the source and target functions for $X +_e E$ are induced by the source and target functions for X and for E.*

Clearly, the precise set of nodes of E does not matter in the construction of $X +_e E$, as long as it contains the source and target of every edge of E and is contained in $|X|$. This notation is extended to morphisms: let $f_1 : X_1 \to Y_1$ and $f_2 : X_2 \to Y_2$, then $f_1 + f_2 : X_1 + X_2 \to Y_1 + Y_2$ is defined piecewise from f_1 and f_2. Similarly, let $f : X \to Y$ and $g : E \to F$ with $|E| \subseteq |X|$ and $|F| \subseteq |Y|$, then $f +_e g : X +_e E \to Y +_e F$ is defined as f on the nodes and piecewise from f and g on the edges.

Remark 1. Let X be a subgraph of a graph Y. Let $\overline{X}$ denote the subgraph of Y induced by the nodes outside $|X|$ and $\widetilde{X}$ the subgraph of Y induced by the edges which are neither in X nor in $\overline{X}$, that is, the edges that are incident to a node (at least) in X but do not belong to X. For all nodes n, p in Y let $\widetilde{X}_{n\to p}$ denote the subgraph of Y induced by the edges from n to p in $\widetilde{X}$ (so that $\widetilde{X}_{n\to p}$ is empty whenever both n and p are in $\overline{X}$). Then Y can be expressed as $Y = (X + \overline{X}) +_e \widetilde{X}$ with $\widetilde{X}_{\to} = \sum_{n\in|Y|,p\in|Y|} \widetilde{X}_{n\to p}$ which can also be written as $|Y| = |X| + |\overline{X}|$ and $Y_{\to} = X_{\to} + \overline{X}_{\to} + \sum_{n\in|Y|,p\in|Y|} \widetilde{X}_{n\to p}$

Definition 3. *A* matching of graphs *is a monomorphism of graphs. Given a matching $m : L \to G$, the nodes and edges in $m(L)$ are called the* matching nodes *and the* matching edges*, respectively.*

Thus, a morphism of graphs is a matching if and only if it is *injective*, in the sense that both underlying functions (on nodes and on edges) are injections. So, up to isomorphism, every matching of graphs is an inclusion. For simplicity of notations, we now assume that all matchings of graphs are inclusions.

2.2 Polarized Node Cloning, Algorithmically

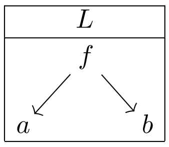
The *polarized node cloning of graphs* is a graph transformation which allows one to perform flexible cloning of nodes and their incident edges. Given a rewriting rule with a left-hand side L and a right-hand side R and a matching m of L in a graph G, the transformation of the nodes and the matching edges of G is provided by the rule, while the transformation of the non-matching edges (i.e., edges of G not in the image of L) is rather flexible: a node n can be cloned either with all its non-matching edges, or with all its outgoing non-matching edges, or with all its incoming non-matching edges, or with none of its non-matching edges. Definition 4 below provides an algorithmic definition of polarized node cloning (AlgoPC for short). An algebraic approach, more abstract, is presented in Section 3.

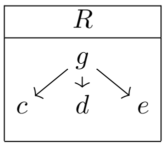
Definition 4. *An* AlgoPC rewrite rule *consists of a tuple $\mu = (L, R, C^+, C^-)$, where L and R are graphs and $C^+, C^- : |L| \times |R| \to \mathbb{N}$ are mappings. Then L and R are called the* left-hand side *and the* right-hand side*, respectively, and C^+, C^- are called the* cloning multiplicities *of μ. Let $\mu = (L, R, C^+, C^-)$ be an AlgoPC rewrite rule, G a graph and $m : L \to G$ a matching. Thus $|G| = |L| + |\overline{L}|$ and $G_\to = L_\to + \overline{L}_\to + \widetilde{L}_\to$. The* AlgoPC rewrite step *applying the rule μ to the matching m builds the graph H and the matching $h : R \to H$ such that h is the inclusion and $|H| = |R| + |\overline{L}|$ and $H_\to = R_\to + \overline{L}_\to + \sum_{n\in|H|, p\in|H|} E_{n,p}$ where:*

1. *if $n \in |R|$ and $p \in |R|$ then there is an edge $n \overset{(e,i)}{\to} p$ in $E_{n,p}$ for each edge $n_L \overset{e}{\to} p_L$ in $\widetilde{L}_\to$ and each $i \in \{1, \ldots, C^+(n_L, n) \times C^-(p_L, p)\}$;*
2. *if $n \in |R|$ and $p \in |\overline{L}|$ then there is an edge $n \overset{(e,i)}{\to} p$ in $E_{n,p}$ for each edge $n_L \overset{e}{\to} p$ in $\widetilde{L}_\to$ and each $i \in \{1, \ldots, C^+(n_L, n)\}$;*
3. *if $n \in |\overline{L}|$ and $p \in |R|$ then there is an edge $n \overset{(e,i)}{\to} p$ in $E_{n,p}$ for each edge $n \overset{e}{\to} p_L$ in $\widetilde{L}_\to$ and each $i \in \{1, \ldots, C^-(p_L, p)\}$;*
4. *if $n \in |\overline{L}|$ and $p \in |\overline{L}|$ then $E_{n,p}$ is empty.*

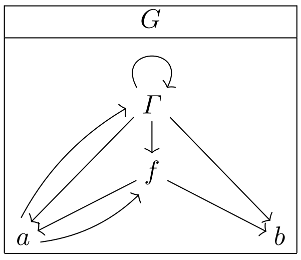
So, when an AlgoPC rule $\mu = (L, R, C^+, C^-)$ is applied to a matching of L in G, the image of L in G is erased and replaced by R, the subgraph $\overline{L}$ remains unchanged, and the edges in $\widetilde{L}$ are handled according to the cloning multiplicities. The subtleties in building clones lie in the treatment of the edges in $\widetilde{L}$.

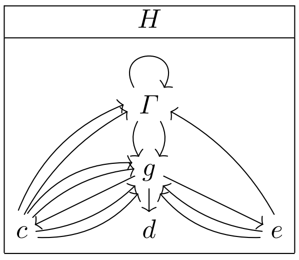
Example 1. Let us consider the following rule $\mu = (L, R, C^+, C^-)$ where





$C^+(a, c) = 2$, $C^+(a, e) = 1$, $C^-(f, g) = 2$, and every other cloning multiplicity is 0. Now let us consider the graphs G and H:





Then G rewrites into H using the rule μ and the matching $L \to G$ defined by the inclusion. Indeed, as specified by the cloning multiplicities, the edge going out of node a towards Γ is cloned three times, two times by the edges going out from c towards Γ ($C^+(a,c) = 2$) and a third time by the edge going out from e ($C^+(a,e) = 1$), the node b is erased as well as all its incident edges, and the incoming edges of f are duplicated ($C^-(f,g) = 2$) and redirected towards g. The edge from a towards f is copied four times ($C^+(a,c) \times C^-(f,g) = 4$) from c to g and two times ($C^+(a,e) \times C^-(f,g) = 2$) from e to g.

3 Polarized Sesqui-Pushout of Graphs

In this section, in order to provide an algebraic version of the polarized node cloning of graphs defined in Section 2.2, we introduce polarized graphs, we study their sesqui-pushout rewriting and we use it for defining the notion of *polarized sesqui-pushout* of graphs.

3.1 Polarized Graphs

A polarized graph is a graph where every node may be polarized in the sense that it may be marked either with a "+", with a "−", with both "±" or with no mark. The polarizations will be used as cloning instructions.

Definition 5. *A* polarization $X^\pm$ *of a graph* X *is a pair* $X^\pm = (|X|^+, |X|^-)$ *of subsets of* $|X|$*. A node* n *may be denoted* n^+ *if it is in* $|X|^+$*,* n^- *if it is in* $|X|^-$ *and* $n^\pm$ *if it is in* $|X|^+ \cap |X|^-$*. A* polarized graph $\mathbb{X} = (X, X^\pm)$ *is a graph* X *together with a polarization* $X^\pm$ *of* X *such that the source of each edge* e *of* $X_\to$ *is in* $|X|^+$ *and the target of* e *is in* $|X|^-$*. A* morphism *of polarized graphs* $f : \mathbb{X} \to \mathbb{Y}$*, where* $\mathbb{X} = (X, X^\pm)$ *and* $\mathbb{Y} = (Y, Y^\pm)$*, is a morphism of graphs* $f : X \to Y$ *such that* $f(|X|^+) \subseteq |Y|^+$ *and* $f(|X|^-) \subseteq |Y|^-$*. This provides the category* $\mathbf{Gr}^\pm$ *of polarized graphs. The notations in defintion 1 are extended to polarized graphs: when* $\mathbb{X} = (X, X^\pm)$ *then* $|\mathbb{X}| = |X|$ *and* $\mathbb{X}_\to = X_\to$*.*

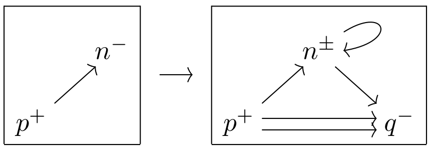
Definition 6. *Given two polarized graphs* $\mathbb{X}_1$ *and* $\mathbb{X}_2$*, their* sum *is the polarized graph* $\mathbb{X}_1 + \mathbb{X}_2$ *made of the graph* $X_1 + X_2$ *with the polarization* $|X_1 + X_2|^+ = |X_1|^+ + |X_2|^+$ *and* $|X_1 + X_2|^- = |X_1|^- + |X_2|^-$*. Given two polarized graphs* $\mathbb{X}$ *and* $\mathbb{E}$ *such that* $|E| \subseteq |X|$*,* $|E|^+ \subseteq |X|^+$ *and* $|E|^- \subseteq |X|^-$*, their* edge-sum *is the polarized graph* $\mathbb{X} +_e \mathbb{E}$ *made of the graph* $X +_e E$ *with the polarization* $|X +_e E|^+ = |X|^+$ *and* $|X +_e E|^- = |X|^-$*.*

Definition 7. *A* matching *of polarized graphs is a monomorphism* $f : \mathbb{X} \to \mathbb{Y}$ *such that* $f(|X|^+) = f(|X|) \cap |Y|^+$ *and* $f(|X|^-) = f(|X|) \cap |Y|^-$ *(we say that* f strictly preserves *the polarization).*

Thus, a matching of polarized graphs is a matching of graphs which strictly preserves the polarization. We now assume that all matchings of polarized graphs are inclusions, which is the case up to isomorphism.

Remark 2. Let $f : \mathbb{X} \to \mathbb{Y}$ be a matching of polarized graphs. Analogously to Remark 1, using the fact that f strictly preserves the polarization, we can express $\mathbb{Y}$ as $\mathbb{Y} = (\mathbb{X} + \overline{\mathbb{X}}) +_e \widetilde{\mathbb{X}}$ with $\widetilde{\mathbb{X}}_{\to} = \sum_{n \in |\mathbb{Y}|, p \in |\mathbb{Y}|} \widetilde{\mathbb{X}}_{n \to p}$, where $\widetilde{\mathbb{X}}_{n \to p}$ denotes the polarized graph made of the graph $\widetilde{X}_{n \to p}$ as in Remark 1 with its nodes polarized as in $\mathbb{Y}$.

Example 2. Here is a morphism of polarized graphs which is an inclusion although it is not a matching (the condition $f(|X|^+) = f(|X|) \cap |Y|^+$ is not fulfilled):



Definition 8. *The* underlying *graph of a polarized graph* $\mathbb{X} = (X, X^\pm)$ *is* X*. This defines a functor* $\mathrm{Depol} : \mathbf{Gr}^\pm \to \mathbf{Gr}$*. The polarized graph* $\mathbb{X}$ induced by *a graph* X *is* $\mathbb{X} = (X, X^\pm)$ *where* $|X|^+ = |X|^- = |X|$*. This defines a functor* $\mathrm{Pol} : \mathbf{Gr} \to \mathbf{Gr}^\pm$*, which is a right adjoint to* Depol *(this is denoted* $\mathrm{Depol} \dashv \mathrm{Pol}$*). Moreover, the functor* $\mathrm{Depol} \circ \mathrm{Pol}$ *is the identity of* $\mathbf{Gr}$*.*

3.2 Sesqui-Pushout Rewriting of Polarized Graphs

In this section we describe the sesqui-pushout of polarized graphs. The sesqui-pushout rewriting [4] relies on the well-known categorical notions of pushout (PO) and pullback (PB): a sesqui-pushout rewriting step is made of a final pullback complement followed by a pushout. Pushouts and final pullback complements of polarized graphs are described in Propositions 1 and 2, respectively.

Proposition 1. *Let* $r : \mathbb{K} \to \mathbb{R}$ *be a morphism of polarized graphs and* $d : \mathbb{K} \to \mathbb{D}$ *a matching of polarized graphs. The following square, where* h *is the inclusion, is a pushout of* d *and* r *in* $\mathbf{Gr}^\pm$*.*

$$\begin{tikzcd}
\mathbb{K} \arrow[r, "r"] \arrow[d, "d"'] & \mathbb{R} \arrow[d, "h"] \arrow[dl, phantom, "\ulcorner", very near end] \\
\mathbb{D} = (\mathbb{K} + \overline{\mathbb{K}}) +_e \widetilde{\mathbb{K}} \arrow[r, "r_1 = (r + \mathrm{id}_{\overline{\mathbb{K}}}) +_e \widetilde{r}"'] & \mathbb{H} = (\mathbb{R} + \overline{\mathbb{K}}) +_e \widetilde{\mathbb{R}}
\end{tikzcd}$$

where $\widetilde{\mathbb{R}}_{n \to p} = \sum_{n_D \in r_1^{-1}(n), p_D \in r_1^{-1}(p)} \widetilde{\mathbb{K}}_{n_D \to p_D}$ *for all* $n, p \in |\mathbb{H}|$, $n_D \in |\mathbb{D}|^+$, $p_D \in |\mathbb{D}|^-$*, and where* $\widetilde{r} : \widetilde{\mathbb{K}} \to \widetilde{\mathbb{R}}$ *maps* $n_D \xrightarrow{e} p_D$ *to* $r_1(n_D) \xrightarrow{e} r_1(p_D)$*.*

Remark 3. Pushouts are preserved by Depol, because Depol is left adjoint to Pol. With the notations as in Proposition 1, this implies that $\mathrm{Depol}(h)$ can also be obtained by computing a pushout of $\mathrm{Depol}(d)$ and $\mathrm{Depol}(r)$ in $\mathbf{Gr}$.

Let us now define final pullback complements in the naive way, this definition coincides with the one in [9, 4] when both exist.

Definition 9. *In a category $\mathcal{M}$, let $a : X \to Y$ and $g : Y \to Y_1$ be consecutive morphisms. A* pullback complement *(PBC) of a and g is an object X_1 with a pair of morphisms $f : X \to X_1$, $a_1 : X_1 \to Y_1$ such that there is a pullback:*

$$\begin{tikzcd}
Y \arrow[d, "g"'] & X \arrow[l, "a"'] \arrow[d, "f"] \arrow[dl, phantom, "\lrcorner", very near start] \\
Y_1 & X_1 \arrow[l, "a_1"]
\end{tikzcd}$$

A morphism $k : (X_1, f, a_1) \to (X_1', f', a_1')$ of pullback complements of a and g is a morphism $k : X_1 \to X_1'$ in $\mathcal{M}$ such that $k \circ f = f'$ and $a_1' \circ k = a_1$. This yields the category of pullback complements of a and g, and the final pullback complement *(FPBC) of a and g is defined as the final object in this category, if it does exist.*

Proposition 2. *Let $l : \mathbb{K} \to \mathbb{L}$ be a morphism and $m : \mathbb{L} \to \mathbb{G}$ a matching of polarized graphs. The following square, where d is the inclusion, is a FPBC of l and m in $\mathbf{Gr}^{\pm}$:*

$$\begin{tikzcd}
\mathbb{L} \arrow[d, "m"'] & \mathbb{K} \arrow[l, "l"'] \arrow[d, "d"] \arrow[dl, phantom, "\lrcorner", very near start] \\
\mathbb{G} = (\mathbb{L} + \overline{\mathbb{L}}) +_e \widetilde{\mathbb{L}} & \mathbb{D} = (\mathbb{K} + \overline{\mathbb{L}}) +_e \widetilde{\mathbb{K}} \arrow[l, "l_1 = (l + \mathrm{id}_{\overline{\mathbb{L}}}) +_e \widetilde{l}"] \arrow[ul, phantom, "\ulcorner", very near start]
\end{tikzcd}$$

where $\widetilde{\mathbb{K}}_{n_D \to p_D} = \widetilde{\mathbb{L}}_{l_1(n_D) \to l_1(p_D)}$ for all $n_D \in |\mathbb{D}|^+, p_D \in |\mathbb{D}|^-$ (otherwise $\widetilde{\mathbb{K}}_{n_D \to p_D} = \emptyset$) and where $\widetilde{l} : \widetilde{\mathbb{K}} \to \widetilde{\mathbb{L}}$ maps $n_D \xrightarrow{e} p_D$ to $l_1(n_D) \xrightarrow{e} l_1(p_D)$.

The next definition is the usual definition of SqPO rewriting [4], applied to the category of polarized graphs.

Definition 10. *A* SqPO rewrite rule of polarized graphs *is a span of polarized graphs. Let $\rho = \mathbb{L} \xleftarrow{l} \mathbb{K} \xrightarrow{r} \mathbb{R}$ be a SqPO rewrite rule of polarized graphs and $m : \mathbb{L} \to \mathbb{G}$ a matching of polarized graphs. The* SqPO rewrite step *applying the rule ρ to the matching m builds the polarized graph $\mathbb{H}$ and the matching of polarized graphs $h : \mathbb{R} \to \mathbb{H}$ such that h is the inclusion, in two steps. First a FPBC of m and l is built as in Proposition 2, which gives rise to a polarized graph $\mathbb{D}$, a morphism $l_1 : \mathbb{D} \to \mathbb{G}$ and a matching $d : \mathbb{K} \to \mathbb{D}$ in $\mathbf{Gr}^{\pm}$. Then a pushout of $d : \mathbb{K} \to \mathbb{D}$ and $r : \mathbb{K} \to \mathbb{R}$ is constructed as in Proposition 1, which gives rise to a graph $\mathbb{H}$, a morphism $r_1 : \mathbb{D} \to \mathbb{H}$ and a matching $h : \mathbb{R} \to \mathbb{H}$ in $\mathbf{Gr}^{\pm}$.*

We represent a SqPO rewrite step of polarized graphs by the following diagram:

$$\begin{tikzcd}
\mathbb{L} \arrow[d, "m"'] & \mathbb{K} \arrow[l, "l"'] \arrow[r, "r"] \arrow[d, "d"] \arrow[dl, phantom, "\lrcorner", very near start] & \mathbb{R} \arrow[d, "h"] \\
\mathbb{G} & \mathbb{D} \arrow[l, "l_1"] \arrow[r, "r_1"'] \arrow[ul, phantom, "\ulcorner", very near start] & \mathbb{H} \arrow[ul, phantom, "\ulcorner", very near start]
\end{tikzcd}$$

Merging Propositions 1 and 2 yields the following result, which provides an explicit description of a SqPO rewrite step of polarized graphs.

Theorem 1. *In the category of polarized graphs, let $\rho = (\mathbb{L} \xleftarrow{l} \mathbb{K} \xrightarrow{r} \mathbb{R})$ be a SqPO rewrite rule and $m : \mathbb{L} \to \mathbb{G}$ a matching, so that $\mathbb{G} = (\mathbb{L} + \overline{\mathbb{L}}) +_e \widetilde{\mathbb{L}}$. The SqPO rewrite step applying ρ to m builds the matching $h : \mathbb{R} \to \mathbb{H}$ where h is the inclusion and $\mathbb{H} = (\mathbb{R} + \overline{\mathbb{L}}) +_e \widetilde{\mathbb{R}}$ where, for all nodes n, p in $|\mathbb{H}|$:*

$$\widetilde{\mathbb{R}}_{n\to p} = \begin{cases} \sum_{n_K^+ \in r^{-1}(n), p_K^- \in r^{-1}(p)} \widetilde{\mathbb{L}}_{l(n_K)\to l(p_K)} & \text{when } n, p \in |\mathbb{R}| \\ \sum_{n_K^+ \in r^{-1}(n)} \widetilde{\mathbb{L}}_{l(n_K)\to p} & \text{when } n \in |\mathbb{R}|,\ p \in |\overline{\mathbb{L}}| \\ \sum_{p_K^- \in r^{-1}(p)} \widetilde{\mathbb{L}}_{n\to l(p_K)} & \text{when } n \in |\overline{\mathbb{L}}|,\ p \in |\mathbb{R}| \\ \emptyset & \text{when } n, p \in |\overline{\mathbb{L}}| \end{cases}$$

3.3 Polarized Node Cloning of Graphs, Algebraically

In this section we show that the polarized node cloning of graphs can easily be performed using the sesqui-pushout rewriting of polarized graph. This is called the *polarized sesqui-pushout* rewriting system (PSqPO). In a PSqPO rewriting step, the given matching $m : L \to G$ and the resulting matching $h : R \to H$ are matchings of ordinary graphs, while the interface matching $d : \mathbb{K} \to \mathbb{D}$ is a matching of polarized graphs where the polarization of a node indicates how the rewriting step acts on the non-matching edges incident to this node. The adjoint functors $\mathrm{Pol} : \mathbf{Gr} \to \mathbf{Gr}^{\pm}$ (right adjoint) and $\mathrm{Depol} : \mathbf{Gr}^{\pm} \to \mathbf{Gr}$ (left adjoint) from Definition 8 are used for moving between categories $\mathbf{Gr}$ and $\mathbf{Gr}^{\pm}$. It should be reminded that $\mathrm{Depol} \circ \mathrm{Pol}$ is the identity of $\mathbf{Gr}$.

Definition 11. *A* PSqPO rewrite rule *of graphs is a span of graphs $L \xleftarrow{l} K \xrightarrow{r} R$ together with a polarized graph $\mathbb{K}$ such that $K = \mathrm{Depol}(\mathbb{K})$. This is denoted by $L \xleftarrow{l} \mathbb{K} \xrightarrow{r} R$. Thanks to the adjunction $\mathrm{Depol} \dashv \mathrm{Pol}$, each PSqPO rewrite rule $L \xleftarrow{l} \mathbb{K} \xrightarrow{r} R$ gives rise to a SqPO rewrite rule $\mathbb{L} \xleftarrow{l} \mathbb{K} \xrightarrow{r} \mathbb{R}$ in $\mathbf{Gr}^{\pm}$ where $\mathbb{L} = \mathrm{Pol}(L)$ and $\mathbb{R} = \mathrm{Pol}(R)$. The* PSqPO rewrite step *applying a PSqPO rewrite rule $\rho = (L \xleftarrow{l} \mathbb{K} \xrightarrow{r} R)$ to a matching of graphs $m : L \to G$ is the following construction of a matching of graphs $h : R \to H$.*

(i) *Let $m' = \mathrm{Pol}(m) : \mathbb{L} \to \mathbb{G}$, so that m' is a matching of polarized graphs.*
(ii) *Let $h' : \mathbb{R} \to \mathbb{H}$ be the matching of polarized graphs obtained by applying the SqPO rewriting rule ρ' to the matching m' in $\mathbf{Gr}^{\pm}$; note that $\mathrm{Depol}(\mathbb{R}) = \mathrm{Depol}(\mathrm{Pol}(R)) = R$.*
(iii) *Let $H = \mathrm{Depol}(\mathbb{H})$ and $h = \mathrm{Depol}(h') : R \to H$, this is the required matching of graphs.*

This means that H is made of a copy of R together with the non-matching nodes of G (i.e., nodes of G which are not in the image of the matching) and with an edge $n \xrightarrow{(n_D,p_D,e)} p$ for each n_D in $|K|^+ + |\overline{L}|$ such that $r_1(n_D) = n$, each p_D in $|K|^- + |\overline{L}|$ such that $r_1(p_D) = p$ and each $n_G \xrightarrow{e} p_G$ in $G_\to$ where $n_G = l_1(n_D)$ and $p_G = l_1(p_D)$. Since l_1 and r_1 are the identity on $|\overline{L}|$, whenever both n and p are in $|\overline{L}|$ then $H_{n\to p} = G_{n\to p}$.

A PSqPO rewrite step of graphs can be represented by the following diagram:

$$\begin{tikzcd}
L \arrow[d, "m"'] & & \mathbb{L} \arrow[d, "m'"'] & \mathbb{K} \arrow[l, "l"'] \arrow[r, "r"] \arrow[d, "d'"] & \mathbb{R} \arrow[d, "h'"] & & R \arrow[d, "h"] \\
G \arrow[r, phantom, "\overset{\text{Pol}}{\longmapsto}"] & & \mathbb{G} & \mathbb{D} \arrow[l, "l_1"] \arrow[r, "r_1"'] & \mathbb{H} \arrow[r, phantom, "\overset{\text{Depol}}{\longmapsto}"] & & H
\end{tikzcd}$$

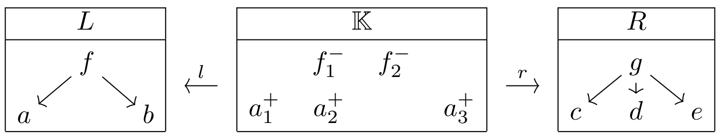
Remark 4. According to Definitions 11 and 10, applying a PSqPO rewrite rule ρ' to a matching m' can be decomposed in four steps: (i) m is mapped to $m' = \mathrm{Pol}(m)$, (ii-a) the FPBC of m and l in $\mathbf{Gr}^\pm$ provides d', (ii-b) the PO of d' and r in $\mathbf{Gr}^\pm$ yields h', (iii) h' is mapped to $h = \mathrm{Depol}(h')$. Thanks to remark 3, steps (ii-b) and (iii) can be "permuted", in the following way: first d' is mapped to $d = \mathrm{Depol}(d')$, then the PO of d and r in $\mathbf{Gr}$ yields h. Thus, a PSqPO rewrite step of graphs can also be represented by the following diagram:

$$\begin{tikzcd}
L \arrow[d, "m"'] & & \mathbb{L} \arrow[d, "m'"'] & \mathbb{K} \arrow[l, "l"'] \arrow[d, "d'"] & & K \arrow[r, "r"] \arrow[d, "d"'] & R \arrow[d, "h"] \\
G \arrow[r, phantom, "\overset{\text{Pol}}{\longmapsto}"] & & \mathbb{G} & \mathbb{D} \arrow[l, "l_1"] \arrow[r, phantom, "\overset{\text{Depol}}{\longmapsto}"] & & D \arrow[r, "r_1"'] & H
\end{tikzcd}$$

The next result shows that the PSqPO rewriting of graphs does provide an algebraic version of the polarized node cloning of graphs. A proof of a more precise result is provided in [8].

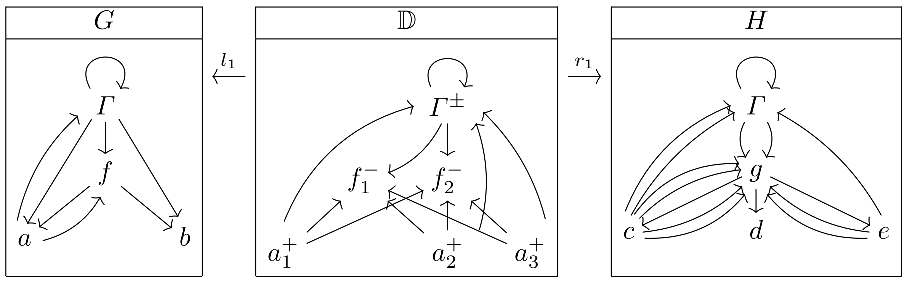
Proposition 3. *Let $\mu = (L, R, C^+, C^-)$ be an AlgoPC rewrite rule. Let $\mathbb{K}$ be the polarized graph without edges and with, for each $\star \in \{+, -\}$, a node $(n_L, n_R)_{i,\star}{}^\star$ for each pair of nodes $(n_L, n_R) \in |L| \times |R|$ and each $i \in \{1, \dots, C^\star(n_L, n_R)\}$. Let $\rho = L \xleftarrow{l} \mathbb{K} \xrightarrow{r} R$ be the PSqPO rewrite rule where $l((n_L, n_R)_{i,\star}) = n_L$ and $r((n_L, n_R)_{i,\star}) = n_R$. Then the rules μ and ρ are equivalent, in the sense that for each matching of graphs $m : L \to G$, the AlgoPC rewrite step applying μ to m and the PSqPO rewrite step applying ρ to m yield the same matching $h : R \to H$.*

Example 3. The rewrite rule μ of Example 1 can be translated to the following PSqPO rule:



where $l(f_1) = l(f_2) = f$, $l(a_1) = l(a_2) = l(a_3) = a$, $r(f_1) = r(f_2) = g$, $r(a_1) = r(a_2) = c$ and $r(a_3) = e$. As in Example 1, the matching is the inclusion of L in

G (below) and the PSqPO rewrite step builds:



The resulting graph H and matching $h : R \to H$ are the same as in Example 1.

4 An Extension to Labeled Polarized Graphs

For several modeling purposes, it is useful to add labels to nodes and edges. In this section we discuss an extension of our proposal in order to perform polarized sesqui-pushout graph transformation on labeled graphs. We provide syntactic conditions which ensure the existence of the constructions involved in the rewriting process. Hereafter, two sets $\mathcal{L}_N$ and $\mathcal{L}_E$ are given, they are called the set of *labels* for nodes and for edges, respectively. Moreover, all the constructions are considered up to isomorphism.

Definition 12. *A* labeled graph (X, lab) *is a graph X together with two partial functions* $\mathrm{lab} : |X| \rightharpoonup \mathcal{L}_N$ *for the* labeling *of nodes and* $\mathrm{lab} : X_{\to} \rightharpoonup \mathcal{L}_E$ *for the* labeling *of edges. A* morphism *of labeled graphs $f : (X, \mathrm{lab}_X) \to (Y, \mathrm{lab}_Y)$ is a morphism of graphs $f : X \to Y$ which preserves the labels, in the sense that if a node or an edge x in X is labeled with a then $f(x)$ in Y is labeled with a (if x is unlabeled there is no restriction on the labeling of $f(x)$). This provides the category* $\mathbf{LGr}$ *of labeled graphs (with labels in $\mathcal{L}_N$ and $\mathcal{L}_E$).*

A labeled graph (X, lab) is often simply denoted X. A node x is denoted $x\colon a$ if it is labeled with a and $x\colon \circ$ if it is unlabeled. An edge $x \to y$ is denoted $x \xrightarrow{a} y$ if it is labeled with a and simply $x \to y$ if it is unlabeled. A *matching* of labeled graphs is a matching of graphs which preserves the labels. Since polarizations and labelings do not interfere, these definitions and results are easily combined with the definitions and results in Section 3.1. This provides the category $\mathbf{LGr}^{\pm}$ of *labeled polarized graphs*, and Proposition 1 and Proposition 2 are generalized to labeled polarized graphs as follows.

Proposition 4. *Let $r : \mathbb{K} \to \mathbb{R}$ be a morphism of labeled polarized graphs and $d : \mathbb{K} \to \mathbb{D}$ a matching of labeled polarized graphs. Let us assume that:*

- *For each node or edge x in $\mathbb{K}$, if $r(x) : a$ and $d(x) : b$, then $a = b$.*
- *For each distinct nodes or edges x, y in $\mathbb{K}$, if $r(x) = r(y)$, $d(x) : a$ and $d(y) : b$, then $a = b$.*

Then the pushout of d and r in $\mathbf{LGr}^{\pm}$ exists, its underlying diagram of polarized graphs is the pushout of d and r in $\mathbf{Gr}^{\pm}$ and each node or edge x in $\mathbb{H}$ is labeled if and only if it is the image of a labeled node or edge in $\mathbb{R}$ or in $\mathbb{D}$.

Thanks to the assumptions, no conflict may arise when labeling the graph $\mathbb{H}$: if a node or edge x in $\mathbb{H}$ is the image of several nodes or edges in $\mathbb{R}$ or in $\mathbb{D}$ (at most one in $\mathbb{R}$ and maybe several in $\mathbb{D}$), then all of them have the same label, which becomes the label of x.

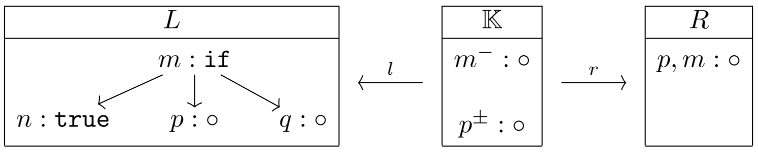
Proposition 5. *Let $l : \mathbb{K} \to \mathbb{L}$ be a morphism of labeled polarized graphs and $m : \mathbb{L} \to \mathbb{G}$ a matching of labeled polarized graphs. Then the FPBC of l and m exists, its underlying diagram of polarized graphs is the FPBC in $\mathbf{Gr}^{\pm}$ and each node or edge x_D in the graph $\mathbb{D}$ is labeled as follows: if x_D is not in the image of $\mathbb{K}$ then x_D is labeled in $\mathbb{D}$ like $l_1(x_D)$ in $\mathbb{G}$, otherwise $x_D = d(x_K)$ for a unique x_K in $\mathbb{K}$ and the label of x_D in $\mathbb{D}$ is determined by the labels of x_K in $\mathbb{K}$, $x_L = l(x_K)$ in $\mathbb{L}$ and $x_G = m(x_L)$ in $\mathbb{G}$ according to the following patterns:*

$$\begin{tikzcd}
x_L : a \arrow[d, mapsto] & x_K : a \arrow[l, mapsto] \arrow[d, mapsto] & x_L : \circ \arrow[d, mapsto] & x_K : \circ \arrow[l, mapsto] \arrow[d, mapsto] & x_L : \circ \arrow[d, mapsto] & x_K : \circ \arrow[l, mapsto] \arrow[d, mapsto] & x_L : a \arrow[d, mapsto] & x_K : \circ \arrow[l, mapsto] \arrow[d, mapsto] \\
x_G : a & x_D : a \arrow[l, mapsto] & x_G : a & x_D : a \arrow[l, mapsto] & x_G : \circ & x_D : \circ \arrow[l, mapsto] & x_G : a & x_D : \circ \arrow[l, mapsto]
\end{tikzcd}$$

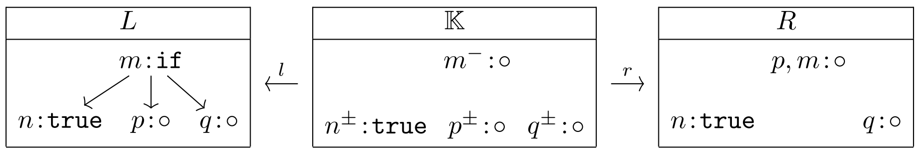
The labeled PSqPO rewrite rules cannot be defined simply as PSqPO rewrite rules where the graphs are labeled and the morphisms preserve the labels: indeed, in order to avoid conflicts in labeling the pushout, the assumptions in Proposition 4 must be satisfied after the construction of the polarized FPBC (Proposition 5). This leads to the following definition.

Definition 13. *A* labeled PSqPO rewrite rule *is a PSqPO rewrite rule $L \xleftarrow{l} \mathbb{K} \xrightarrow{r} R$ (Definition 11) where the graphs are labeled and the morphisms preserve the labels, such that the following conditions are fulfilled (where $K = \mathrm{Depol}(\mathbb{K})$) : (i) for each unlabeled node or edge x in K, if $l(x)$ is unlabeled in L then $r(x)$ is unlabeled in R and (ii) for each distinct unlabeled nodes or edges x, y in K, if $l(x) \neq l(y)$ and $l(x)$ or $l(y)$ is unlabeled in L then $r(x) \neq r(y)$ in R.*

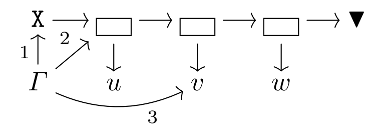
Example 4. The behavior of the "`if` b `then...else...`" operator in imperative languages can be modelled thanks to two polarized PSqPO rewrite rules, one when b is `true` and another one when b is `false`. Here is a possible choice when b is `true` (morphisms are represented via node name sharing, for instance $r(m) = r(p) = p, m$ and $l(m) = m$):



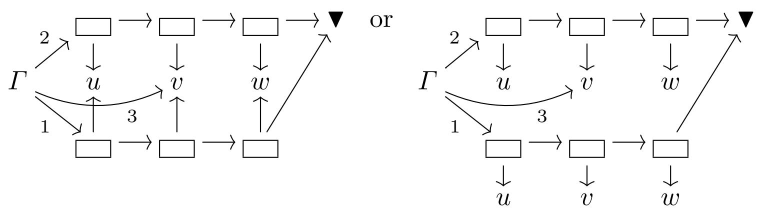
These rules for modeling "`if...then...else...`" are destructive, in the sense that nodes n and q disappear during the rewrite step. Non-destructive rules can also be chosen, here is such a rule for `true`.



Example 5. The problem of copying objects in object-oriented languages has been thoroughly examined. Two generic ways of copying objects are usually considered: shallow cloning, which is the basic cloning of Java (see the reference of method `clone` in class `Object` [15]) and deep cloning, which is implemented by `deep_copy` in Eiffel [20]. These two ways of cloning can be modelled by our approach. We restrict ourselves here to the cloning of a particular data, say *linked lists of constants* (a constant is implemented as a node without any outgoing edge) and consider two cloning routines: `sc` (shallow cloning) and `dc` (deep cloning). Intuitively, we would like to implement rules that transform, for instance, the following graph, where ▼ denotes the end of the list:



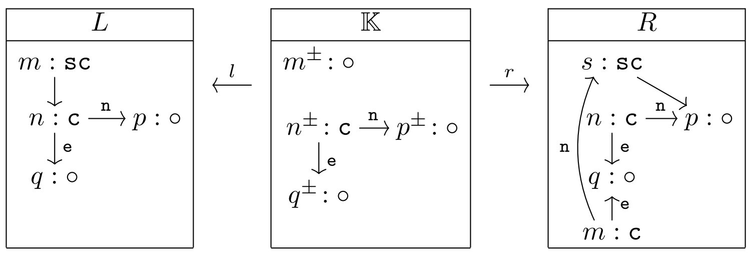
into the following ones, depending whether `X` is replaced by `sc` or by `dc`:



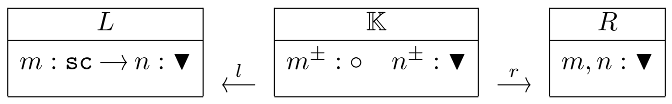
Notice that in the case of deep cloning, the edge from Γ to v is not cloned to point to the "new" occurrence of v (the graph on the right). Indeed, the deep cloning primitives do not modify the environment.

Let us consider labeled graphs. Label `c` is used to represent the usual `cons` constructor of lists. The parameters of `c` are identified by edge labels, the `next` cell of the list is pointed by an edge labeled `n` and the `element` of the cell is pointed by an edge labeled `e`.

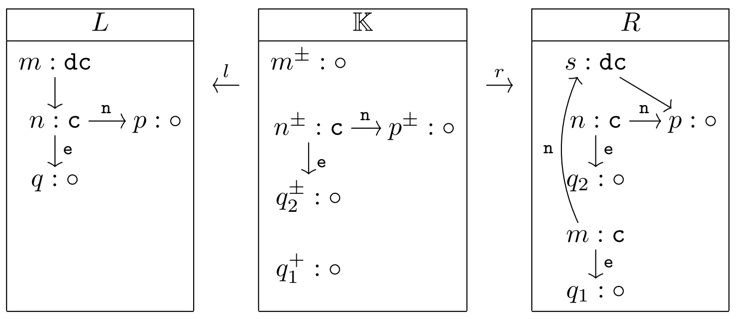
For the shallow cloning (`sc`) the recursive case is implemented by the following rule where morphisms l, r are represented by node name sharing. All edges that point to m before the execution of this rule will point to the new `c` node (the node m : `c` in R). The function `sc` is recursively called by the node s in R.



The base (halt) case is implemented by the following rule where the node m, n is the image of both m and n by r:



For deep cloning (dc), the recursive case is implemented by the following rule:



In this case node q is cloned twice in $\mathbb{K}$: the incoming edges of q are not cloned as incoming edges of q_1^+ (as it is the case for the edge from Γ to v).
The base (halt) case for dc is implemented by substituting sc with dc in the corresponding rule.

5 Related Work

Polarized sesqui-pushout graph rewriting (PSqPO) is a new way to perfom graph transformations which offers different possibilities to clone nodes and their incident edges, in addition to classical graph transformations (addition and deletion of nodes and edges). In this section the PSqPO approach is compared with other approaches for graph transformations.

In [7] an algebraic approach of termgraph transformation, based on heterogeneous pushouts (HPO), has been proposed. With respect to cloning abilities, the HPO approach offers the possibility to make one or more copies of a node together with its outgoing edges. Therefore, this way of cloning nodes is limited to the outgoing edges only and contrasts with the flexible possibilities of cloning edges proposed in the present paper. In fact, whenever a graph G rewrites into H according to the HPO approach using a rule (L, R, τ, σ) [7, Definition 5], the graph G can also be rewritten into H according to a rule $L \xleftarrow{l} \mathbb{K} \xrightarrow{r} R$ where morphisms l and r encode the functions τ and σ.

Cloning is also one of the features of the sesqui-pushout approach (SqPO) to graph transformation [4]. The SqPO and PSqPO approaches mainly differ in the way of handling cloning. In [4], the cloning of a node is performed by copying all its incident edges. This is a particular case of PSqPO. The use of polarized graphs helped us to specify for every clone, the way incident edges can be copied. Therefore, a SqPO rewrite step can be simulated by a PSqPO rewrite step by polarizing every node n in the interface graph $\mathbb{K}$ as $n^{\pm}$, but the converse does not hold in general. For instance, in both rules defining the shallow cloning operator `sc` (see Example 5), all nodes are cloned with polarities $\pm$, thus these rules can be implemented using the SqPO approach in the category of graphs. However, in the recursive case implementing the deep cloning operator, `dc`, one node, q_1^{+}, in $\mathbb{K}$ is polarized as $+$ only; it follows that this rule cannot be modelled with a standard SqPO transformation of graphs. Furthermore, in [4], the sesqui-pushout approach is compared to the classical DPO and SPO approaches. Therefore, for comparing our approach with the DPO and SPO, we may rely on [4, Propositions 12 and 14].

Cloning is also subject of interest in [6]. The authors consider rewrite rules of the form $S := R$ where S is a *star*, i.e., S is a (nonterminal) node surrounded by its adjacent nodes together with the edges that connect them. Rewrite rules which perform the cloning of a node are given in [6, Definition 6]. These rules show how a star can be removed, kept identical to itself or copied (cloned) more than once. Here again, unlike our approach, each node is cloned together with all its incoming and outgoing edges.

6 Conclusion

We have investigated a new way to perform node cloning in graph transformation with some flexibility in copying incident edges. To obtain this result, we have used an auxiliary category of polarized graphs which allows one to declare how incident edges are cloned. The algebraic definition of a graph rewriting step is based on a sesqui-pushout transformation in the auxiliary category. In [8], the reader may find more results such as the equivalence of the algorithmic and the algebraic definitions of PSqPO as well as the vertical composition of transformations.

In [18], Löwe proposes a general framework of graph rewriting in span-categories. He shows how classical algebraic graph transformation approaches can be seen as instances of his framework. Our approach, which is close to the sesqui-pushout rewriting, could be presented also as an instance of Löwe's framework up to some particular considerations due to the use of two kinds of graphs in our spans, namely polarized and not polarized graphs. Details of the instance, including the complete definitions of abstract spans and matching of abstract spans are matter of further investigation.

References

1. F. Baader and T. Nipkow. *Term rewriting and all that.* Cambridge University Press, 1998.
2. H. Barendregt, M. van Eekelen, J. Glauert, R. Kenneway, M. J. Plasmeijer, and M. Sleep. Term graph rewriting. In *PARLE'87*, pages 141–158. LNCS 259, 1987.
3. R. V. Book and F. Otto. *String-rewriting systems.* Springer-Verlag, 1993.
4. A. Corradini, T. Heindel, F. Hermann, and B. König. Sesqui-pushout rewriting. In *ICGT'06*, pages 30–45. LNCS 4178, 2006.
5. A. Corradini, U. Montanari, F. Rossi, H. Ehrig, R. Heckel, and M. Löwe. Algebraic approaches to graph transformation - part I: Basic concepts and double pushout approach. In *Handbook of Graph Grammars*, pages 163–246, 1997.
6. F. Drewes, B. Hoffmann, D. Janssens, M. Minas, and N. V. Eetvelde. Adaptive star grammars. In *ICGT'06*, pages 77-91. LNCS 4178, 2006.
7. D. Duval, R. Echahed, and F. Prost. A heterogeneous pushout approach to term-graph transformation. In *RTA'09*, pages 194-208. LNCS 5595, 2009.
8. D. Duval, R. Echahed, and F. Prost. Graph rewriting with polarized cloning. arXiv:0911.3786,V3, 2012.
9. R. Dyckhoff and W. Tholen. Exponentiable morphisms, partial products and pullback complements. *Journal of Pure and Applied Algebra*, 1987.
10. R. Echahed. Inductively sequential term-graph rewrite systems. In *ICGT'08*, pages 84-98. LNCS 5214, 2008.
11. H. Ehrig, G. Engels, H.-J. Kreowski, and G. Rozenberg, editors. *Handbook of Graph Grammars and Computing by Graph Transformations, Volume 2: Applications, Languages and Tools.* World Scientific, 1999.
12. H. Ehrig, R. Heckel, M. Korff, M. Löwe, L. Ribeiro, A. Wagner, and A. Corradini. Algebraic approaches to graph transformation - part II: Single pushout approach and comparison with double pushout approach. In *Handbook of Graph Grammars*, pages 247–312, 1997.
13. H. Ehrig, H.-J. Kreowski, U. Montanari, and G. Rozenberg, editors. *Handbook of Graph Grammars and Computing by Graph Transformations, Volume 3: Concurrency, Parallelism and Distribution.* World Scientific, 1999.
14. H. Ehrig, M. Pfender, and H. J. Schneider. Graph-grammars: An algebraic approach. In *14th Annual Symposium on Foundations of Computer Science (FOCS), 15-17 October 1973, The University of Iowa, USA*, pages 167–180. IEEE, 1973.
15. J. Gosling, B. Joy, G. Steele, G. Bracha, and A. Buckley. *The Java(TM) Language Specification. Java SE 7 Edition.* Oracle documentation, 2005.
16. R. Kennaway. On "on graph rewritings". *Theoretical Computer Science*, 52:37–58, 1987.
17. M. Löwe. Algebraic approach to single-pushout graph transformation. *Theoretical Computer Science*, 109(1&2):181–224, 1993.
18. M. Löwe. Graph rewriting in span-categories. In *ICGT'10*, pages 218-233. LNCS 6372, 2010.
19. S. Mac Lane. *Categories for the Working Mathematician*, volume 5. Springer-Verlag, second edition, 1998.
20. F. Miller, A. Vandome, and J. McBrewster. *Eiffel (programming language).* Alphascript publishing, 2010.
21. J. C. Raoult. On graph rewriting. *Theoretical Computer Science*, 32:1–24, 1984.
22. G. Rozenberg, editor. *Handbook of Graph Grammars and Computing by Graph Transformations, Volume 1: Foundations.* World Scientific, 1997.