

Graph rewriting with polarized cloning

CCS - IC 2009

D. Duval and R. Echahed and F. Prost

LJK - LIG - Université de Grenoble

26 Nov. 2009

Introduction 1

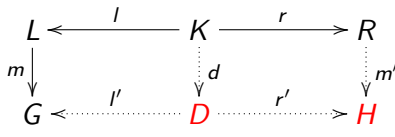
- Unlike term rewriting, there is no “natural” graph rewriting framework (tree vs graph).

Introduction 1

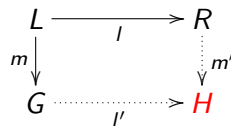
- Unlike term rewriting, there is no “natural” graph rewriting framework (tree vs graph).
- Two streams:
 - ⇒ Algorithmic approaches.
 - ⇒ Algebraic approaches.

Introduction 1

- Unlike term rewriting, there is no “natural” graph rewriting framework (tree vs graph).
- Two streams:
 - \Rightarrow Algorithmic approaches.
 - \Rightarrow Algebraic approaches.
- Pushouts everywhere:



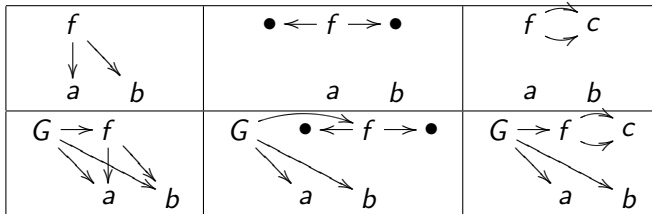
Double pushout



Single pushout

Introduction 2

- Pushouts are useful to glue together different parts.



Introduction 3

- The rule $f(x) \rightarrow f(b)$ can be implemented in DPO by:

$$\begin{array}{ccccc} f(x) & \longleftarrow & K & \longrightarrow & f(b) \\ \downarrow & & \downarrow & & \downarrow \\ G[f(a)] & \longleftarrow & D[a] & \longrightarrow & H[\textcolor{red}{a}, b] \end{array}$$

Introduction 3

- The rule $f(x) \rightarrow f(b)$ can be implemented in DPO by:

$$\begin{array}{ccccc} f(x) & \longleftarrow & K & \longrightarrow & f(b) \\ \downarrow & & \downarrow & & \downarrow \\ G[f(a)] & \longleftarrow & D[a] & \longrightarrow & H[\textcolor{red}{a}, b] \end{array}$$

- Deletion of nodes can be done thanks to partial morphisms (deletion of incoming edges).

Introduction 3

- The rule $f(x) \rightarrow f(b)$ can be implemented in DPO by:

$$\begin{array}{ccccc} f(x) & \longleftarrow & K & \longrightarrow & f(b) \\ \downarrow & & \downarrow & & \downarrow \\ G[f(a)] & \longleftarrow & D[a] & \longrightarrow & H[\textcolor{red}{a}, b] \end{array}$$

- Deletion of nodes can be done thanks to partial morphisms (deletion of incoming edges).
- Cloning nodes and edges with polarized graphs

Introduction 3

- The rule $f(x) \rightarrow f(b)$ can be implemented in DPO by:

$$\begin{array}{ccccc} f(x) & \longleftarrow & K & \longrightarrow & f(b) \\ \downarrow & & \downarrow & & \downarrow \\ G[f(a)] & \longleftarrow & D[a] & \longrightarrow & H[\textcolor{red}{a}, b] \end{array}$$

- Deletion of nodes can be done thanks to partial morphisms (deletion of incoming edges).
- Cloning nodes and edges with polarized graphs
 - 0 clone stands for deletion.

Introduction 3

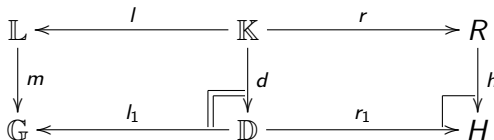
- The rule $f(x) \rightarrow f(b)$ can be implemented in DPO by:

$$\begin{array}{ccccc} f(x) & \longleftarrow & K & \longrightarrow & f(b) \\ \downarrow & & \downarrow & & \downarrow \\ G[f(a)] & \longleftarrow & D[a] & \longrightarrow & H[\textcolor{red}{a}, b] \end{array}$$

- Deletion of nodes can be done thanks to partial morphisms (deletion of incoming edges).
- **Cloning nodes and edges with polarized graphs**
 - 0 clone stands for deletion.
 - + clones : output clones.
 - - clones : input clones.

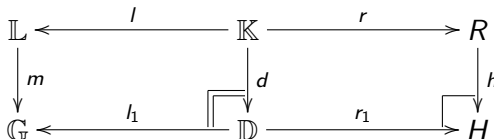
One and a half pushout

- Our rewrite step is heterogeneous:
 - 1 One pushback and one pushout
 - 2 Pushback is done on polarized graphs, pushout on graphs.

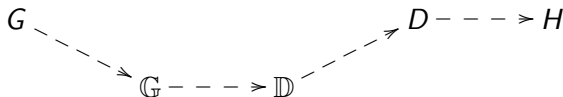


One and a half pushout

- Our rewrite step is heterogeneous:
 - 1 One pushback and one pushout
 - 2 Pushback is done on polarized graphs, pushout on graphs.

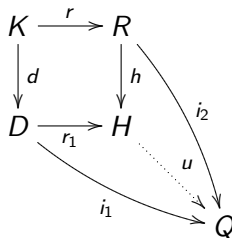


- Rewriting is done through 4 steps:



Pushout

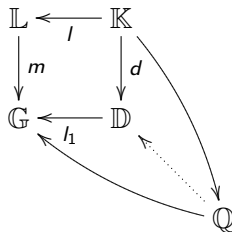
- Pushout : colimit of a span



(r_1, h) is the pushout of (d, r) .

Pushback

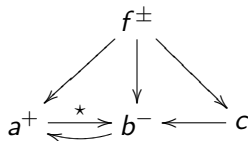
- Pullback : dual of a pushout
- Pushback : Final pullback complement.



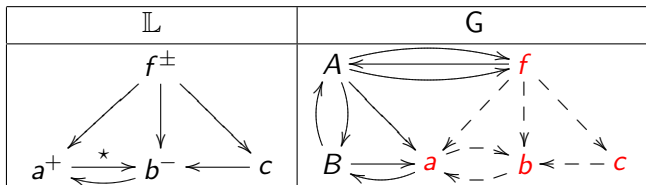
(d, l_1) is the pushback of (l, m)

Polarized Graphs

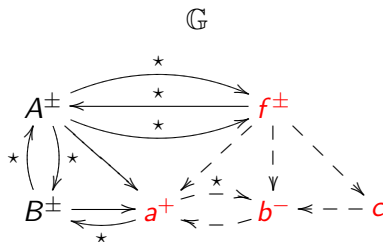
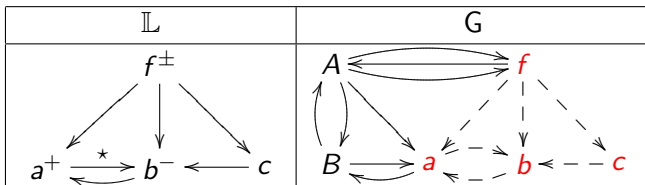
- A *polarization* X° of a graph X is a triple $(|X|^+, |X|^-, X_\rightarrow^*)$ such that $|X|^+ \subseteq |X|$, $|X|^- \subseteq |X|$ and $X_\rightarrow^* \subseteq X_\rightarrow$, such that each $n \xrightarrow{e} p$ in X_\rightarrow^* has its source $n \in |X|^+$ and its target $p \in |X|^-$.



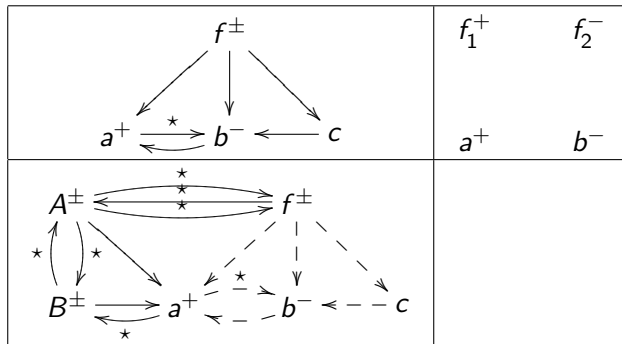
Polarized Graph induced by matching



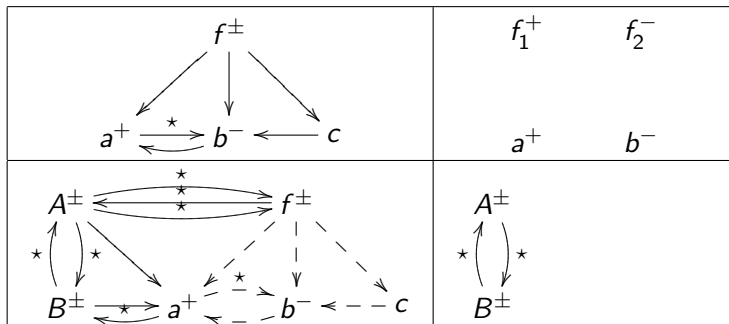
Polarized Graph induced by matching



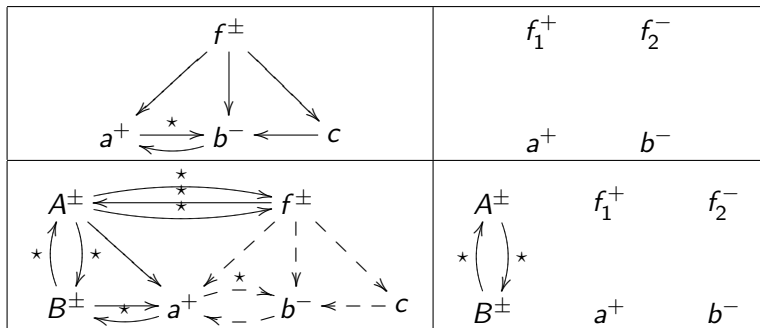
Pushback on Polarized Graphs



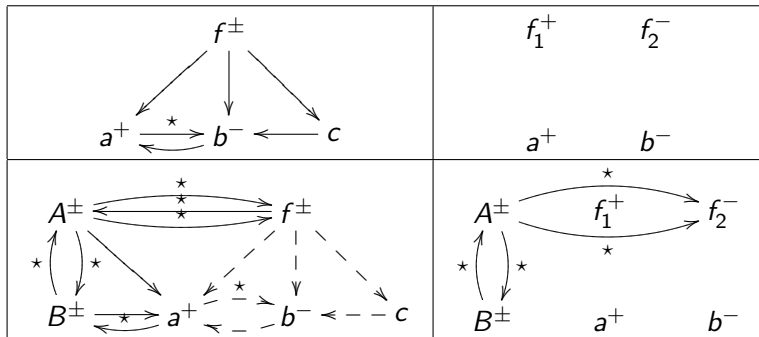
Pushback on Polarized Graphs



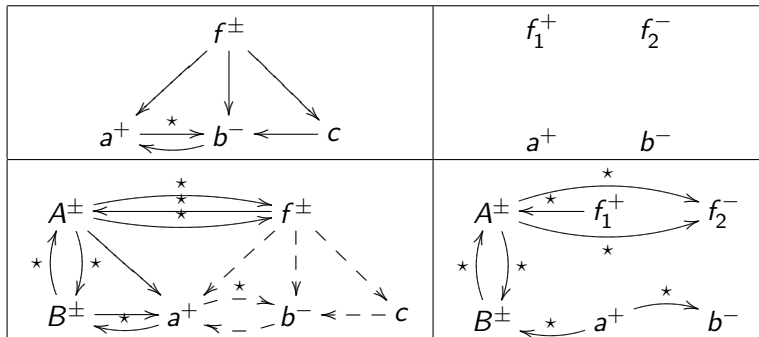
Pushback on Polarized Graphs



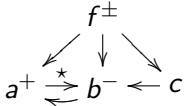
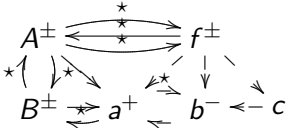
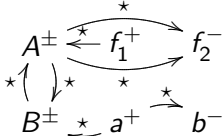
Pushback on Polarized Graphs



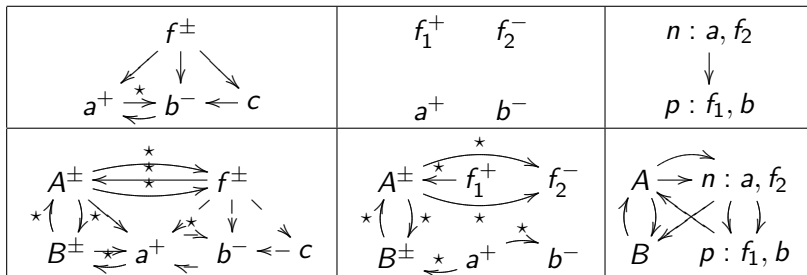
Pushback on Polarized Graphs



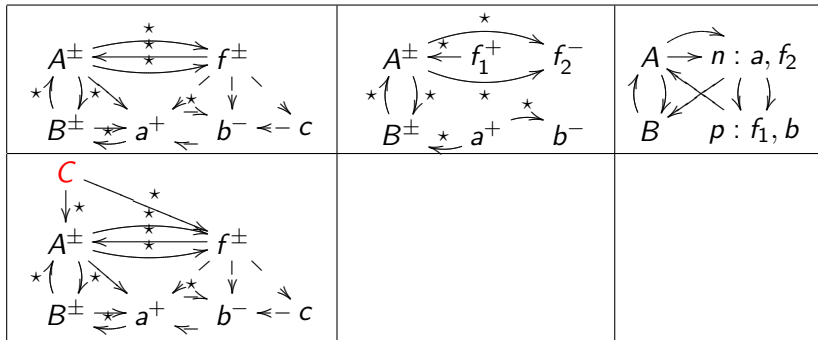
A complete rewrite step

	$\begin{array}{cc} f_1^+ & f_2^- \\ a^+ & b^- \end{array}$	$\begin{array}{c} n : a, f_2 \\ \downarrow \\ p : f_1, b \end{array}$
		

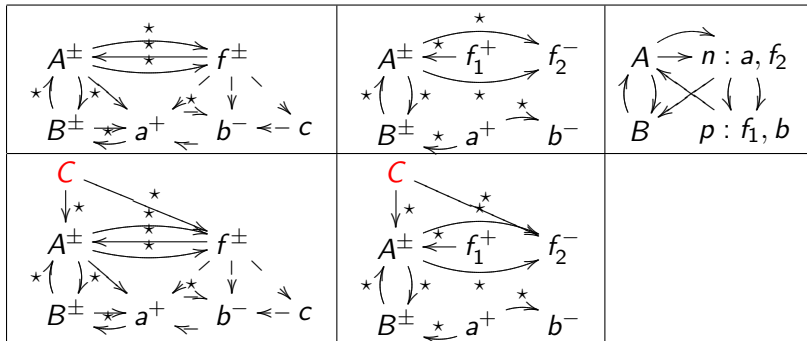
A complete rewrite step



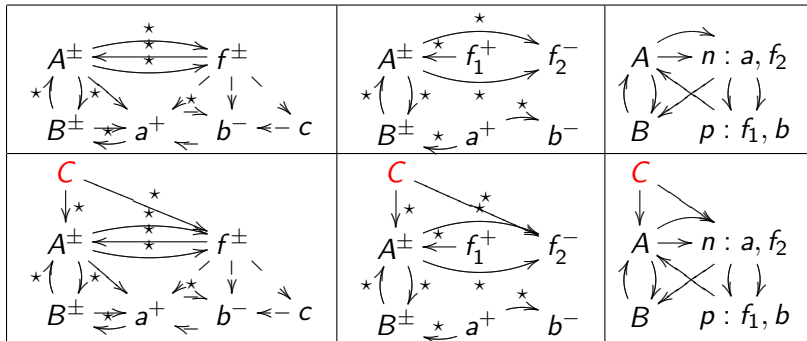
Contextual closure of rewrite rules



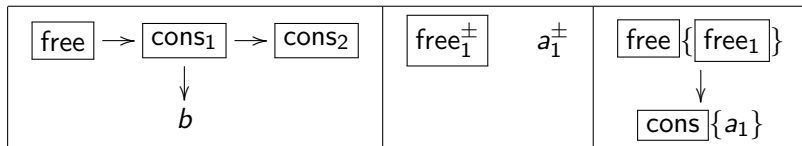
Contextual closure of rewrite rules



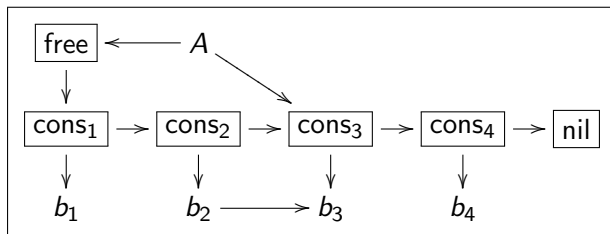
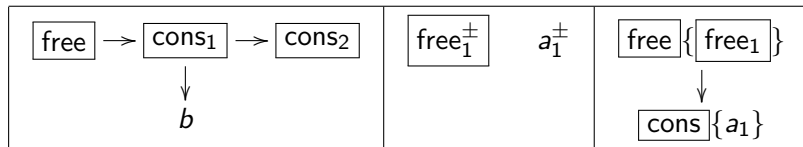
Contextual closure of rewrite rules



Node deletion

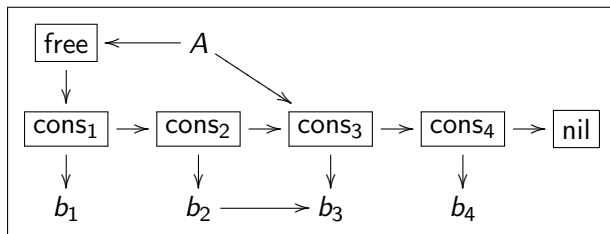
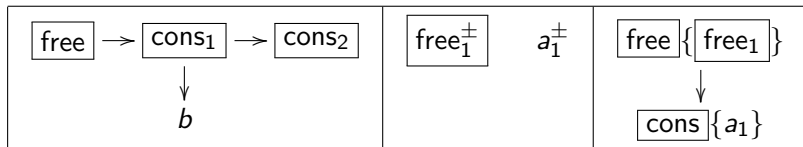


Node deletion

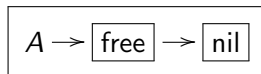


rewrites to

Node deletion



rewrites to



Global update

<div><div>update</div><div>↓</div><div>new^{\pm}</div></div> <div>$\rightarrow old^{-}$</div>	$new_1^{\pm} \quad old_1^{-}$	$g\{new_1, old_1\}$

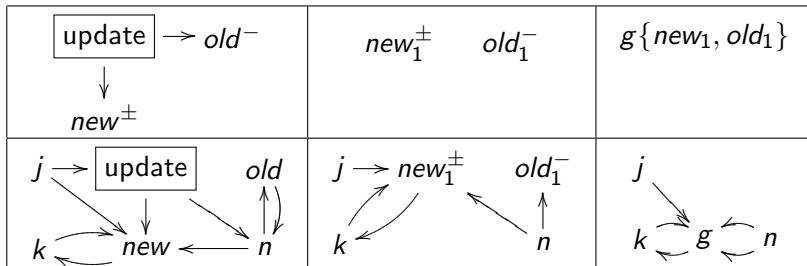
Global update

$\boxed{\text{update}} \rightarrow old^-$ \downarrow new^\pm	$new_1^\pm \quad old_1^-$	$g\{new_1, old_1\}$
<p>Diagram illustrating a global update operation. A box labeled <code>update</code> has an incoming arrow from <code>j</code> and an outgoing arrow to <code>new</code>. <code>new</code> has an incoming arrow from <code>k</code> and an outgoing arrow to <code>n</code>. <code>n</code> has an incoming arrow from <code>old</code> and an outgoing arrow to <code>old</code> (forming a loop).</p>		

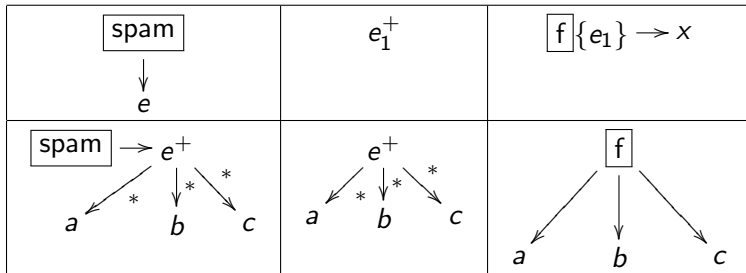
Global update

$\boxed{\text{update}} \rightarrow old^-$ \downarrow new^\pm	$new_1^\pm \quad old_1^-$	$g\{new_1, old_1\}$
<p>Diagram illustrating the state after a global update. A box labeled <code>update</code> has an arrow pointing to <code>new</code>. <code>j</code> points to <code>update</code>. <code>k</code> points to <code>new</code>. <code>old</code> and <code>n</code> are connected by a double-headed arrow. <code>n</code> points to <code>new</code>.</p>	<p>Diagram illustrating the state before a global update. <code>j</code> points to <code>new_1^\pm</code>. <code>k</code> points to <code>new_1^\pm</code>. <code>old_1^-</code> and <code>n</code> are connected by a double-headed arrow. <code>n</code> points to <code>new_1^\pm</code>.</p>	

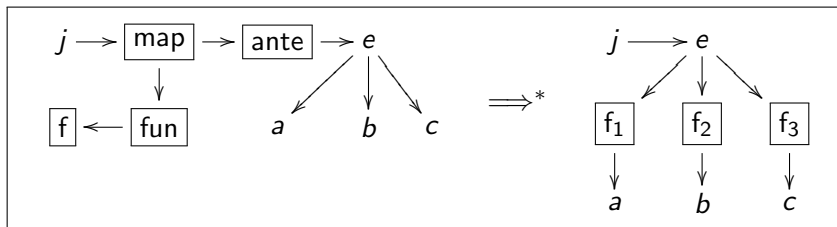
Global update



Spamming



Using both $-$ and $+$ cloning



Conclusion

- This framework allows a flexible approach to cloning.
- It subsumes many previous works

Conclusion

- This framework allows a flexible approach to cloning.
- It subsumes many previous works
 - Heterogeneous pushouts.

Conclusion

- This framework allows a flexible approach to cloning.
- It subsumes many previous works
 - Heterogeneous pushouts.
 - Sesqui-pushout.

Conclusion

- This framework allows a flexible approach to cloning.
- It subsumes many previous works
 - Heterogeneous pushouts.
 - Sesqui-pushout.
 - Adaptative star grammars.

Conclusion

- This framework allows a flexible approach to cloning.
- It subsumes many previous works
 - Heterogeneous pushouts.
 - Sesqui-pushout.
 - Adaptative star grammars.
- Termgraphs.