Termination of Injective DPO Graph Rewriting Systems using Subgraph Counting

Qi QIU

Supervisor: Xavier URBAIN
Universite Claude Bernard Lyon 1, France
Univ. Grenoble Alpes, France

Double-pushout (DPO) rewriting with injective rules and matches

- Graph : finite edge-labeled directed multigraph
- Rules $\varphi = (L \overset{l}{\leftarrowtail} K \overset{r}{\rightarrowtail} R)$ consist of two inclusions l and r.
- Rule $\varphi' = (L' \overset{l'}{\leftarrowtail} K' \overset{r'}{\rightarrowtail} R')$ and φ are equivalent if there are isomorphisms a, b, c such that:

$$\begin{tikzcd}
L' \arrow[d, tail, "a"] \arrow[dr, phantom, "="] & K' \arrow[l, tail, "l'"'] \arrow[r, tail, "r'"] \arrow[d, tail, "b"] \arrow[dr, phantom, "="] & R' \arrow[d, tail, "c"'] \\
L & K \arrow[l, tail, "l"'] \arrow[r, tail, "r"] & R
\end{tikzcd}$$

- Rewriting steps $G \Rightarrow_{\varphi} H$ are double-pushouts

$$\begin{tikzcd}
L' \arrow[d, tail] \arrow[dr, phantom, "\mathrm{PO}"] & K' \arrow[l, tail, "l'"'] \arrow[r, tail, "r'"] \arrow[d, tail] \arrow[dr, phantom, "\mathrm{PO}"] & R' \arrow[d, tail] \\
G & C \arrow[l, tail] \arrow[r, tail] & H
\end{tikzcd}$$

where all arrows are inclusions.

Termination [2]

- $\mathcal{R}$: set of DPO graph rewriting rules
- impossibility of transforming any graph G_0 indefinitely

  $$G_0 \Rightarrow_{\mathcal{R}} G_1 \Rightarrow_{\mathcal{R}} G_2 \Rightarrow_{\mathcal{R}} \cdots$$

  with the non-deterministic strategy

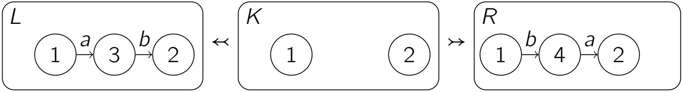
  "apply rules as long as possible"
- Corresponds to program termination on all inputs in conventional programming languages
- Undecidable in general[1]

[1] **plump1998terminationundecidable**.

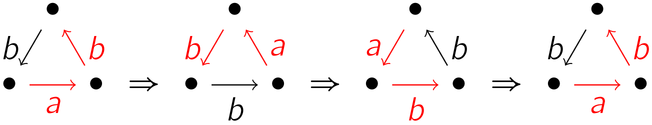
[2] this slide is from the slides of a presentation given by Plump

One-rule examples

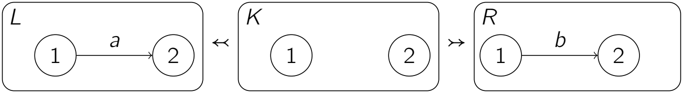
Rule a :



Looping:



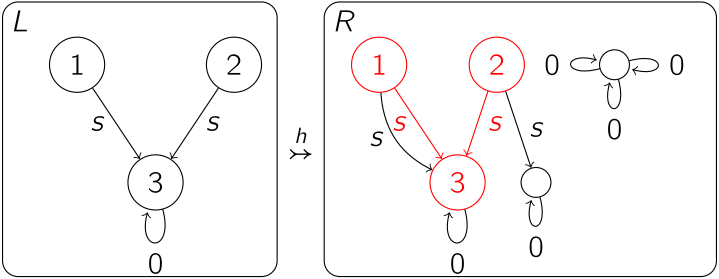
Rule b:



Terminating: the number of edges labeled by “a” strictly decreases.

Visual Notation for Graph Morphisms

Visualization of an injective graph morphism[3]:

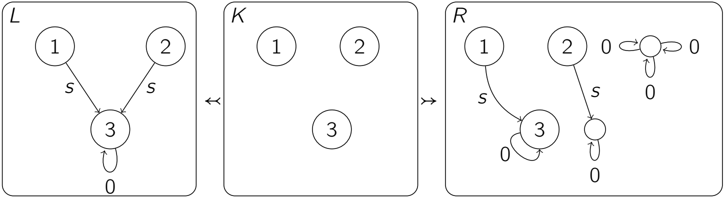


- Graphs are contained in boxes
- Graph name is placed in the top left corner
- Codomain is represented as a supergraph of domain
- Some nodes are marked by numbers for clarity
- Morphism name is placed between the boxes

[3] **overbeek2023apbpotutorial**.

Termination by Subgraph Counting

- Motivating rule φ:



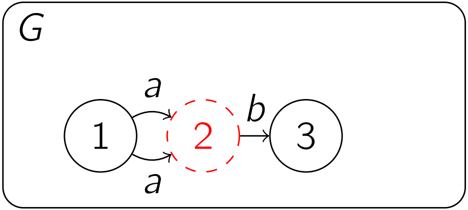
- Termination by strict decrease of the number of occurrences of $\bullet \xrightarrow{s} \bullet \xleftarrow{s} \bullet$
- Can we prove its termination by a machine-checkable condition?

Plan

- Pre-graphs
- Analysis of subgraph change before and after rewriting
- Sufficient conditions for termination by subgraph counting

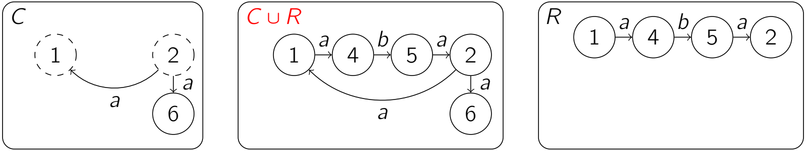
Pre-graphs

- Pre-graphs are graphs with missing nodes and dangling edges.
- Example:

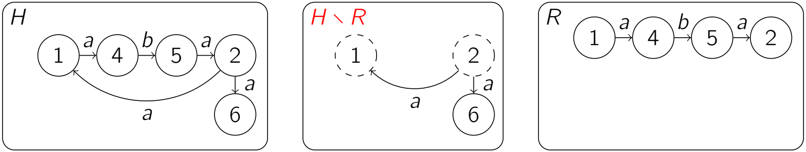


Pre-graph operations

Union $C \cup R$ of two pre-graphs $C \subseteq G$ and $R \subseteq G$

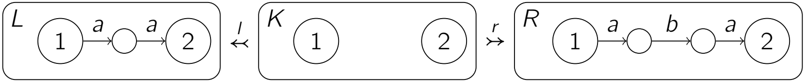


Relative complement of R in H where $R \subseteq H$, denoted $H \smallsetminus R$:



Running example

We will use the following running example to analyze the subgraph changes before and after a rewriting step.

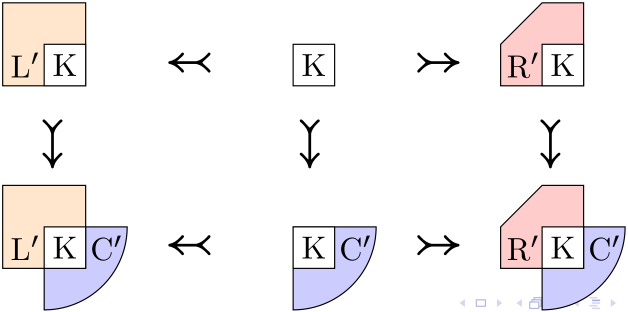


It replaces chain $\bullet \xrightarrow{a} \bullet \xrightarrow{a} \bullet$ with $\bullet \xrightarrow{a} \bullet \xrightarrow{b} \bullet \xrightarrow{a} \bullet$.

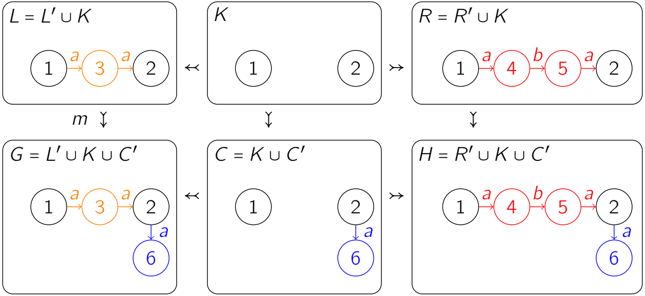
Analysis of rewriting steps

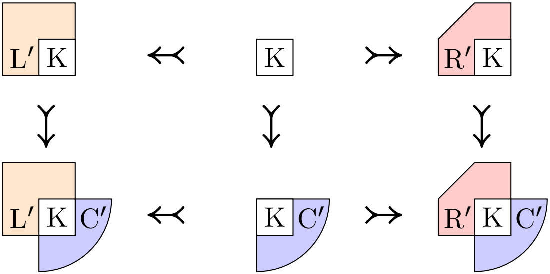
$$\begin{tikzcd}
L \arrow[d, tail] & K \arrow[l, tail] \arrow[r, tail] \arrow[d, tail] & R \arrow[d, tail] \\
G & C \arrow[l, tail] \arrow[r, tail] & H
\end{tikzcd}$$

All arrows are inclusions by definition.
Graphs can be decomposed as unions of pre-graphs:



Example

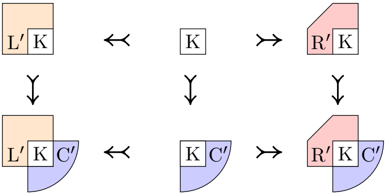

$L = L' \cup K$
1 a 3 a 2
K
1 2
$R = R' \cup K$
1 a 4 b 5 a 2
m
$G = L' \cup K \cup C'$
1 a 3 a 2 a 6
$C = K \cup C'$
1 2 a 6
$H = R' \cup K \cup C'$
1 a 4 b 5 a 2 a 6


L′ K
K
R′ K
L′ K C′
K C′
R′ K C′

Implicit, Explicit and Shared Occurrences

An X-occurrence in a graph G is a subgraph of G isomorphic to X.

$$\begin{tikzcd} L \arrow[d, tail] & K \arrow[l, tail] \arrow[r, tail] \arrow[d, tail] & R \arrow[d, tail] \\ G & C \arrow[l, tail] \arrow[r, tail] & H \end{tikzcd}$$



X : a subgraph of L

X-occurrence in G is

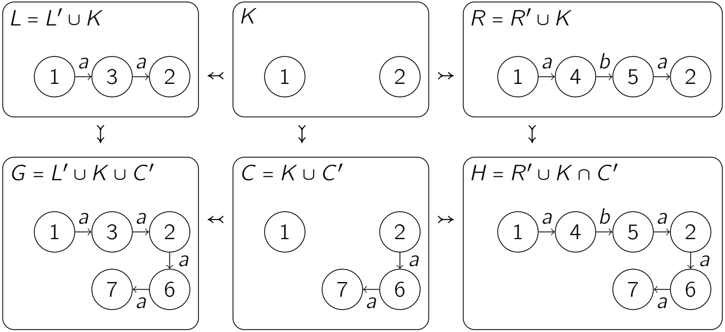
- shared if included in C
- explicit if included in L
- implicit if having elements in both C' and L'

X-occurrence in H is

- shared if included in C
- explicit if included in R
- implicit if having elements in both C' and R'

Example

Consider the occurrences of the graph $\bullet \xrightarrow[a]{} \bullet \xrightarrow[a]{} \bullet$ in



- Occurrence shared by G and H: 2-6-7
- Explicit occurrence in G: 1-3-2
- Implicit occurrences in G: 3-2-6
- 0 explicit occurrence in H
- Implicit occurrences in H: 5-2-6

X-occurrences in a graph G

Remark:

$$\begin{aligned}|X\text{-occurrences}| = &|\text{explicit } X\text{-occurrences}| + \\ &|\text{shared } X\text{-occurrences}| + \\ &|\text{implicit } X\text{-occurrences}|\end{aligned}$$

Termination of a graph rewriting rule φ

- φ : rewriting rule
- X: a subgraph of the left-hand side graph of φ
- φ terminates if for all $G \Rightarrow_{\varphi} H$, the number of X-occurrences strictly decreases.
- φ terminates if for all $G \Rightarrow_{\varphi} H$,

  $$(|\text{explicit } X\text{-occurrences in } G| - |\text{explicit } X\text{-occurrences in } H|) + (|\text{implicit } X\text{-occurrences in } G| - |\text{implicit } X\text{-occurrences in } H|) > 0$$

  because shared X-occurrences in G and H are the same.
- φ terminates if for all $G \Rightarrow_{\varphi} H$,

  $$|\text{explicit } X\text{-occurrences in } G| > |\text{explicit } X\text{-occurrences in } H|$$
  $$|\text{implicit } X\text{-occurrences in } G| \geq |\text{implicit } X\text{-occurrences in } H|$$

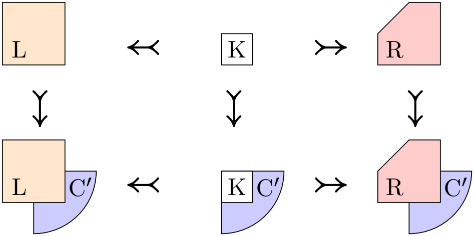
Termination of a graph rewriting rule

The first condition

$$|\text{explicit } X\text{-occurrences in } G| > |\text{explicit } X\text{-occurrences in } H|$$

is equivalent to

$$|X\text{-occurrences in } L| > |X\text{-occurrences in } R|$$



Therefore, the key challenge is to show

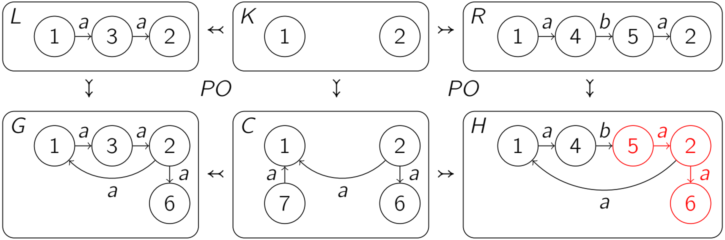
$$|\text{implicit } X\text{-occurrences in } G| \geq |\text{implicit } X\text{-occurrences in } H|$$

for all rewriting steps $G \Rightarrow_{\varphi} H$.

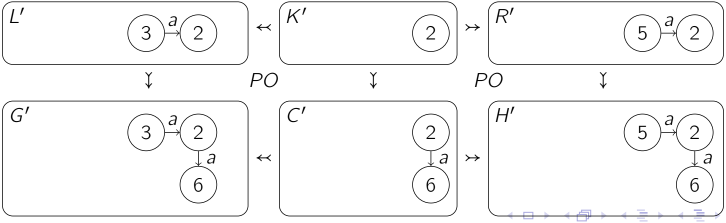
We construct an injection from the implicit X-occurrences in H to the implicit X-occurrences in G.

Analysis of Implicit Occurrences

Consider the implicit occurrence of $\bullet \xrightarrow[a]{} \bullet \xrightarrow[a]{} \bullet$ in H:

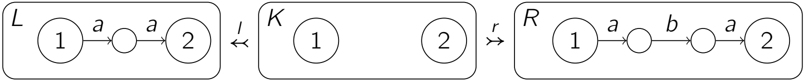


The occurrence is $C' \cup R'$. C' is shared by G and H. R' is not in G but there L' is in G and $C' \cup L'$ is an implicit occurrence in G.

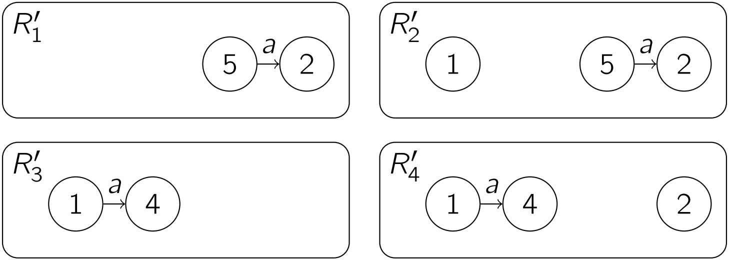


Distinguished Subgraph $D(R, X)$: subgraphs of R which can form an implicit X-occurrence in some rewriting step

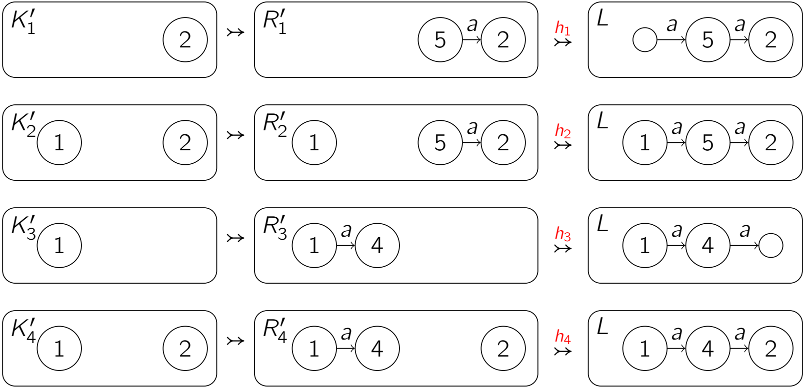
Running example:



$D(R, X)$ consists of :



Observation: For every $R_i \in D(R, X)$, there is a morphism $h_i : R_i \rightarrowtail L$ which preserves the interface elements



X-non-increasing rule

Definition

A rule φ is X-non-increasing rule if

1. For every $R_i \in D(R, X)$, there is a morphism $h_i : R_i \rightarrowtail L$ which preserves the interface elements,
2. three more conditions on h_i

Condition 1 guarantees every implicit X-occurrence in H has a corresponding implicit X-occurrence in G with the same interface elements

Other conditions guarantee: different implicit X-occurrences in H have different corresponding implicit X-occurrences in G

Main Results

Let φ be a X-non-increasing rule.

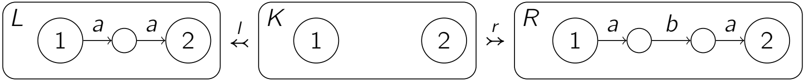
Lemma (More X-occurrences before rewriting)

For all $G \Rightarrow_{\varphi} H$, there are more implicit X-occurrences in G than in H.

Theorem (Sufficient Termination Condition)

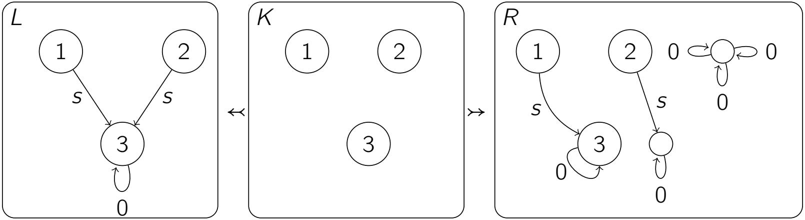
φ is terminating if there are strictly more explicit X-occurrences in L than in R.

Terminating of Running Example



- X : $\bullet \xrightarrow[a]{} \bullet \xrightarrow[a]{} \bullet$
- X-non-increasing rule
- Strictly more explicit X-occurrences in L than in R: $1 > 0$.

Termination of Motivating Rule



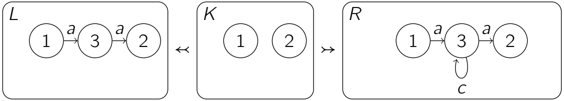
- X : $\bullet \xrightarrow{s} \bullet \xleftarrow{s} \bullet$
- $D(R, X)$ consists of R_1: $1 \xrightarrow{s} 3$ and R_2: $1 \xrightarrow{s} 3 \quad 2$
- X-non-increasing rule because inclusions from $h_1 : R_1 \rightarrowtail L$ and $h_2 : R_2 \rightarrowtail L$ satisfy all conditions.
- Strictly more explicit X-occurrences in L than in R : $1 > 0$
- Terminating

Relative work

- Termination of PBPO+ rewriting using weighted subgraph counting [**overbeek2024termination˙lmcs**]
  - same idea
  - more general
  - cannot prove termination of the motivating rule
- Forward Closure Method [**plump1995ontermination**]
  - proves termination of the motivating rule
  - not easy to apply: necessary and sufficient termination condition
- Termination of rewriting systems using weighted type graphs [**zantema2014termination**; **bruggink2014termination**; **bruggink2015proving**; **endrullis2024generalized˙arxiv˙v2**; **qiu2025termination˙nwf˙v2˙acceptedgcm**]
  - more general
  - cannot prove termination of the motivating rule
- Modular Termination Method [**plump2018modular**]
  - termination of the union of two rule sets
  - our method complements this method

An extension for counting subgraphs with antipattern

An extension has been developed and implemented for termination of the following rule:



by counting L-occurrences which are not included in R-occurrences.

Future Work

- Extension to rules with negative application conditions

Conclusion

Subgraph Counting method

- machine-checkable sufficient termination condition
- for injective DPO graph rewriting
- by counting occurrences of a subgraph X
- implemented in **LyonParallel** [4]

[4] `github.com/Qi-tchi/LyonParallel/tree/icgt2025`, MIT License

Acknowledgements

- Thanks to reviewers for their valuable comments and suggestions.
- Termination section uses a slide from a presentation by Plump in 2018.

References I