Termination of Graph Rewriting using Weighted Type Graphs over Non-well-founded Semirings

Qi QIU

Supervisor: Xavier URBAIN
Université Claude Bernard Lyon 1, France
Université de Lorraine, France

Plan (for myself)

- Introduction
  - Graphs and Graph Morphisms
  - Double Pushout Diagram (DPO)
  - Graph rewriting using DPO
    - Example of DPO graph rewriting : grsaa
  - Termination of DPO graph rewriting systems
    - Example: looping and terminating examples
- Type graph method in theory
- Type graph method in practice and the introduction of the main problem
- Morphism weight (by a example)
- Graph weight (by a example)
- A condition that weighted type graphs must satisfy
- Rule φ
- Example with rule φ
- Sufficient condition for termination using weighted type graph
- Sufficient condition simplified
- Example

Outline

Graphs

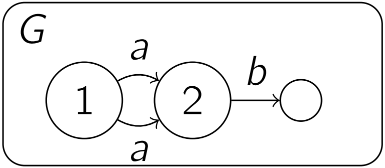
Graphs are finite and directed with:

- Parallel edges,
- Labeled edges,
- Finite labels.

Example:

- Graph G with 3 nodes.
- 2 edges from node 1 to node 2 labeled by a.
- 1 edge from node 2 to node 3 labeled by b.

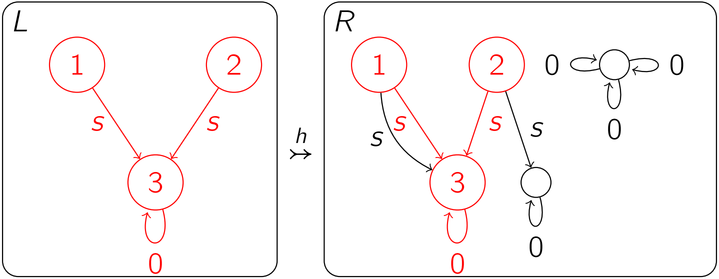
Notation:



- Graphs are contained in boxes
- Graph name G is placed in the top left corner
- Numbers in nodes are identifiers, ommitted when not relevant

Inclusion (morphism)

Morphisms: structure-preserving functions between graphs
Isomorphisms: bijective morphisms
Inclusions: morphisms that are indentity functions
Example in a viasual notion:



- $\overset{h}{\rightarrowtail}$ between the boxes marques an inclusion named h.

Graph rewriting rule

Rules $\varphi = (L \overset{l}{\leftarrowtail} K \overset{r}{\rightarrowtail} R)$ consist of inclusions l and r.

- Running example φ:

$$\boxed{L\quad 1 \xrightarrow{a} 3 \xrightarrow{a} 2} \overset{l}{\leftarrowtail} \boxed{K\quad 1 \qquad\qquad 2} \overset{r}{\rightarrowtail} \boxed{R\quad 1 \xrightarrow{a} 4 \xrightarrow{b} 5 \xrightarrow{a} 2}$$

Rule $\varphi' = (L' \overset{l'}{\leftarrowtail} K' \overset{r'}{\rightarrowtail} R')$ and φ are equivalent if there are isomorphisms a, b, c such that $a \circ l' = l \circ b$ and $c \circ r' = r \circ b$:

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

A rule equivalent to φ:

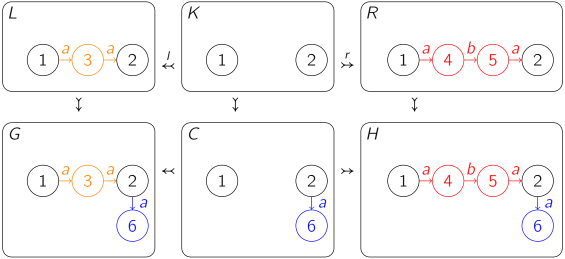
$$\boxed{L'\quad 6 \xrightarrow{a} 7 \xrightarrow{a} 8} \overset{l}{\leftarrowtail} \boxed{K'\quad 6 \qquad\qquad 7} \overset{r}{\rightarrowtail} \boxed{R'\quad 6 \xrightarrow{a} 9 \xrightarrow{b} 10 \xrightarrow{a} 7}$$

Graph Rewriting

Rewriting steps $G \Rightarrow_\varphi H$ using rule φ are commutative diagrams with an equivalent rule $L' \overset{l'}{\leftarrowtail} K' \overset{r'}{\rightarrowtail} R'$ where all morphisms are inclusions:

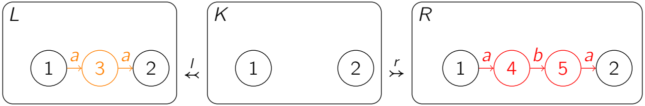
$$\begin{tikzcd} L' \arrow[d, tail] & K' \arrow[l, tail, "l'"'] \arrow[r, tail, "r'"] \arrow[d, tail] & R' \arrow[d, tail] \\ G & C \arrow[l, tail] \arrow[r, tail] & H \end{tikzcd}$$

A rewriting step using φ:

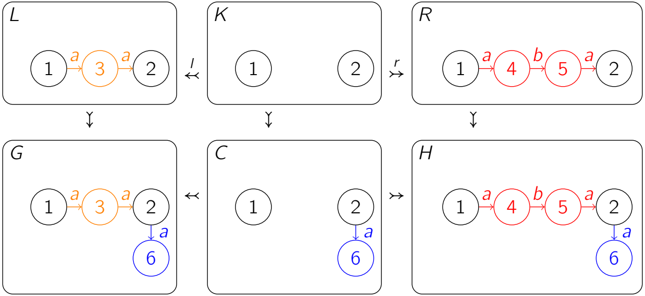


A rewriting step of the running rule (to do : delete)

Rewriting rule φ:



A rewriting step $G \Rightarrow_\varphi H$:



Termination of DPO Graph Rewriting Systems

- $\mathcal{R}$: a set of rules
- No graph G_0 can be rewritten forever:

$$G_0 \underset{\mathcal{R}}{\Rightarrow} G_1 \underset{\mathcal{R}}{\Rightarrow} G_2 \underset{\mathcal{R}}{\Rightarrow} \cdots$$

  when using the non-deterministic strategy

  "apply rules as long as possible"

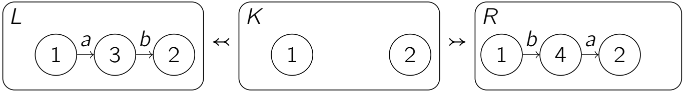
- Aligns with the standard notion of program termination:

  "every execution (on any input) eventually halts."

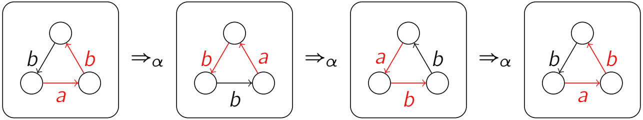
- Undecidable in general

One-rule examples

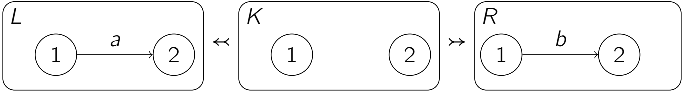
Rule α :



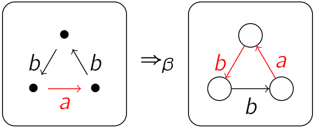
Looping:



Rule b:



Termination by the number of edges labeled by “a”.



Introduction

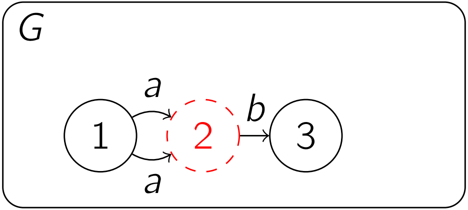
Termination of Injective DPO Graph Rewriting using Morphism Counting

Etending the Type Graph Method to Non-well-founded Semirings

Termination of Injective DPO Graph Rewriting using Morphism Counting with antipatterns

Pre-graphs

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

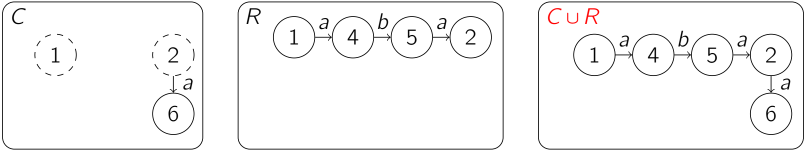


The pregraph G has

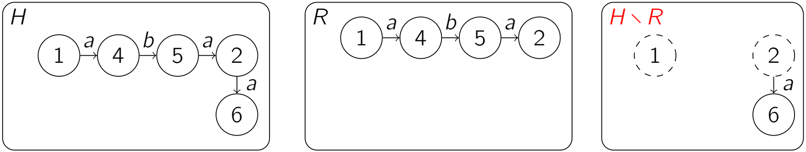
- 2 existing nodes,
- 1 missing node,
- 3 dangling edges.

Pre-graph operations

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



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



test

Introduction

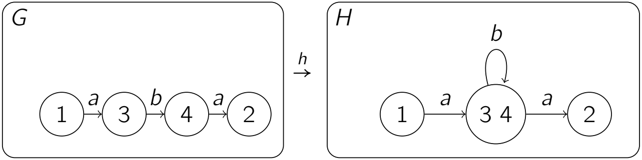
Termination of Injective DPO Graph Rewriting using Morphism Counting

Etending the Type Graph Method to Non-well-founded Semirings

Termination of Injective DPO Graph Rewriting using Morphism Counting with antipatterns

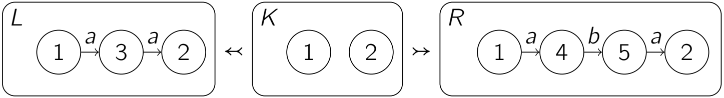
Graphs and Graph Morphisms

- **Graph morphisms**: structure-preserving mappings.
  Ex:

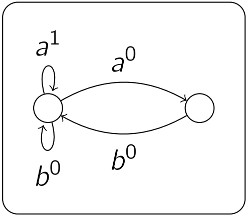


Type graph method with weighted type graphs over natural numbers with an example

- Rule α:

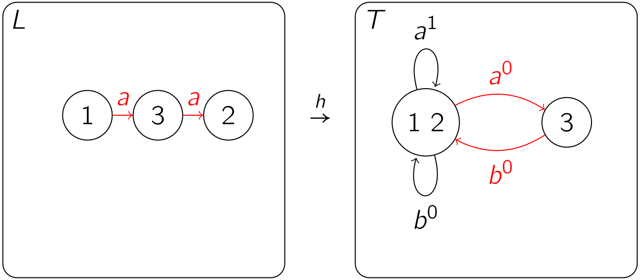


- Weighted type graph T over natural numbers:



Morphism Weight

- The weight of $h : L \to T$ is the sum of weights of all edges in $\mathrm{Im}(h)$.
- Example:

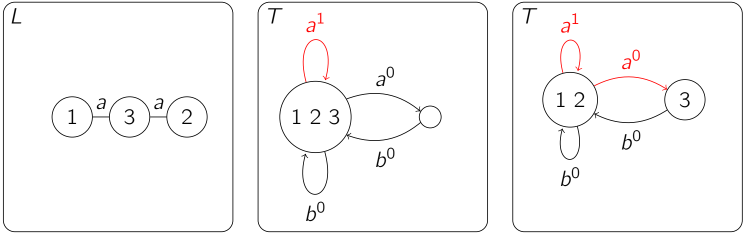


$w_T(h) = 0 + 0 = 0$

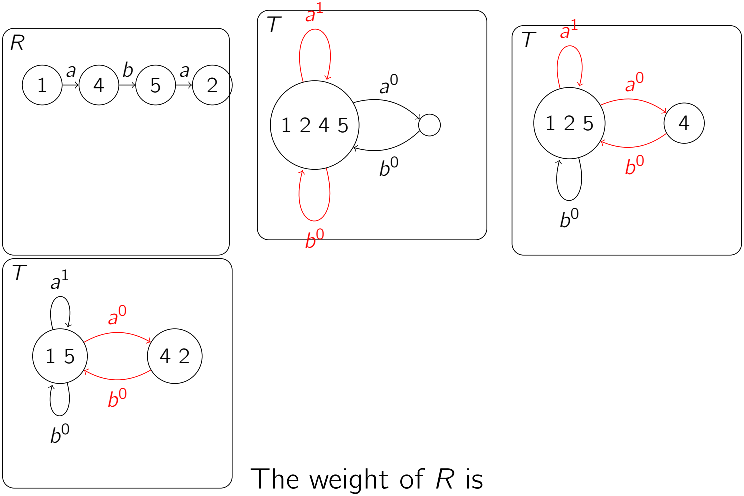
Graph Weight

$w_T(L)$: the minimum weight $w_T(h)$ of all morphisms $h : L \to T$

Example :



$w_T(L) = \min\{1 + 1, 1 + 0\} = 1$



The weight of R is

$$\min\{1+0+1, 0+0+1, 0+0+0\} = 0$$

Since for all $G \Rightarrow_\alpha H$, we have $w_T(G) \in \mathbb{N}$, it remains to show that every rewriting step strictly decreases the weight.
Not every weighted graph can be a weighted type graph It guarantees: for all $G \Rightarrow_\alpha H$, the weight of G is defined and $w_T(G) \in \mathbb{N}$

Termination Condition

$$L: \; 1 \xrightarrow{a} 3 \xrightarrow{a} 2 \quad \leftarrowtail \quad K: \; 1 \quad 2 \quad \rightarrowtail \quad R: \; 1 \xrightarrow{a} 4 \xrightarrow{b} 5 \xrightarrow{a} 2$$

For every morphism $t_K : K \to T$, we define

- $S(t_k, L)$: the sum of the weights of the morphisms t_L that extend t_K
- $S(t_k, R)$: the sum of the weights of the morphisms t_R that extend t_K

By [**endrullis2024generalized˙arxiv˙v2**], every rewriting step strictly decreases the weight if

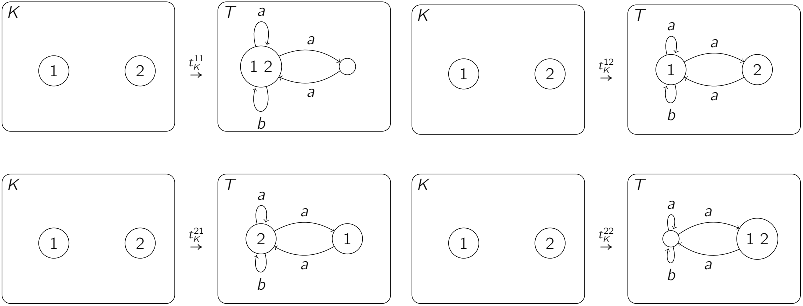
- for all $t_K : K \to T$,

$$S(t_K, L) > S(t_K, R)$$

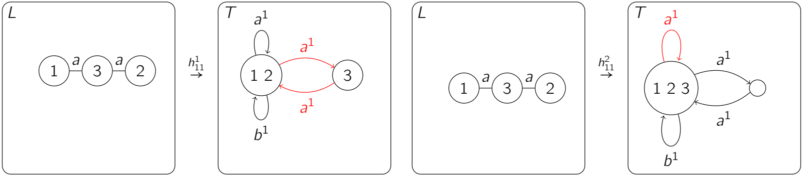
.

Example

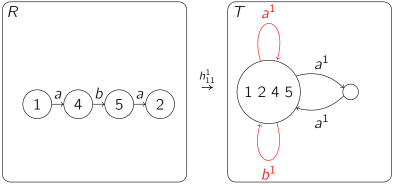
There are $t_K^{11}, t_K^{12}, t_K^{21}, t_K^{22} : K \to T$ as depicted below:



Detail for t_K^{11}



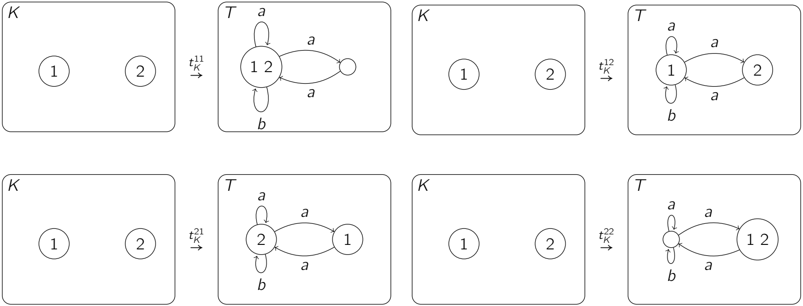
We have $S(t_k^{11}, L) = w_{\mathcal{T}}(h_{11}^1) + w_{\mathcal{T}}(h_{11}^2) = (1 * 1) + (1 * 1) = 2$



we have $S(t_k^{11}, R) = w_{\mathcal{T}}(h_{11}^3) = 1 * 1 * 1 = 1.$
Therefore, for t_K^{11}, we have: $S(t_k^{11}, L) = 2 > 1 = S(t_k^{11}, R).$

Example

There are $t_K^{11}, t_K^{12}, t_K^{21}, t_K^{22} : K \to T$ as depicted below:



- t_K^{11} : $2 > 1$
- t_K^{12} : $1 \geq 1$
- t_K^{21} : $1 \geq 1$
- t_K^{22} : $1 \geq 1$

Therefore, our Running Example is terminating by the weighted type graph T over natural numbers.

Type Graph Method with Weighted Type Graphs over Well-founded Semirings

The existence of suitable weighted type graphs is undecidable in general.

Searching for Weighted Type Graphs over Natural Numbers

User-specified parameters:

- k nodes
- edge weights in $\{0, 1, \ldots, n\}$

Assumption:

- no parallel edges of the same label

The problem amounts to checking the satisfiability of an existential Presburger arithmetic formula:

- $k^2|\Sigma|$ binary variables
- $k^2|\Sigma|$ integer variables

Challenge:

- there are $2^{k^2|\Sigma|} \cdot n^{k^2|\Sigma|}$ possible assignments of weights
- k and n unknown à priori

Solution: using real numbers instead of natural numbers.

Every rewriting step strictly decreases the weight if

- for every $t_K : K \to T$,

$$S(t_K, L) > S(t_K, R)$$

.

- there is $\delta > 0$ such that for some $t_K : K \to T$,

$$S(t_K, L) > S(t_K, R) + \delta$$

.

Searching for Weighted Type Graphs over Real Numbers

User-specified parameters:

- k nodes
- ~~edge weights in $\{0, 1, \ldots, n\}$~~

Assumption:

- no parallel edges of the same label

The problem amounts to checking the satisfiability of an existential Presburger arithmetic formula:

- $k^2|\Sigma|$ binary variables
- $k^2|\Sigma|$ real variables

Challenge:

- ~~there are $2^{k^2|\Sigma|} \cdot n^{k^2|\Sigma|}$ possible assignments of weights~~
- there are $2^{k^2|\Sigma|}$ linear programs which have polynomial-time average-case complexity

Implementation and Experimental Results

Implemented in the tool LyonParallel

- supports natural and real numbers
- searches with natural and real numbers in parallel and cooperates between them

Tested on examples from previous work:

- no need of user-specified upper bound on weights
- Acceleration

Introduction

Termination of Injective DPO Graph Rewriting using Morphism Counting

Etending the Type Graph Method to Non-well-founded Semirings

Termination of Injective DPO Graph Rewriting using Morphism Counting with antipatterns