Rewriting Higher-Order Stack Trees

Vincent Penelle

Journes SDA2, 10 avril 2015

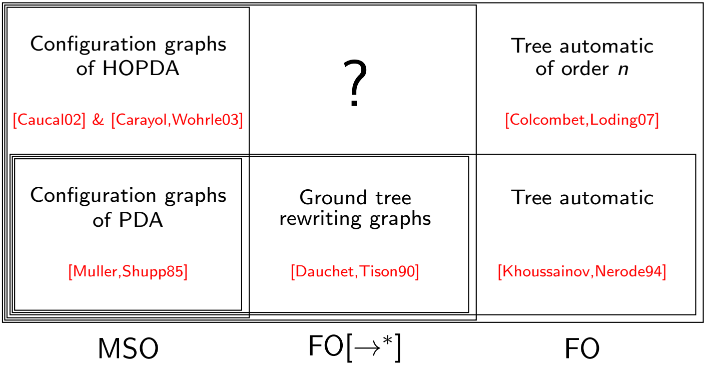
Table of Contents

Context

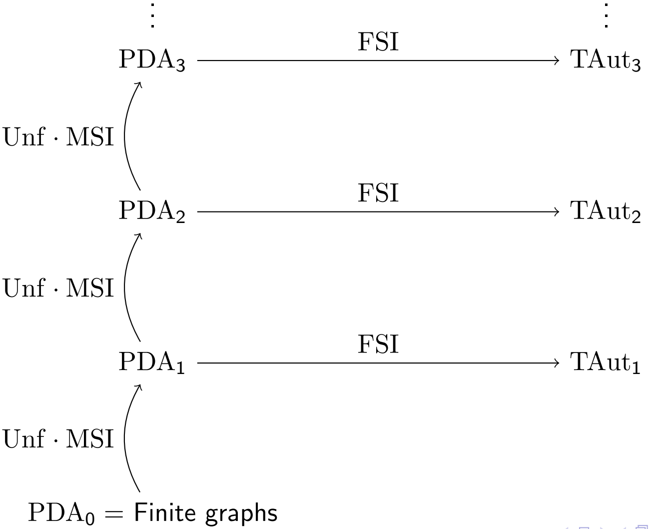
- Model Checking over some classes of infinite graphs: which logic theories (FO,FO[$\rightarrow^*$],MSO,...) are decidable ?

Context

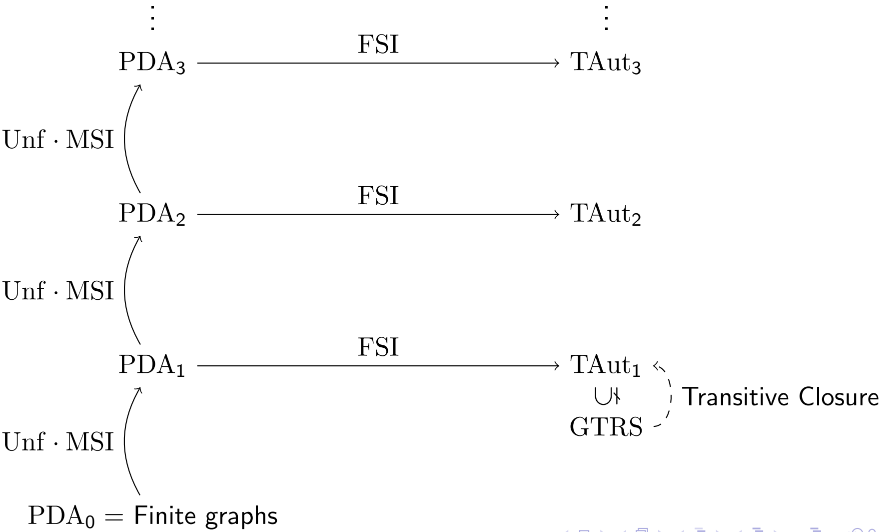
- Model Checking over some classes of infinite graphs: which logic theories (FO,FO[$\to^*$],MSO,...) are decidable ?



HOPDA and TAut hierarchies



HOPDA and TAut hierarchies



HOPDA and TAut hierarchies

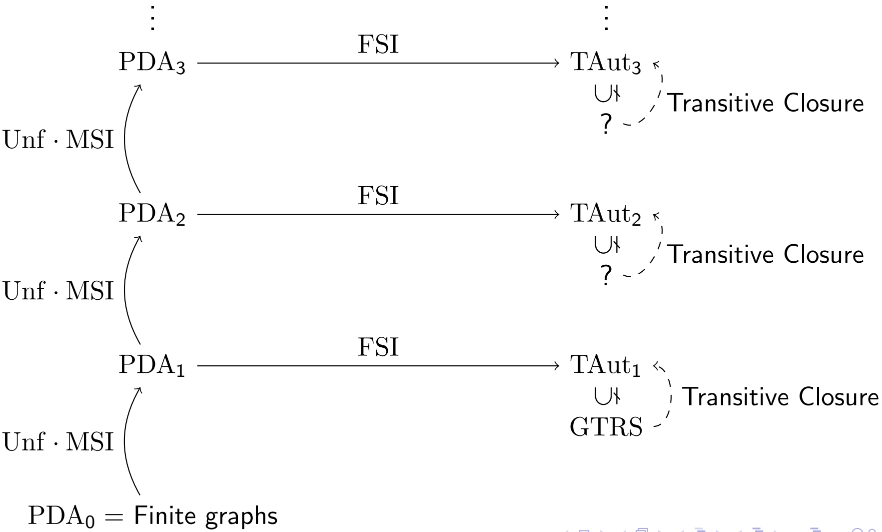
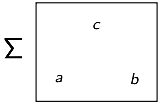
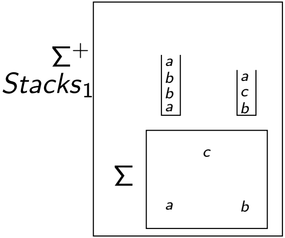


Table of Contents

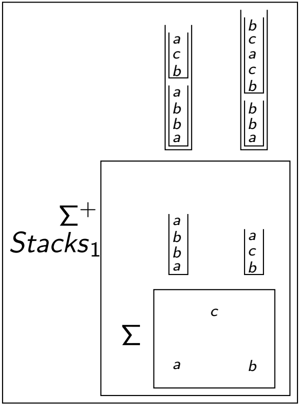
Stacks and Trees



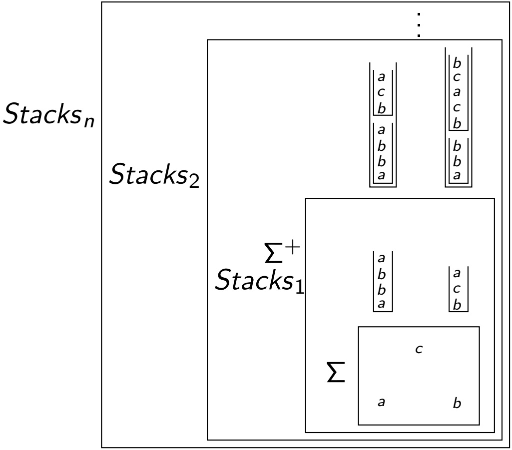

Σ^+
$Stacks_1$
a
b
b
a
a
c
b
Σ
c
a
b

Stacks and Trees

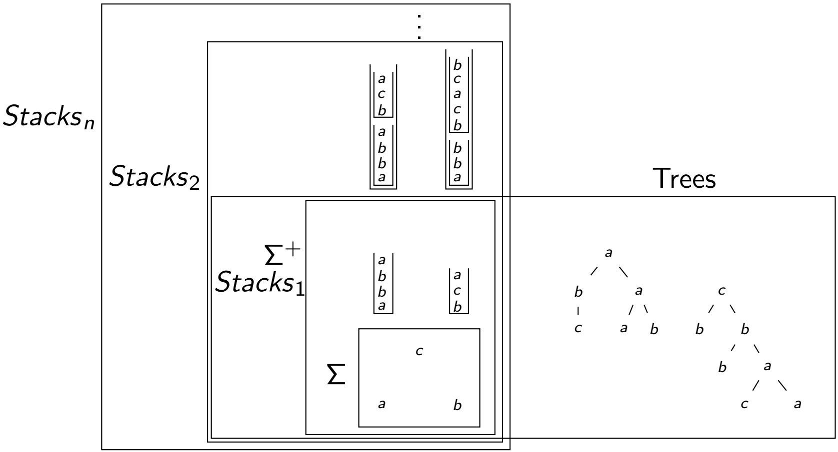
$Stacks_2$



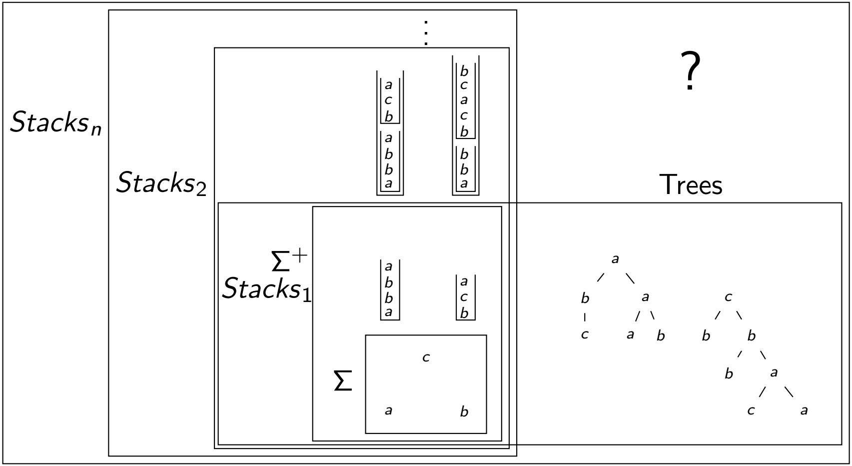
Stacks and Trees



Stacks and Trees

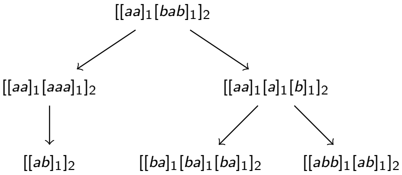


Stacks and Trees



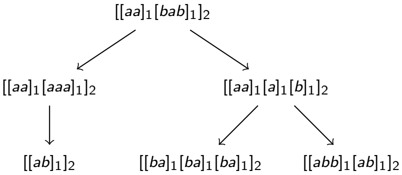
Higher-Order Stack Trees

A n-stack tree is a tree labelled by $(n-1)$-stacks.



Higher-Order Stack Trees

A n-stack tree is a tree labelled by $(n-1)$-stacks.



A unary n-stack tree is a n-stack.
A 1-stack tree is a tree labelled by Σ.

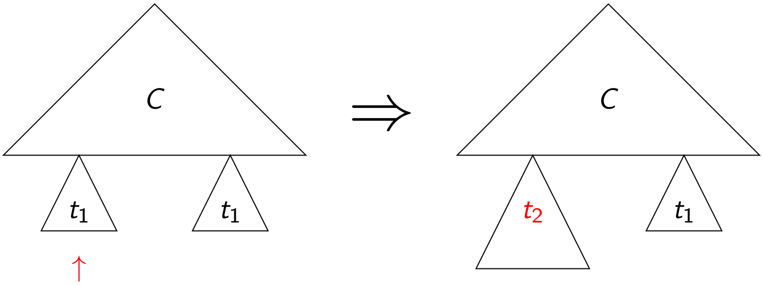
Higher-Order Stacks Operations

- $Stacks_0 = \Sigma$
- $Ops_0 = \{\mathrm{rew_{b,a}} \mid a, b \in \Sigma\}$
  - $\mathrm{rew_{b,a}}(b) = a$
  - $\mathrm{rew_{b,a}}(c)$ is not defined

- $Stacks_n = (Stacks_{n-1})^+$
- $Ops_n = \{\mathrm{copy}_n, \overline{\mathrm{copy}}_n\} \cup Ops_{n-1}$
  - $\mathrm{copy}_n([s_1, \cdots, s_k]_n) = [s_1, \cdots, s_k, s_k]_n$
  - $\overline{\mathrm{copy}}_n([s_1, \cdots, s_k, s_{k+1}]_n) = [s_1, \cdots, s_k]_n$ if $s_k = s_{k+1}$ and not defined otherwise.
  - $\theta([s_1, \cdots, s_k]_n = [s_1, \cdots, s_{k-1}, \theta(s_k)]_n$, for $\theta \in Ops_{n-1}$

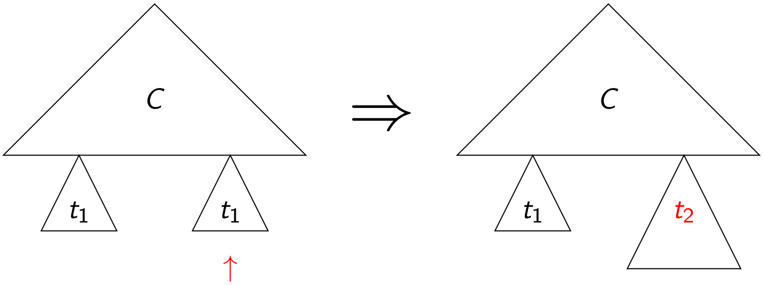
Ground Tree Operation

A ground tree operation is a pair of trees (t_1, t_2)



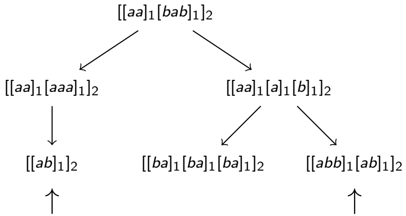
Ground Tree Operation

A ground tree operation is a pair of trees (t_1, t_2)



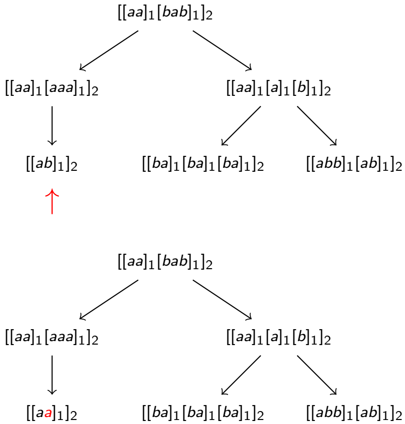
Basic Operations: Level 0

Rewriting operations over Σ: $\mathrm{rew}_{\mathrm{b,a}}$



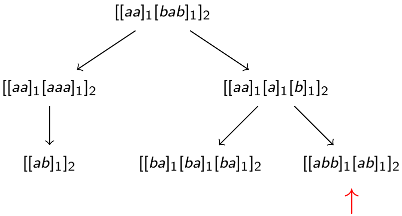
Basic Operations: Level 0

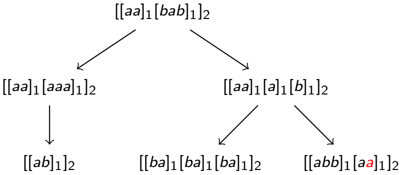
Rewriting operations over Σ: $\mathrm{rew}_{\mathrm{b,a}}$



Basic Operations: Level 0

Rewriting operations over Σ: $\mathrm{rew}_{\mathrm{b,a}}$



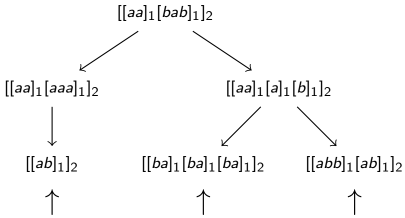


Basic Operations: Level $i < n$

Copy operations: $\mathrm{copy}_i, \overline{\mathrm{copy}}_i$

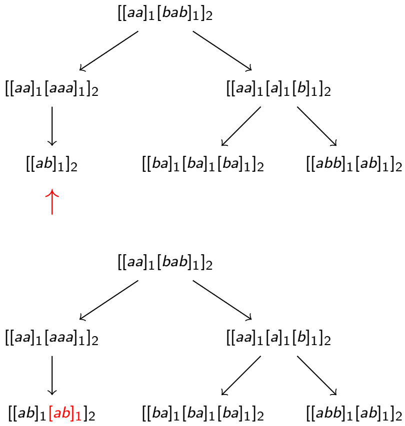
Basic Operations: Level $i < n$

Copy operations: copy_2



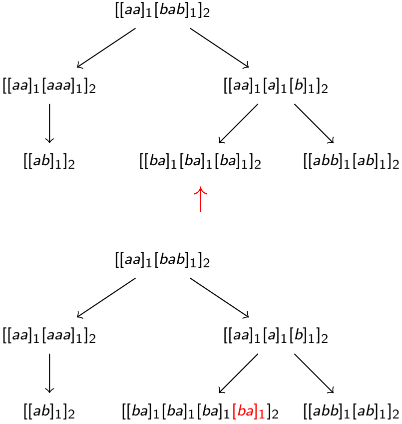
Basic Operations: Level $i < n$

Copy operations: copy_2



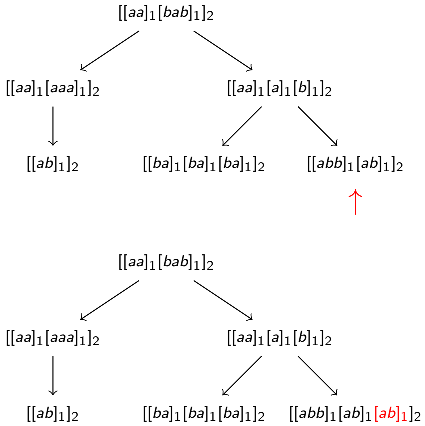
Basic Operations: Level $i < n$

Copy operations: copy_2



Basic Operations: Level $i < n$

Copy operations: copy_2

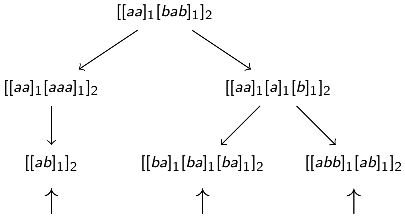


Basic Operations: Level n

Tree copy operations: $\mathrm{copy}_n^i, \overline{\mathrm{copy}}_n^i$

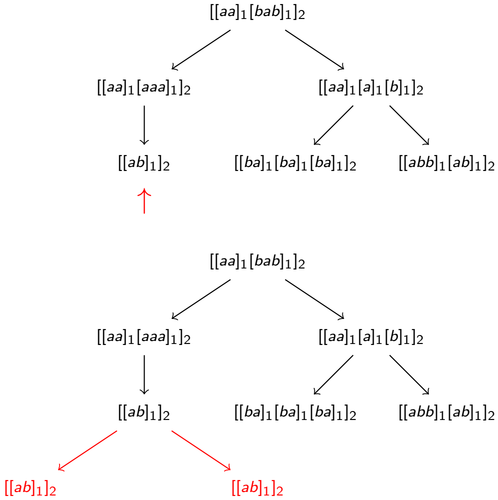
Basic Operations: Level n

Tree copy operations: copy^2_3



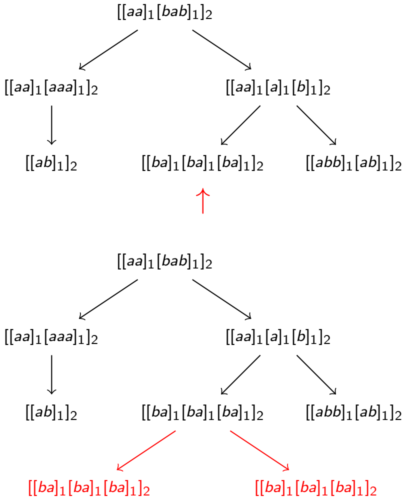
Basic Operations: Level n

Tree copy operations: copy_3^2



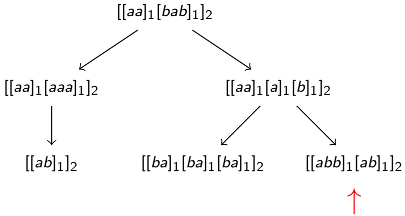
Basic Operations: Level n

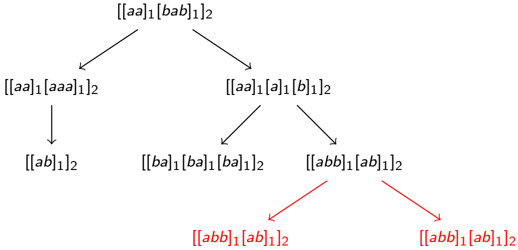
Tree copy operations: copy_3^2



Basic Operations: Level n

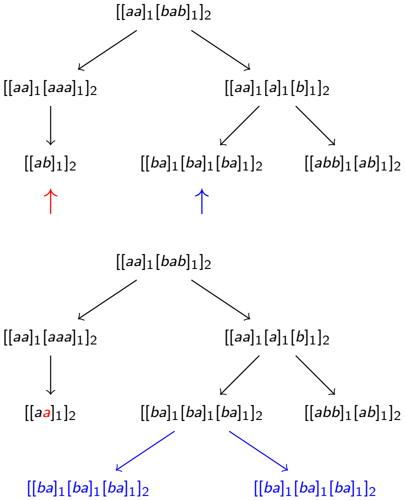
Tree copy operations: copy_3^2





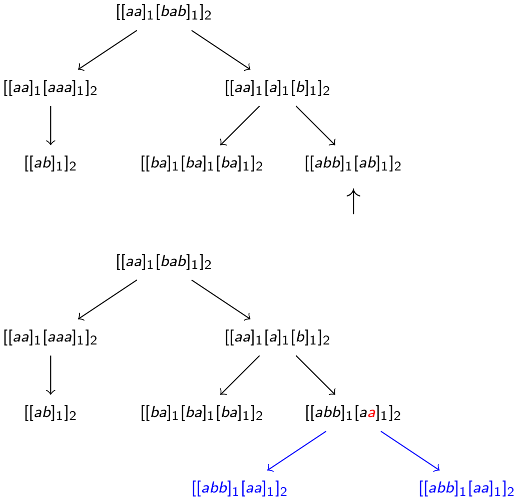
Composition

$\mathrm{rew}_{\mathrm{b,a}} \circ \mathrm{copy}^2_3$:

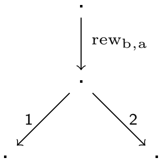


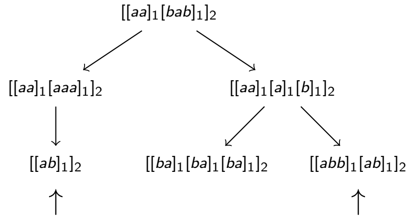
Composition

$\mathrm{rew}_{b,a} \circ \mathrm{copy}^2_3$:

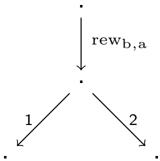


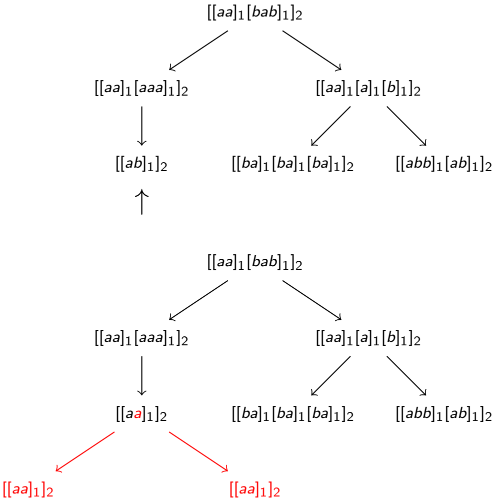
Compound Operations



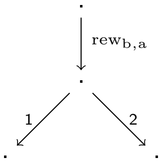


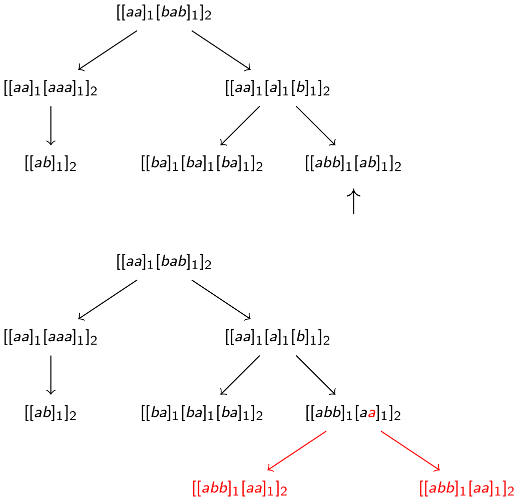
Compound Operations



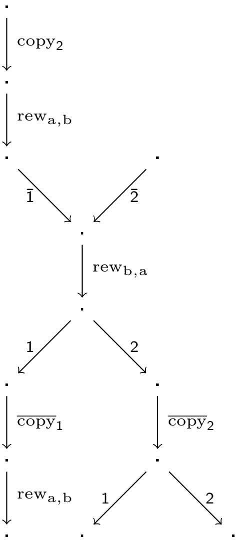


Compound Operations

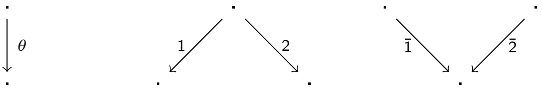




Compound Operations

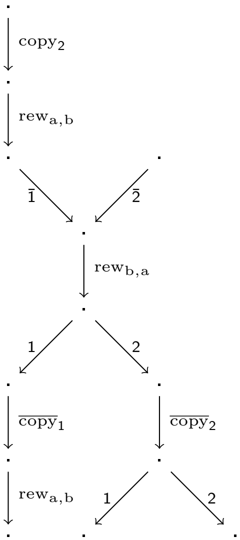


- Characterised by DAGs obtained by concatenations of DAGs whose edges represent basic operations:

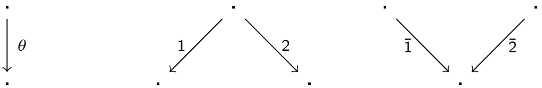


- Only connected operations

Compound Operations



- Characterised by DAGs obtained by concatenations of DAGs whose edges represent basic operations:



- Only connected operations
- Level 1 $\rightarrow$ ground tree rewriting rules
- Unary trees $\rightarrow$ finite composition of higher-order pushdown operations

Compound Operations

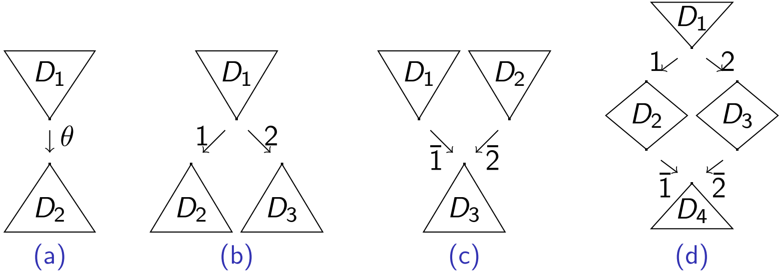


Table of Contents

Main Result

Given a set of compound operations R, its rewriting graph $\mathcal{G}_R$ is:

- $V_{\mathcal{G}_R} = ST_n$
- $E_{\mathcal{G}_R} = \{(t, r, t') \mid r \in R \wedge r(t, t')\}$

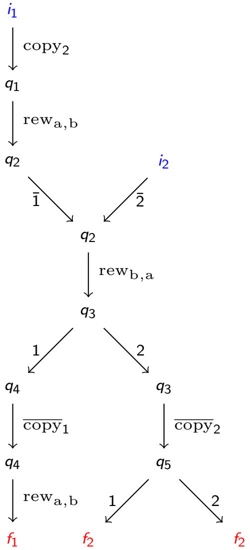
Theorem

Given a finite set of compound operations R, its rewriting graph has a decidable $FO[\xrightarrow{*}]$ theory.

Proof ingredients:

- Notion of recognisability over compound operations
- Finite set interpretation of every stack-tree rewriting graph into a graph with a decidable MSO-theory (the level n treegraph)

Operation automata



- Automaton: $\mathcal{A} = (Q, I, F, \Delta)$.
- Rules of form: $(q_1, \theta, q_2), (q_1, (q_2, q_3)), ((q_1, q_2), q_3)$.
- A run is a labelling of a DAG. It is accepting if it is consistent with Δ, input nodes are labelled by initial states and output nodes by final states.
- *Rec* the set languages of operation automata is closed by union, intersection and iteration.

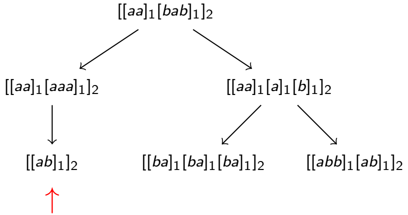
The Finite Set Interpretation

Given a GSTRS R, we interpret $(ST_n, \{\xrightarrow{\alpha} \mid \alpha \in R\}, \xrightarrow[R]{*})$ into the level n treegraph $(Stacks_n(\Sigma \cup \{1,2\}), \{\xrightarrow{\theta} \mid \theta \in Ops_n\})$ with these formulæ:

- $\delta(X)$ which is true if X codes a n-stack tree
- $\psi_\alpha(X_t, X_{t'})$ which is true if $t' \in \alpha(t)$
- $\psi_A(X_t, X_{t'})$ which is true if $\exists \alpha \in L(A), t' \in \alpha(t)$

Coding of Stack Trees

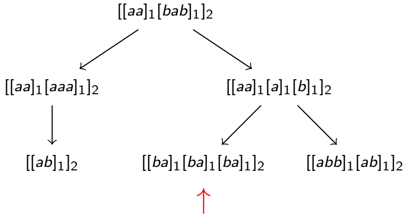
We interpret a *n*-stack tree as the set of *n*-stacks representing the paths from the root to each leaf, in which we added the direction of the path and the number of sons of each node on the path.



$$[[[aa]_1[bab21]_1]_2[[aa]_1[aaa11]_1]_2[[ab]_1]_2]_3$$

Coding of Stack Trees

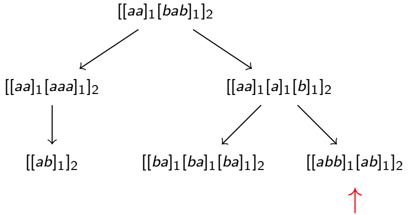
We interpret a *n*-stack tree as the set of *n*-stacks representing the paths from the root to each leaf, in which we added the direction of the path and the number of sons of each node on the path.



$$[[[aa]_1[bab22]_1]_2[[aa]_1[a]_1[b21]_1]_2[[ba]_1[ba]_1[ba]_1]_2]_3$$

Coding of Stack Trees

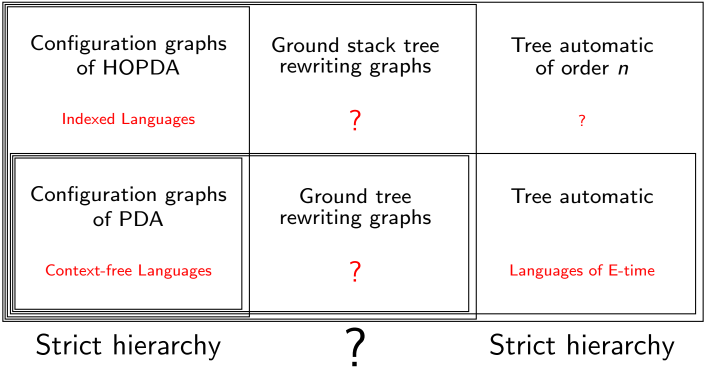
We interpret a *n*-stack tree as the set of *n*-stacks representing the paths from the root to each leaf, in which we added the direction of the path and the number of sons of each node on the path.



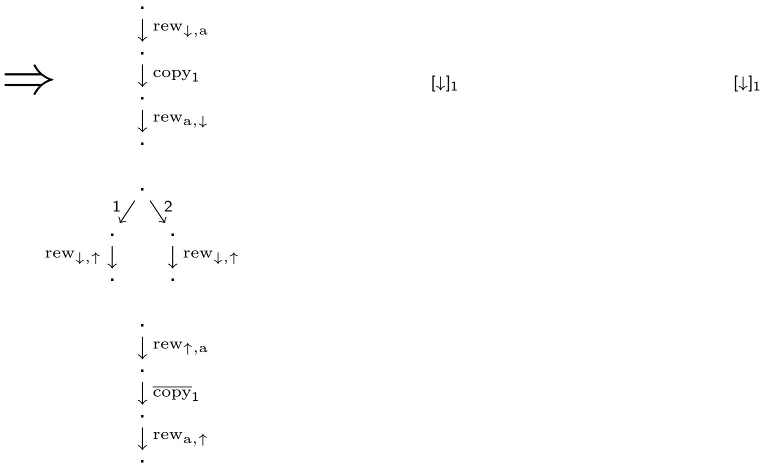
$$[[[aa]_1[bab22]_1]_2[[aa]_1[a]_1[b22]_1]_2[[abb]_1[ab]_1]_2]_3$$

Table of Contents

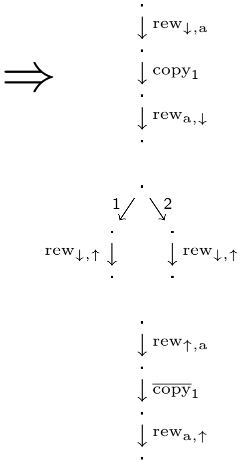
Languages and Strictness



Exemple of a language: $\{u \sqcup u \mid u \in \Sigma^*\}$

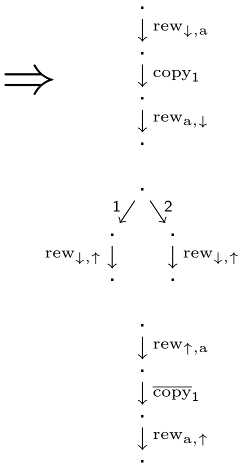


Exemple of a language: $\{u \sqcup u \mid u \in \Sigma^*\}$



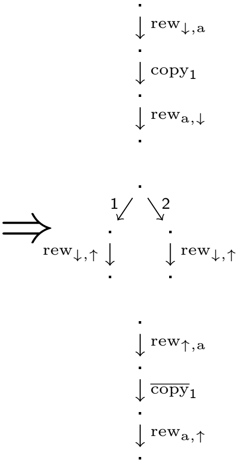
$[a \downarrow]_1$ $[a \downarrow]_1$

Exemple of a language: $\{u \sqcup u \mid u \in \Sigma^*\}$



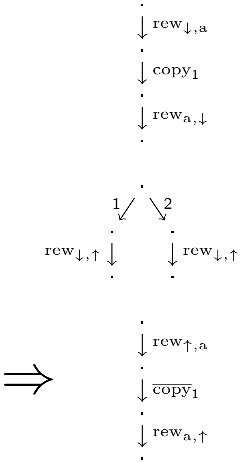
$[ab \downarrow]_1$ $[ab \downarrow]_1$

Exemple of a language: $\{u \sqcup u \mid u \in \Sigma^*\}$

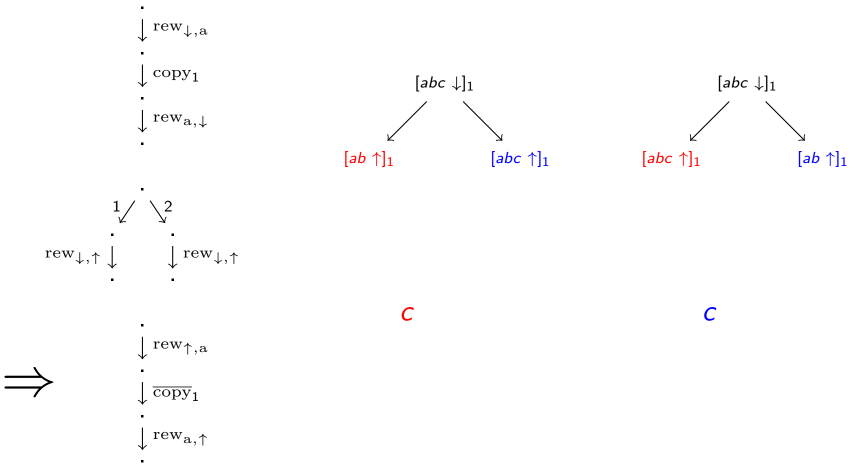


$[abc \downarrow]_1$ $[abc \downarrow]_1$

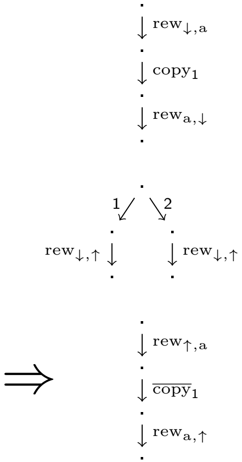
Exemple of a language: $\{u \sqcup u \mid u \in \Sigma^*\}$

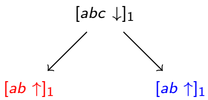


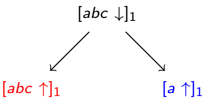
Exemple of a language: $\{u \sqcup u \mid u \in \Sigma^*\}$



Exemple of a language: $\{u \sqcup u \mid u \in \Sigma^*\}$



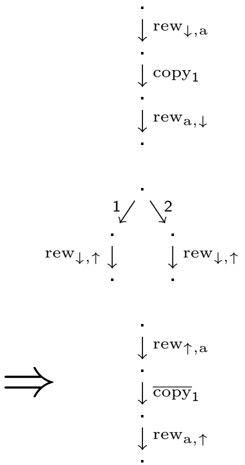


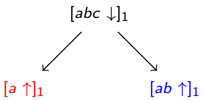


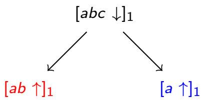
cc

cb

Exemple of a language: $\{u \sqcup u \mid u \in \Sigma^*\}$

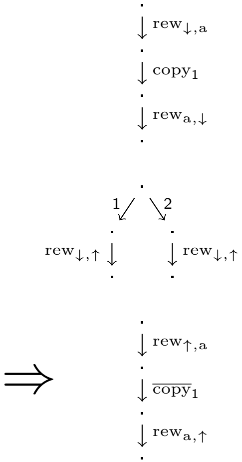


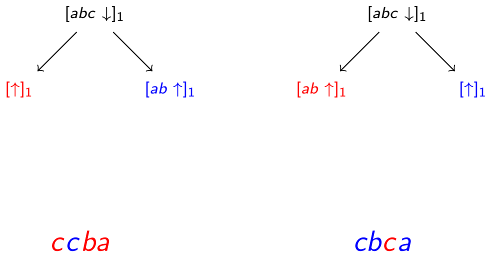




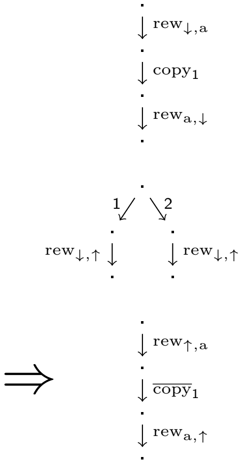
ccb   *cbc*

Exemple of a language: $\{u \sqcup u \mid u \in \Sigma^*\}$



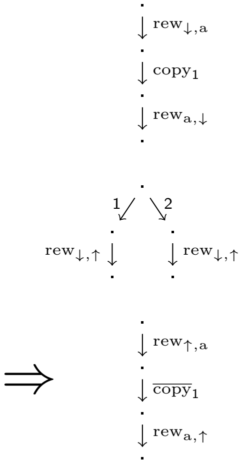


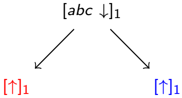
Exemple of a language: $\{u \sqcup u \mid u \in \Sigma^*\}$

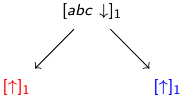


ccbab *cbcab*

Exemple of a language: $\{u \sqcup u \mid u \in \Sigma^*\}$







ccbaba

cbcaba

Higher-Order Trees