Deterministic Transducers over Infinite Terms

Christof Löding
LIAFA (formerly RWTH Aachen)
loeding@liafa.jussieu.fr

joint work with
Thomas Colcombet, Warsaw University
(formerly IRISA, Rennes)
thomas.colcombet@laposte.net

OUTLINE

(1) Basic definitions and terminology

(2) Overview and background

(3) Deterministic top-down tree transducers with rational lookahead

(4) MSO transductions

(5) Main result: comparison of deterministic transducers and MSO transductions

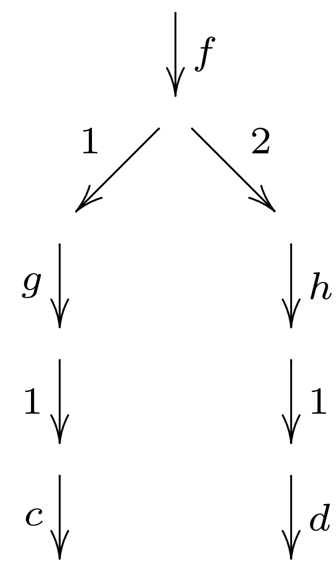
TERMS

- ranked alphabet $\mathcal{F}$ (symbols with arity)
- $|f|$ denotes rank of $f \in \mathcal{F}$
- $|\mathcal{F}|_{\max} = \max\{|f| \mid f \in \mathcal{F}\}$

Terms (possibly infinite) represented as finite edge-labeled trees over the alphabet $\Sigma_{\mathcal{F}} = \mathcal{F} \cup \{1, \ldots, |\mathcal{F}|_{\max}\}$:

Example

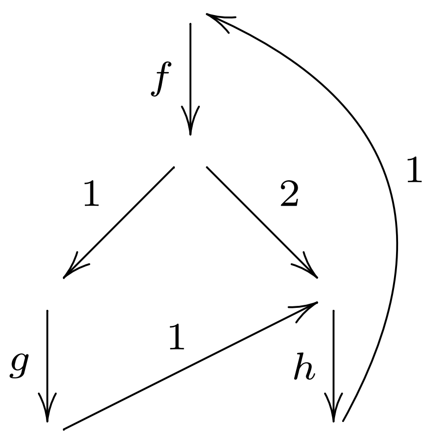
$f(g(c), h(d))$ represented as



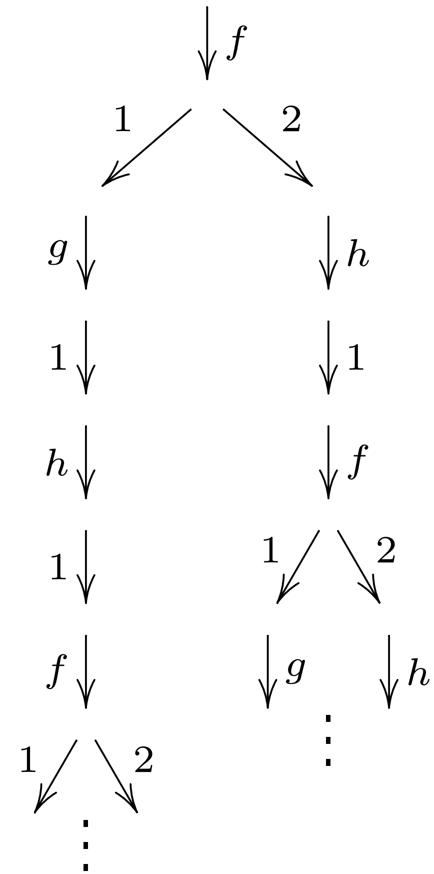
FOLDED TERMS

- rooted graph G (edge labels from $\Sigma_{\mathcal{F}}$)
- unfolding of G from the root denoted by $\mathrm{unfold}(G)$
- G is a folded term if $\mathrm{unfold}(G)$ is a term

Example:



unfolds to



MSO Logic – Rational Sets of Terms

MSO logic over folded terms:

- Signature $(E_a)_{a\in\Sigma_{\mathcal{F}}}$, binary symbols interpreted as the edge relations for each symbol in $\Sigma_{\mathcal{F}}$.
- Quantification over individual vertices.
- Quantification over sets of vertices.

$$\begin{aligned}\phi(x) = \quad & \forall X[x \in X \wedge \forall y, z(y \in X \wedge E(y,z) \rightarrow z \in X) \\ & \rightarrow \exists z', z'' \in X(E_c(z', z''))]\end{aligned}$$

A set of terms is rational

- if it is definable in MSO logic or equivalently
- if it is the set of terms accepted by a Rabin or parity tree automaton or equivalently
- if it is definable in the modal μ-calculus.

Outline

(1) Basic definitions and terminology

terms, folded terms, MSO logic, rational sets of terms

(2) Overview and background

(3) Deterministic top-down tree transducers with rational lookahead

(4) MSO transductions

(5) Main result: comparison of deterministic transducers and MSO transductions

BACKGROUND

- (infinite) terms describe (infinite) objects, e.g., graphs or formal languages

Example – Terms Representing Graphs

Representation of vertex-colored graphs

$\mathcal{F} = \{\oplus, \eta_{i,j}, \rho_{i\to j}, \underline{1}, \ldots, \underline{k}, \perp\}$

$\oplus$	disjoint union
$\eta_{i,j}$	add edges between i-vertices and j-vertices
$\rho_{i\to j}$	make i-vertices to j-vertices
$\underline{i}$	single i-vertex
$\perp$	empty graph

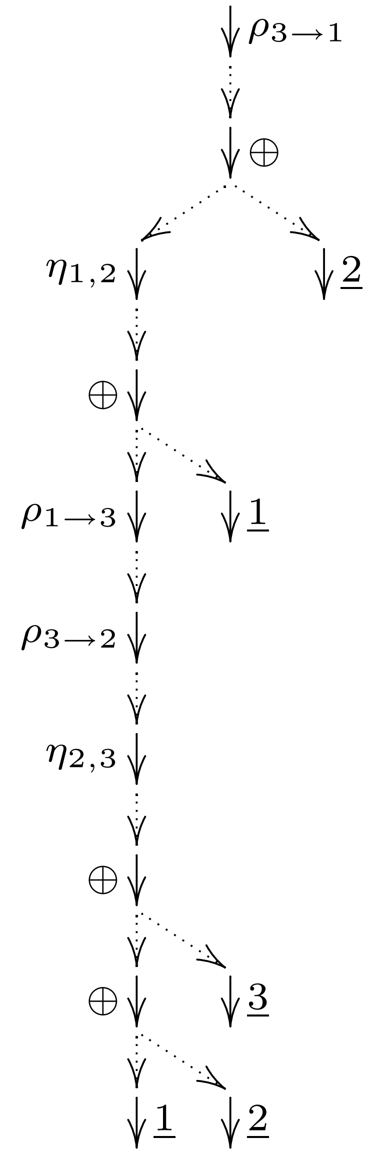
Example – Terms Representing Graphs

Representation of vertex-colored graphs

$\mathcal{F} = \{\oplus, \eta_{i,j}, \rho_{i\to j}, \underline{1}, \ldots, \underline{k}, \perp\}$

$\oplus$	disjoint union
$\eta_{i,j}$	add edges between i-vertices and j-vertices
$\rho_{i\to j}$	make i-vertices to j-vertices
$\underline{i}$	single i-vertex
$\perp$	empty graph

$t:$



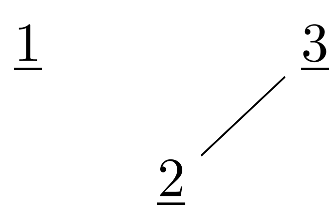
Example – Terms Representing Graphs

Representation of vertex-colored graphs

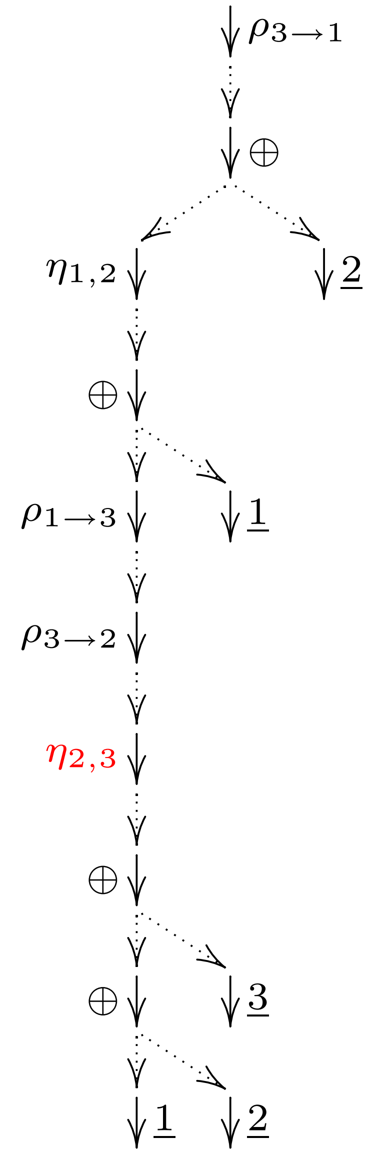
$\mathcal{F} = \{\oplus, \eta_{i,j}, \rho_{i\to j}, \underline{1}, \ldots, \underline{k}, \perp\}$

$\oplus$	disjoint union
$\eta_{i,j}$	add edges between i-vertices and j-vertices
$\rho_{i\to j}$	make i-vertices to j-vertices
$\underline{i}$	single i-vertex
$\perp$	empty graph

$val(t)$:



t :



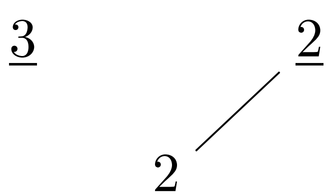
EXAMPLE – TERMS REPRESENTING GRAPHS

Representation of vertex-colored graphs

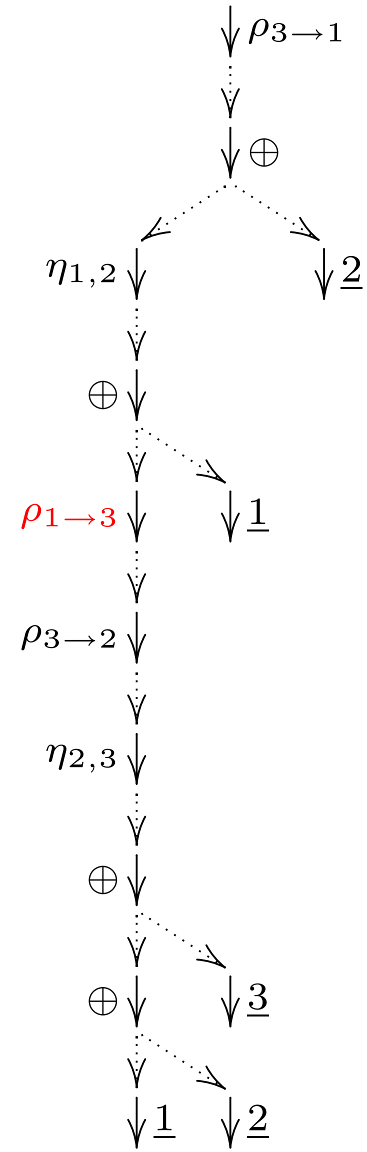
$\mathcal{F} = \{\oplus, \eta_{i,j}, \rho_{i \to j}, \underline{1}, \ldots, \underline{k}, \perp\}$

$\oplus$	disjoint union
$\eta_{i,j}$	add edges between i-vertices and j-vertices
$\rho_{i \to j}$	make i-vertices to j-vertices
$\underline{i}$	single i-vertex
$\perp$	empty graph

$val(t)$:



t :



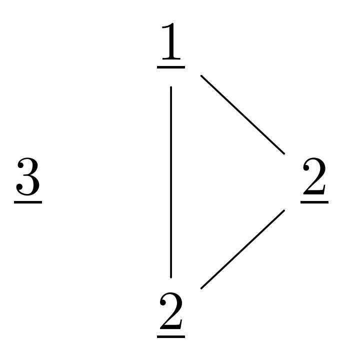
EXAMPLE – TERMS REPRESENTING GRAPHS

Representation of vertex-colored graphs

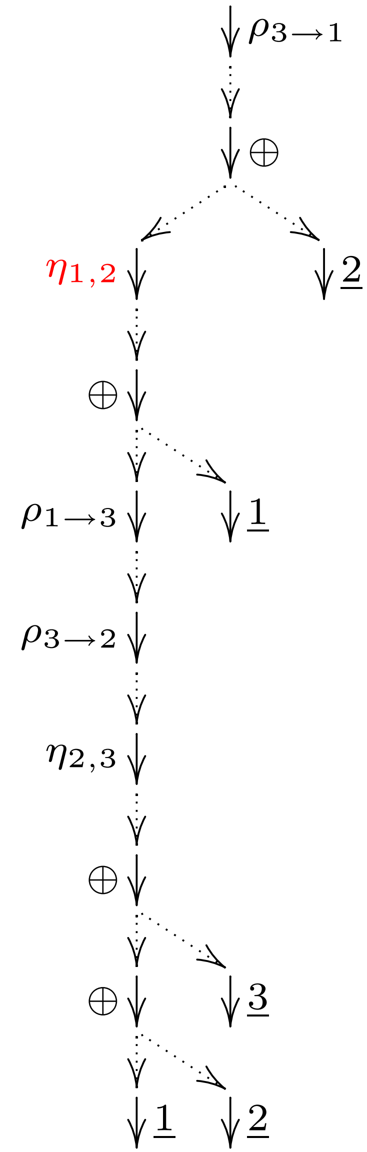
$\mathcal{F} = \{\oplus, \eta_{i,j}, \rho_{i\to j}, \underline{1}, \ldots, \underline{k}, \perp\}$

$\oplus$	disjoint union
$\eta_{i,j}$	add edges between i-vertices and j-vertices
$\rho_{i\to j}$	make i-vertices to j-vertices
$\underline{i}$	single i-vertex
$\perp$	empty graph

$val(t)$:



t :



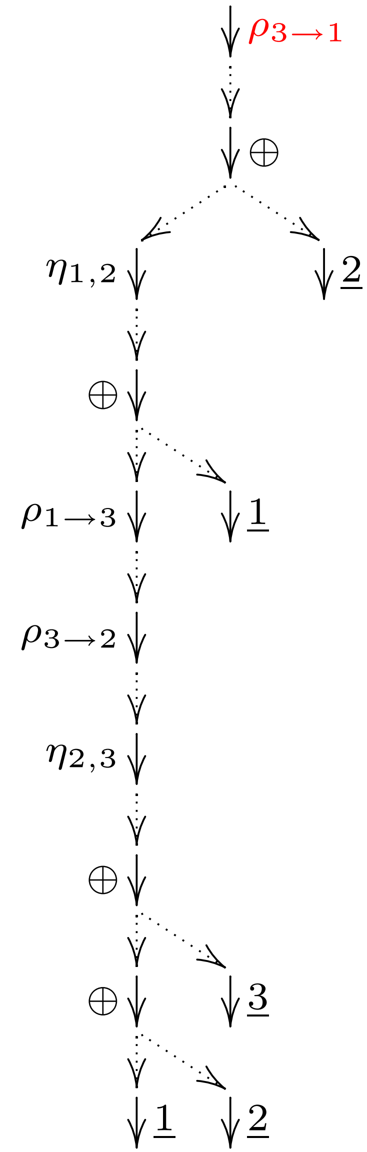
Example – Terms Representing Graphs

Representation of vertex-colored graphs

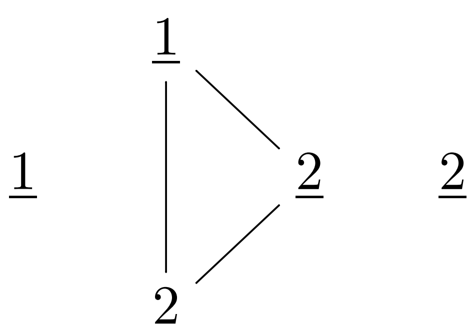
$\mathcal{F} = \{\oplus, \eta_{i,j}, \rho_{i\to j}, \underline{1}, \ldots, \underline{k}, \bot\}$

$\oplus$	disjoint union
$\eta_{i,j}$	add edges between i-vertices and j-vertices
$\rho_{i\to j}$	make i-vertices to j-vertices
$\underline{i}$	single i-vertex
$\bot$	empty graph

t :



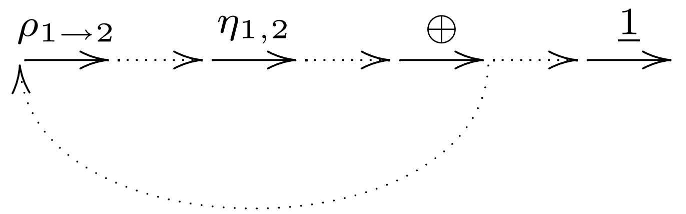
$val(t)$:



Background

- (infinite) terms describe (infinite) objects, e.g., graphs or formal languages
- another way of describing objects is via equational systems
- equational systems can be represented by folded terms

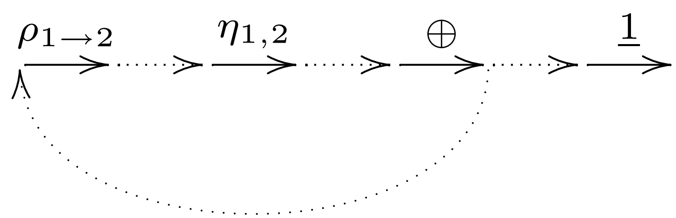
$$G = \rho_{1\to 2}(\eta_{1,2}(\underline{1} \oplus G))$$

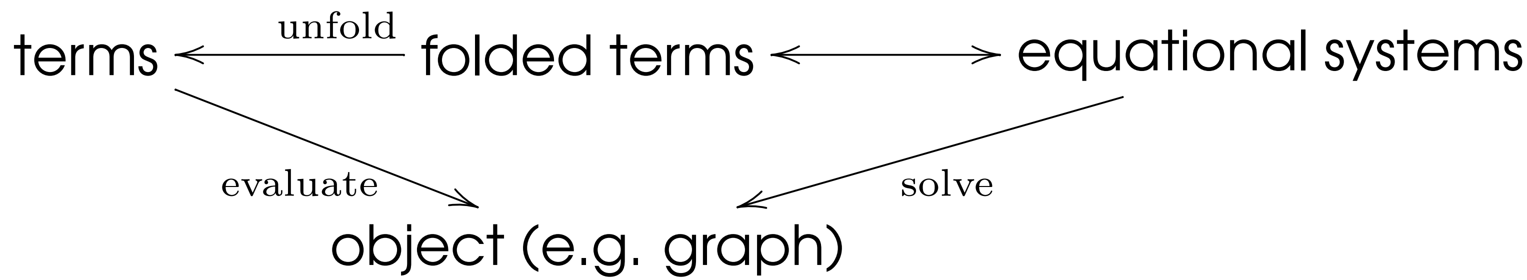


BACKGROUND

- (infinite) terms describe (infinite) objects, e.g., graphs or formal languages
- another way of describing objects is via equational systems
- equational systems can be represented by folded terms

$$G = \rho_{1\to 2}(\eta_{1,2}(\underline{1} \oplus G))$$

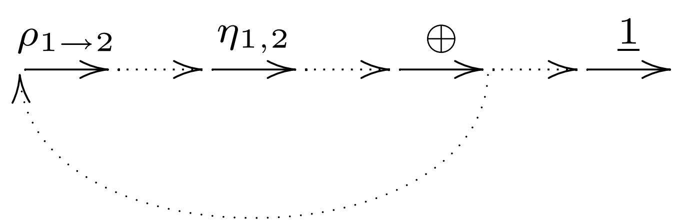


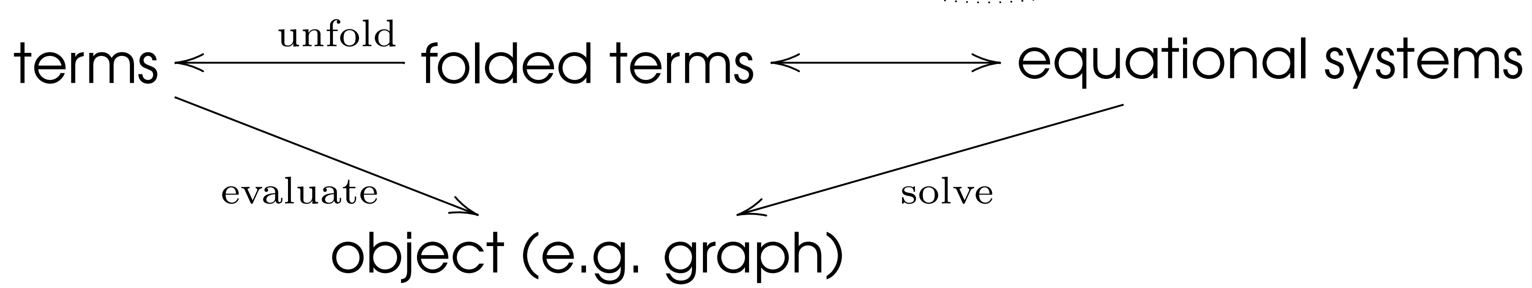


Background

- (infinite) terms describe (infinite) objects, e.g., graphs or formal languages
- another way of describing objects is via equational systems
- equational systems can be represented by folded terms

$$G = \rho_{1\to 2}(\eta_{1,2}(\underline{1} \oplus G))$$





- develop tools to deal with equational systems

OVERVIEW

Objective: apply transformations to the represented objects

Approach: transform the representation

for more details see thesis of Thomas Colcombet

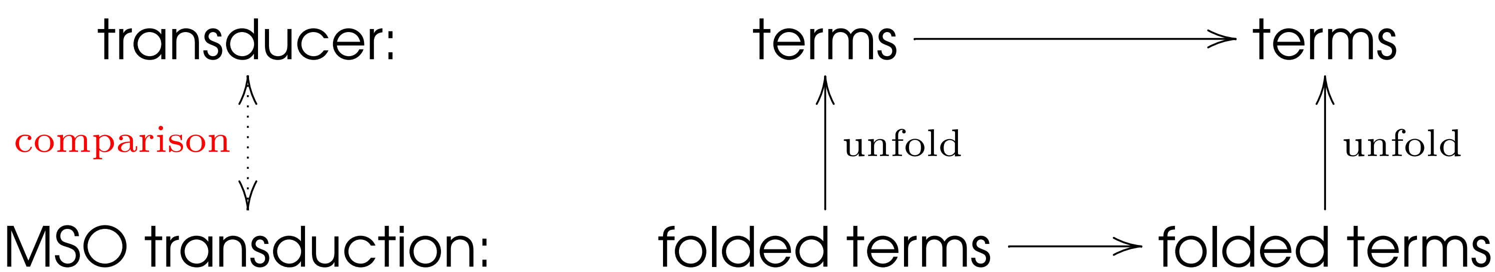
OVERVIEW

Objective: apply transformations to the represented objects

Approach: transform the representation

for more details see thesis of Thomas Colcombet

In this talk:



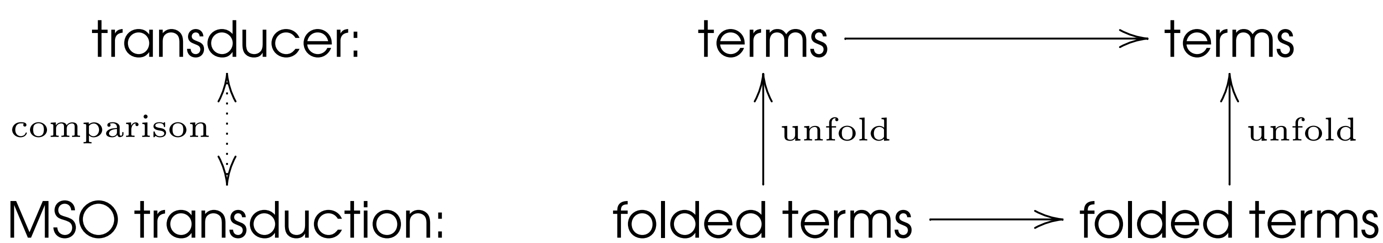
OUTLINE

(1) Basic definitions and terminology

terms, folded terms, MSO logic, rational sets of terms

(2) Overview and background

transformation of objects by transformation of representation



(3) Deterministic top-down tree transducers with rational lookahead

(4) MSO transductions

(5) Main result: comparison of deterministic transducers and MSO transductions

$T = (Q, \mathcal{F}, \mathcal{F}', q_0, \Delta)$ with:

- $\mathcal{F}$, $\mathcal{F}'$ ranked alphabets (input and output alphabet)
- Q a finite set of states
- $q_0 \in Q$ the initial state
- Δ a finite set of rules of one of the following forms:

(production rule): $q(x) \to g(q_1(x), ..., q_{|g|}(x))$
$g \in \mathcal{F}'$, x a variable, and $q_1, \ldots, q_{|g|} \in Q$

(consumption rule): $q(f(x_1, ..., x_{|f|})) \to q'(x_i)$
$f \in \mathcal{F}$, $q, q' \in Q$, and $x_1, \ldots, x_{|f|}$ variables

(lookahead rule): $q(x \in L) \to q'(x)$
L a rational set of $\mathcal{F}$-terms (called lookahead set), $q, q' \in Q$, and x a variable

TOP-DOWN TREE TRANSDUCERS WITH RATIONAL LOOKAHEAD

$T = (Q, \mathcal{F}, \mathcal{F}', q_0, \Delta)$ with:

- $\mathcal{F}$, $\mathcal{F}'$ ranked alphabets (input and output alphabet)
- Q a finite set of states
- $q_0 \in Q$ the initial state
- Δ a finite set of rules of one of the following forms:

  (production rule): $q(x) \to g(q_1(x), ..., q_{|g|}(x))$
  $g \in \mathcal{F}'$, x a variable, and $q_1, \ldots, q_{|g|} \in Q$

  (consumption rule): $q(f(x_1, ..., x_{|f|})) \to q'(x_i)$
  $f \in \mathcal{F}$, $q, q' \in Q$, and $x_1, \ldots, x_{|f|}$ variables

  (lookahead rule): $q(x \in L) \to q'(x)$
  L a rational set of $\mathcal{F}$-terms (called lookahead set), $q, q' \in Q$, and x a variable

Semantics: Start with $q_0(t)$ and ‘apply rewriting rules to infinity’

Determinism: for any q, t no two rules apply to $q(t)$

Example

$\mathcal{F} = \mathcal{F}' = \{\oplus, \eta_{i,j}, \rho_{i\to j}, \underline{1}, \ldots, \underline{k}, \bot\}$

Goal: Remove isolated vertices from $val(t)$

For a set of colors C let f_C be the mapping that removes all vertices from G that are isolated and not of color C. We are interested in $f_\emptyset$.

EXAMPLE

$\mathcal{F} = \mathcal{F}' = \{\oplus, \eta_{i,j}, \rho_{i\to j}, \underline{1}, \ldots, \underline{k}, \perp\}$

Goal: Remove isolated vertices from $val(t)$

For a set of colors C let f_C be the mapping that removes all vertices from G that are isolated and not of color C. We are interested in $f_\emptyset$.

Invariants:

$f_C(\perp) = \perp$. $f_C(\underline{i}) = \underline{i}$ if $i \in C$ and $f_C(\underline{i}) = \perp$, otherwise.

$f_C(G \oplus G') = f_C(G) \oplus f_C(G')$

$f_C(\eta_{i,j}(G)) = f_{C'}(G)$ with $C' = \begin{cases} C \cup \{i,j\} \text{ if } G \text{ contains } i\text{- and } j\text{-vertices} \\ C \text{ otherwise} \end{cases}$

$f_C(\rho_{i\to j}(G)) = f_{C'}(G)$ with $C' = \begin{cases} C \cup \{i\} \text{ if } j \in C \\ C \setminus \{i\} \text{ if } j \notin C \end{cases}$

Implementation: Transducer keeps track of the set C using the invariants.

EXAMPLE

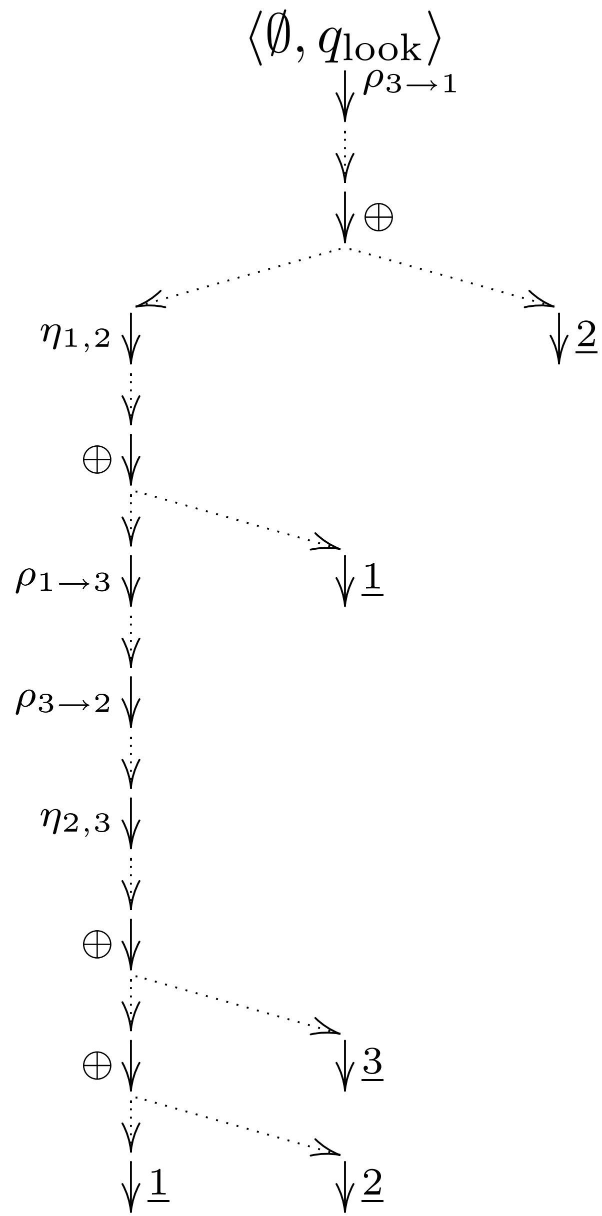
Lookahead sets:

$$
\begin{array}{lllllll}
L_{\underline{i}} & = & \{\underline{i}\} & L_{\perp} & = & \{\perp\} \\
L_{\oplus} & = & \{t \mid t = \oplus(t_1, t_2)\} & L_{\rho_{i\to j}} & = & \{t \mid t = \rho_{i\to j}(t_1)\} \\
L_{\eta_{i,j}} & = & \multicolumn{4}{l}{\{t \mid t = \eta_{i,j}(t_1) \text{ and } val(t_1) \text{ contains } i\text{- and } j\text{-vertices}\}} \\
\overline{L_{\eta_{i,j}}} & = & \multicolumn{4}{l}{\{t \mid t = \eta_{i,j}(t_1) \text{ and } val(t_1) \text{ does not contain } i\text{- and } j\text{-vertices}\}}
\end{array}
$$

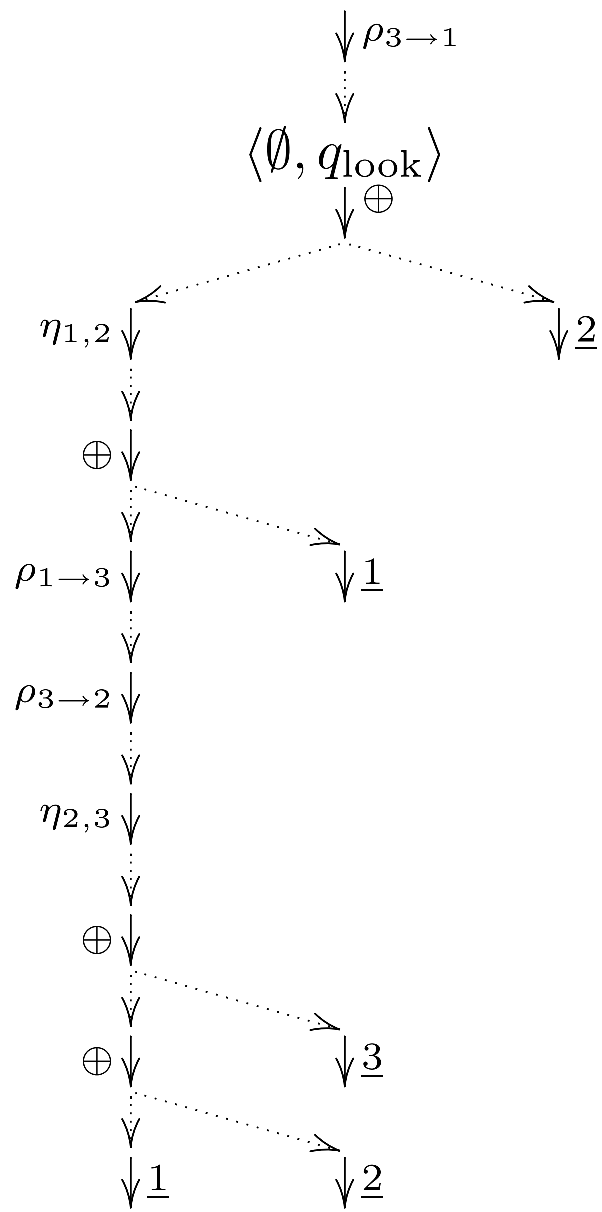
Some of the rewriting rules:

- $\langle C, q_{\text{look}}\rangle\,(x \in L_{\underline{i}}) \to \langle C, q_{\underline{i}}\rangle\,(x)$, $\quad \langle C, q_{\underline{i}}\rangle\,(x \in L_{\underline{i}}) \to \begin{cases} \underline{i} & \text{if } i \in C \\ \perp & \text{otherwise} \end{cases}$
- $\langle C, q_{\text{look}}\rangle\,(x \in \overline{L_{\eta_{i,j}}}) \to \langle C, q_{\text{cons}}\rangle\,(x)$
- $\langle C, q_{\text{look}}\rangle\,(x \in L_{\eta_{i,j}}) \to \langle C \cup \{i, j\}, q_{\eta_{i,j}}\rangle\,(x)$
- $\langle C, q_{\text{look}}\rangle\,(x \in L_{\oplus}) \to \langle C, q_{\oplus}\rangle\,(x)$, $\langle C, q_{\oplus}\rangle\,(x) \to \oplus(\langle C, q_{\oplus,1}\rangle\,(x), \langle C, q_{\oplus,2}\rangle\,(x))$
- $\langle C, q_{\text{look}}\rangle\,(x \in L_{\rho_{i\to j}}) \to \langle C' \cup \{i\}, q_{i\to j}\rangle\,(x)$ with $C' = \begin{cases} C \cup \{i\} \text{ if } j \in C \\ C \setminus \{i\} \text{ if } j \notin C \end{cases}$

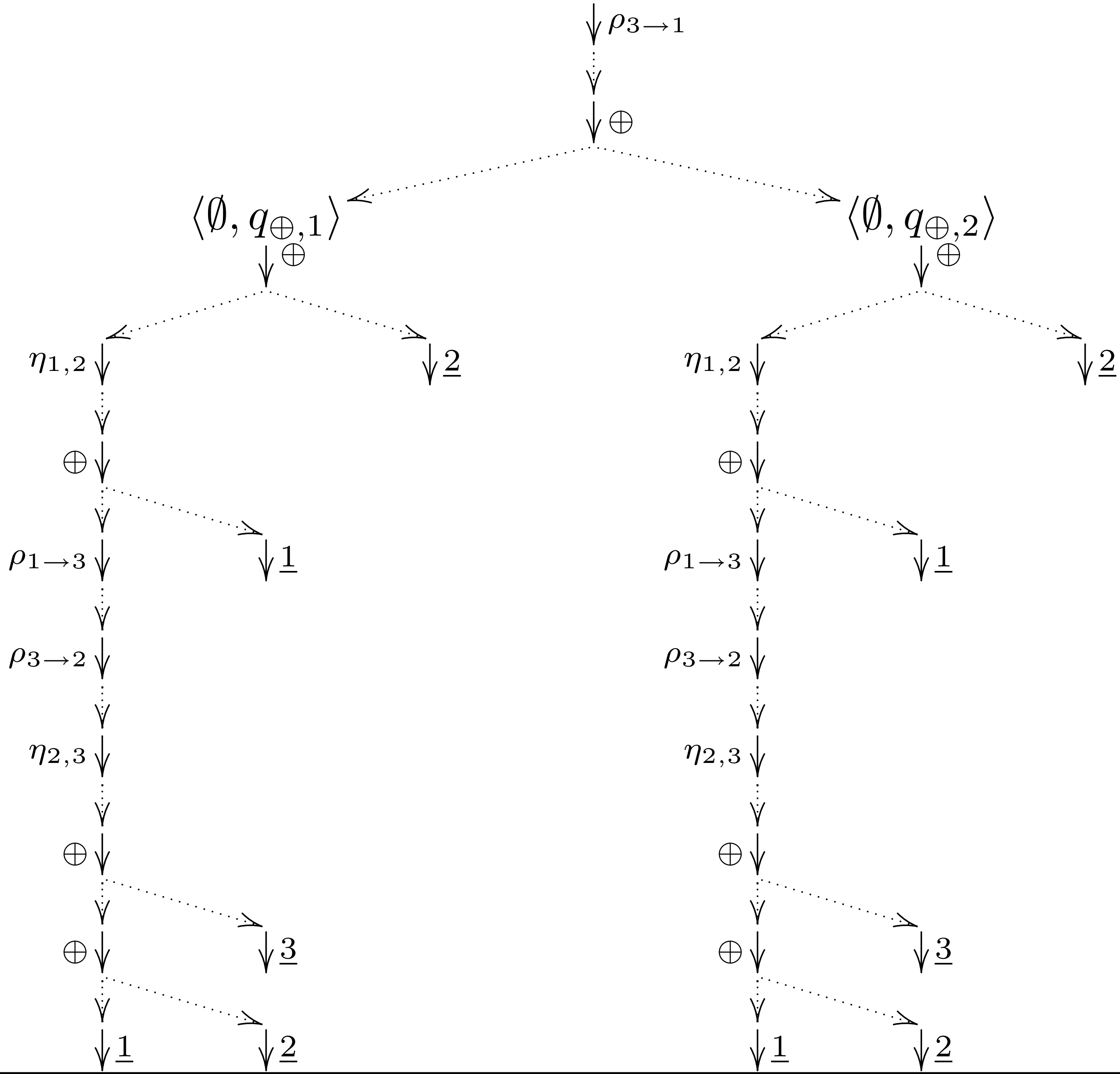
Sample Application



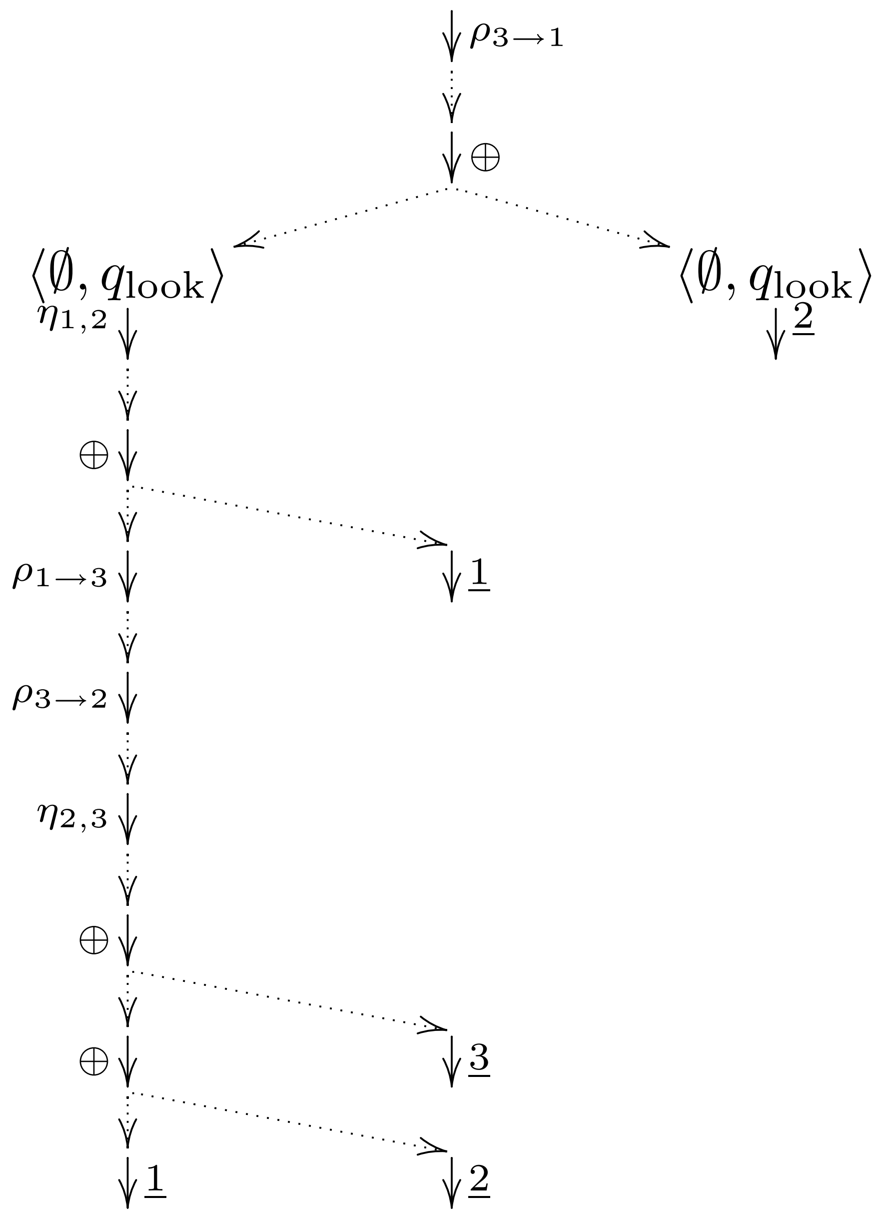
Sample Application



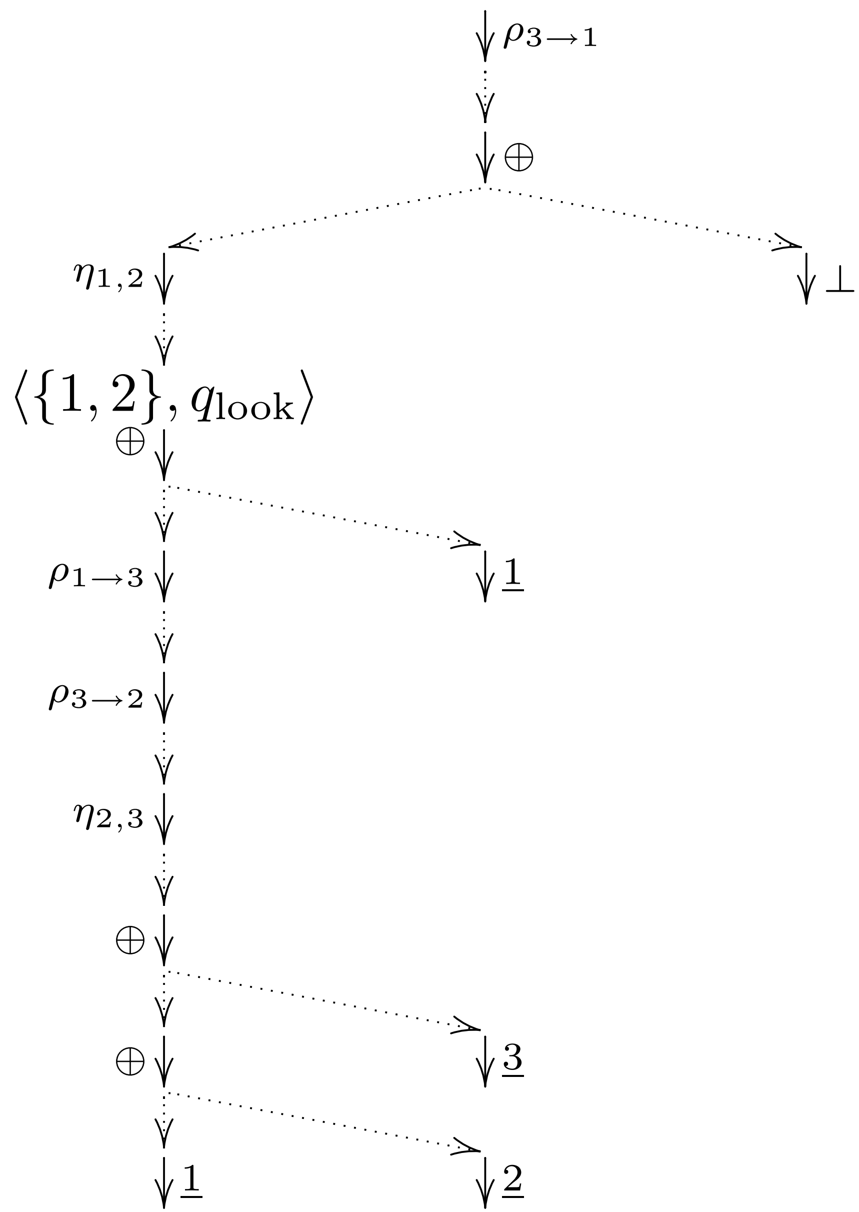
Sample Application



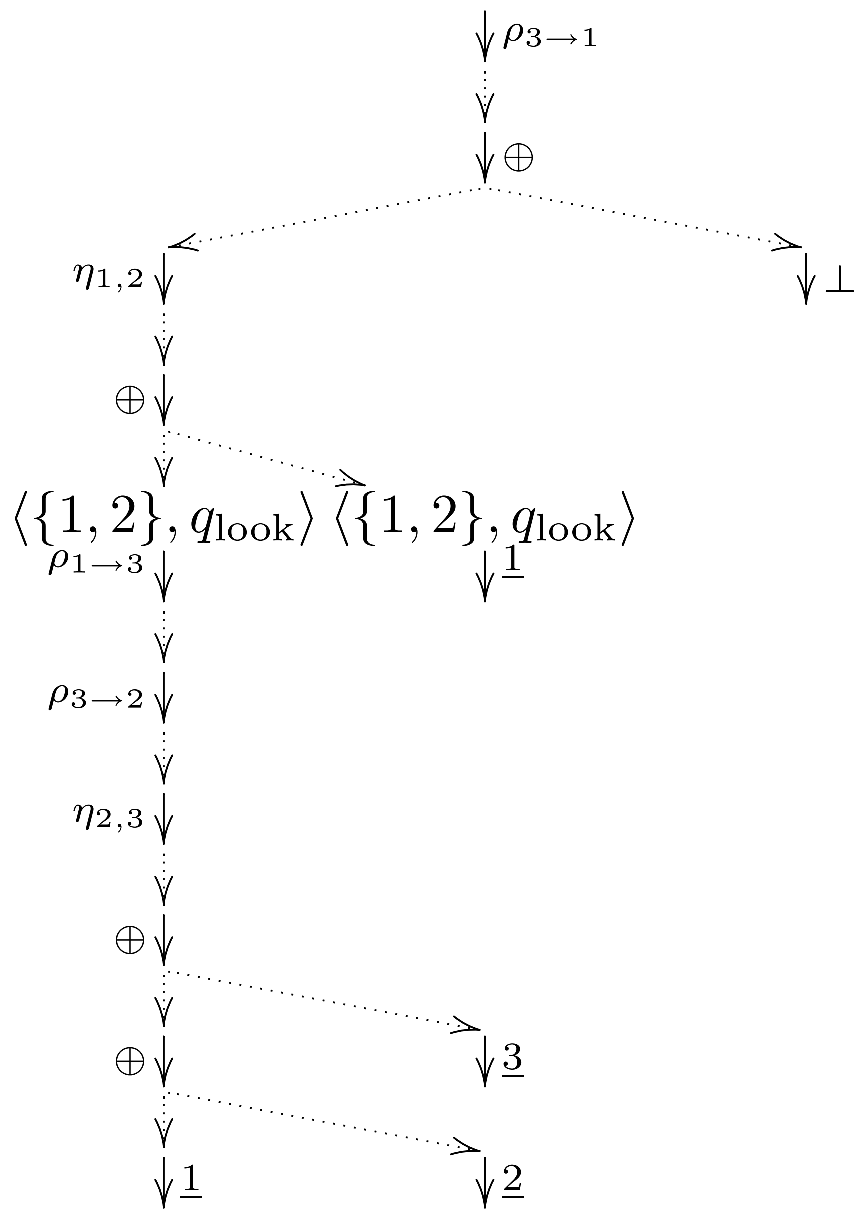
SAMPLE APPLICATION



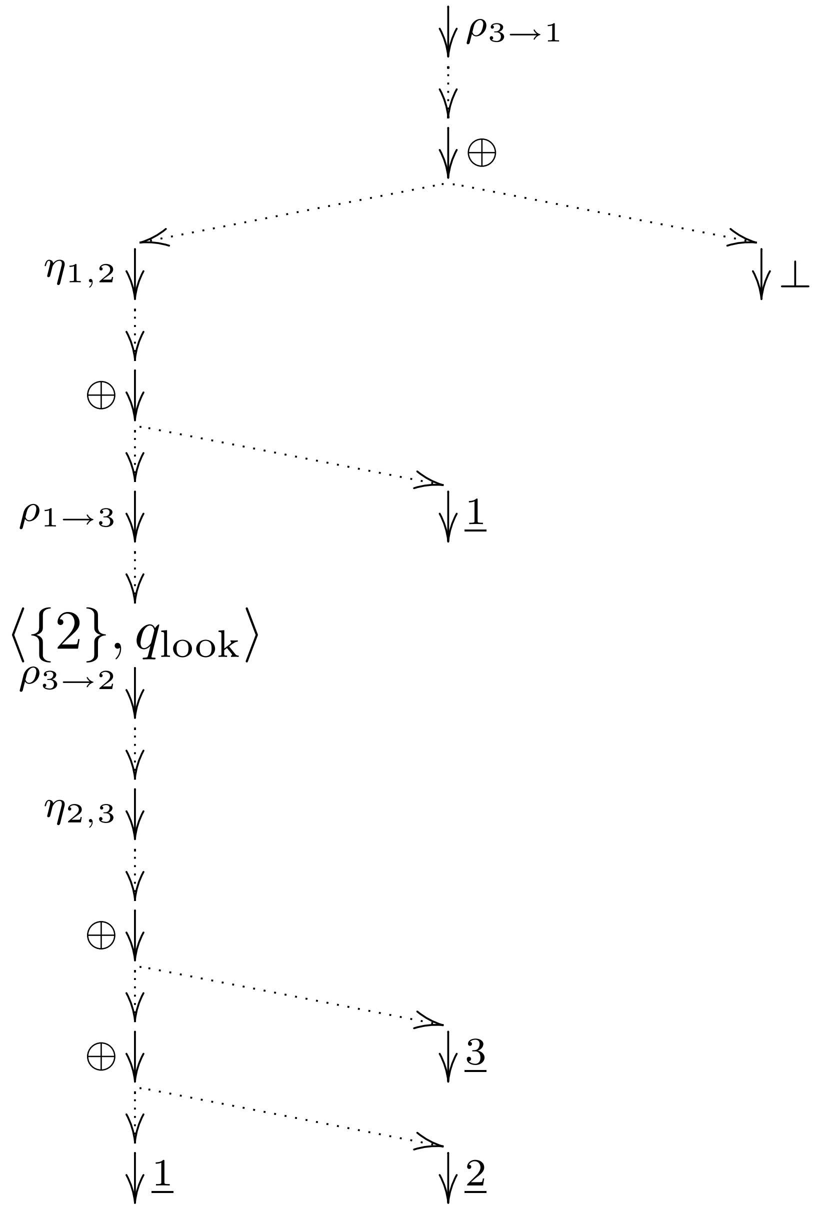
Sample Application



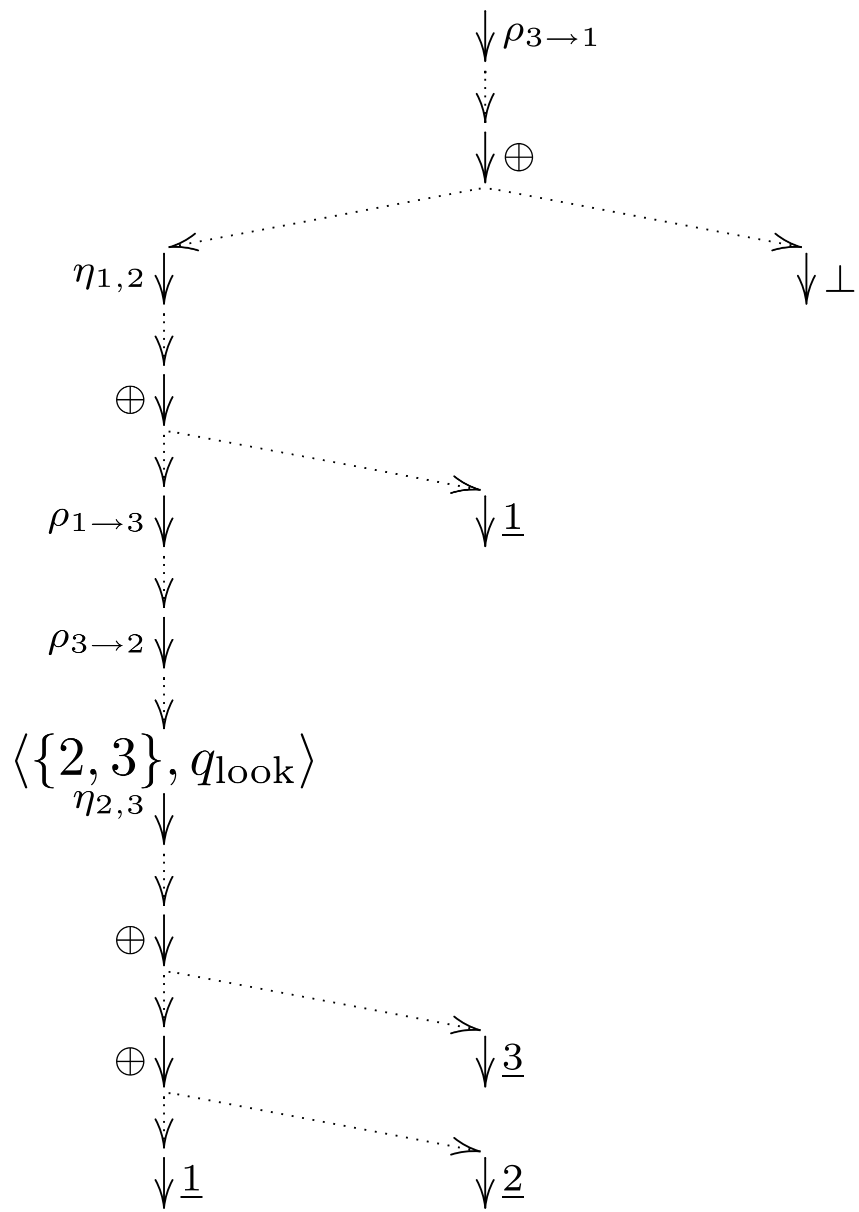
SAMPLE APPLICATION



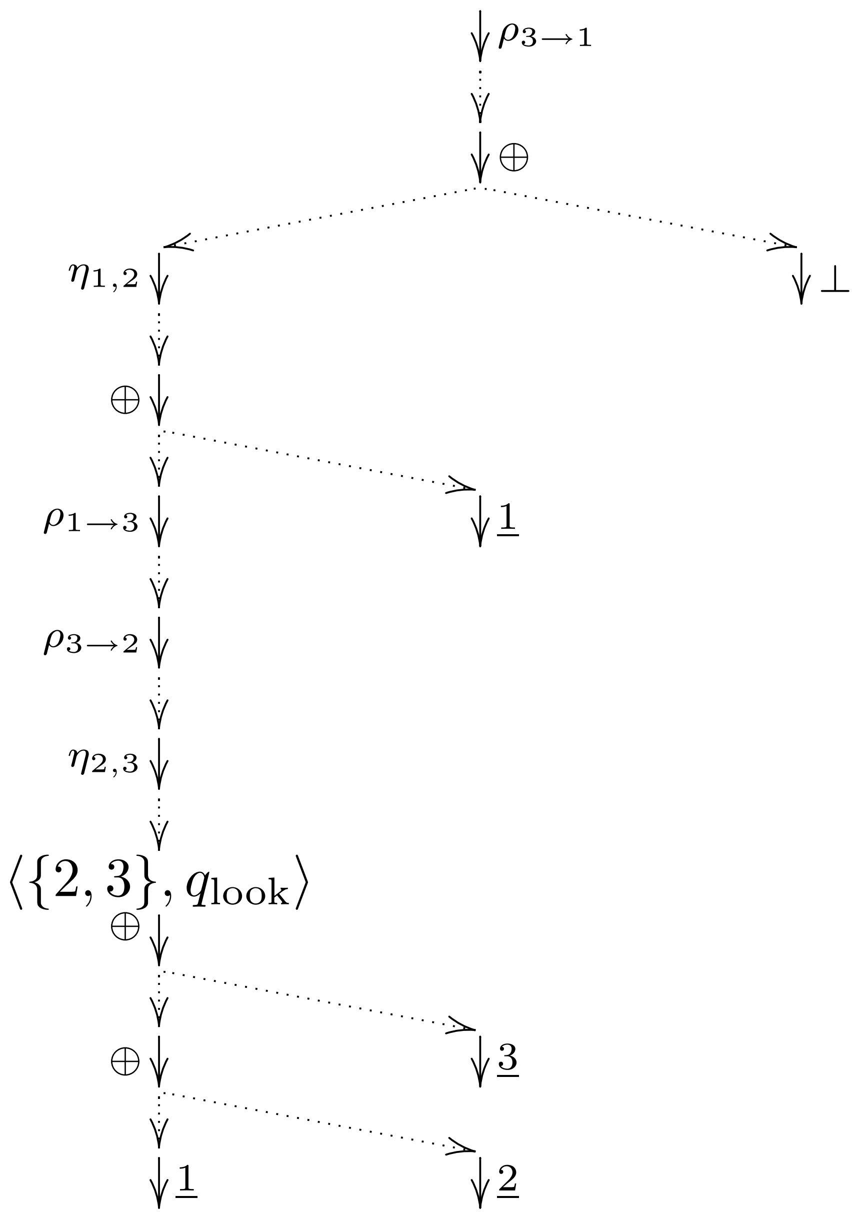
Sample Application



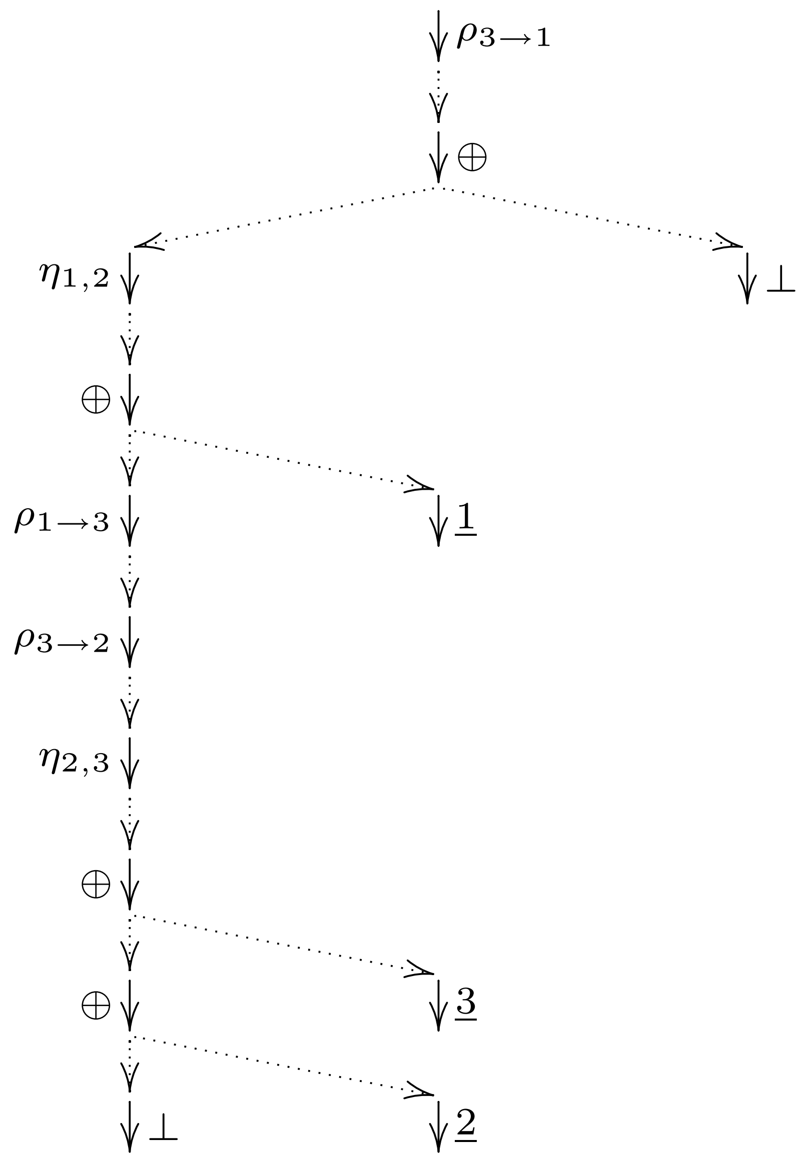
SAMPLE APPLICATION



SAMPLE APPLICATION



Sample Application



PROPERTIES OF DETERMINISTIC TRANSDUCERS

- The inverse image of a rational set of terms by a deterministic transducer is rational.
- The image of a rational set of terms by a deterministic transducer needs not to be rational.
- The image of a regular term (unfolding of a finite folded term) by a deterministic transducer is a regular term.
- Deterministic transducers are closed under composition.

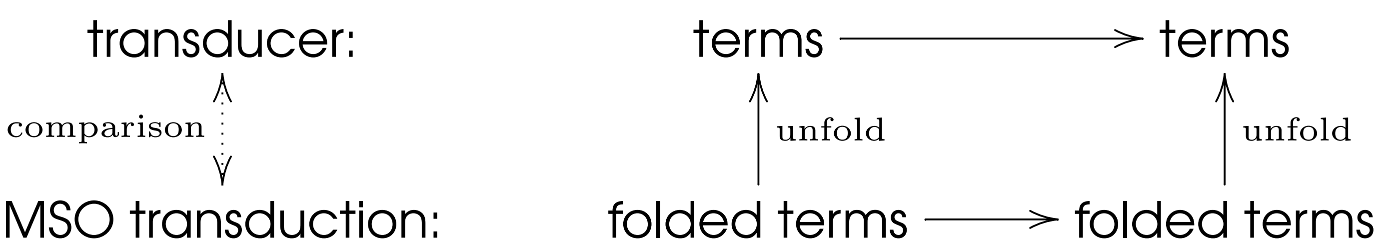
OUTLINE

(1) Basic definitions and terminology

terms, folded terms, MSO logic, rational sets of terms

(2) Overview and background

transformation of objects by transformation of representation



(3) Deterministic top-down tree transducers with rational lookahead

(4) MSO transductions

(5) Main result: comparison of deterministic transducers and MSO transductions

MSO TRANSDUCTIONS

$n \in \mathbb{N}$

$$M = (\Sigma_{\mathcal{F}}, \Sigma_{\mathcal{F}'}, (\phi_{a,i,j}(x,y)), (\rho_i(x,y)), n)$$

$a \in \Sigma_{\mathcal{F}'}, i, j \in \{1, \ldots, n\}$ $i \in \{1, \ldots, n\}$

MSO-formulas $\phi_{a,i,j}(x,y)$ and $\rho_i(x,y)$ over the signature $(E_a)_{a \in \Sigma_{\mathcal{F}}}$

For a folded term $G = (V_G, E_G)$ with root r_G, M defines a folded term $M(G) = (V_{M(G)}, E_{M(G)})$ with root $r_{M(G)}$:

- $V_{M(G)} = V \times [1, n]$
- $((v,i), a, (u,j)) \in E_{M(G)}$ iff $G \models \phi_{a,i,j}(v,u)$
- $r_{M(G)} = (u,i)$ for the unique u and i with $G \models \rho_i(r_G, u)$.

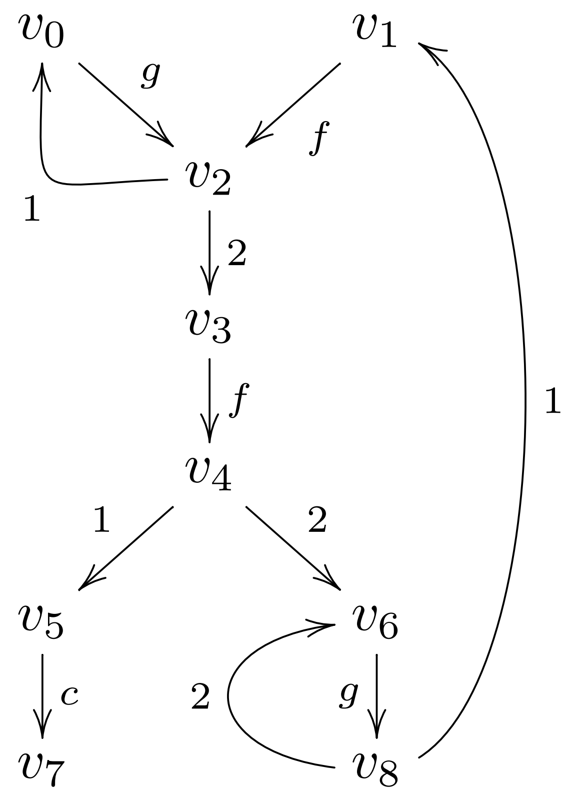
semantic conditions

EXAMPLE

$\mathcal{F} = \mathcal{F}' = \{f, g, c\}$ with f, g binary and c constant.
Swap subterms of f if the right subterm contains c

EXAMPLE

$\mathcal{F} = \mathcal{F}' = \{f, g, c\}$ with f, g binary and c constant.
Swap subterms of f if the right subterm contains c



Example

$\mathcal{F} = \mathcal{F}' = \{f, g, c\}$ with f, g binary and c constant.
Swap subterms of f if the right subterm contains c

Root:

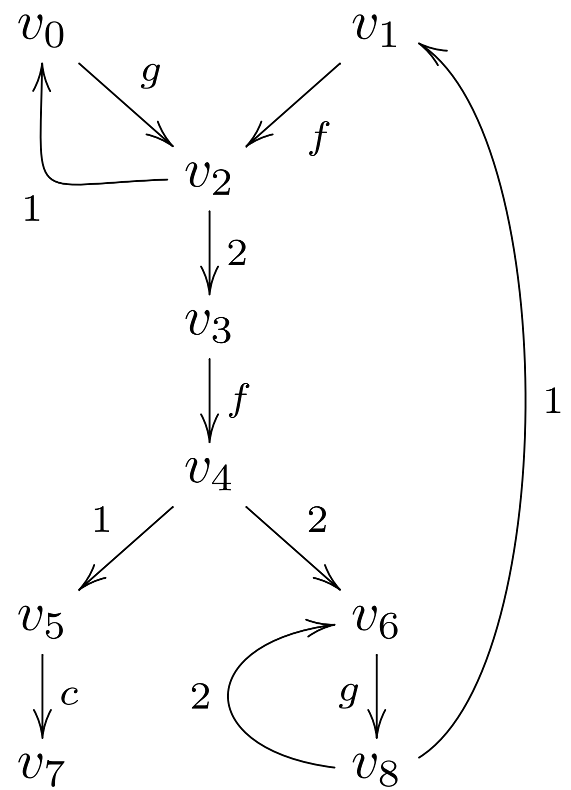
$\rho_1(x, y) = (x = y)$

Edges:

$\phi_{a,1,1}(x, y) = E_a(x, y)$
for $a \in \{g, c, 1, 2\}$

$\phi_{1,2,1}(x, y) = E_2(x, y)$
$\phi_{2,2,1}(x, y) = E_1(x, y)$

$\phi_{f,1,1}(x, y) =$
$E_f(x, y) \wedge \neg \phi_{f,1,2}(x, y)$



$$\phi_{f,1,2}(x, y) = \quad E_f(x, y) \wedge \exists z[E_2(y, z) \wedge$$
$$\forall X(z \in X \wedge \forall z', z''(z' \in X \wedge E(z', z'') \rightarrow z'' \in X)$$
$$\rightarrow \exists z', z'' \in X(E_c(z', z'')))]$$

EXAMPLE

$\mathcal{F} = \mathcal{F}' = \{f, g, c\}$ with f, g binary and c constant.
Swap subterms of f if the right subterm contains c

Root:

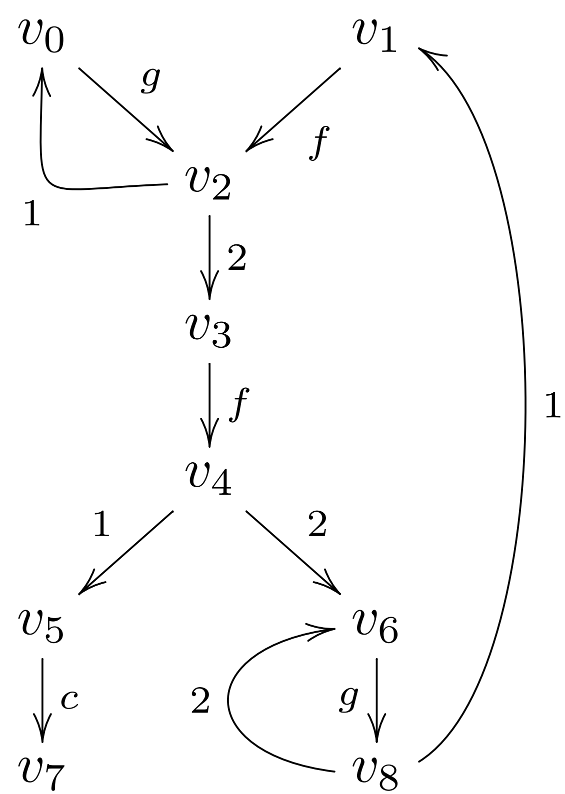
$\rho_1(x, y) = (x = y)$

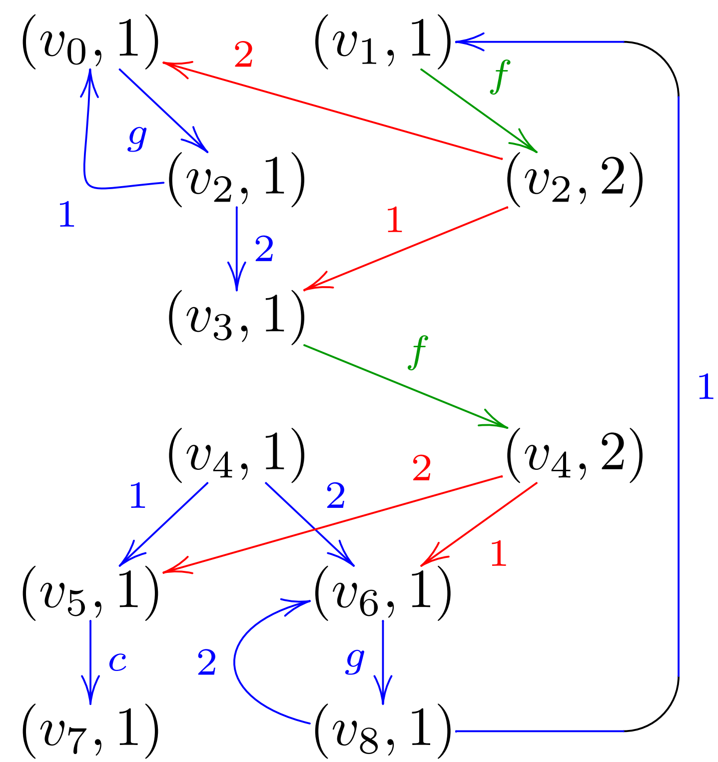
Edges:

$\phi_{a,1,1}(x, y) = E_a(x, y)$
for $a \in \{g, c, 1, 2\}$

$\phi_{1,2,1}(x, y) = E_2(x, y)$
$\phi_{2,2,1}(x, y) = E_1(x, y)$

$\phi_{f,1,1}(x, y) =$
$E_f(x, y) \wedge \neg \phi_{f,1,2}(x, y)$





$$\phi_{f,1,2}(x, y) = \quad E_f(x, y) \wedge \exists z[E_2(y, z) \wedge$$
$$\forall X(z \in X \wedge \forall z', z''(z' \in X \wedge E(z', z'') \to z'' \in X)$$
$$\to \exists z', z'' \in X(E_c(z', z'')))]$$

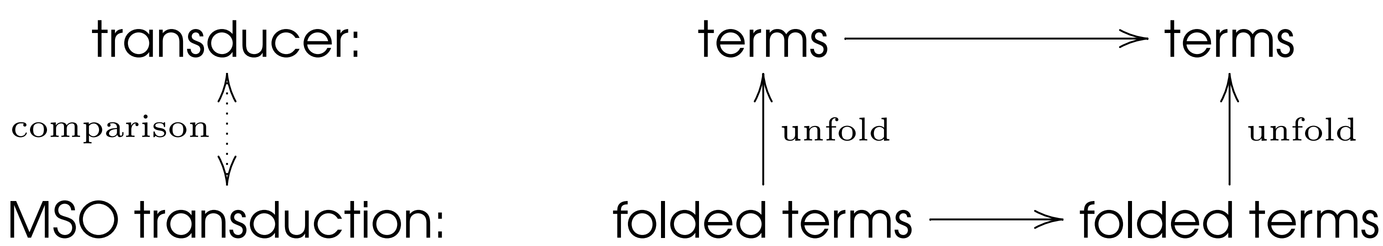
OUTLINE

(1) Basic definitions and terminology

terms, folded terms, MSO logic, rational sets of terms

(2) Overview and background

transformation of objects by transformation of representation



(3) Deterministic top-down tree transducers with rational lookahead

(4) MSO transductions

(5) Main result: comparison of deterministic transducers and MSO transductions

Bisimilarity Preserving Transductions

An MSO Transduction M is bisimilarity preserving if for any two rooted folded terms G, G':

$$\mathrm{unfold}(G) = \mathrm{unfold}(G') \Rightarrow \mathrm{unfold}(M(G)) = \mathrm{unfold}(M(G'))$$

MAIN RESULT

Bisimilarity preserving MSO Transductions and deterministic transducers have the same expressive power.

MAIN RESULT

Bisimilarity preserving MSO Transductions and deterministic transducers have the same expressive power.

More precisely:

(i) For each deterministic transducer T there exists a bisimilarity preserving MSO transduction M_T such that for all folded terms G:

$$\mathrm{unfold}(M_T(G)) = T(\mathrm{unfold}(G))$$

(ii) For each bisimilarity preserving MSO transduction M there exists a deterministic transducer T_M such that for all folded terms G:

$$\mathrm{unfold}(M(G)) = T_M(\mathrm{unfold}(G))$$

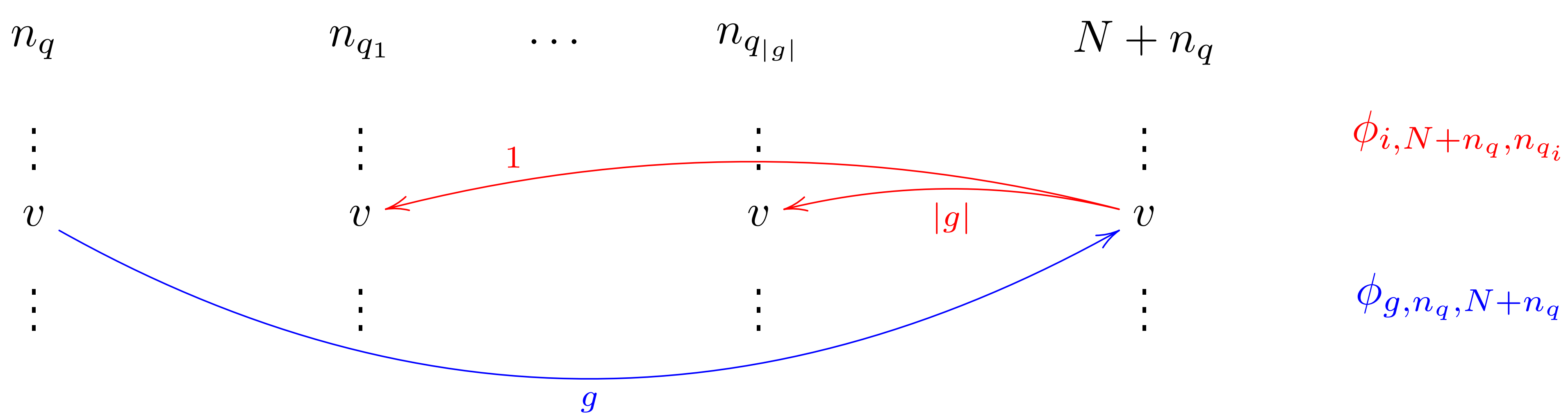
TRANSDUCER $\longrightarrow$ MSO TRANSDUCTION

- If T has N states, then M_T uses $2 \cdot N$ copies of G.
- State q identified uniquely with a number n_q.
- To deal with consumption and lookahead rules a new symbol ε of arity 1 is introduced. This can be removed by a second MSO transduction.

Transducer $\longrightarrow$ MSO Transduction

- If T has N states, then M_T uses $2 \cdot N$ copies of G.
- State q identified uniquely with a number n_q.
- To deal with consumption and lookahead rules a new symbol ε of arity 1 is introduced. This can be removed by a second MSO transduction.

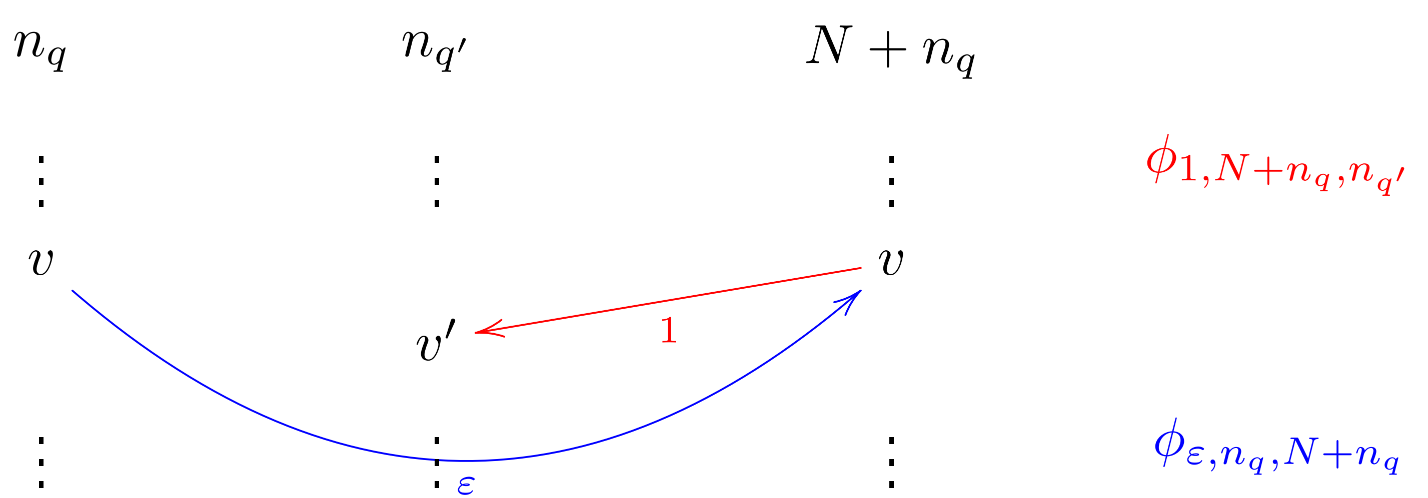
Production rule $q(x) \to g(q_1(x), \ldots, q_{|g|}(x))$



Transducer $\longrightarrow$ MSO Transduction

- If T has N states, then M_T uses $2 \cdot N$ copies of G.
- State q identified uniquely with a number n_q.
- To deal with consumption and lookahead rules a new symbol ε of arity 1 is introduced. This can be removed by a second MSO transduction.

Consumption rule $q(f(x_1, \ldots, x_{|f|})) \rightarrow q'(x_i)$

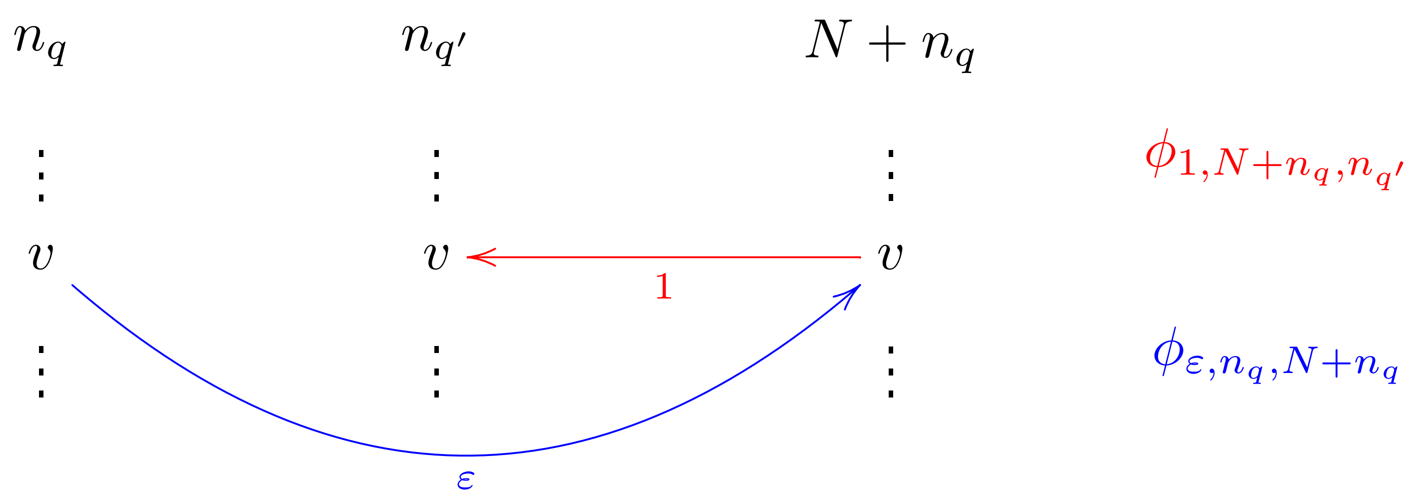


if exists u with $v \xrightarrow{f} u \xrightarrow{i} v'$ in G

Transducer $\longrightarrow$ MSO Transduction

- If T has N states, then M_T uses $2 \cdot N$ copies of G.
- State q identified uniquely with a number n_q.
- To deal with consumption and lookahead rules a new symbol ε of arity 1 is introduced. This can be removed by a second MSO transduction.

Lookahead rule $q(x \in L) \rightarrow q'(x)$



if $\mathrm{unfold}(G, v)$ is in L

MSO TRANSDUCTION $\longrightarrow$ TRANSDUCER

For each bisimilarity preserving MSO transduction M there exists a deterministic transducer T_M such that for all folded terms G:

$$\mathrm{unfold}(M(G)) = T_M(\mathrm{unfold}(G))$$

MSO TRANSDUCTION $\longrightarrow$ TRANSDUCER

For each bisimilarity preserving MSO transduction M there exists a deterministic transducer T_M such that for all folded terms G:

$$\mathrm{unfold}(M(G)) = T_M(\mathrm{unfold}(G))$$

It suffices to consider M on terms:

M bisimilarity preserving $\Rightarrow \mathrm{unfold}(M(G)) = \mathrm{unfold}(M(\mathrm{unfold}(G)))$

MSO TRANSDUCTION $\longrightarrow$ TRANSDUCER

For each bisimilarity preserving MSO transduction M there exists a deterministic transducer T_M such that for all terms t:

$$\mathrm{unfold}(M(t)) = T_M(t)$$

MSO Transduction $\longrightarrow$ Transducer

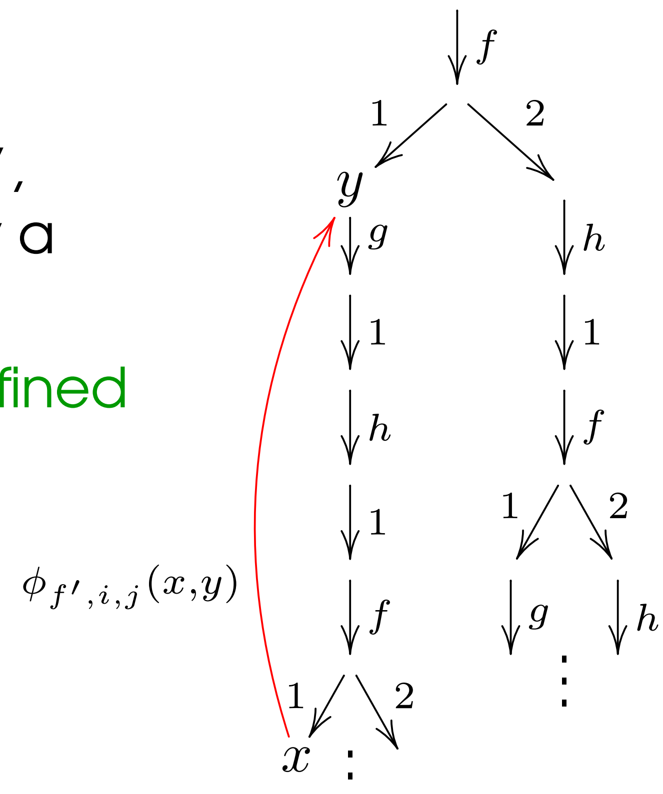
For each bisimilarity preserving MSO transduction M there exists a deterministic transducer T_M such that for all terms t:

$$\mathrm{unfold}(M(t)) = T_M(t)$$

Main difficulty:

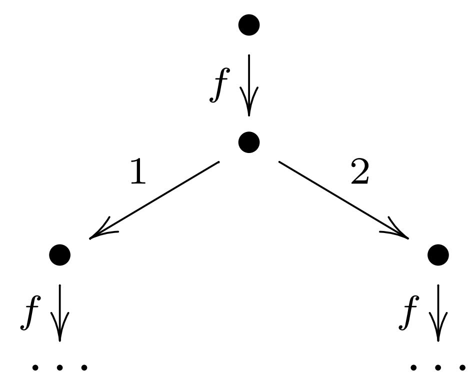
- Transducers work top-down.
- If M defines new edges 'going upward', these edges cannot be constructed by a finite state transducer.

$\Rightarrow$ In a first step normalize M such that defined edges are 'going downward'.



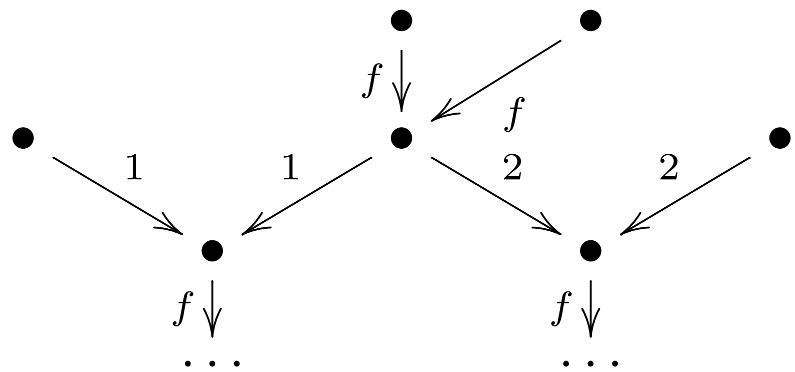
TOP-DOWN NORMALIZATION

t :

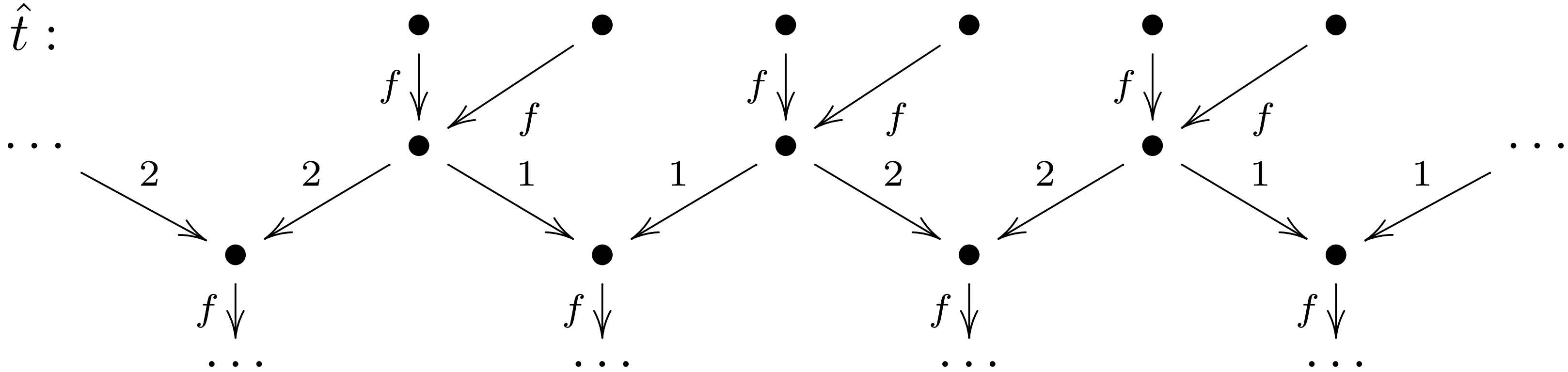


TOP-DOWN NORMALIZATION

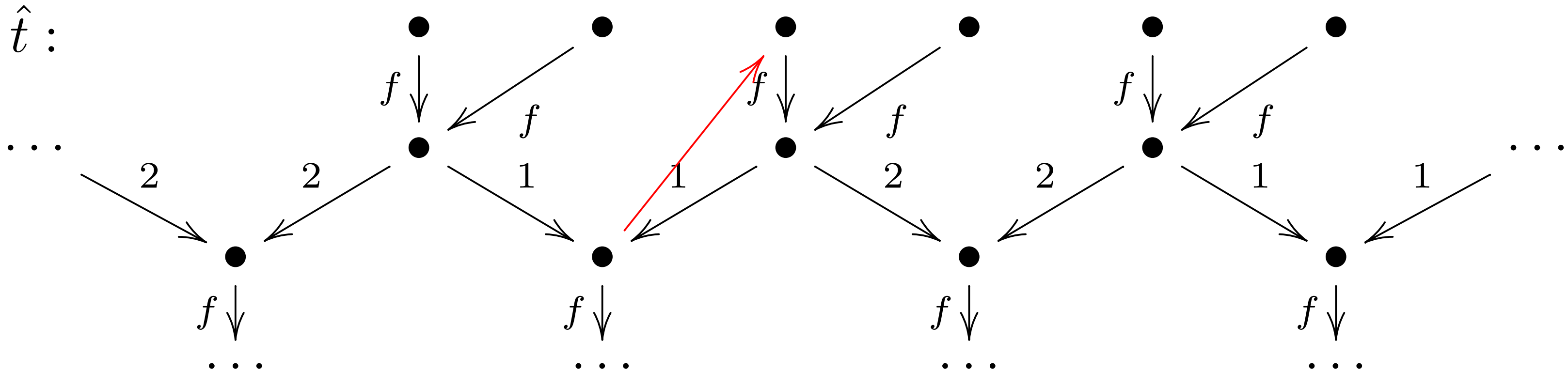
t :



TOP-DOWN NORMALIZATION

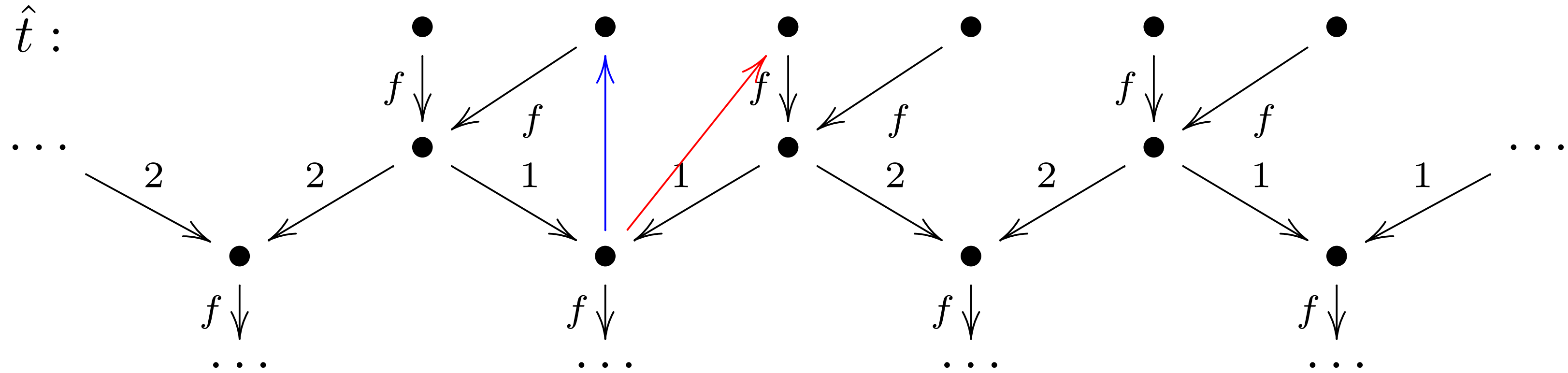


TOP-DOWN NORMALIZATION



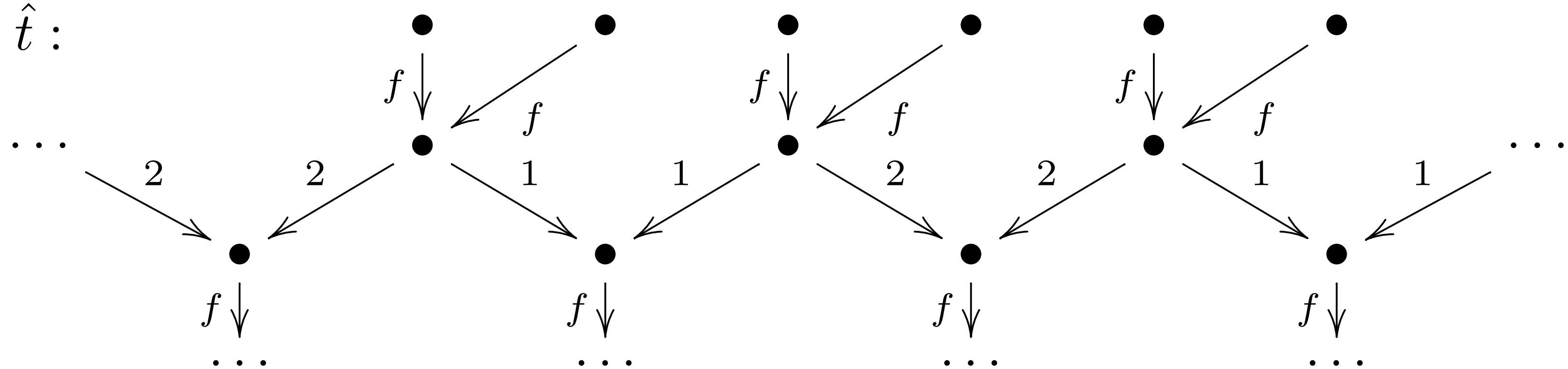
Consider M on $\hat{t}$ (with root inherited from t) and assume a new edge goes upward.

Top-Down Normalization



Consider M on $\hat{t}$ (with root inherited from t) and assume a new edge goes upward.
Then the same formula defines another edge with the same origin.
Hence $M(\hat{t})$ is not a folded term.

Top-Down Normalization

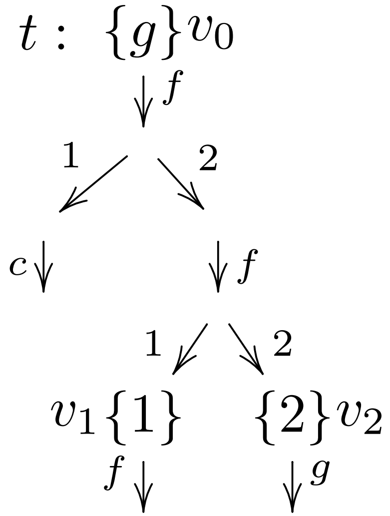


- In $\hat{t}$ the edges defined by M are going downward.
- The formulas $\phi_{a,i,j}$ on $\hat{t}$ can be transformed into formulas $\hat{\phi}_{a,i,j}$ on t ($\hat{t}$ can be obtained from t by the Muchnik/Walukiewicz construction).
- The new MSO transduction $\hat{M}$ using the formulas $\hat{\phi}_{a,i,j}$ has the following properties:
  - $\mathrm{unfold}(M(t)) = \mathrm{unfold}(\hat{M}(t))$
  - The edges defined by $\hat{M}$ are going downward.

NORMALIZED TRANSDUCTION → TRANSDUCER

Rough sketch:

- Normalized Transduction $M = (\Sigma_{\mathcal{F}}, \Sigma_{\mathcal{F}'}, (\phi_{a,i,j}(x,y)), (\rho_i(x,y)), n)$
- Transform formulas $\phi_{a,i,j}(x,y)$ into (Rabin) tree automata accepting 'marked terms':



$\mathcal{A}_{g,i,j_1,j_2}$ accepts t if for some ℓ and v

$$\begin{aligned} t &\models \phi_{g,i,\ell}(v_0, v) \\ t &\models \phi_{1,\ell,j_1}(v, v_1) \\ t &\models \phi_{1,\ell,j_2}(v, v_2) \end{aligned}$$

- Transducer T_M keeps track of the states of the automata $\mathcal{A}_{a,i,j_1,\ldots,j_k}$ while going through the term.
- The lookahead is used to check for which automaton there exists a marking that is accepted. This information is used to construct the next edge.

CONCLUSION

- For every deterministic transducer there is an equivalent MSO transduction.
  $\leadsto$ decidability of the MSO theory of terms is preserved
- For every bisimilarity preserving MSO transduction there is an equivalent deterministic transducer.
  $\leadsto$ deterministic transducers are expressively complete for MSO logic
- Transducers are more handy than MSO transductions concerning their construction and the proofs of correctness (cf. thesis of T. Colcombet)

Open:

- We assume that M is bisimilarity preserving for finite and infinite folded terms. Can one transfer the result if M has this property only for finite folded terms?
- Transfer (and analyze) other models of transducers that have been defined for finite terms to the infinite world.