

The Equivalence Problem for Deterministic MSO Tree Transducers is Decidable

Sebastian Maneth

EPFL, Switzerland

Joint Work with Joost Engelfriet

Outline

- Equivalence Problems
- Parikh's Theorem and Semilinear Sets
- MSO Tree Transducers
- Deciding Equivalence of MSOTTs
- Future

Equivalence Problems

- *nondeterministic (one-way) finite state transducers*

undecidable

[Griffiths68]

(→ reduction from PCP, use complement and union)

Equivalence Problems

- *nondeterministic (one-way) finite state transducers* **undecidable**
[Griffiths68]
(→ reduction from PCP, use complement and union)
- *deterministic (one-way) finite state transducers* **decidable** [Gurari82]
(→

Equivalence Problems

- *nondeterministic (one-way) finite state transducers* **undecidable**
[Griffiths68]
(→ reduction from PCP, use complement and union)
- *deterministic (one-way) finite state transducers* **decidable** [Gurari82]
(→
- *deterministic top-down tree transducers* **decidable** [Esik80]

Equivalence Problems

- *nondeterministic (one-way) finite state transducers* **undecidable**
[Griffiths68]
(→ reduction from PCP, use complement and union)
- *deterministic (one-way) finite state transducers* **decidable** [Gurari82]
(→
- *deterministic top-down tree transducers* **decidable** [Esik80]
- *nonnested, seperated attributed/marco tree transducers* **decidable**
[Courcelle/Franchi-Zannetacci82]

(→ seperated = can be evaluated in two phases,
(1) only inherited, over Δ_{inh}
(2) synthsized, over Δ_{syn})

Equivalence Problems

What about (deterministic) *macro tree transducers*?

Even for linear in the parameters and in the input variables

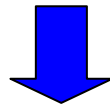
→ OPEN!

Equivalence Problems

What about (deterministic) *macro tree transducers*?

Even for linear in the parameters and in the input variables

→ OPEN!



monadic output

What about (det.) *top-down tree-to-string transducers*?

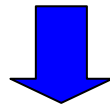
Even for linear transducers → OPEN!

Equivalence Problems

What about (deterministic) *macro tree transducers*?

Even for linear in the parameters and in the input variables

→ OPEN!



monadic output

What about (det.) *top-down tree-to-string transducers*?

Even for linear transducers → OPEN!

EXAMPLE

$q, g(x_1, x_2) \rightarrow a \langle q, x_1 \rangle \langle q, x_2 \rangle$

$q, e \rightarrow a$

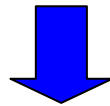
Translates *bin-tree with n nodes* into *string aaa... a*

Equivalence Problems

What about (deterministic) *macro tree transducers*?

Even for linear in the parameters and in the input variables

→ OPEN!



monadic output

What about (det.) *top-down tree-to-string transducers*?

Even for linear transducers → OPEN!

EXAMPLE

$q, g(x_1, x_2) \rightarrow a \langle q, x_1 \rangle \langle q, x_2 \rangle$

$q, e \rightarrow a$

Translates *bin-tree with n nodes* into *string aaa... a*

$p, g(x_1, x_2) \rightarrow \langle p, x_2 \rangle a \langle p, x_1 \rangle$

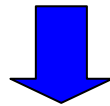
$p, e \rightarrow a$

Equivalence Problems

What about (deterministic) *macro tree transducers*?

Even for linear in the parameters and in the input variables

→ OPEN!



monadic output

What about (det.) *top-down tree-to-string transducers*?

Even for linear transducers → OPEN!

$$\begin{aligned} \text{DMTTlin} \subseteq \mathbf{DMTT^{Rfc}} = \mathbf{DMSOTT} &\supseteq \text{yDTfc} \supseteq \text{yDTlin} \\ &= \mathbf{ATTsur^R} \\ &= \mathbf{DMTT^* \cap LSI} \end{aligned}$$

Equivalence Problems

What about (deterministic) *macro tree transducers*?

Even for linear in the parameters and in the input variables

→ **decidable!**

What about (det.) *top-down tree-to-string transducers*?

Even for linear transducers → **decidable!**

$$\text{DMTTlin} \subseteq \mathbf{DMTT^{\text{Rfc}} = \text{DMSOTT}} \supseteq \text{yDTfc} \supseteq \text{yDTlin}$$

MAIN THEOREM

The Equivalence problem for **DMSOTT** is **decidable**.

FACTS

DMSOTT

- Closed under composition
- inverses preserve REGT

DMSOTT(REGT)

$$= \text{MSOTT}(\text{REGT})$$

- Closed under MSOTT
- path languages(MSOTT(REGT))
= string languages(MSOTT(REGT))
= MSOTS(REGT)
- are **Parikh**

FACTS

DMSOTT

- Closed under composition
- inverses preserve REGT

DMSOTT(REGT)

$$= \text{MSOTT}(\text{REGT})$$

- Closed under MSOTT
- path languages(MSOTT(REGT))
 - = string languages(MSOTT(REGT))
 - = MSOTS(REGT)

- are **Parikh**

Languages in MSOTT(REGT)

→ are **Parikh**

Let $\Sigma = \{ \sigma_1, \dots, \sigma_k \}$.

For a discrete graph / string / tree g over Σ , define

$$\text{Par}(g) = (n_1, \dots, n_k) \in \mathbf{N}^k$$

such that n_i is the number of σ_i 's in g .

A set $P \subseteq \mathbf{N}^k$ is **semilinear** if there exists

a *regular language* R such that $P = \text{Par}(R)$.

A set S (of graphs/stings) is **Parikh**, if $\text{Par}(S)$ is semilinear.

Languages in MSOTT(REGT)

→ are **Parikh**

Let $\Sigma = \{ \sigma_1, \dots, \sigma_k \}$.

For a discrete graph / string / tree g over Σ , define

$$\text{Par}(g) = (n_1, \dots, n_k) \in \mathbf{N}^k$$

such that n_i is the number of σ_i 's in g .

A set $P \subseteq \mathbf{N}^k$ is **semilinear** if there exists

a *regular language* R such that $P = \text{Par}(R)$.

A set S (of graphs/stings) is **Parikh**, if $\text{Par}(S)$ is semilinear.

PARIKH'S THEOREM

Every context-free language is **Parikh**.

Parikh's Theorem and Semilinear Sets

Context-free languages are **Parikh**

$$L1 = \{ a^n b^n \mid n \geq 1 \}$$

Parikh's Theorem and Semilinear Sets

Context-free languages are **Parikh**

$$L1 = \{ a^n b^n \mid n \geq 1 \}$$

$$\text{Par}(L1) = \{ (n, n) \mid n \geq 1 \}$$

Parikh's Theorem and Semilinear Sets

Context-free languages are **Parikh**

$$L1 = \{ a^n b^n \mid n \geq 1 \}$$

$$\text{Par}(L1) = \{ (n, n) \mid n \geq 1 \} = \text{Par}((ab)^*)$$

Parikh's Theorem and Semilinear Sets

Context-free languages are **Parikh**

$$L1 = \{ a^n b^n \mid n \geq 1 \}$$

$$\text{Par}(L1) = \{ (n, n) \mid n \geq 1 \} = \text{Par}((ab)^*)$$

$$L2 = \{ w w \mid w \in \{ a, b \}^* \}$$

$$\text{Par}(L2) =$$

Parikh's Theorem and Semilinear Sets

Context-free languages are **Parikh**

$$L1 = \{ a^n b^n \mid n \geq 1 \}$$

$$\text{Par}(L1) = \{ (n, n) \mid n \geq 1 \} = \text{Par}((ab)^*)$$

$$L2 = \{ w w \mid w \in \{ a, b \}^* \}$$

$$\text{Par}(L2) = \text{Par}((ab)^*)$$

$$L3 = \{ w \in \{ a, b \}^* \mid \#a(w) = \#b(w) \}$$

→ Is $L3$ context-free??

Parikh's Theorem and Semilinear Sets

Context-free languages are **Parikh**

$$L1 = \{ a^n b^n \mid n \geq 1 \}$$

$$\text{Par}(L1) = \{ (n, n) \mid n \geq 1 \} = \text{Par}((ab)^*)$$

$$L2 = \{ w w \mid w \in \{ a, b \}^* \}$$

$$\text{Par}(L2) = \text{Par}((ab)^*)$$

$$L3 = \{ w \in \{ a, b \}^* \mid \#a(w) = \#b(w) \}$$

→ Is L3 context-free?? SURE! ...even deterministic...

$$\text{And, again, } \text{Par}(L3) = \text{Par}(L2) = \text{Par}(L1)$$

Languages in MSOTT(REGT)

→ are **Parikh**

Let $\Sigma = \{ \sigma_1, \dots, \sigma_k \}$.

For a discrete graph / string / tree g over Σ , define

$$\text{Par}(g) = (n_1, \dots, n_k) \in \mathbf{N}^k$$

such that n_i is the number of σ_i 's in g .

A set $P \subseteq \mathbf{N}^k$ is **semilinear** if there exists
a *regular language* R such that $P = \text{Par}(R)$.

A set S (of graphs/stings) is **Parikh**, if $\text{Par}(S)$ is semilinear.

PARIKH'S THEOREM

Every context-free language is **Parikh**.

Languages in $MSOTT(REGT) = ATTsur^R(REGT)$

→ are **Parikh**

Let $\Sigma = \{ \sigma_1, \dots, \sigma_k \}$.

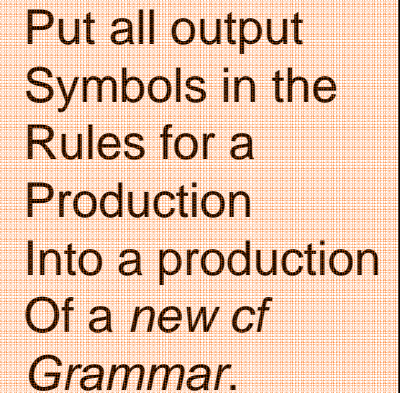
For a discrete graph / string / tree g over Σ , define

$$Par(g) = (n_1, \dots, n_k) \in \mathbf{N}^k$$

such that n_i is the number of σ_i 's in g .

A set $P \subseteq \mathbf{N}^k$ is **semilinear** if there exists
a *regular language* R such that $P = Par(R)$.

A set S (of graphs/stings) is **Parikh**, if $Par(S)$ is semilinear.



Put all output
Symbols in the
Rules for a
Production
Into a production
Of a *new cf*
Grammar.

PARIKH'S THEOREM

Every context-free language is **Parikh**.

Languages L in $MSOTT(REGT) = ATTsur^R(REGT)$
→ are **Parikh**

$Par(L)$ is semilinear (i.e., $= Par(R)$ for some reg.l. R)

Lemma It is decidable for a **semilinear set** $S \subseteq \mathbf{N}^2$,
whether there exists $n \in \mathbf{N}$ such that $(n,n) \in S$.

Proof. Let $P = \{ (n,n) \mid n \in \mathbf{N} \} = Par((ab)^*)$

Then $S \cap P$ is semilinear, and semilinear sets have
decidable emptiness. [[GinsburgSpanier64](#), hard!]

Alternative Proof??

Languages L in $MSOTT(REGT) = ATTsur^R(REGT)$
→ are **Parikh**

$Par(L)$ is semilinear (i.e., $= Par(R)$ for some reg.l. R)

Lemma It is decidable for a **semilinear set** $S \subseteq \mathbf{N}^2$,
whether there exists $n \in \mathbf{N}$ such that $(n,n) \in S$.

Alternative Proof

$$L3 = \{ w \in \{ a, b \}^* \mid \#a(w) = \#b(w) \}$$

Let R be reg. language s.t. $Par(S) = R$.

Then $R \cap L3$ is context-free, and
cf. languages have decidable emptiness.

MAIN THEOREM

The Equivalence problem for **DMSOTT** is **decidable**.

Let's first do string output

MAIN THEOREM

The Equivalence problem for **DMSOTS** is **decidable**.

Let's first do string output

MAIN THEOREM

The Equivalence problem for **DMSOTS** is **decidable**.

M1, M2 deterministic MSO tree-to-string transducers

(1) Add an end marker \$ (gives dmsots's N1/N2)

$$N1(s) = M1(s) \$$$

$$N2(s) = M2(s) \$$$

(2) M1 is equiv. to M2

iff $\neg (\exists s \exists n \ N1(s) / n \neq N2(s) / n)$

Let's first do string output

MAIN THEOREM

The Equivalence problem for **DMSOTS** is **decidable**.

M1, M2 deterministic MSO tree-to-string transducers

(1) Add an end marker \$ (gives dmsots's N1/N2)

$$N1(s) = M1(s) \$$$

$$N2(s) = M2(s) \$$$

(2) M1 is equiv. to M2

iff $\neg (\exists s \exists n \ N1(s) / n \neq N2(s) / n)$

iff $\neg (\exists a, b: a \neq b \wedge \exists s \exists n \ N1(s) / n = a \wedge N2(s) / n = b)$

Let's first do string output

MAIN THEOREM

The Equivalence problem for **DMSOTS** is **decidable**.

M1, M2 deterministic MSO tree-to-string transducers

(1) Add an end marker \$ (gives dmsots's N1/N2)

$$N1(s) = M1(s) \$$$

$$N2(s) = M2(s) \$$$

(2) M1 is equiv. to M2 (gives $N1^a(s) = \{ a^n \mid N1(s) / n = a \}$
and $N2^b(s)$)

iff $\neg (\exists s \exists n \ N1(s) / n \neq N2(s) / n)$

iff $\neg (\exists a, b: a \neq b \wedge \exists s \exists n \ N1(s) / n = a \wedge N2(s) / n = b)$

iff $\neg (\dots \wedge \exists s \exists n \ a^n \in N1^a(s) \wedge b^n \in N2^b(s))$

Let's first do string output

MAIN THEOREM

The Equivalence problem for **DMSOTS** is **decidable**.

M1, M2 deterministic MSO tree-to-string transducers

(1) Add an end marker \$ (gives dmsots's N1/N2)

$$N1(s) = M1(s) \$$$

$$N2(s) = M2(s) \$$$

(2) M1 is equiv. to M2 (gives $N12^{a,b}(s) = \{ a^n b^m \mid N1(s) / n = a, N2(s) / m = b \}$)

iff $\neg (\exists s \exists n \ N1(s) / n \neq N2(s) / n)$

iff $\neg (\exists a, b: a \neq b \wedge \exists s \exists n \ N1(s) / n = a \wedge N2(s) / n = b)$

iff $\neg (\dots \wedge \exists s \exists n \ a^n \in N1^a(s) \wedge b^n \in N2^b(s))$

iff $\neg (\dots \wedge \text{Par}(\text{Out}(N12^{a,b})) \cap \{(n, n) \mid n \geq 1\} \neq \emptyset)$

Let's first do string output

MAIN THEOREM

The Equivalence problem for **DMSOTS** is **decidable**.

M1, M2 deterministic MSO tree-to-string transducers

(1) Add an end marker \$ (gives dmsots's N1/N2)

$$N1(s) = M1(s) \$$$

$$N2(s) = M2(s) \$$$

(2) M1 is equiv. to M2 (gives $N12^{a,b}(s) = \{ a^n b^m \mid N1(s) / n = a, N2(s) / m = b \}$)

iff $\neg (\exists a, b: a \neq b \wedge \text{Par}(\text{Out}(N12^{a,b})) \cap \{ (n, n) \mid n \geq 1 \} \neq \emptyset)$
 $\in \Delta \cup \{ \$ \}$

for finitely many a,b

(A) construct $N12^{a,b}$

(B) Construct 'Parikh-I' $R_{a,b}$ for $\text{Out}(N12_{a,b})$

(C) Check Intersection-Emptiness w. **L3**

□

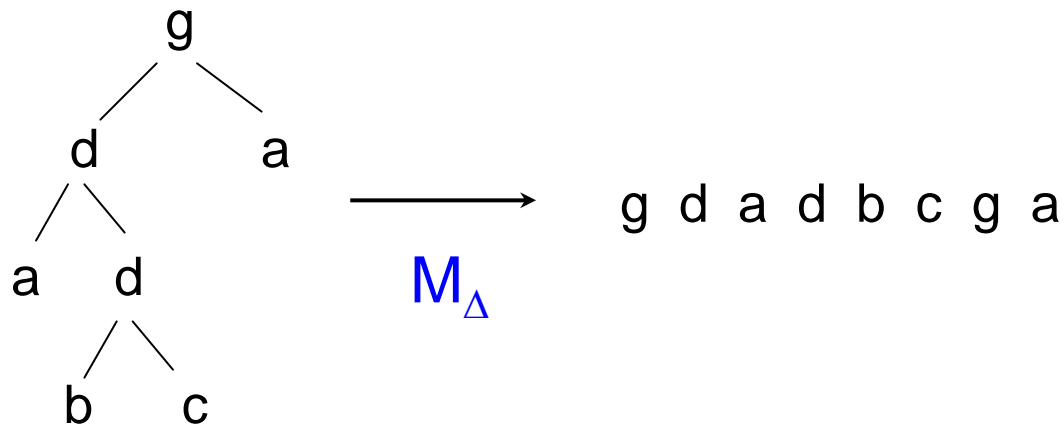
Let's first do string output

MAIN THEOREM

The Equivalence problem for **DMSOTS** is **decidable**.

DMSOTT

→ Translate tree t over Δ into string $\text{pre}(t)$ of its node labels in pre-order.



Let's first do string output

MAIN THEOREM

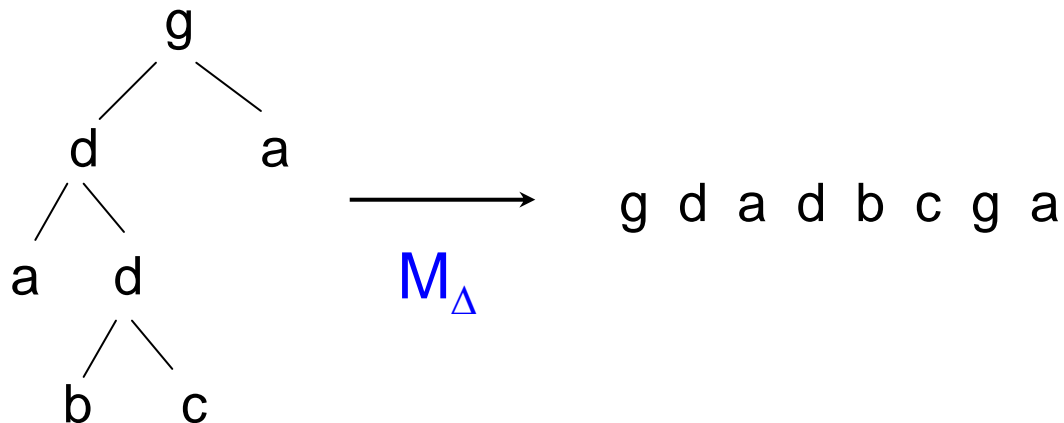
The Equivalence problem for **DMSOTS** is **decidable**.

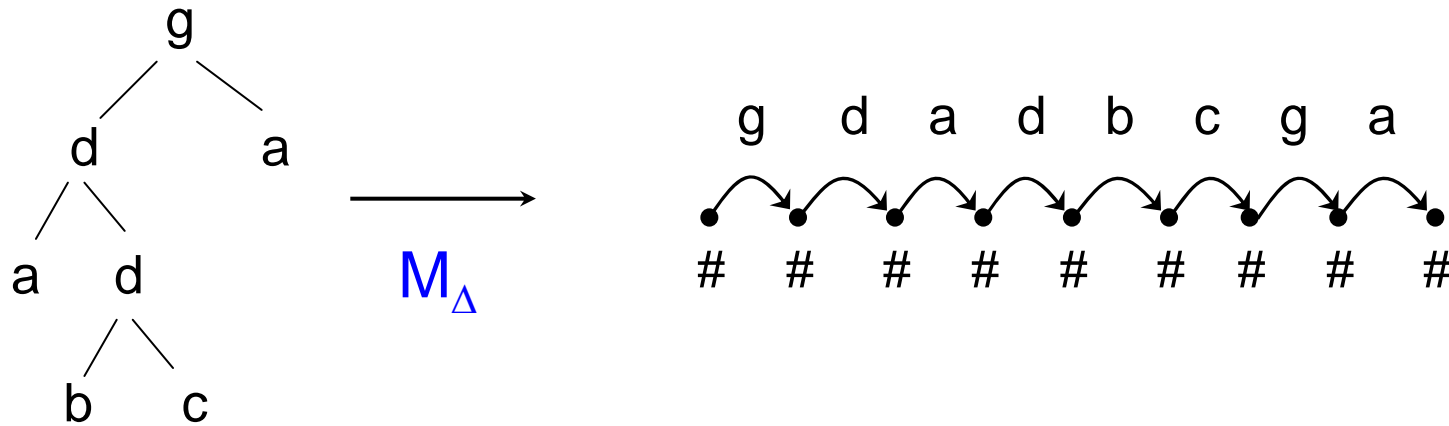
DMSOTT

→ $M_{\Delta} \in \text{DMSOTS}$

Given DMSOTTs M_1, M_2 , construct $N_1 = M_1 ; M_{\Delta}$ and $N_2 = M_2 ; M_{\Delta}$

M_1 equiv. to M_2 iff N_1 equiv. to N_2 □





DMSOTS $M_\Delta = (\{ 1, 2 \}, \text{true}, \{ \psi_{1,\#}, \psi_{2,\#} \}, \{ \chi_{c,c',\delta} \}_{c,c' \in \{1,2\}, \delta \in \Delta})$

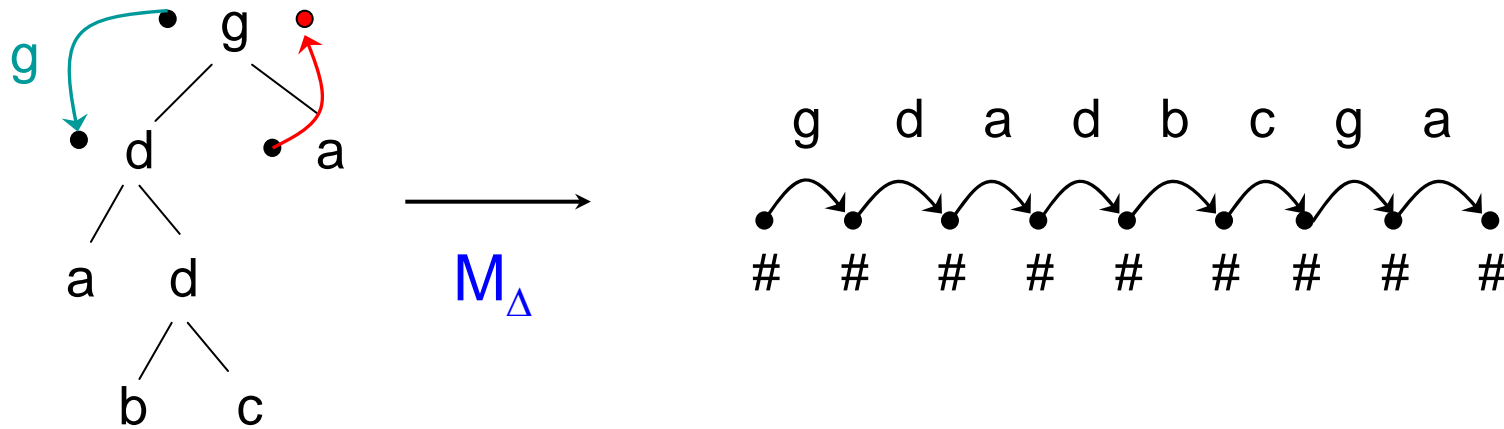
$$\psi_{1,\#}(x) = \text{true}$$

$$\psi_{2,\#}(x) = \text{root}(x) = \neg(\exists y) (\text{edg}_1(y,x) \vee \text{edg}_2(y,x))$$

$$\chi_{1,1,\delta}(x, y) = \text{lab}_\delta(x) \wedge \pi(x,y)$$

$$\chi_{1,2,\delta}(x, y) = \text{lab}_\delta(x) \wedge \text{root}(y) \wedge \neg(\exists z) \pi(x,z)$$

$$\pi(x,y) = \text{“}y \text{ is successor of } x \text{ in pre-order”}$$



DMSOTS $M_\Delta = (\{ 1, 2 \}, \text{true}, \{ \psi_{1,\#}, \psi_{2,\#} \}, \{ \chi_{c,c',\delta} \}_{c,c' \in \{1,2\}, \delta \in \Delta})$

$$\psi_{1,\#}(x) = \text{true}$$

$$\psi_{2,\#}(x) = \text{root}(x) = \neg(\exists y) (\text{edg}_1(y,x) \vee \text{edg}_2(y,x))$$

$$\chi_{1,1,\delta}(x, y) = \text{lab}_\delta(x) \wedge \pi(x,y)$$

$$\chi_{1,2,\delta}(x, y) = \text{lab}_\delta(x) \wedge \text{root}(y) \wedge \neg(\exists z) \pi(x,z)$$

$$\pi(x,y) = \text{“}y \text{ is successor of } x \text{ in pre-order”}$$

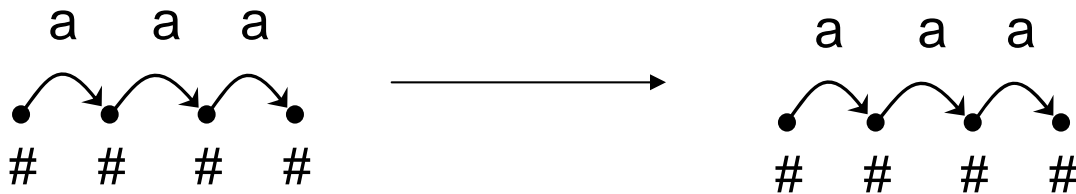
EXAMPLE

Det. MSO string-to-string transducers

Identity on strings over $\Delta = \{a\}$

$$\psi_{\#}(x) = \text{true}$$

$$\chi_a(x,y) = \text{edg}_a(x,y)$$



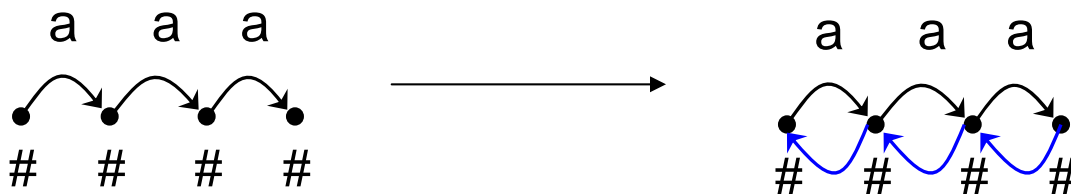
EXAMPLE

Det. MSO string-to-string transducers

Identity on strings over $\Delta = \{a\}$

$$\begin{aligned} \text{M1: } \psi_{\#}(x) &= \text{true} \\ \chi_a(x,y) &= \text{edg}_a(x,y) \end{aligned}$$

$$\begin{aligned} \text{M2: } \psi_{\#}(x) &= \text{true} \\ \chi_a(x,y) &= \text{edg}_a(y,x) \end{aligned}$$



EXAMPLE

Identity on strings over $\Delta = \{ a \}$

$$\begin{aligned} \text{M1: } \psi_{\#}(x) &= \text{true} \\ \chi_a(x,y) &= \text{edg}_a(x,y) \end{aligned}$$

$$\begin{aligned} \text{M2: } \psi_{\#}(x) &= \text{true} \\ \chi_a(x,y) &= \text{edg}_a(y,x) \end{aligned}$$

(1) Add end marker \$ (\rightarrow N1 / N2)

(2) For $x,y \in \Delta \cup \{ \$ \}$ with $x \neq y$:

\rightarrow construct $N_{12^{a,\$}}$

\rightarrow construct automaton A for Parikh-regl $\text{Out}(N_{12^{a,\$}})$

\rightarrow check intersection emptiness with
 $G = \{ S \rightarrow SS \mid aSb \mid bSa \mid \lambda \}$

Equivalence of LSI Macro Tree Transducers

Future