

Logic and Automata

Lecture 5
Properties of MSO Translations

Sebastian Maneth
NICTA & UNSW

Logic Summer School - Canberra, December 2006

Outline

1. Composition Closure of MSOT
2. Inverses of MSOT preserve MSO definability
3. Parikh's Theorem
4. Decidability of equivalence

MSO Definable Translations

MSO graph translation $f: GR(\Sigma_1, \Gamma_1) \rightarrow GR(\Sigma_2, \Gamma_2)$

dom: domain formula $L(\text{dom}) = \text{dom}(f)$
 $N_a(x)$: node formulas for every $a \in \Sigma_2$
 $E_i(x, y)$: edge formulas for every $i \in \Gamma_2$

Input graph $g = (V, E)$ $\rightarrow \text{size}(f(g)) = |U|$
 output graph $h = f(g) = (U, F)$ $\text{is} \leq \text{size}(g) = |V|$

$U = \{u \in V \mid \text{there is a unique } \sigma \in \Sigma_2 \text{ such that } (g, u) \models N_\sigma(x)\}$
 $E = \{(u, \gamma, v) \mid \exists! \gamma \in \Gamma_2: (g, u, v) \models E_\gamma(x, y)\}$
 $\forall u \in U: \text{lab-}\sigma(u) \text{ iff } (g, u) \models N_\sigma(x)$

C: finite "copy" set $(\text{dom}, C, \{N_{a,c}\}_{a \in \Sigma_2, c \in C}, \{E_{\gamma,c,d}\}_{\gamma \in \Gamma_2, c,d \in C})$

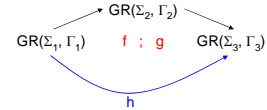
$U = \{(u, c) \in V \times C \mid \exists! \sigma \in \Sigma_2: (g, u) \models N_{\sigma,c}(x)\}$

$E = \{((u, c), \gamma, (v, d)) \mid \exists! \gamma \in \Gamma_2: (g, u, v) \models E_{\gamma,c,d}(x, y)\}$

$\forall u \in U: \text{lab-}\sigma(u) \text{ iff } (g, u) \models N_\sigma(x) \rightarrow \text{size}(f(g)) \leq |C| \cdot \text{size}(g)$

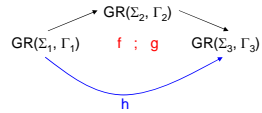
1. Composition Closure

Theorem $MSOT; MSOT \subseteq MSOT$



1. Composition Closure

Theorem $MSOT; MSOT \subseteq MSOT$



First: no copy sets.

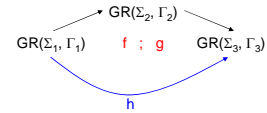
$g = (\text{dom}, \{c\}, \{N_{a,c}\}_{a \in \Sigma_2, c \in C}, \{E_{\gamma,c,d}\}_{\gamma \in \Gamma_2, c,d \in C})$

Obtain h from g

\rightarrow replace $\text{lab}_\sigma(x)$ by g 's definition for $N_\sigma(x)$ and $\text{edge}_i(x, y)$ by g 's definition for $E_i(x, y)$

1. Composition Closure

Theorem $MSOT; MSOT \subseteq MSOT$



First: no copy sets.

$g = (\text{dom}, \{c\}, \{N_{a,c}\}_{a \in \Sigma_2, c \in C}, \{E_{\gamma,c,d}\}_{\gamma \in \Gamma_2, c,d \in C})$

Obtain h from g

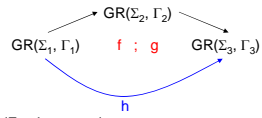
\rightarrow replace $\text{lab}_\sigma(x)$ by g 's definition for $N_\sigma(x)$ and $\text{edge}_i(x, y)$ by g 's definition for $E_i(x, y)$

\rightarrow replace $(\exists x) \phi$ by $(\exists x)(p(x) \wedge \phi)$, $p(x)$ says that there is exactly one $\sigma \in \Sigma_2$ s.t. $N_\sigma(x)$

\rightarrow replace $(\exists X) \phi$ by $(\exists X)(\forall x)(x \in X \rightarrow p(x) \wedge \phi)$

1. Composition Closure

Theorem $MSOT; MSOT \subseteq MSOT$



$$g = (\text{dom}, D, \{N_{\sigma, c}\}_{\sigma \in \Sigma_2, c \in C}, \{E_{\gamma, c, d}\}_{\gamma \in \Gamma_2, c, d \in C})$$

Obtain h from g

- replace $\text{lab}_\sigma(x)$ by $\bigvee_{c \in C} \{g\text{'s definition for } N_\sigma(x)\}$ and $\text{edge}_\gamma(x, y)$ by $\bigvee_{c \in C} \{g\text{'s definition for } E_{\gamma, c, d}\}$
- replace $(\exists x) \phi$ by $(\exists x)(p(x) \wedge \phi)$, $p(x)$ says that there is exactly one $\sigma \in \Sigma_2$ and one $c \in C$ s.t. $N_{\sigma, c}(x)$
- replace $(\exists X) \phi$ by $(\exists X)(\forall x)(x \in X \rightarrow p(x) \wedge \phi)$

2. Inverses perserve REGT

REGT = class of regular tree languages
= class of MSO definable tree languages

Theorem $MSOT^{-1}(\text{REGT}) \subseteq \text{REGT}$

2. Inverses perserve REGT

REGT = class of regular tree languages
= class of MSO definable tree languages

Theorem $MSOT^{-1}(\text{REGT}) \subseteq \text{REGT}$

ψ closed MSO formula

2. Inverses perserve REGT

REGT = class of regular tree languages
= class of MSO definable tree languages

Theorem $MSOT^{-1}(\text{REGT}) \subseteq \text{REGT}$

ψ closed MSO formula

Same as composition!

In ψ

- replace $\text{lab}_\sigma(x)$ by $\bigvee_{c \in C} \{g\text{'s definition for } N_\sigma(x)\}$ and $\text{edge}_\gamma(x, y)$ by $\bigvee_{c \in C} \{g\text{'s definition for } E_{\gamma, c, d}\}$
- replace $(\exists x) \phi$ by $(\exists x)(p(x) \wedge \phi)$, $p(x)$ says that there is exactly one $\sigma \in \Sigma_2$ and one $c \in C$ s.t. $N_{\sigma, c}(x)$
- replace $(\exists X) \phi$ by $(\exists X)(\forall x)(x \in X \rightarrow p(x) \wedge \phi)$

3. Parikh's Theorem

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

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

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

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

A set $P \subseteq \mathbb{N}^k$ 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.

3. Parikh's Theorem

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

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

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

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

A set $P \subseteq \mathbb{N}^k$ 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**

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

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

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

$$\text{Par}(L2) = \text{Par}((aa)^+(bb)^+)$$

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

→ Is L3 context-free??

Parikh's Theorem
and Semilinear Sets

Context-free languages are **Parikh**

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

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

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

$$\text{Par}(L2) = \text{Par}((aa)^+(bb)^+)$$

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

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

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

4. Deciding Equivalence of MSOTT

MSOTT = MSO definable **Tree-to-Tree** Translations

→ Languages in MSOTT(REGT) are **Parikh**

Theorem Given to MSOTT's f and g , it is **decidable** whether or not they are **equivalent** (=describe the same function).

4. Deciding Equivalence of MSOTT

MSOTT = MSO definable **Tree-to-Tree** Translations

→ Languages in MSOTT(REGT) are **Parikh**

Theorem Given to MSOTT's f and g , it is **decidable** whether or not they are **equivalent** (=describe the same function).

→ Deciding equivalence is very *special property*, and has many applications (caching, verification, optimization, etc)

Note that, even for *context-free languages*, **equivalence is undecidable!**

4. Deciding Equivalence of MSOTT

MSOTT = MSO definable **Tree-to-Tree** Translations

→ Languages in MSOTT(REGT) are **Parikh**

Theorem Given to MSOTT's f and g , it is **decidable** whether or not they are **equivalent** (=describe the same function).

→ Deciding equivalence is very *special property*, and has many applications (caching, verification, optimization, etc)

Note that, even for *context-free languages*, **equivalence is undecidable!**

Corollary
Equivalence for **structural XML queries of linear size increase** is decidable!

4. Deciding Equivalence of MSOTT

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

4. Deciding Equivalence of MSOTT

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

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

Then $S \cap P$ is semilinear [**GinsburgSpanier64, difficult!**], and semilinear sets have decidable emptiness.

Alternative Proof??

4. Deciding Equivalence of MSOTT

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

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

Then $S \cap P$ is semilinear [**GinsburgSpanier64, difficult!**], and semilinear sets have decidable emptiness.

Alternative Proof??

$L3 = \{ w \in \{a, b\}^* \mid \#a(w) = \#b(w) \}$
Let R be reg. language s.t. $\text{Par}(S) = R$.

Then $R \cap L3$ is context-free ("triple-construction") and cf. languages have decidable emptiness.

4. Deciding Equivalence of MSOTT

Let's first do string output.
The Equivalence problem for MSOTS is decidable.

Given MSOTS M1 and M2:
 Add an end marker \$ (gives MSOTS N1/N2)
 $N1(s) = M1(s) \$$
 $N2(s) = M2(s) \$$

M1 is equiv. to M2
 iff $\neg(\exists s \exists n \ N1(s)/n \neq N2(s)/n)$

4. Deciding Equivalence of MSOTT

Let's first do string output.
The Equivalence problem for MSOTS is decidable.

Given MSOTS M1 and M2:
 Add an end marker \$ (gives MSOTS N1/N2)
 $N1(s) = M1(s) \$$
 $N2(s) = M2(s) \$$

M1 is equiv. to M2
 iff $\neg(\exists s \exists n \ N1(s)/n \neq N2(s)/n)$

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

4. Deciding Equivalence of MSOTT

Let's first do string output.
The Equivalence problem for MSOTS is decidable.

Given MSOTS M1 and M2:
 Add an end marker \$ (gives MSOTS N1/N2)
 $N1(s) = M1(s) \$$
 $N2(s) = M2(s) \$$

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

4. Deciding Equivalence of MSOTT

Let's first do string output.
The Equivalence problem for MSOTS is decidable.

Given MSOTS M1 and M2:
 Add an end marker \$ (gives MSOTS N1/N2)
 $N1(s) = M1(s) \$$
 $N2(s) = M2(s) \$$

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

4. Deciding Equivalence of MSOTT

Let's first do string output.
The Equivalence problem for MSOTS is decidable.

Given MSOTS M1 and M2:
 Add an end marker \$ (gives MSOTS N1/N2)
 $N1(s) = M1(s) \$$
 $N2(s) = M2(s) \$$

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 $\neg(\dots \wedge \text{Par}(\text{Out}(N12^{a,b})) \cap \{(n, n) \mid n \geq 1\} \neq \emptyset?)$ **Decidable By Lemma.** □

4. Deciding Equivalence of MSOTT

The Equivalence problem for MSOTT is decidable.

Do you remember the MSOTS transducer "PRE" that translates a tree into its yield/frontier (leaf symbols in left-to-right order)?

Simply compose M1 and M2 w. the **MSOTS PRE!**
 → Gives **MSOTS N1, N2.** □

Lots of **open questions**....

- What is the complexity of the equivalence check.
- Implementation
- Can it be extended to larger classes of XML transformations?

Other things

How to find the smallest of grammar that generates a given tree?
(→ **tree compression**)

Same, but for graph grammars! → "**graph compression**"
How to run algorithms on the compressed output.

MAYBE: how to **find a small MSO formula** for a given structure!?!


If you find this interesting, please consider doing your PhD @NICTA in Sydney.
→ Currently searching for PhDs to work on these most exciting topics! ☺