Using “Hilbert Methods” to decide Equivalence for Transducers

Adrien Boiret
joint work with Mikołaj Bojańczyk, Janusz Schmude, Radosław Piórkowski

March 27th 2018

Table of contents

Plan

1. Polynomial Register Automata
   - Zeroness Problem
   - Polynomial Reduction

2. Positive and Negative Results
   - Unranked Unordered Forests are well-behaved
   - Polynomials with composition are not well-behaved

Polynomial Operations

Algebra $\mathcal{A}$, operations $\phi : \mathcal{A}^k \to \mathcal{A}$

Polynomial operations: $p : \mathcal{A}^n \to \mathcal{A}$ (or $p : \mathcal{A}^n \to \mathcal{A}^m$ by product)
Combination of operations ϕ

Polynomial Operations

Algebra $\mathcal{A}$, operations $\phi : \mathcal{A}^k \to \mathcal{A}$

Polynomial operations: $p : \mathcal{A}^n \to \mathcal{A}$ (or $p : \mathcal{A}^n \to \mathcal{A}^m$ by product)
Combination of operations ϕ

Examples

$(\mathbb{Q}, +, \times)$, $p : (x, y) \mapsto (x^2 + xy, y)$

$(\mathbb{Q}[X], +, \times, -(-))$, $p : P \mapsto P(P)$

$(\Sigma^*, .)$, $p : (u, v) \mapsto u.v.u$

Register Automata

Bottom-up Tree Automata with Registers over $\mathcal{A}$ ($\mathcal{A}$-RA)
Finite ranked alphabet Σ, States Q, vector of n registers $\bar{r}$

Register Automata

Bottom-up Tree Automata with Registers over $\mathcal{A}$ ($\mathcal{A}$-RA)
Finite ranked alphabet Σ, States Q, vector of n registers $\overline{r}$

- Transition $a(q_1(\overline{r_1}), \ldots, q_k(\overline{r_k})) \rightarrow q(p(\overline{r_1}, \ldots, \overline{r_k}))$
- Final output: $q(\overline{r}) \rightarrow p(\overline{r})$

Register Automata

Bottom-up Tree Automata with Registers over $\mathcal{A}$ ($\mathcal{A}$-RA)
Finite ranked alphabet Σ, States Q, vector of n registers $\overline{r}$

- Transition $a(q_1(\overline{r_1}), \ldots, q_k(\overline{r_k})) \rightarrow q(p(\overline{r_1}, \ldots, \overline{r_k}))$
- Final output: $q(\overline{r}) \rightarrow p(\overline{r})$

Examples

Tree to String transducers are register automata on finite words with concatenation.

Macro Tree Transducers are register automata on trees with leaf substitution.

Some Register Automata

On $(\Sigma^*, .)$

Example: Visibly Pushdown

One state q of dimension 1, output $q(r) \to r$
$a() \to q(< a >< /a >)$
$g(q(r_1), q(r_2)) \to q(< g > r_1 \cdot r_2 < /g >)$

Some Register Automata

On $(\Sigma^*, .)$

Example: Visibly Pushdown

One state q of dimension 1, output $q(r) \to r$
$a() \to q(< a >< /a >)$
$g(q(r_1), q(r_2)) \to q(< g > r_1 \cdot r_2 < /g >)$

Example: Exponential Blowup

One state q of dimension 1, output $q(r) \to r$
$a() \to (A), \ \ g(q(r)) \to q(r.r)$

Plan

1. Polynomial Register Automata
   - Zeroness Problem
   - Polynomial Reduction

2. Positive and Negative Results
   - Unranked Unordered Forests are well-behaved
   - Polynomials with composition are not well-behaved

Zeroness Problem

Input:
M register automata of dimension n over a ring $\mathcal{R}$

Output:
Does M compute a constant 0 function?

Zeroness Problem

Input:
M register automata of dimension n over a ring $\mathcal{R}$

Output:
Does M compute a constant 0 function?

Theorem

If zeroness is decidable for $\mathcal{R}$-RA :

- Functionality is decidable for $\mathcal{R}$-RA
- Equivalence is decidable for functional $\mathcal{R}$-RA

Zeroness Problem

Input:
M register automata of dimension n over a ring $\mathcal{R}$

Output:
Does M compute a constant 0 function?

Theorem

If zeroness is decidable for $\mathcal{R}$-RA :

- Functionality is decidable for $\mathcal{R}$-RA
- Equivalence is decidable for functional $\mathcal{R}$-RA

Theorem

The zeroness problem is decidable for $(\mathbb{Q}, +, \times)$ and $(\mathbb{Q}[X], +, \times)$

Polynomial Closure

For $S \subseteq \mathbb{Q}^n$

$\text{pol}(S) = \{p \mid \forall s \in S,\ p(s) = 0\}$ Ideal of $\overline{\mathbb{Q}}[X_1, \ldots, X_n]$

Closure of S: $\overline{S} = \{(x_1, \ldots, x_n) \mid \forall p \in \text{pol}(S)\} \subseteq \overline{\mathbb{Q}}^n$

Polynomial Closure

For $S \subseteq \mathbb{Q}^n$

$\mathrm{pol}(S) = \{p \mid \forall s \in S,\ p(s) = 0\}$ Ideal of $\overline{\mathbb{Q}}[X_1, \dots, X_n]$

Closure of S: $\overline{S} = \{(x_1, \dots, x_n) \mid \forall p \in \mathrm{pol}(S)\} \subseteq \overline{\mathbb{Q}}^n$

Proposition

- For p polynomial, $p(S_1) \subseteq S_2 \implies p(\overline{S_1}) \subseteq \overline{S_2}$
- $\overline{X_1} \times \cdots \times \overline{X_n} \subseteq \overline{X_1 \times \cdots \times X_n}$

Polynomial Closure

For $S \subseteq \mathbb{Q}^n$

$\mathrm{pol}(S) = \{p \mid \forall s \in S,\ p(s) = 0\}$ Ideal of $\overline{\mathbb{Q}}[X_1, \ldots, X_n]$

Closure of S: $\overline{S} = \{(x_1, \ldots, x_n) \mid \forall p \in \mathrm{pol}(S)\} \subseteq \overline{\mathbb{Q}}^n$

Proposition

- For p polynomial, $p(S_1) \subseteq S_2 \implies p(\overline{S_1}) \subseteq \overline{S_2}$
- $\overline{X_1} \times \cdots \times \overline{X_n} \subseteq \overline{X_1 \times \cdots \times X_n}$

Corollary

If a set of equations $\{S \supseteq p(S_1, \ldots, S_n) \ldots\}$ has a solution $S_1, \ldots, S_n$, then $\overline{S}_1, \ldots, \overline{S}_n$ is a solution.

Decide Zeroness

Two semi-decidabilities:

Decide Zeroness

Two semi-decidabilities:

M does not compute a constant 0 function

Try runs until we find a counterexample

Decide Zeroness

Two semi-decidabilities:

M does not compute a constant 0 function

Try runs until we find a counterexample

M computes a constant 0 function

Find a closed set $\eta(q)$ of $\overline{\mathbb{Q}}^n$ for each state q of M

Decide Zeroness

Two semi-decidabilities:

M does not compute a constant 0 function

Try runs until we find a counterexample

M computes a constant 0 function

Find a closed set $\eta(q)$ of $\overline{\mathbb{Q}}^n$ for each state q of M

- If $a(q_1(\overline{r_1}), \ldots, q_k(\overline{r_k})) \to q(p(\overline{r_1}, \ldots, \overline{r_k}))$

$$\eta(q) \supseteq p(\eta(q_1), \ldots, \eta(q_k))$$

Decide Zeroness

Two semi-decidabilities:

M does not compute a constant 0 function

Try runs until we find a counterexample

M computes a constant 0 function

Find a closed set $\eta(q)$ of $\overline{\mathbb{Q}}^n$ for each state q of M

- If $a(q_1(\overline{r_1}), \ldots, q_k(\overline{r_k})) \to q(p(\overline{r_1}, \ldots, \overline{r_k}))$

$$\eta(q) \supseteq p(\eta(q_1), \ldots, \eta(q_k))$$

- Final output $q(\bar{r}) \to p(\bar{r})$

$$\{0\} \supseteq p(\eta(q))$$

Semi-Decide Zeroness

M computes a constant 0 function

Find a closed set $\eta(q)$ of $\overline{\mathbb{Q}}^n$ for each state q of M

- If $a(q_1(\overline{r_1}), \ldots, q_k(\overline{r_k})) \to q(p(\overline{r_1}, \ldots, \overline{r_k}))$

$$\eta(q) \supseteq p(\eta(q_1), \ldots, \eta(q_k))$$

- Final output $q(\bar{r}) \to p(\bar{r})$

$$\{0\} \supseteq p(\eta(q))$$

Semi-Decide Zeroness

M computes a constant 0 function

Find an ideal $\eta(q) =< P_1, \ldots, P_m >$ for each state q of M
Ideals enumerable thanks to Hilbert's Theorem

- If $a(q_1(\overline{r_1}), \ldots, q_k(\overline{r_k})) \to q(p(\overline{r_1}, \ldots, \overline{r_k}))$

$$\eta(q) \supseteq p(\eta(q_1), \ldots, \eta(q_k))$$

- Final output $q(\bar{r}) \to p(\bar{r})$

$$\{0\} \supseteq p(\eta(q))$$

Semi-Decide Zeroness

M computes a constant 0 function

Find an ideal $\eta(q) =< P_1, \ldots, P_m >$ for each state q of M

Ideals enumerable thanks to Hilbert's Theorem

- If $a(q_1(\overline{r_1}), \ldots, q_k(\overline{r_k})) \to q(p(\overline{r_1}, \ldots, \overline{r_k}))$

  $$\eta(q) \supseteq p(\eta(q_1), \ldots, \eta(q_k))$$

  Inclusion testable thanks to Groebner Bases

- Final output $q(\overline{r}) \to p(\overline{r})$

  $$\{0\} \supseteq p(\eta(q))$$

  Inclusion testable thanks to Groebner Bases

Plan

1. Polynomial Register Automata
   - Zeroness Problem
   - Polynomial Reduction

2. Positive and Negative Results
   - Unranked Unordered Forests are well-behaved
   - Polynomials with composition are not well-behaved

Polynomial Reduction

Algebras $(\mathcal{A}, \phi_1, \ldots, \phi_k)$, and $(\mathcal{B}, \psi_1, \ldots, \psi_m)$,

Polynomial Reduction

Algebras $(\mathcal{A}, \phi_1, \ldots, \phi_k)$, and $(\mathcal{B}, \psi_1, \ldots, \psi_m)$,

Polynomial reduction: $f : \mathcal{A} \to \mathcal{B}^n$ injective function such that every ϕ_i in $\mathcal{A}$ is represented by a polynomial p_i in $\mathcal{B}$

$$\forall \phi_i \exists p_i \text{ polynomial in } \mathcal{B} \mid f(\phi_i(a_1, \ldots, a_n)) = p_i(f(a_1), \ldots, f(a_n))$$

Polynomial Reduction

Algebras $(\mathcal{A}, \phi_1, \ldots, \phi_k)$, and $(\mathcal{B}, \psi_1, \ldots, \psi_m)$,

Polynomial reduction: $f : \mathcal{A} \to \mathcal{B}^n$ injective function such that every ϕ_i in $\mathcal{A}$ is represented by a polynomial p_i in $\mathcal{B}$

$$\forall \phi_i \exists p_i \text{ polynomial in } \mathcal{B} \mid f(\phi_i(a_1, \ldots, a_n)) = p_i(f(a_1), \ldots, f(a_n))$$

$\mathcal{A} \preceq_\pi \mathcal{B}$: $\mathcal{A}$ reduces to $\mathcal{B}$
$\preceq_\pi$ is a transitive relation

From Words to Integers

We reduce words on alphabet $\Sigma = \{0, 1\}$ into pairs of integers:

$(\Sigma^*, .)$	$(\mathbb{Z}, +, -, \times)$
u	$([u]_2, 2^{\|u\|})$

From Words to Integers

We reduce words on alphabet $\Sigma = \{0, 1\}$ into pairs of integers:

$(\Sigma^*, .)$	$(\mathbb{Z}, +, -, \times)$
u	$([u]_2, 2^{\lvert u \rvert})$
$u.v$	$([u]_2 \times 2^{\lvert v \rvert} + [v]_2, 2^{\lvert u+v \rvert})$

$$01011.011 = 01011000 + 011$$

From Words to Integers

We reduce words on alphabet $\Sigma = \{0, 1\}$ into pairs of integers:

$(\Sigma^*, .)$	$(\mathbb{Z}, +, -, \times)$
u	$([u]_2, 2^{\lvert u \rvert})$
$u.v$	$([u]_2 \times 2^{\lvert v \rvert} + [v]_2, 2^{\lvert u+v \rvert})$

$$01011.011 = 01011000 + 011$$

Any Word-RA can be reduced to a $\mathbb{Z}$-RA

Reduction of Register Automata

$\mathcal{A}$ with operation $\phi_1, \ldots, \phi_l$, $\mathcal{B}$ such that $\mathcal{A} \preceq_\pi \mathcal{B}$,
Polynomial reduction: $f : \mathcal{A} \to \mathcal{B}^n$

$\mathcal{A}$-RA M into $\mathcal{B}$-RA States Q, m $\mathcal{A}$ registers: $m \times n$ $\mathcal{B}$ registers

Reduction of Register Automata

$\mathcal{A}$ with operation $\phi_1, \ldots, \phi_l$, $\mathcal{B}$ such that $\mathcal{A} \preceq_\pi \mathcal{B}$,
Polynomial reduction: $f : \mathcal{A} \rightarrow \mathcal{B}^n$

$\mathcal{A}$-RA M into $\mathcal{B}$-RA States Q, m $\mathcal{A}$ registers: $m \times n$ $\mathcal{B}$ registers

- Transition $a(q_1(\overline{r_1}), \ldots, q_k(\overline{r_k})) \rightarrow q(p(\overline{r_1}, \ldots, \overline{r_k}))$
- Final output: $q(\overline{r}) \rightarrow p(\overline{r})$

Reduction of Register Automata

$\mathcal{A}$ with operation $\phi_1, \ldots, \phi_l$, $\mathcal{B}$ such that $\mathcal{A} \preceq_\pi \mathcal{B}$,
Polynomial reduction: $f : \mathcal{A} \to \mathcal{B}^n$

$\mathcal{A}$-RA M into $\mathcal{B}$-RA States Q, m $\mathcal{A}$ registers: $m \times n$ $\mathcal{B}$ registers

- Transition $a(q_1(\overline{r_1}), \ldots, q_k(\overline{r_k})) \to q(p(\overline{r_1}, \ldots, \overline{r_k}))$
  $\exists p'$ polynomial in $\mathcal{B} \mid f(p) = p'(f, \ldots, f)$
- Final output: $q(\overline{r}) \to p(\overline{r})$
  $\exists p'$ polynomial in $\mathcal{B} \mid f(p) = p'(f, \ldots, f)$

Decide Equivalence with "Hilbert Methods"

Theorem - Reduction

If $\mathcal{A} \preceq_\pi \mathcal{B}$ and functionality (equivalence) is decidable for register automata on $\mathcal{B}$, then functionality (equivalence) is decidable for register automata on $\mathcal{A}$.

Decide Equivalence with "Hilbert Methods"

Theorem - Reduction

If $\mathcal{A} \preceq_\pi \mathcal{B}$ and functionality (equivalence) is decidable for register automata on $\mathcal{B}$, then functionality (equivalence) is decidable for register automata on $\mathcal{A}$.

Theorem - "Hilbert Methods"

- Functionality is decidable for $(\mathbb{Q}[X], +, \times)$-RA
- Equivalence is decidable for functional $(\mathbb{Q}[X], +, \times)$-RA

Decide Equivalence with "Hilbert Methods"

Theorem - Reduction

If $\mathcal{A} \preceq_\pi \mathcal{B}$ and functionality (equivalence) is decidable for register automata on $\mathcal{B}$, then functionality (equivalence) is decidable for register automata on $\mathcal{A}$.

Theorem - "Hilbert Methods"

- Functionality is decidable for $(\mathbb{Q}[X], +, \times)$-RA
- Equivalence is decidable for functional $(\mathbb{Q}[X], +, \times)$-RA

Corollary (SMK 2015)

Equivalence is decidable for Tree to String transducers.

Plan

1. Polynomial Register Automata
   - Zeroness Problem
   - Polynomial Reduction

2. Positive and Negative Results
   - Unranked Unordered Forests are well-behaved
   - Polynomials with composition are not well-behaved

Plan

Unranked Unordered Forests

Finite alphabet Σ, Algebra of forests $\mathcal{H}_\Sigma$.

Operations: Binary concatenation $\cdot$ (associative and commutative), for each $a \in \Sigma$, place a forest under a root of label a: unary root_a.

Unranked Unordered Forests

Finite alphabet Σ, Algebra of forests $\mathcal{H}_\Sigma$.

Operations: Binary concatenation $\cdot$ (associative and commutative), for each $a \in \Sigma$, place a forest under a root of label a: unary root_a.

Can be modeled with alphabet with one symbol.

$$\mathrm{root}_{a_i}(h) \to \mathrm{root}^i(\mathrm{root}\{\} \cdot \mathrm{root}(h))$$

Reduction from Forests to Polynomials

$\mathcal{H}_\Sigma$	$(\mathbb{Q}[X], +, \times)$
$h.h'$	$f(h) \times f(h')$

Reduction from Forests to Polynomials

$\mathcal{H}_\Sigma$	$(\mathbb{Q}[X], +, \times)$
$h.h'$	$f(h) \times f(h')$
$\mathrm{root}(h)$	$2 + X \times f(h)$

Reduction from Forests to Polynomials

$\mathcal{H}_\Sigma$	$(\mathbb{Q}[X], +, \times)$
$h.h'$	$f(h) \times f(h')$
$\mathrm{root}(h)$	$2 + X \times f(h)$

Eisenberg Criterion

For $P(X) = a_0 + \cdots + a_k X^k \in \mathbb{Q}[X]$, if $\exists n$ prime such that

$$\forall 0 \leqslant i < k,\ n | a_i, \quad n \not| a_k, \quad n^2 \not| a_0$$

then $P(X)$ is irreducible in $\mathbb{Q}[X]$

0-1 Contexts

Unary alphabet, one variable y
Algebra: Forests contexts with one or no y
Operations: Concatenation, root, substitution $h[y \leftarrow h']$

0-1 Contexts

Unary alphabet, one variable y
Algebra: Forests contexts with one or no y
Operations: Concatenation, root, substitution $h[y \leftarrow h']$

Encoded as pairs of polynomials (P_Σ, P_y)

- P_Σ is the encoding of the tree without y
- P_y is an encoding of "where" y is

0-1 Contexts

Unary alphabet, one variable y
Algebra: Forests contexts with one or no y
Operations: Concatenation, root, substitution $h[y \leftarrow h']$

Encoded as pairs of polynomials (P_Σ, P_y)

- P_Σ is the encoding of the tree without y
- P_y is an encoding of "where" y is

Can be extended to $0 - n$ with the same methods

Typed Algebra-RA

Typed algebra $\mathcal{A} = \mathcal{A}_1 \sqcup \cdots \sqcup \mathcal{A}_n$
Operations $\phi : \mathcal{A}_{i_1} \times \cdots \times \mathcal{A}_{i_k} \to \mathcal{A}_j$

Typed Algebra-RA

Typed algebra $\mathcal{A} = \mathcal{A}_1 \sqcup \cdots \sqcup \mathcal{A}_n$
Operations $\phi : \mathcal{A}_{i_1} \times \cdots \times \mathcal{A}_{i_k} \to \mathcal{A}_j$

$\mathcal{A}$-RA with n registers: each state q has typed registers $\overline{r}$
Type $\mathcal{A}_{i_1} \times \cdots \times \mathcal{A}_{i_n}$

Typed Algebra-RA

Typed algebra $\mathcal{A} = \mathcal{A}_1 \sqcup \cdots \sqcup \mathcal{A}_n$
Operations $\phi : \mathcal{A}_{i_1} \times \cdots \times \mathcal{A}_{i_k} \to \mathcal{A}_j$

$\mathcal{A}$-RA with n registers: each state q has typed registers $\overline{r}$
Type $\mathcal{A}_{i_1} \times \cdots \times \mathcal{A}_{i_n}$

- Transition $a(q_1(\overline{r_1}), \ldots, q_k(\overline{r_k})) \to q(p(\overline{r_1}, \ldots, \overline{r_k}))$
- Final output: $q(\overline{r}) \to p(\overline{r})$

Typed Algebra-RA

Typed algebra $\mathcal{A} = \mathcal{A}_1 \sqcup \cdots \sqcup \mathcal{A}_n$
Operations $\phi : \mathcal{A}_{i_1} \times \cdots \times \mathcal{A}_{i_k} \to \mathcal{A}_j$

$\mathcal{A}$-RA with n registers: each state q has typed registers $\overline{r}$
Type $\mathcal{A}_{i_1} \times \cdots \times \mathcal{A}_{i_n}$

- Transition $a(q_1(\overline{r_1}), \ldots, q_k(\overline{r_k})) \to q(p(\overline{r_1}, \ldots, \overline{r_k}))$
- Final output: $q(\overline{r}) \to p(\overline{r})$

Polynomial reduction: $f : \mathcal{A} \to \mathcal{B}^n$
If a, a' same type in $\mathcal{A}$ **then** $f(a), f(a')$ same type in $\mathcal{B}$

Reduction from Contexts to Polynomials

Extend reduction to $\mathbb{Q}[X, Y]$ with $Y \leftarrow P$:

$(\text{Contexts}, \cdot, \text{root}, y \leftarrow h)$	$\mathbb{Q}[X, Y], +, \times, Y \leftarrow P$
$h.h'$	$f(h) \times f(h')$
$\text{root}(h)$	$2 + X \times f(h)$

Reduction from Contexts to Polynomials

Extend reduction to $\mathbb{Q}[X, Y]$ with $Y \leftarrow P$:

$(\text{Contexts}, \cdot, \text{root}, y \leftarrow h)$	$\mathbb{Q}[X, Y], +, \times, Y \leftarrow P$
y	Y
$h.h'$	$f(h) \times f(h')$
$\text{root}(h)$	$2 + X \times f(h)$
$h[y \leftarrow h']$	$f(h)[Y \leftarrow f(h')]$

Finite Degree Composition as $+, \times$

Polynomial of $\mathbb{Q}[X, Y]$ for a 0-1 context
y appears at most once: $P(X) + Y.P'(X)$

Finite Degree Composition as $+, \times$

Polynomial of $\mathbb{Q}[X, Y]$ for a 0-1 context
y appears at most once: $P(X) + Y.P'(X)$

Replace Y of $P(X) + Y.P'(X)$ by $Q(X) + Y.Q'(X)$

$$(P + P'.Q)(X) + Y.(P'.Q')(X)$$

Finite Degree Composition as $+, \times$

Polynomial of $\mathbb{Q}[X, Y]$ for a 0-1 context
y appears at most once: $P(X) + Y.P'(X)$

Replace Y of $P(X) + Y.P'(X)$ by $Q(X) + Y.Q'(X)$

$$(P + P'.Q)(X) + Y.(P'.Q')(X)$$

Can **encode** $P(X) + Y.P'(X)$ as (P, P') in $(\mathbb{Q}[X], +, \times)$

Finite Degree Composition as $+, \times$

Polynomial of $\mathbb{Q}[X, Y]$ for a 0-1 context
y appears at most once: $P(X) + Y.P'(X)$

Replace Y of $P(X) + Y.P'(X)$ by $Q(X) + Y.Q'(X)$

$$(P + P'.Q)(X) + Y.(P'.Q')(X)$$

Can **encode** $P(X) + Y.P'(X)$ as (P, P') in $(\mathbb{Q}[X], +, \times)$

$$\begin{aligned} P(X, Y, Z) = P_0(X) + \quad & Y.P_1(X) + \quad & Y^2.P_2(X) \\ + \quad & Z.P_3(X) + \quad & Z^2.P_4(X) \\ & + \quad & Y.Z.P_5(X) \end{aligned}$$

Forests Register Automata

Theorem - Forests Register Automata

Functionality and equivalence are decidable for register automata on Unranked Unordered 0-n Contexts.

Forests Register Automata

Theorem - Forests Register Automata

Functionality and equivalence are decidable for register automata on Unranked Unordered 0-n Contexts.

Example: "FCNS"

One state q of dimension 1, output $q(r) \rightarrow r$

$a() \rightarrow q(\alpha()), \quad g(q(r_1), q(r_2)) \rightarrow q(\gamma(r_1) \cdot r_2)$

Forests Register Automata

Theorem - Forests Register Automata

Functionality and equivalence are decidable for register automata on Unranked Unordered 0-n Contexts.

Example: "FCNS"

One state q of dimension 1, output $q(r) \to r$
$a() \to q(\alpha()), \;\; g(q(r_1), q(r_2)) \to q(\gamma(r_1) \cdot r_2)$

Example: Vertical Exponential Blowup

One state q of dimension 1, output $q(r) \to r[y \leftarrow \alpha()]$
$a() \to (\beta(y)), \;\; g(q(r)) \to q(r[y \leftarrow r])$

MSO Unranked Unordered Forests Transformations

MSO on Unranked Unordered Forests: MSO + $\mathrm{Child}(x, y)$ $(+\mathrm{Sibling}(x, y))$

MSO Forest Transformation

MSO Unranked Unordered Forests Transformations

MSO on Unranked Unordered Forests: MSO $+ \mathrm{Child}(x, y)$ $(+\mathrm{Sibling}(x, y))$

MSO Forest Transformation

- One copy of the input $\rightarrow$ n copies of the output
  $x \rightarrow x_1, \ldots, x_n$

MSO Unranked Unordered Forests Transformations

MSO on Unranked Unordered Forests: MSO + $\mathrm{Child}(x, y)$ $(+\mathrm{Sibling}(x, y))$

MSO Forest Transformation

- One copy of the input → n copies of the output
  $x \to x_1, \ldots, x_n$
- $\mathrm{Child}_{i,j}(x, y) := \phi(x, y)$
  ϕ MSO-formula in the input

MSO Unranked Unordered Forests Transformations

MSO on Unranked Unordered Forests: MSO + $\mathrm{Child}(x, y)$ ($+\mathrm{Sibling}(x, y)$)

MSO Forest Transformation

- One copy of the input → n copies of the output
  $x \to x_1, \ldots, x_n$
- $\mathrm{Child}_{i,j}(x, y) := \phi(x, y)$
  ϕ MSO-formula in the input
- $\mathrm{Sibling}_{i,j}(x, y) := \psi(x, y)$
  ψ MSO-formula in the input

MSO Unranked Unordered Forests Transformations

MSO on Unranked Unordered Forests: MSO + $\mathrm{Child}(x, y)$ $(+\mathrm{Sibling}(x, y))$

MSO Forest Transformation

- One copy of the input → n copies of the output
  $x \rightarrow x_1, \ldots, x_n$
- $\mathrm{Child}_{i,j}(x, y) := \phi(x, y)$
  ϕ MSO-formula in the input
- $\mathrm{Sibling}_{i,j}(x, y) := \psi(x, y)$
  ψ MSO-formula in the input

Claim

Forests Register Automata can express all MSO Unranked Unordered Forests Transformations

MSO Unranked Unordered Forests Transformations

MSO Transformation
Unordered to Unordered

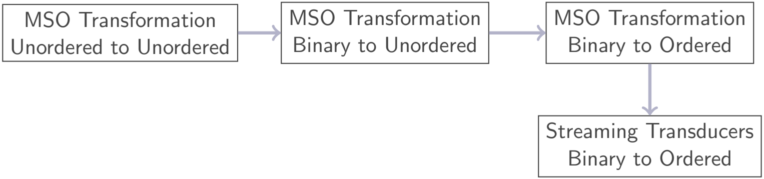
MSO Unranked Unordered Forests Transformations

MSO Transformation Unordered to Unordered → MSO Transformation Binary to Unordered

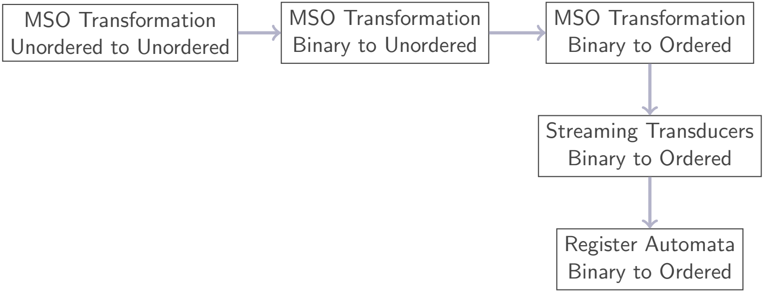
MSO Unranked Unordered Forests Transformations

MSO Transformation Unordered to Unordered → MSO Transformation Binary to Unordered → MSO Transformation Binary to Ordered

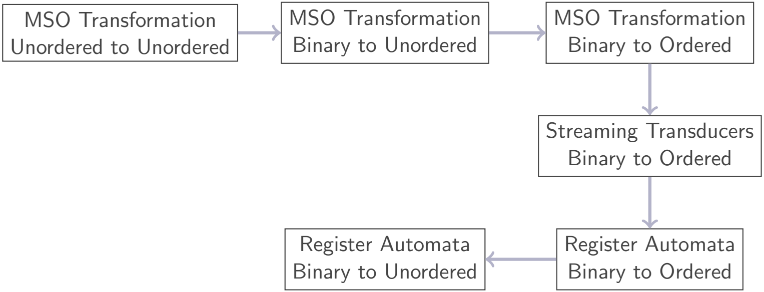
MSO Unranked Unordered Forests Transformations



MSO Unranked Unordered Forests Transformations



MSO Unranked Unordered Forests Transformations



Plan

1. Polynomial Register Automata
   - Zeroness Problem
   - Polynomial Reduction

2. Positive and Negative Results
   - Unranked Unordered Forests are well-behaved
   - Polynomials with composition are not well-behaved

Polynomials with Composition

Algebra: $(\mathbb{Q}[X], +, \times, X \to P)$
Register automata equivalence on this algebra is undecidable.

Polynomials with Composition

Algebra: $(\mathbb{Q}[X], +, \times, X \to P)$
Register automata equivalence on this algebra is undecidable.

Reduction to accessibility in 2-counter machines

Polynomials with Composition

Algebra: $(\mathbb{Q}[X], +, \times, X \to P)$
Register automata equivalence on this algebra is undecidable.

Reduction to accessibility in 2-counter machines

2-counter machines (2CM)
$Q = \{q_0, \ldots, q_n\}$, configuration $(q, c_1, c2)$, $c_1, c_2 \in \mathbb{N}$
$\delta : (q, b_1, b_2) \to (q', -1/0/+1, -1/0/+1)$, $\;b_i : \; c_i = 0?$
Initial: $(q_0, 0, 0)$, Question: can we reach q_n?

This problem is undecidable.

Registers for 2-Counter Machine Simulation

Register word automata with alphabet δ

Reading $(q, b_1, b_2) \rightarrow (q', d_1, d_2)$: make the step in the 2CM

1-state Register Automaton:

Registers for 2-Counter Machine Simulation

Register word automata with alphabet δ
Reading $(q, b_1, b_2) \rightarrow (q', d_1, d_2)$: make the step in the 2CM
1-state Register Automaton:

Registers to encode a 2CM configuration

- r_q: if the current state is q_i, then $r_q = i$
- r_1, r_2: if c_i is at value j, then $r_i = j$

Registers for 2-Counter Machine Simulation

Register word automata with alphabet δ
Reading $(q, b_1, b_2) \to (q', d_1, d_2)$: make the step in the 2CM
1-state Register Automaton:

Registers to encode a 2CM configuration

- r_q: if the current state is q_i, then $r_q = i$
- r_1, r_2: if c_i is at value j, then $r_i = j$

Update after reading $(i, b_1, b_2) \to (j, d_1, d_2)$

- $r_q \leftarrow j$
- $r_1 \leftarrow r_1 + d_1$, $r_2 \leftarrow r_2 + d_2$

Registers for 2-Counter Machine Simulation

Register word automata with alphabet δ
Reading $(q, b_1, b_2) \to (q', d_1, d_2)$: make the step in the 2CM
1-state Register Automaton:

Registers to encode a 2CM configuration

- r_q: if the current state is q_i, then $r_q = i$
- r_1, r_2: if c_i is at value j, then $r_i = j$

Update after reading $(i, b_1, b_2) \to (j, d_1, d_2)$

- $r_q \leftarrow j$
- $r_1 \leftarrow r_1 + d_1,\ r_2 \leftarrow r_2 + d_2$

Were we **allowed** to read $(i, b_1, b_2) \to (j, d_1, d_2)$?

Witness Register

r_w witness: $r_w = 0$ iff mistake.

Witness Register

r_w witness: $r_w = 0$ iff mistake.

Update after reading $(i, b_1, b_2) \to (j, d_1, d_2)$
$r_w \leftarrow r_w . T_i(r_q) . T_1 . T_2$

Witness Register

r_w witness: $r_w = 0$ iff mistake.

Update after reading $(i, b_1, b_2) \to (j, d_1, d_2)$
$r_w \leftarrow r_w . T_i(r_q) . T_1 . T_2$

Test for states: $T_i = \prod_{0 \leqslant j \leqslant n}^{i \neq j} X - j : \begin{cases} \neq 0 \text{ if } X = i \\ = 0 \text{ if } X \neq i \end{cases}$

Witness Register

r_w witness: $r_w = 0$ iff mistake.

Update after reading $(i, b_1, b_2) \to (j, d_1, d_2)$
$r_w \leftarrow r_w . T_i(r_q) . T_1 . T_2$

Test for states: $T_i = \prod_{0 \leqslant j \leqslant n}^{i \neq j} X - j : \begin{cases} \neq 0 \text{ if } X = i \\ = 0 \text{ if } X \neq i \end{cases}$

Test that $c_i \neq 0$: $T_i = r_i$

Test that $c_i = 0$: $T_i = \prod_{1 \leqslant j \leqslant k} X - j : \begin{cases} \neq 0 \text{ if } X = 0 \\ = 0 \text{ if } X \neq 0 \end{cases}$

Witness Register

r_w witness: $r_w = 0$ iff mistake.

Update after reading $(i, b_1, b_2) \rightarrow (j, d_1, d_2)$
$r_w \leftarrow r_w . T_i(r_q) . T_1 . T_2$

Test for states: $T_i = \prod_{0 \leqslant j \leqslant n}^{i \neq j} X - j : \begin{cases} \neq 0 \text{ if } X = i \\ = 0 \text{ if } X \neq i \end{cases}$

Test that $c_i \neq 0$: $T_i = r_i$

Test that $c_i = 0$: $T_i = \prod_{1 \leqslant j \leqslant k} X - j : \begin{cases} \neq 0 \text{ if } X = 0 \\ = 0 \text{ if } X \neq 0 \end{cases}$

Test that $c_i = 0$ must be stored and updated into its own register

Polynomials with Composition are not well-behaved

Output: $P_n(r_q).r_w$

Polynomials with Composition are not well-behaved

Output: $P_n(r_q).r_w$

This register automaton produces anything other than 0 $\iff$ q_n accessible.

Theorem

The equivalence problem for register automata over $(\mathbb{Q}[X], +, \times, X \to P)$ is undecidable.

Some Parting Words

What we've done:

- Abstraction of "Hilbert Methods" to decide some Transducer Equivalence
- Positive Result: Unranked Unordered Forests
- Negative Result: $\mathbb{Q}[X]$ with composition

Some Parting Words

What we've done:

- Abstraction of "Hilbert Methods" to decide some Transducer Equivalence
- Positive Result: Unranked Unordered Forests
- Negative Result: $\mathbb{Q}[X]$ with composition

What's to come:

- New reductions? (DAG, Graphs with bounded tree width...)
- New targets for Zeroness results?
- "Stratified" registers with a dangerous operation
  Order on states, increased when using dangerous operation

That's it!

Thank you for your attention!