# Equivalence of Deterministic Tree-to-String Transducers Is Decidable

Helmut Seidl, Sebastian Maneth, Gregor Kemper

TU München, U. of Edinburgh

GT-ALGA, April 12, 2016

# Overview

Part 1: The General Setting

Part 2: Tree-to-Int Transducers

Part 3: Affine Spaces

Part 4: Polynomial Invariants

# Overview

Part 1: The General Setting

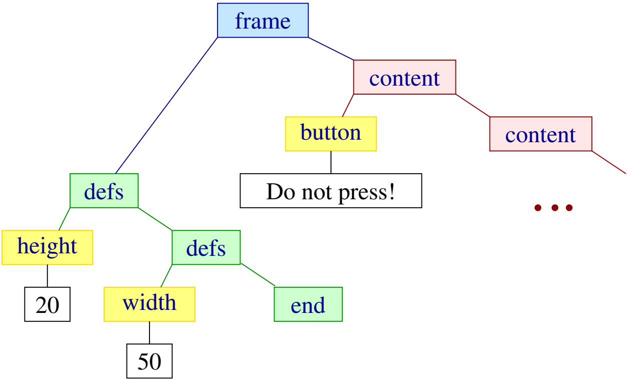
Part 2: Tree-to-Int Transducers

Part 3: Affine Spaces

Part 4: Polynomial Invariants

# Tree-to-String Translation

## Input



# Tree-to-String Translation

## Output

```
<frame height=20 width=50>
    <button>Do not press!</button>
    ...
</frame>
```

# Tree-to-String Translation

## Output

```
<frame height=20 width=50>
    <button>Do not press!</button>
    ...
</frame>
```

## Realized by:

$$
\begin{array}{lcl}
q(\text{frame}(x_1, x_2)) & \rightarrow & \texttt{<frame } q_1(x_1)q(x_2)\texttt{</frame>} \\
q_1(\text{end}) & \rightarrow & \texttt{>} \\
q_1(\text{defs}(x_1, x_2)) & \rightarrow & q(x_1)q_1(x_2) \\
q(\text{height}(x_1)) & \rightarrow & \texttt{height = } q(x_1) \\
 & \ldots &
\end{array}
$$

# Tree-to-String Translation

## Output

```
<frame height=20 width=50>
    <button>Do not press!</button>
    ...
</frame>
```

## Or realized by:

$$
\begin{array}{lcl}
q'(\text{frame}(x_1, x_2)) & \rightarrow & \texttt{<frame } q'_1(x_1)\texttt{>}\ q'(x_2)\texttt{</frame>} \\
q'_1(\text{end}) & \rightarrow & \epsilon \\
q'_1(\text{defs}(x_1, x_2)) & \rightarrow & q'(x_1) q'_1(x_2) \\
q'(\text{height}(x_1)) & \rightarrow & \texttt{height = } q'(x_1) \\
 & \ldots &
\end{array}
$$

# Tree-to-String Translation

These two translations are equivalent.

# Tree-to-String Translation

These two translations are equivalent.

Unstructured output, though, can be generated in surprisingly different ways ...

$$
\begin{array}{lcl}
q(f(x_1, x_2, x_3)) & \to & q(x_3)\ a\ q_1(x_2)\ b\ q(x_2) \\
q_1(f(x_1, x_2, x_3)) & \to & q_1(x_3) q_1(x_2) q_1(x_2)\ ba \\
q_1(e) & \to & ba \\
q(e) & \to & ab
\end{array}
$$

# Tree-to-String Translation

These two translations are equivalent.

Unstructured output, though, can be generated in surprisingly different ways ...

$$
\begin{array}{lcl}
q(f(x_1, x_2, x_3)) & \to & q(x_3)\ a\ q_1(x_2)\ b\ q(x_2) \\
q_1(f(x_1, x_2, x_3)) & \to & q_1(x_3) q_1(x_2) q_1(x_2)\ ba \\
q_1(e) & \to & ba \\
q(e) & \to & ab
\end{array}
$$

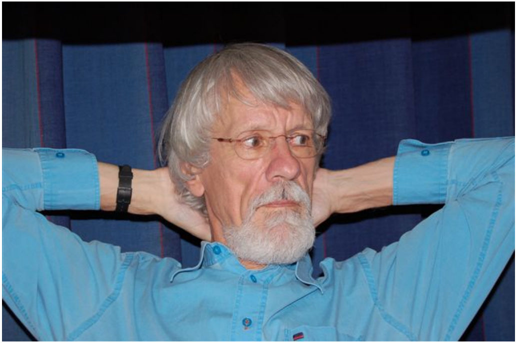
versus

$$
\begin{array}{lcl}
q'(f(x_1, x_2, x_3)) & \to & ab\ q'(x_2) q'(x_2) q'(x_3) \\
q'(e) & \to & ab
\end{array}
$$

# Related Work

problem statement

Engelfriet, 1980



# Related Work (cont.)

with monadic input

Culik II, Karhumäki, 1986
Ruohonen, 1986
Honkala, 2000

# Related Work (cont.)

| | |
|---|---|
| with monadic input | Culik II, Karhumäki, 1986<br>Ruohonen, 1986<br>Honkala, 2000 |
| MSO-definable | Engelfriet, Maneth, 2006 |
| sequential | Staworko et al., 2009 |

# Related Work (cont.)

| | |
|---|---|
| with monadic input | Culik II, Karhumäki, 1986 |
| | Ruohonen, 1986 |
| | Honkala, 2000 |
| MSO-definable | Engelfriet, Maneth, 2006 |
| sequential | Staworko et al., 2009 |
| polynomial program invariants | Letichevsky, Lvov, 1996 |
| | Müller-Olm, S., 2004 |

# General Idea

Obvious:

In-equivalence can be verified by counter example

# General Idea

Obvious:

In-equivalence can be verified by counter example

Required:

Complete proof system for equivalence

# Overview

Part 1: The General Setting

Part 1: Tree-to-Int Transducers

Part 3: Affine Spaces

Part 2: Polynomial Invariants

# Simplification

- A single transducer with states $Q = \{1, \ldots, n\}$.
- The transducer is total.

# Simplification

- A single transducer with states $Q = \{1, \ldots, n\}$.
- The transducer is total.
- There is a topdown-deterministic automaton $B$ with states $p \in P$

dom($p$) is the set of trees accepted at state $p$

# Simplification

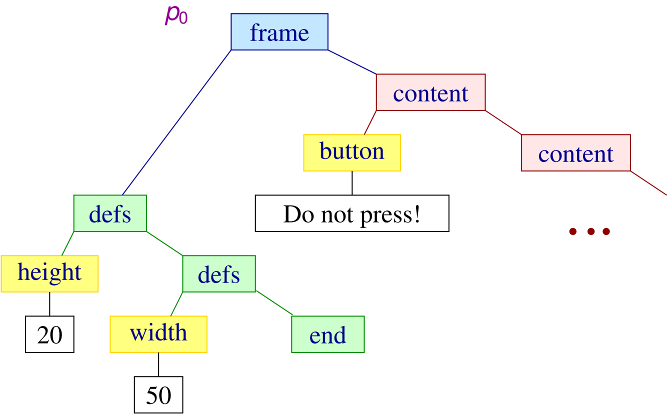
- A single transducer with states $Q = \{1, \ldots, n\}$.
- The transducer is total.
- There is a topdown-deterministic automaton $B$ with states $p \in P$

  // generalization of algebraic data type

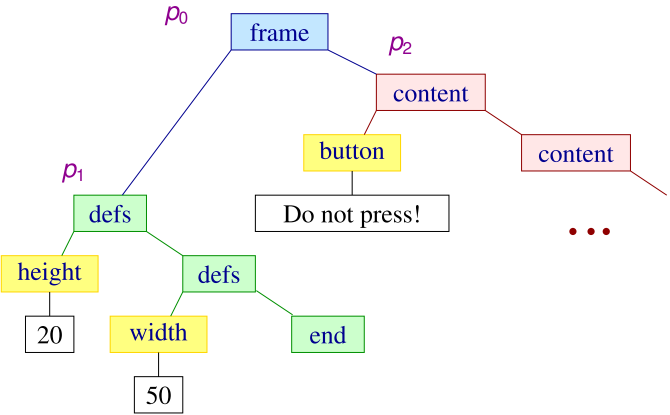
  dom($p$) is the set of trees accepted at state $p$

  // trees of type $p$

# Topdown Automaton



# Topdown Automaton



# Simplified Question

For states $q, q'$ of the transducer, $p_0 \in P$, does it hold that

$$[\![q]\!](t) = [\![q']\!](t) \qquad (t \in \mathrm{dom}(p_0))$$

# From Arbitrary Output to Ints

## Unary Output

$$q(f(x_1, x_2)) \rightarrow dd\, q_1(x_1)\, d\, q_1(x_1)\, q_2(x_2)$$

# From Arbitrary Output to Ints

## Unary Output

$$q(f(x_1, x_2)) \;\rightarrow\; dd\, q_1(x_1)\, d\, q_1(x_1)\, q_2(x_2)$$

Succinct representation: Tree-to-int transducer

$$q(f(x_1, x_2)) \;\rightarrow\; 3 + 2 \cdot q_1(x_1) + q_2(x_2)$$

# From Arbitrary Output to Ints

## Unary Output

$$q(f(x_1, x_2)) \;\to\; dd\, q_1(x_1)\, d\, q_1(x_1)\, q_2(x_2)$$

## Succinct representation:

Tree-to-int transducer

$$q(f(x_1, x_2)) \;\to\; 3 + 2 \cdot q_1(x_1) + q_2(x_2)$$

## Arbitary Output

| | | |
|---|---|---|
| letters $a, b, c, \ldots$ | $\hat{=}$ | digits $1, \ldots, h-1$ |
| string $aabc$ | $\hat{=}$ | $1 + h \cdot (1 + h \cdot (2 + h \cdot 3))$ |

# Transformation

### Wanted

Transformation of tree-to-string into tree-to-int ...

# Transformation

## Wanted

Transformation of tree-to-string into tree-to-int ...

$$q(f(x_1, x_2)) \;\to\; a\; q_1(x_1)\; b\; q_2(x_2)$$

is simulated by:

$$q(f(x_1, x_2) \quad ) \;\to$$

# Transformation

### Wanted

Transformation of tree-to-string into tree-to-int ...

$$q(f(x_1, x_2)) \;\to\; a\; q_1(x_1)\; b\; q_2(x_2)$$

is simulated by:

$$q(f(x_1, x_2), y) \;\to$$

// now with an accumulating parameter $y$
// for right context

# Transformation

## Wanted

Transformation of tree-to-string into tree-to-int ...

$$q(f(x_1, x_2)) \;\to\; a\, q_1(x_1)\; b\; q_2(x_2)$$

is simulated by:

$$q(f(x_1, x_2), y) \;\to\; 1 + 3\,\cdot$$

// now with an accumulating parameter $y$
// for right context

# Transformation

## Wanted

Transformation of tree-to-string into tree-to-int ...

$$q(f(x_1, x_2)) \;\to\; a\, q_1(x_1)\; b\; q_2(x_2)$$

is simulated by:

$$q(f(x_1, x_2), y) \;\to\; 1 + 3 \cdot q_1(x_1, \qquad\qquad\qquad\qquad )$$

// now with an accumulating parameter $y$
// for right context

# Transformation

## Wanted

Transformation of tree-to-string into tree-to-int ...

$$q(f(x_1, x_2)) \;\rightarrow\; a\; q_1(x_1)\; b\; q_2(x_2)$$

is simulated by:

$$q(f(x_1, x_2), y) \;\rightarrow\; 1 + 3 \cdot q_1(x_1, 2 + 3 \cdot q_2(x_2, y))$$

// now with an accumulating parameter $y$
// for right context

# Overview

# Equivalence of Tree-to-Int Transducers

## Idea

- The semantics of a tree $t$ can be seen as

$$[\![t]\!] = ([\![1]\!](t), \ldots, [\![n]\!](t)) \in \mathbb{Q}^n$$

# Equivalence of Tree-to-Int Transducers

## Idea

- The semantics of a tree $t$ can be seen as

$$[\![t]\!] = ([\![1]\!](t), \ldots, [\![n]\!](t)) \in \mathbb{Q}^n$$

- For state $p$ of $B$, let $V_p = \{[\![t]\!] \mid t \in \mathrm{dom}(p)\}$.

# Equivalence of Tree-to-Int Transducers

## Idea

- The semantics of a tree $t$ can be seen as

$$[\![t]\!] = ([\![1]\!](t), \ldots, [\![n]\!](t)) \in \mathbb{Q}^n$$

- For state $p$ of $B$, let $V_p = \{[\![t]\!] \mid t \in \mathrm{dom}(p)\}$.
- Consider $H(\mathbf{v}) = \mathbf{v}_q - \mathbf{v}_{q'}$.
- The following statements are equivalent:
  1. $q, q'$ agree on inputs from $\mathcal{L}(B)$
  2. $H(\mathbf{v}) = 0 \qquad (\mathbf{v} \in V_{p_0})$

# Equivalence of Tree-to-Int Transducers

## Idea

- The semantics of a tree $t$ can be seen as

$$[\![t]\!] = ([\![1]\!](t), \ldots, [\![n]\!](t)) \in \mathbb{Q}^n$$

- For state $p$ of $B$, let $V_p = \{[\![t]\!] \mid t \in \mathrm{dom}(p)\}$.
- Consider $H(\mathbf{v}) = \mathbf{v}_q - \mathbf{v}_{q'}$.
- The following statements are equivalent:

  1. $q, q'$ agree on inputs from $\mathcal{L}(B)$
  2. $H(\mathbf{v}) = 0 \qquad (\mathbf{v} \in V_{p_0})$
  3. $H(\mathbf{v}) = 0 \qquad (\mathbf{v} \in \mathrm{Aff}(V_{p_0}))$

     // affine closure

# Computing Affine Closures

Define

$$\begin{array}{rcl} [\![f]\!](\mathbf{x}_1,\ldots,\mathbf{x}_k) & = & ([\![T_1]\!](\mathbf{x}),\ldots,[\![T_n]\!](\mathbf{x})) \quad \text{where} \\ & & q(f(x_1,\ldots,x_k)) \rightarrow T_q \end{array}$$

# Computing Affine Closures

Define

$$\begin{array}{rcll} [\![f]\!](\mathbf{x}_1,\ldots,\mathbf{x}_k) & = & ([\![T_1]\!](\mathbf{x}),\ldots,[\![T_n]\!](\mathbf{x})) & \text{where} \\ & & q(f(x_1,\ldots,x_k)) \;\rightarrow\; T_q & \text{and} \end{array}$$

$$[\![3\cdot q(x_1)+q'(x_1)+2\cdot q'(x_2)+5]\!](\mathbf{x}_1,\mathbf{x}_2) \quad =$$

# Computing Affine Closures

Define

$$\begin{aligned}
[\![f]\!](\mathbf{x}_1, \ldots, \mathbf{x}_k) \quad &= \quad ([\![T_1]\!](\mathbf{x}), \ldots, [\![T_n]\!](\mathbf{x})) \quad \text{where} \\
& \quad\;\; q(f(x_1, \ldots, x_k)) \;\to\; T_q \quad \text{and}
\end{aligned}$$

$$\begin{aligned}
[\![3 \cdot q(x_1) + q'(x_1) + 2 \cdot q'(x_2) + 5]\!](\mathbf{x}_1, \mathbf{x}_2) \quad &= \\
3 \cdot \mathbf{x}_{1q} + \mathbf{x}_{1q'} + 2 \cdot \mathbf{x}_{2q'} + 5 &
\end{aligned}$$

# Computing Affine Closures

Define

$$\begin{array}{rcll} [\![f]\!](\mathbf{x}_1,\ldots,\mathbf{x}_k) & = & ([\![T_1]\!](\mathbf{x}),\ldots,[\![T_n]\!](\mathbf{x})) & \text{where} \\ & & q(f(x_1,\ldots,x_k)) \;\to\; T_q & \text{and} \end{array}$$

$$[\![3\cdot q(x_1) + q'(x_1) + 2\cdot q'(x_2) + 5]\!](\mathbf{x}_1,\mathbf{x}_2) \quad = \quad 3\cdot \mathbf{x}_{1q} + \mathbf{x}_{1q'} + 2\cdot \mathbf{x}_{2q'} + 5$$

$\Longrightarrow$ $\quad [\![f]\!] : \mathbb{Q}^n \times \ldots \times \mathbb{Q}^n \to \mathbb{Q}^n$ is affine.

# Computing Affine Closures (cont.)

## Consequence

$V'_p = \mathrm{Aff}(V_p)$ is the least solution of:

$$V'_p \supseteq [\![f]\!](V'_{p_1}, \ldots, V'_{p_k})$$

$((p, f) \mapsto p_1 \ldots p_k$ transition of $B)$ over the complete lattice of affine sub-spaces of $\mathbb{Q}^n$ !

# Computing Affine Closures (cont.)

## Consequence

$V'_p = \mathsf{Aff}(V_p)$ is the least solution of:

$$V'_p \supseteq [\![f]\!](V'_{p_1}, \ldots, V'_{p_k})$$

($(p, f) \mapsto p_1 \ldots p_k$ transition of $B$) over the complete lattice of affine sub-spaces of $\mathbb{Q}^n$ !

## Theorem

- Equivalence of total tree-to-int transducers relative to some $B$ is decidable in polynomial time.

# Computing Affine Closures (cont.)

## Consequence

$V'_p = \mathrm{Aff}(V_p)$ is the least solution of:

$$V'_p \supseteq [\![f]\!](V'_{p_1}, \ldots, V'_{p_k})$$

($(p, f) \mapsto p_1 \ldots p_k$ transition of $B$) over the complete lattice of affine sub-spaces of $\mathbb{Q}^n$ !

## Theorem

- Equivalence of total tree-to-int transducers relative to some $B$ is decidable in polynomial time.
- In-Equivalence of linear tree-to-string transducers is decidable in randomized polynomial time.

# Overview

Part 1: The General Setting

Part 2: Tree-to-Int Transducers

Part 3: Affine Spaces

Part 4: Polynomial Invariants

# Tree-to-int Transducers with Parameters

$$
\begin{array}{lcl}
q(f(x_1, x_2), y) & \rightarrow & q_1(x_1, q_1(x_2, 1)) \\
q_1(a(x_1), y) & \rightarrow & y + q_1(x_1, y) \\
q_1(e, y) & \rightarrow & 0
\end{array}
$$

# Tree-to-int Transducers with Parameters

$$
\begin{array}{lcl}
q(f(x_1, x_2), y) & \rightarrow & q_1(x_1, q_1(x_2, 1)) \\
q_1(a(x_1), y) & \rightarrow & y + q_1(x_1, y) \\
q_1(e, y) & \rightarrow & 0 \\
q'(f(x_1, x_2), y) & \rightarrow & q_1(x_2, q_1(x_1, 1))
\end{array}
$$

# Tree-to-int Transducers with Parameters

$$\begin{array}{lcl} q(f(x_1,x_2),y) & \to & q_1(x_1,q_1(x_2,1)) \\ q_1(a(x_1),y) & \to & y + q_1(x_1,y) \\ q_1(e,y) & \to & 0 \\ q'(f(x_1,x_2),y) & \to & q_1(x_2,q_1(x_1,1)) \end{array}$$

The semantics of a tree $t$ is a vector

$$[\![t]\!] : (\mathbb{Q}^l \to \mathbb{Q})^n$$

of affine functions in the parameters

$\Longrightarrow$

can be represented by a matrix $([\![t]\!]_{qi}) \in \mathbb{Q}^{n\times(l+1)}$

# The Semantics of Constructors

$$[\![f]\!] : (\mathbb{Q}^{n\times(l+1)} \times \ldots \times \mathbb{Q}^{n\times(l+1)}) \to \mathbb{Q}^{n\times(l+1)}$$

thus is of the form:

$$([\![f]\!](\mathbf{x}_1, \ldots, \mathbf{x}_k))_{qj} = \text{polynomial in the } \mathbf{x}_{iq'j'}$$

# In the Example

$$\begin{array}{lcl} q(f(x_1, x_2), y) & \rightarrow & q_1(x_1, q_1(x_2, 1)) \\ q_1(a(x_1), y) & \rightarrow & y + q_1(x_1, y) \\ q_1(e, y) & \rightarrow & 0 \\ q'(f(x_1, x_2), y) & \rightarrow & q_1(x_2, q_1(x_1, 1)) \end{array}$$

# In the Example

$$
\begin{array}{lcl}
q(f(x_1, x_2), y) & \to & q_1(x_1, q_1(x_2, 1)) \\
q_1(a(x_1), y) & \to & y + q_1(x_1, y) \\
q_1(e, y) & \to & 0 \\
q'(f(x_1, x_2), y) & \to & q_1(x_2, q_1(x_1, 1))
\end{array}
$$

$$
\begin{array}{lcl}
([\![f]\!](\mathbf{x}_1, \mathbf{x}_2))_{q0} & = & \mathbf{x}_{1q_10} + \mathbf{x}_{1q_11} \cdot (\mathbf{x}_{2q_10} + \mathbf{x}_{2q_11} \cdot 1) \\
([\![f]\!](\mathbf{x}_1, \mathbf{x}_2))_{q'0} & = & \mathbf{x}_{2q_10} + \mathbf{x}_{2q_11} \cdot (\mathbf{x}_{1q_10} + \mathbf{x}_{1q_11} \cdot 1)
\end{array}
$$

# Polynomial Invariant

polynomial equality:

$$\mathbf{z}_{q1} \cdot \mathbf{z}_{q'1} \cdot \mathbf{z}_{q'0} - 2 \cdot \mathbf{z}_{q0} + 3 \doteq 0$$

# Polynomial Invariant

polynomial equality:

$$\mathbf{z}_{q1} \cdot \mathbf{z}_{q'1} \cdot \mathbf{z}_{q'0} - 2 \cdot \mathbf{z}_{q0} + 3 \doteq 0$$

$r_1 \doteq 0 \wedge \ldots \wedge r_m \doteq 0$ invariant at $p$ iff

$$r_1(\llbracket t \rrbracket) = \ldots = r_m(\llbracket t \rrbracket) = 0 \qquad (t \in \mathbf{dom}(p))$$

# Polynomial Invariant

polynomial equality:

$$\mathbf{z}_{q1} \cdot \mathbf{z}_{q'1} \cdot \mathbf{z}_{q'0} - 2 \cdot \mathbf{z}_{q0} + 3 \doteq 0$$

$r_1 \doteq 0 \wedge \ldots \wedge r_m \doteq 0$ invariant at $p$ iff

$$r_1(\llbracket t \rrbracket) = \ldots = r_m(\llbracket t \rrbracket) = 0 \qquad (t \in \mathbf{dom}(p))$$

can be described by polynomial ideals ...

# Polynomial Ideals: A Primer

$R$ ring. $I \subseteq R$ ideal, if

- $a + b \in I$ whenever $a, b \in I$;
- $r \cdot a \in I$ whenever $a \in I$ and $r \in R$.

# Polynomial Ideals: A Primer

$R$ ring. $I \subseteq R$ ideal, if

- $a + b \in I$ whenever $a, b \in I$;
- $r \cdot a \in I$ whenever $a \in I$ and $r \in R$.

$I$ is finitely generated, if

$$I = \langle a_1, \ldots, a_s \rangle_R = \{\sum_{i=1}^{s} r_i \cdot a_i \mid r_i \in R\}$$

# Polynomial Ideals: A Primer

$R$ ring. $I \subseteq R$ ideal, if

- $a + b \in I$ whenever $a, b \in I$;
- $r \cdot a \in I$ whenever $a \in I$ and $r \in R$.

$I$ is finitely generated, if

$$I = \langle a_1, \ldots, a_s \rangle_R = \{ \sum_{i=1}^{s} r_i \cdot a_i \mid r_i \in R \}$$

$R = \mathbb{Q}[\mathbf{z}]$ polynomial ring

# Polynomial Ideals — Basis Theorem

David Hilbert (1890)

Every ideal of $\mathbb{Q}[\mathbf{z}]$ is finitely generated !

# Consequence

- Polynomial Invariants can be represented by polynomial ideals!
- Finite conjunctions suffice!

# Consequence

- Polynomial Invariants can be represented by polynomial ideals!
- Finite conjunctions suffice!
- There are effective algorithms for
  - membership
  - inclusion
  - equality

# Inductive Invariant

Notation: $r_{qj}^{(f)} = ([\![f]\!](\mathbf{x}_1, \ldots, \mathbf{x}_k))_{qj}$

$\mathbf{z}$ fresh set of variables

# Inductive Invariant

Notation: $r_{qj}^{(f)} = (\llbracket f \rrbracket(\mathbf{x}_1, \ldots, \mathbf{x}_k))_{qj}$

$\mathbf{z}$ fresh set of variables

$p \mapsto I_p \subseteq \mathbb{Q}[\mathbf{z}]$

# Inductive Invariant

Notation: $r_{qj}^{(f)} = ([\![f]\!](\mathbf{x}_1, \ldots, \mathbf{x}_k))_{qj}$

$\mathbf{z}$ fresh set of variables

$p \mapsto I_p \subseteq \mathbb{Q}[\mathbf{z}]$ is inductive if for $p \to f(p_1, \ldots, p_k)$,

$$I_p \subseteq \{ r \in \mathbb{Q}[\mathbf{z}] \mid r[r^{(f)}/\mathbf{z}] \in I_{p_1} \oplus \ldots \oplus I_{p_k} \}$$

holds.

# Inductive Invariant

Notation: $r^{(f)}_{qj} = (\llbracket f \rrbracket(\mathbf{x}_1, \ldots, \mathbf{x}_k))_{qj}$

$\mathbf{z}$ fresh set of variables

$p \mapsto I_p \subseteq \mathbb{Q}[\mathbf{z}]$ is inductive if for $p \to f(p_1, \ldots, p_k)$,

$$I_p \subseteq \{r \in \mathbb{Q}[\mathbf{z}] \mid r[r^{(f)}/\mathbf{z}] \in I_{p_1}(\mathbf{x}_1) \oplus \ldots \oplus I_{p_k}(\mathbf{x}_k) \}$$

holds.

# Inductive Invariant

**Notation:** $r^{(f)}_{qj} = (\llbracket f \rrbracket(\mathbf{x}_1, \ldots, \mathbf{x}_k))_{qj}$

$\mathbf{z}$ fresh set of variables

$p \mapsto I_p \subseteq \mathbb{Q}[\mathbf{z}]$ is inductive if for $p \to f(p_1, \ldots, p_k)$,

$$I_p \subseteq \{r \in \mathbb{Q}[\mathbf{z}] \mid r[r^{(f)}/\mathbf{z}] \in \langle I_{p_1}(\mathbf{x}_1)\rangle_{\mathbb{Q}[\mathbf{x}]} \oplus \ldots \oplus \langle I_{p_k}(\mathbf{x}_k)\rangle_{\mathbb{Q}[\mathbf{x}]}\}$$

holds.

# Main Theorem

- Assume $p \mapsto I_p$ is inductive. Then for every $r \in I_p$,

$$r([\![t]\!]) = 0 \qquad (t \in \mathrm{dom}(p))$$

# Main Theorem

- Assume $p \mapsto I_p$ is inductive. Then for every $r \in I_p$,
$$r(\llbracket t \rrbracket) = 0 \qquad (t \in \mathrm{dom}(p))$$
- For $p \in P$, let
$$\bar{I}_p = \{r \in \mathbb{Q}[\mathbf{z}] \mid r(\llbracket t \rrbracket) = 0 \quad (t \in \mathrm{dom}(p))\}$$
Then $p \mapsto \bar{I}_p$ is inductive.

# Main Theorem

- Assume $p \mapsto I_p$ is inductive. Then for every $r \in I_p$,
$$r([\![t]\!]) = 0 \qquad (t \in \mathrm{dom}(p))$$
- For $p \in P$, let
$$\bar{I}_p = \{r \in \mathbb{Q}[\mathbf{z}] \mid r([\![t]\!]) = 0 \quad (t \in \mathrm{dom}(p))\}$$
Then $p \mapsto \bar{I}_p$ is inductive.

## Corollary

Let $H(\mathbf{z}) = \mathbf{z}_{q0} - \mathbf{z}_{q'0}$. Then $q, q'$ are equivalent (relative to $p_0$) iff
$$H \in I_{p_0}$$
for some inductive invariant.

# Discussion

- The best inductive invariant $p \mapsto \bar{I}_p$ is a greatest fixpoint.

  Greatest fixpoint iteration may not terminate.

# Discussion

- The best inductive invariant $p \mapsto \bar{I}_p$ is a greatest fixpoint.

  Greatest fixpoint iteration may not terminate.

- All inductive invariants, though, can be recursively enumerated!

# Discussion

- The best inductive invariant $p \mapsto \bar{I}_p$ is a greatest fixpoint.

  Greatest fixpoint iteration may not terminate.

- All inductive invariants, though, can be recursively enumerated!

- All potential counter examples can be enumerated ...

# Wrap-up

## Theorem

- Equivalence of deterministic tree-to-int transducers with parameters is decidable.

# Wrap-up

## Theorem

- Equivalence of deterministic tree-to-int transducers with parameters is decidable.
- Equvalence of general deterministic tree-to-string transducers is decidable.

# Summary

Parameters allow to encode general output alphabets by means of unaries, i.e., numbers.

# Summary

Parameters allow to encode general output alphabets by means of unaries, i.e., numbers.

Equivalence for unary transducers can be handled by means of techniques from precise program analysis, i.e., program proving.

Thank you!