Equivalence of Deterministic Tree-to-String Transducers Is Decidable

Helmut Seidl

Joint work with:

Sebastian Maneth, Gregor Kemper

PUMA, May 22, 2015
Overview

Part 1: The General Setting
Part 2: Tree-to-Int Transducers
Part 3: Affine Spaces
Part 4: Polynomial Ideals
Overview

Part 1: The General Setting
Part 2: Tree-to-Int Transducers
Part 3: Affine Spaces
Part 4: Polynomial Ideals
Tree-to-String Translation

Input

```
frame
defs
height
  20
width
  50
defs
content
button
  Do not press!
content
...```

Technische Universität München
Tree-to-String Translation

Output

```xml
<frame height=20 width=50>
  <button>Do not press!</button>
  ...
</frame>
```
Tree-to-String Translation

Output

\[
\text{<frame height=20 width=50>}
\text{<button>Do not press!</button>}
\text{...}
\text{</frame>}
\]

Realized by:

\[
\begin{align*}
q(frame(x_1, x_2)) & \rightarrow \text{<frame } q_1(x_1)q(x_2)\text{)}</frame> \\
q_1(\text{end}) & \rightarrow > \\
q_1(\text{defs}(x_1, x_2)) & \rightarrow q_2(x_1)q_1(x_2) \\
q_2(\text{height}(x_1)) & \rightarrow \text{height } = q_3(x_1) \\
\end{align*}
\]
Tree-to-String Translation (cont.)

Output

\[
\begin{align*}
\text{<frame height=20 width=50>}
\text{<button>Do not press!</button>}
\text{...}
\text{</frame>}
\end{align*}
\]

Or realized by:

\[
\begin{align*}
q(\text{frame}(x_1, x_2)) & \rightarrow \text{<frame } q_1(x_1)\text{>q}(x_2)\text{</frame>}
q_1(\text{end}) & \rightarrow \epsilon
q_1(\text{defs}(x_1, x_2)) & \rightarrow q_2(x_1)q_1(x_2)
q_2(\text{height}(x_1)) & \rightarrow \text{height } = q_3(x_1)
\text{...}
\end{align*}
\]
Question

Are these two translations equivalent?
Question

Are these two translations equivalent?

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

\[ q(f(x_1, x_2, x_3)) \rightarrow q(x_3)aq_1(x_2)bq(x_2) \]
\[ q_1(f(x_1, x_2, x_3)) \rightarrow q_1(x_3)q_1(x_2)q_1(x_2)ba \]
\[ q_1(e) \rightarrow ba \]
\[ q(e) \rightarrow ab \]
Question

Are these two translations equivalent?

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

\[
q(f(x_1, x_2, x_3)) \rightarrow q(x_3) a q_1(x_2) b q(x_2)
\]

\[
q_1(f(x_1, x_2, x_3)) \rightarrow q_1(x_3) q_1(x_2) q_1(x_2) b a
\]

\[
q_1(e) \rightarrow b a
\]

\[
q(e) \rightarrow a b
\]

versus

\[
q'(f(x_1, x_2, x_3)) \rightarrow a b q'(x_2) q'(x_2) q'(x_3)
\]

\[
q'(e) \rightarrow a b
\]
Notation

\( y_{DT} \) — det. topdown tree-to-string transducer

\[ q(f(x_1, \ldots, x_k)) \rightarrow T \]
Notation

\( y_{DT} \) — det. topdown tree-to-string transducer

\[ q(f(x_1, \ldots, x_k)) \rightarrow T \]

\( y_{MDT} \) — det. topdown macro tree-to-string transducer

\[ q(f(x_1, \ldots, x_k), y_1, \ldots, y_l) \rightarrow T \]

// initially \( y_i = \epsilon \)
Related Work

- Problem statement: Engelfriet, 1980
- MSO-definable: Engelfriet, Maneth, 2006
- With monadic input: Honkala, 2000
- Sequential: Staworko et al., 2009

Technische Universität München
Related Work

- problem statement: Engelfriet, 1980
- MSO-definable: Engelfriet, Maneth, 2006
- with monadic input: Honkala, 2000
- sequential: Staworko et al., 2009
- affine program invariants: MMO, S., 2004
- polynomial program invariants: Letichevsky, Lvov, 1996; MMO, S., 2004
Overview

Part 1: The General Setting
Part 2: Tree-to-Int Transducers
Part 3: Affine Spaces
Part 4: Polynomial Ideals
From Arbitrary to Unary Output

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

\[ q(f(x_1, x_2)) \rightarrow 3 + 2 \cdot q_1(x_1) + q_2(x_2) \]
From Arbitrary to Unary Output

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

\[ q(f(x_1, x_2)) \rightarrow 3 + 2 \cdot q_1(x_1) + q_2(x_2) \]

Encoding

letters \( a, b, c, \ldots \)  \( \triangleq \) digits \( 1, \ldots, h - 1 \)

string \( aabc \)  \( \triangleq \) \[ 1 + h \cdot (1 + h \cdot (2 + h \cdot 3)) \]
From Arbitrary to Unary Output

Unary Output

\[ q(f(x_1, x_2)) \rightarrow dd q_1(x_1) \cdot d q_1(x_1) \cdot q_2(x_2) \]

Succinct representation

\[ q(f(x_1, x_2)) \rightarrow 3 + 2 \cdot q_1(x_1) + q_2(x_2) \]

Encoding

letters \( a, b, c, \ldots \) \( \equiv \) digits \( 1, \ldots, h - 1 \)

string \( aabc \) \( \equiv \) \( 1 + h \cdot (1 + h \cdot (2 + h \cdot 3)) \)

Wanted

Transformation \( y_{DT} \rightarrow \) tree-to-int transducer
Transformation

The \textit{yDT} rule

\[ q(f(x_1, \ldots, x_k)) \rightarrow T \]

is simulated with the \textit{yMDT} rule:

\[ q(f(x_1, \ldots, x_k), y) \rightarrow [T] \]

where

\[
\begin{align*}
[\epsilon] &= y \\
[a \; T] &= a + h \cdot [T] \\
[q(x_i) \; T] &= q(x_i, [T])
\end{align*}
\]
Overview

Part 1: The General Setting
Part 2: Tree-to-Int Transducers
Part 3: Affine Spaces
Part 4: Polynomial Ideals
Unary Transducers

Simplification

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

Simplification

- A single transducer with states $Q = \{1, \ldots, n\}$.
- The transducer is total.
- There is a topdown-deterministic automaton $B$.
  - $\text{dom}(p)$ is the set of trees accepted at state $p$.
  - $\mathcal{L}(B) = \text{dom}(p_0)$ for initial state $p_0$. 
Unary Transducers

Simplification

- A single transducer with states $Q = \{1, \ldots, n\}$.
- The transducer is total.
- There is a topdown-deterministic automaton $B$. $\text{dom}(p)$ is the set of trees accepted at state $p$. $\mathcal{L}(B) = \text{dom}(p_0)$ for initial state $p_0$.

Question

For states $q, q'$ of the transducer, does it hold that

$$[q](t) = [q'](t) \quad (t \in \mathcal{L}(B))$$
Unary Transducers

Idea

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

$$[t] = ([1](t), \ldots, [n](t)) \in \mathbb{Q}^n$$
Unary 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 $V_p = \{[[t]] \mid t \in \text{dom}(p)\}$. 
Unary 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 $V_p = \{[[t]] \mid t \in \text{dom}(p)\}$.

• Consider $H(v) = v_q - v_{q'}$.

• The following statements are equivalent:

1. $q, q'$ agree on inputs from $L(B)$
2. $H(v) = 0$ \hspace{1cm} ($v \in V_{p_0}$)
Unary 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 \( V_p = \{ [t] \mid t \in \text{dom}(p) \} \).

- Consider \( H(v) = v_q - v_{q'} \).

- The following statements are equivalent:
  1. $q, q'$ agree on inputs from $\mathcal{L}(B)$
  2. $H(v) = 0$ \quad ($v \in V_{p_0}$)
  3. $H(v) = 0$ \quad ($v \in \text{Aff}(V_{p_0})$)

// affine closure
Computing Affine Closures

Define

\[ [f](x_1, \ldots, x_k) = ([T_1](x), \ldots, [T_n](x)) \]

where

\[ q(f(x_1, \ldots, x_k)) \rightarrow T_q \]
Computing Affine Closures

Define

\[ [f](x_1, \ldots, x_k) = ([T_1](x), \ldots, [T_n](x)) \quad \text{where} \quad q(f(x_1, \ldots, x_k)) \rightarrow T_q \]

\[
\begin{align*}
[z](x) &= z \\
[j(x_i)](x) &= x_{ij} \\
[c \cdot T](x) &= c \cdot [T](x) \\
[T_1 + T_2](x) &= [T_1](x) + [T_2](x)
\end{align*}
\]
Computing Affine Closures

Define

\[ [f](x_1, \ldots, x_k) = ([T_1](x), \ldots, [T_n](x)) \text{ where } q(f(x_1, \ldots, x_k)) \rightarrow T_q \]

\[ [z](x) = z \]
\[ [j(x_i)](x) = x_{ij} \]
\[ [c \cdot T](x) = c \cdot [T](x) \]
\[ [T_1 + T_2](x) = [T_1](x) + [T_2](x) \]

Observation

\[ [f] : \mathbb{Q}^n \times \ldots \times \mathbb{Q}^n \rightarrow \mathbb{Q}^n \text{ is affine.} \]
Computing Affine Closures (cont.)

Consequence

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

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

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

Consequence

\[ V'_p = \text{Aff}(V_p) \] is the least solution of:

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

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

Theorem

- Equivalence of total unary \( yDTs \) relative to some \( B \) is decidable in polynomial time.
Computing Affine Closures (cont.)

Consequence

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

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

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

Theorem

- Equivalence of total unary \( y\text{DTs} \) relative to some \( B \) is decidable in polynomial time.
- In-Equivalence of linear \( y\text{DTs} \) 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 Ideals
Unary Transducers with Parameters

\[ q(f(x_1, x_2), y) \rightarrow q_1(x_1, q_1(x_2, 1)) \]
\[ q_1(a(x_1), y) \rightarrow y + q_1(x_1, y) \]
\[ q_1(e, y) \rightarrow 0 \]
\[ q'(f(x_1, x_2), y) \rightarrow q_1(x_2, q_1(x_1, 1)) \]
Unary Transducers with Parameters

\[
q(f(x_1, x_2), y) \rightarrow q_1(x_1, q_1(x_2, 1)) \\
q_1(a(x_1), y) \rightarrow y + q_1(x_1, y) \\
q_1(e, y) \rightarrow 0 \\
q'(f(x_1, x_2), y) \rightarrow q_1(x_2, q_1(x_1, 1))
\]

The semantics of a tree \( t \) is now a vector

\[[t] : (\mathbb{Q} \rightarrow \mathbb{Q})^n\]

of affine functions in the parameters.
Unary Transducers with Parameters

\[ q(f(x_1, x_2), y) \rightarrow q_1(x_1, q_1(x_2, 1)) \]
\[ q_1(a(x_1), y) \rightarrow y + q_1(x_1, y) \]
\[ q_1(e, y) \rightarrow 0 \]
\[ q'(f(x_1, x_2), y) \rightarrow q_1(x_2, q_1(x_1, 1)) \]

The semantics of a tree \( t \) is now a vector

\[ [t] : (\mathbb{Q} \rightarrow \mathbb{Q})^n \]

of affine functions in the parameters.

\[ \Rightarrow \]

It can be represented by a matrix \((\mathbb{Q} \rightarrow \mathbb{Q})^n\).
The Semantics of Constructors

$\lfloor f \rfloor : (\mathbb{Q}^{n \times (l+1)} \times \ldots \times \mathbb{Q}^{n \times (l+1)}) \to \mathbb{Q}^{n \times (l+1)}$

thus is of the form:

$(\lfloor f \rfloor(x))_{qj} = \text{polynomial in the } x_{iq'j'}$
The Semantics of Constructors

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

thus is of the form:

\( ([f](x))_{qj} = \text{polynomial in the } x_{i q' j'} \)

\[ \Rightarrow \] The affine closure trick fails :-(

Technische Universität München
In the Example

\[
\begin{align*}
q(f(x_1, x_2), y) & \rightarrow q_1(x_1, q_1(x_2, 1)) \\
q_1(a(x_1), y) & \rightarrow y + q_1(x_1, y) \\
q_1(e, y) & \rightarrow 0 \\
q'(f(x_1, x_2), y) & \rightarrow q_1(x_2, q_1(x_1, 1))
\end{align*}
\]
In the Example

\[
q(f(x_1, x_2), y) \rightarrow q_1(x_1, q_1(x_2, 1))
\]
\[
q_1(a(x_1), y) \rightarrow y + q_1(x_1, y)
\]
\[
q_1(e, y) \rightarrow 0
\]
\[
q'(f(x_1, x_2), y) \rightarrow q_1(x_2, q_1(x_1, 1))
\]

\[
([f](x))_{q_0} = x_{1q_0} + x_{1q_1} \cdot (x_{2q_0} + x_{2q_1} \cdot 1)
\]
\[
([f](x))_{q'0} = x_{2q_0} + x_{2q_1} \cdot (x_{1q_0} + x_{1q_1} \cdot 1)
\]
In the Example

\[
q(f(x_1, x_2), y) \rightarrow q_1(x_1, q_1(x_2, 1))
\]

\[
q_1(a(x_1), y) \rightarrow y + q_1(x_1, y)
\]

\[
q_1(e, y) \rightarrow 0
\]

\[
q'(f(x_1, x_2), y) \rightarrow q_1(x_2, q_1(x_1, 1))
\]

\[
(f(x))_{q_0} = x_{1q_0} + x_{1q_1} \cdot (x_{2q_0} + x_{2q_1} \cdot 1)
\]

\[
(f(x))_{q_0}' = x_{2q_0} + x_{2q_1} \cdot (x_{1q_0} + x_{1q_1} \cdot 1)
\]

\[
(f(x))_{q_1} = 0 = (f(x))_{q_1}'
\]
In the Example

\[ q(f(x_1, x_2), y) \rightarrow q_1(x_1, q_1(x_2, 1)) \]

\[ q_1(a(x_1), y) \rightarrow y + q_1(x_1, y) \]

\[ q_1(e, y) \rightarrow 0 \]

\[ q'(f(x_1, x_2), y) \rightarrow q_1(x_2, q_1(x_1, 1)) \]

\[
\begin{align*}
[[f](x)]_{q_0} &= x_1q_{0} + x_1q_{1} \cdot (x_2q_{0} + x_2q_{1} \cdot 1) \\
[[f](x)]_{q_0}' &= x_2q_{0} + x_2q_{1} \cdot (x_1q_{0} + x_1q_{1} \cdot 1) \\
[[f](x)]_{q_1} &= 0 = ([[f](x)]_{q_1}') \\
[[a](x)]_{q_10} &= x_1q_{0} \\
[[a](x)]_{q_11} &= 1 + x_1q_{1} \\
[[e]]_{q_10} &= 0 = ([[e]]_{q_11})
\end{align*}
\]
Polynomial Invariants

Idea

- polynomial equality:

\[ z_{q_1} \cdot z_{q_1'} \cdot z_{q_0'} - 2 \cdot z_{q_0} + 3 = 0 \]
Polynomial Invariants

Idea

- **polynomial equality:**

\[ z_{q_1} \cdot z_{q'_1} \cdot z_{q'_0} - 2 \cdot z_{q_0} + 3 \neq 0 \]

- \( q_1 \neq 0 \land \ldots \land q_r \neq 0 \) **invariant at** \( p \) **iff**

\[ q_1(\llbracket t \rrbracket) = \ldots = q_r(\llbracket t \rrbracket) = 0 \quad (t \in \text{dom}(p)) \]
Polynomial Invariants

Idea

• polynomial equality:

\[ z_{q_1} \cdot z_{q_1'} \cdot z_{q_0'} - 2 \cdot z_{q_0} + 3 = 0 \]

• \( q_1 \not=} 0 \land \ldots \land q_r \not=} 0 \quad \text{invariant at } p \quad \text{iff}

\[ q_1([t]) = \ldots = q_r([t]) = 0 \quad (t \in \text{dom}(p)) \]

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 $r \in R$. 

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

Hilbert (1890): Every ideal of $\mathbb{Q}[z]$ is finitely generated!
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

A ring $R$ 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

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

$R = \mathbb{Q}[z]$ polynomial ring
Polynomial Ideals: A Primer

Let $R$ be a ring. An ideal $I \subseteq R$ is:

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

$I$ is finitely generated, if

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

Let $R = \mathbb{Q}[z]$ be the polynomial ring.

Hilbert (1890):

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

Vanishing Ideal:

\[ \mathcal{I}(V) = \{ q \in \mathbb{Q}[z] \mid \forall v \in V. q(v) = 0 \} \]
Consequences

Vanishing Ideal:

\[ \mathcal{I}(V) = \{ q \in \mathbb{Q}[z] \mid \forall v \in V. q(v) = 0 \} \]

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

Vanishing Ideal:

\[ \mathcal{I}(V) = \{ q \in \mathbb{Q}[z] \mid \forall v \in V. q(v) = 0 \} \]

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

Notation: \( q^{(f)}_{aq} = ([f](x))_{aq} \)

\( z \) fresh set of variables
Inductive Invariants

Notation: \[ q_{aq}^{(f)} = ([f](x))_{aq} \]

\( z \)  fresh set of variables

\( p \mapsto I_p \) is inductive if for every transition \((p, f) \mapsto p_1 \ldots p_k\) of the automaton \( B \),

\[
I_p \subseteq \{ q \in \mathbb{Q}[z] \mid q[q^{(f)}/z] \in \langle I_{p_1}(x_1) \rangle_{\mathbb{Q}[x]} \oplus \ldots \oplus \langle I_{p_k}(x_k) \rangle_{\mathbb{Q}[x]} \}
\]

holds.
Main Result

Theorem

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

$$q([t]) = 0 \quad (t \in \text{dom}(p))$$
Main Result

Theorem

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

$$q(\llbracket t \rrbracket) = 0 \quad (t \in \text{dom}(p))$$

- For $\bar{I}_p = \mathcal{I}(\{ \llbracket t \rrbracket \mid t \in \text{dom}(p) \})$, $p \mapsto \bar{I}_p$ is inductive.
Main Result

Theorem

• Assume $p \mapsto I_p$ is inductive. Then for every $q \in I_p$,

$$q([t]) = 0 \quad (t \in \text{dom}(p))$$

• For $\bar{I}_p = I(\{[t] \mid t \in \text{dom}(p)\})$, $p \mapsto \bar{I}_p$ is inductive.

Corollary

Let $H(z) = z_{q_0} - z_{q'}_0$. Then $q, q'$ are equivalent (relative to $L(B)$) iff

$$H \in \bar{I}_{p_0}$$
Discussion

- Inductive $\rho \mapsto I_\rho$ with $H \in I_{\rho_0}$ proves that $H$ holds.
Discussion

- Inductive $p \rightarrow I_p$ with $H \in I_{p_0}$ proves that $H$ holds.
- If $H$ holds, it can be proven (somehow).
Discussion

- Inductive $p \mapsto I_p$ with $H \in I_{p_0}$ proves that $H$ holds.
- If $H$ holds, it can be proven (somehow).
- $p \mapsto \overline{I}_p$ is a greatest fixpoint.

Greatest fixpoint iteration may not terminate ...
Discussion

- Inductive $p \hookrightarrow I_p$ with $H \in I_{p_0}$ proves that $H$ holds.
- If $H$ holds, it can be proven (somehow).
- $p \hookrightarrow \overline{I}_p$ is a greatest fixpoint.

Greatest fixpoint iteration may not terminate ...

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

- Inductive $p \mapsto I_p$ with $H \in I_{p_0}$ proves that $H$ holds.
- If $H$ holds, it can be proven (somehow).
- $p \mapsto \overline{I}_p$ is a greatest fixpoint.

Greatest fixpoint iteration may not terminate ...

- All inductive invariants, though, can be recursively enumerated!
- All potential counter examples can also be enumerated!!

Technische Universität München
Wrap-up

Theorem

- Equivalence of unary \texttt{yMDTs} is decidable.
Wrap-up

Theorem

- Equivalence of unary $y$MDTs is decidable.
- Equivalence of general $y$DTs is decidable.
Summary

Parameters allow to encode general output alphabets by means of unaries.
**Summary**

Parameters allow to encode general output alphabets by means of unaries.

Equivalence for unary transducers can be handled by means of techniques from precise program analysis, i.e., program proving.
Parameters allow to encode general output alphabets by means of unaries.

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

<table>
<thead>
<tr>
<th>Unary</th>
<th>General</th>
<th>Equalities</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y_{DT}$</td>
<td></td>
<td>affine</td>
</tr>
</tbody>
</table>
Parameters allow to encode general output alphabets by means of unaries.

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

<table>
<thead>
<tr>
<th>Unary</th>
<th>General</th>
<th>Equalities</th>
</tr>
</thead>
<tbody>
<tr>
<td>yDT</td>
<td>linear yDT</td>
<td>affine</td>
</tr>
<tr>
<td>nsn yMDT</td>
<td></td>
<td>affine</td>
</tr>
</tbody>
</table>
Parameters allow to encode general output alphabets by means of unaries.

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

<table>
<thead>
<tr>
<th>Unary</th>
<th>General</th>
<th>Equalities</th>
</tr>
</thead>
<tbody>
<tr>
<td>yDT</td>
<td>linear yDT</td>
<td>affine</td>
</tr>
<tr>
<td>nsn yMDT</td>
<td>yDT</td>
<td>affine</td>
</tr>
<tr>
<td>yMDT</td>
<td>polynomial</td>
<td></td>
</tr>
</tbody>
</table>
Thank you!