First-Order Transformations of Finite Words

Emmanuel Filiot
Krishna Shankara Narayanan
A. Trivedi

ULB-FNRS
IIT Bombay
IIT Bombay

Highlights of Logic, Games and Automata, 2014
Overview

- $\Sigma$: finite alphabet

**Theorem (Engelfriet, Hoogeboom, 01)**

A function $f: \Sigma^* \rightarrow \Sigma^*$ is (Courcelle) MSO-definable iff it is definable by a deterministic two-way transducer.

**Theorem (Alur, Cerny, 10)**

A function $f: \Sigma^* \rightarrow \Sigma^*$ is (Courcelle) MSO-definable iff it is definable by a streaming string transducer (SST).

**Theorem (Main result of this talk)**

A function $f: \Sigma^* \rightarrow \Sigma^*$ is (Courcelle) FO-definable iff it is definable by an aperiodic streaming string transducer.

(Filiot, S.N. Krishna, Trivedi) FO Transformations
Overview

* ∈ finite alphabet

**Theorem (Engelfriet, Hoogeboom, 01)**

A function $f : \Sigma^* \rightarrow \Sigma^*$ is (Courcelle) **MSO-definable** iff it is definable by a **deterministic two-way transducer**.

**Theorem (Alur, Cerny, 10)**

A function $f : \Sigma^* \rightarrow \Sigma^*$ is (Courcelle) **MSO-definable** iff it is definable by a **streaming string transducer (SST)**.
Overview

- $\Sigma$: finite alphabet

<table>
<thead>
<tr>
<th>Theorem (Engelfriet, Hoogeboom, 01)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A function $f : \Sigma^* \rightarrow \Sigma^*$ is (Courcelle) MSO-definable iff it is definable by a deterministic two-way transducer.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Theorem (Alur, Cerny, 10)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A function $f : \Sigma^* \rightarrow \Sigma^*$ is (Courcelle) MSO-definable iff it is definable by a streaming string transducer (SST).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Theorem (Main result of this talk)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A function $f : \Sigma^* \rightarrow \Sigma^*$ is (Courcelle) FO-definable iff it is definable by an aperiodic streaming string transducer.</td>
</tr>
</tbody>
</table>

(Filiot, S.N. Krishna, Trivedi)  FO Transformations
Examples of Transformations

- $f_{del}$: delete all ‘a’ positions
  \[
  abbabaa \mapsto bbb
  \]

- $f_{rev}$: reverse the input word
  \[
  stressed \mapsto desserts
  \]

- $f_{halve}$: maps all inputs $a^n$ to $a^{\lfloor \frac{n}{2} \rfloor}$.
  \[
  a^5 \mapsto a^2
  \]

- $f_{copy}$: copy the input word twice
  \[
  ab# \mapsto ab#ab#
  \]
(Courcelle) MSO Transformations

- words as a structures over \( \{ \text{succ}, (\text{lab}_a)_{a \in \Sigma} \} \)
- output predicates defined by MSO formulas interpreted over the input structure
(Courcelle) MSO Transformations

- words as a structures over \( \{ \text{succ}, (\text{lab}_a)_{a \in \Sigma} \} \)
- output predicates defined by MSO formulas interpreted over the input structure
(Courcelle) MSO Transformations

- words as a structures over $\{\text{succ}, (\text{lab}_a)_{a \in \Sigma}\}$
- output predicates defined by MSO formulas interpreted over the input structure

$\phi_{\text{succ}}(x, y) \equiv \text{succ}(y, x)$
$\phi_{\text{lab}_a}(x) \equiv \text{lab}_a(x)$
(Courcelle) MSO Transformations

- words as a structures over $\{\text{succ}, (\text{lab}_a)_{a \in \Sigma}\}$
- output predicates defined by MSO formulas interpreted over the input structure

$$\phi_{\text{succ}}(x, y) \equiv \text{succ}(y, x)$$
$$\phi_{\text{lab}_a}(x) \equiv \text{lab}_a(x)$$
(Courcelle) MSO Transformations

- words as a structures over \( \{ succ, (lab_a)_{a \in \Sigma} \} \)
- output predicates defined by MSO formulas interpreted over the input structure

\[
\phi_{succ}(x, y) \equiv succ(y, x) \\
\phi_{lab_a}(x) \equiv lab_a(x)
\]
(Courcelle) MSO Transformations

- words as structures over \( \{succ, (lab_a)_{a \in \Sigma}\} \)
- output predicates defined by MSO formulas interpreted over the input structure

\[
\phi_{succ}(x, y) \equiv succ(y, x) \\
\phi_{lab_a}(x) \equiv lab_a(x)
\]

- more generally, input structure can be copied a fixed number of times \( (w \mapsto ww) \)
- **FO-transformations**: MSO replaced by FO over \( \{\leq, (lab_a)_{a \in \Sigma}\} \).
Streaming String Transducers (SST)

- one-way, deterministic model
- extend finite automata with a finite set of word variables $X, Y \ldots$
  - appending a word $u$: $X := Xu$
  - prepending a word: $X := uX$
  - concatenating two variables: $X := YZ$

Theorem (Alur, Cerny, 10)

A function $f: \Sigma^* \rightarrow \Sigma^*$ is MSO-definable iff it is definable by an SST with copyless variable update.

Question: What restriction to put on SST to capture FO?

(Filiot, S.N. Krishna, Trivedi)
Streaming String Transducers (SST)

- one-way, deterministic model
- extend finite automata with a finite set of word variables $X, Y \ldots$
  - appending a word $u$: $X := Xu$
  - prepending a word: $X := uX$
  - concatenating two variables: $X := YZ$

\[
\sigma | X := \sigma . X
\]

reverse :

\[
\begin{array}{c}
\sigma \in \Sigma
\end{array}
\]
Streaming String Transducers (SST)

- one-way, deterministic model
- extend finite automata with a finite set of word variables $X, Y \ldots$
  - appending a word $u$: $X := Xu$
  - prepending a word: $X := uX$
  - concatenating two variables: $X := YZ$

$$\sigma | X := \sigma \cdot X$$

reverse:

Theorem (Alur, Cerny, 10)

A function $f : \Sigma^* \rightarrow \Sigma^*$ is MSO-definable iff it is definable by an SST with copyless variable update.
Streaming String Transducers (SST)

- one-way, deterministic model
- extend finite automata with a finite set of word variables $X, Y \ldots$
  - appending a word $u$: $X := Xu$
  - prepending a word: $X := uX$
  - concatenating two variables: $X := YZ$

\[
\sigma | X := \sigma . X
\]

reverse : $X \quad \sigma \in \Sigma$

Theorem (Alur, Cerny, 10)

A function $f : \Sigma^* \rightarrow \Sigma^*$ is MSO-definable iff it is definable by an SST with copyless variable update.

**Question:** What restriction to put on SST to capture FO?
Aperiodic Finite Automata

Among several characterizations of FO languages\(^1\), we use the following:

**Theorem**

A language \( L \subseteq \Sigma^* \) is FO-definable iff it is definable by an aperiodic finite automaton (AFA).

\(^1\) *First-order definable languages*, V. Diekert and P. Gastin. 2007.
Aperiodic Finite Automata

Among several characterizations of FO languages\(^1\), we use the following:

**Theorem**

A language \( L \subseteq \Sigma^* \) is FO-definable iff it is definable by an aperiodic finite automaton (AFA).

- AFA = finite automaton with aperiodic transition monoid \( T(A) \)
- \( T(A) = \{ M_w \mid w \in \Sigma^* \} \)
- for any two states \( p, q \), \( M_w[p][q] = 1 \) iff \( p \rightsquigarrow^w q \).
- \( T_A \) is aperiodic if \( \exists m \geq 0 \), for all \( M \in T_A \), \( M^m = M^{m+1} \)

**Examples:**

\[ \begin{array}{c}
\text{not aperiodic} \\
\begin{array}{c}
\text{not aperiodic} \\
\end{array}
\end{array} \]

\[ \begin{array}{c}
\text{aperiodic} \\
\begin{array}{c}
\text{aperiodic} \\
\end{array}
\end{array} \]

\(^1\)First-order definable languages, V. Diekert and P. Gastin. 2007.
Towards a restriction: $f_{\text{halve}} : a^n \mapsto a^{\lfloor \frac{n}{2} \rfloor}$ again

- not FO-definable
- definable by:

\[ T_1 : \]

\[ a \mid X := aX \]
\[ a \mid X := X \]

(Filiot, S.N. Krishna, Trivedi) FO Transformations
Towards a restriction:  $f_{halve} : a^n \mapsto a^{\lfloor \frac{n}{2} \rfloor}$ again

- not FO-definable
- definable by:

\begin{align*}
T_1 : & \quad a \mid X := aX \\
& \quad a \mid X := X
\end{align*}

- aperiodicity of the underlying input automaton is not sufficient:

\begin{align*}
T_0 : & \quad a \mid X := aY \\
& \quad Y := X
\end{align*}
Variable flow

\[ T_0 : \text{ } a \rightarrow X \rightarrow Y \rightarrow X \rightarrow Y \rightarrow a \mid X := aY \]

Y := X

Dependency graph

input: \( a \quad a \quad a \quad a \quad a \quad a \)

\( X \rightarrow X \rightarrow X \rightarrow X \rightarrow X \rightarrow X \rightarrow X \)

\( Y \rightarrow Y \rightarrow Y \rightarrow Y \rightarrow Y \rightarrow Y \rightarrow Y \)
Variable flow

\[ T_0 : \quad \rightarrow a \quad X := aY \]
\[ Y := X \]

\[ \Rightarrow \text{impose aperiodicity of the variable flow!} \]
SST Transition Monoid

- set of Boolean matrices $M_w$ indexed by pairs $(q, X)$
- coefficients in $\mathbb{N} \cup \{\bot\}$
- $M_w[p, X][q, Y] = \bot$ if there no run from $p$ to $q$ on $w$
- $M_w[p, X][q, Y] = n \in \mathbb{N}$ if
  - there is a run $r$ from $p$ to $q$ on $w$
  - on this run, $X$ “flows” $n$ times to $Y$
SST Transition Monoid

- set of Boolean matrices \( M_w \) indexed by pairs \((q, X)\)
- coefficients in \( \mathbb{N} \cup \{\perp\} \)
- \( M_w[p, X][q, Y] = \perp \) if there no run from \( p \) to \( q \) on \( w \)
- \( M_w[p, X][q, Y] = n \in \mathbb{N} \) if
  - there is a run \( r \) from \( p \) to \( q \) on \( w \)
  - on this run, \( X \) “flows” \( n \) times to \( Y \)

Example:

\[
\begin{align*}
X &:= aXb \\
Y &:= bY
\end{align*}
\]

Then \( M_{aa}[q_0, Y][q_2, X] = 2. \)
Results and Perspectives

Theorem

- A function $f : \Sigma^* \rightarrow \Sigma^*$ is MSO-definable iff it is definable by a SST with finite transition monoid.
- A function $f : \Sigma^* \rightarrow \Sigma^*$ is FO-definable iff it is definable by a SST with finite and aperiodic transition monoid.
Results and Perspectives

**Theorem**

- A function $f: \Sigma^* \rightarrow \Sigma^*$ is MSO-definable iff it is definable by a SST with finite transition monoid.
- A function $f: \Sigma^* \rightarrow \Sigma^*$ is FO-definable iff it is definable by a SST with finite and aperiodic transition monoid.

**Open question**

Give an effective, machine-independent, characterisation of FOT.

Related to M. Bojanczyk’s work on a weaker semantics (with origin).