Verification of infinite state systems

Angelo Montanari and Gabriele Puppis

Department of Mathematics and Computer Science
University of Udine, Italy
{montana,puppis}@dimi.uniud.it

Go

In this part

We present other two relevant classes of transition systems:

- Rational graphs
  described by (unrestricted) **word transducers**
- Automatic graphs
  described by **left-synchronized word transducers**

We study the decidability of their FO-theories and we provide alternative representations of graphs in both classes.

Definition (Word transducer)

A **word transducer** is a tuple $\mathcal{A} = (Q, \Gamma, \Delta, I, F)$, where

- Q is a finite set of states
- Γ is a finite alphabet
- $\Delta \subseteq Q \times \Gamma^* \times \Gamma^* \times Q$
- $I \subseteq Q$ is the set of initial states
- $F \subseteq Q$ is the set of final states.

Definition (Word transducer)

A **word transducer** is a tuple $\mathcal{A} = (Q, \Gamma, \Delta, I, F)$, where

- Q is a finite set of states
- Γ is a finite alphabet
- $\Delta \subseteq Q \times \Gamma^* \times \Gamma^* \times Q$
- $I \subseteq Q$ is the set of initial states
- $F \subseteq Q$ is the set of final states.

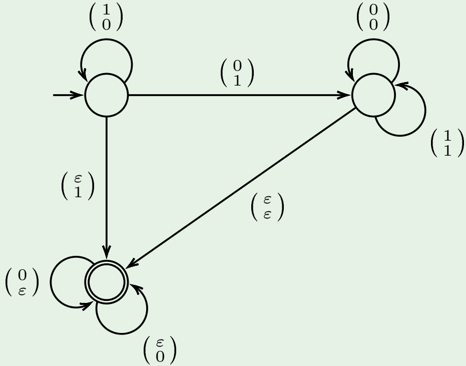
Word transducers are used as acceptors of pairs of finite words over Γ: $(u, v) \in \mathscr{L}(\mathcal{A})$ if there are $u_1, \dots, u_n$, $v_1, \dots, v_n$, $s_0, \dots, s_n$ s.t.

- $u = u_1 \cdot \ldots \cdot u_n$ and $v = v_1 \cdot \ldots \cdot v_n$
- $s_0 \in I$ and $s_n \in F$
- $(s_{i-1}, u_i, v_i, s_i) \in \Delta$ for all $1 \leq i \leq n$

$$\text{(shortly, } I \xrightarrow[\mathcal{A}]{(u,v)} F\text{)}.$$

Example

A word transducer recognizing the set of all reversed binary expansions of pairs of numbers of the form $(n, n+1)$, with $n \in \mathbb{N}$ (e.g., $(111, 0001)$)



If we use finite words over Γ to represent graph vertices,
then we can use word transducers to represent edge relations:

If we use finite words over Γ to represent graph vertices,
then we can use word transducers to represent edge relations:

Definition (Rational graph)

Given an alphabet Γ, a finite set A of edge labels,
a finite state automaton $\mathcal{A}_{dom}$ over Γ, and
a tuple $(\mathcal{A}_a)_{a \in A}$ of word transducers over Γ,
the generated graph, called **rational graph**, is of the form

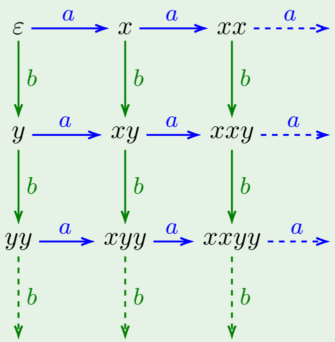
$$\mathcal{T} = (S, (\delta_a)_{a \in A})$$

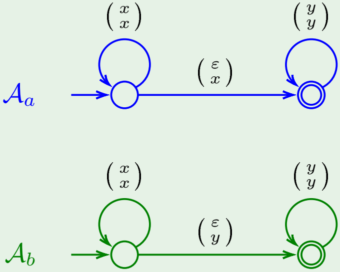
- $S := \mathscr{L}(\mathcal{A}_{dom})$ *(vertices := words accepted by $\mathcal{A}_{dom}$)*
- $\delta_a := \mathscr{L}(\mathcal{A}_a)$ *(edges := pairs accepted by transducers)*

Example

The infinite grid is a *rational graph*:

- vertices are words over $\{x, y\}$ of the form $x^i y^j$
- a-labeled edges connect a word $x^i y^j$ to $x^{i+1} y^j$
  b-labeled edges connect a word $x^i y^j$ to $x^i y^{j+1}$





The previous example shows that the model checking problem for MSO logic over rational graphs is undecidable.

The previous example shows that the model checking problem for MSO logic over rational graphs is undecidable.

Actually, things go bad even for FO logic:

Theorem

The model checking problem for FO logic over rational graphs is undecidable.

Proof

We reduce the Post's Correspondence Problem (**PCP**) to the model checking problem for FO logic over rational graphs.

Proof

We reduce the Post's Correspondence Problem (**PCP**) to the model checking problem for FO logic over rational graphs.

A **PCP-instance** is a tuple $(u_1, \ldots, u_k, v_1, \ldots, v_k)$ of words.
A PCP-instance is **positive** iff there are indices $i_1, \ldots, i_n$ s.t.

$$w = u_{i_1} \cdot \ldots \cdot u_{i_n} \quad = \quad v_{i_1} \cdot \ldots \cdot v_{i_n} = w'$$

(checking if a given PCP-instance $(\bar{u}, \bar{v})$ is positive
is an *undecidable problem*)

Proof

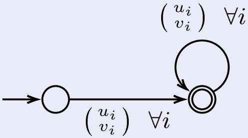
We reduce the Post's Correspondence Problem (**PCP**) to the model checking problem for FO logic over rational graphs.

A **PCP-instance** is a tuple $(u_1, \ldots, u_k, v_1, \ldots, v_k)$ of words.
A PCP-instance is **positive** iff there are indices $i_1, \ldots, i_n$ s.t.

$$w = u_{i_1} \cdot \ldots \cdot u_{i_n} \quad = \quad v_{i_1} \cdot \ldots \cdot v_{i_n} = w'$$

(checking if a given PCP-instance $(\bar{u}, \bar{v})$ is positive
is an *undecidable problem*)

Given $(\bar{u}, \bar{v})$, we define the transducer $\mathcal{A}_{\bar{u},\bar{v}}$:



Proof

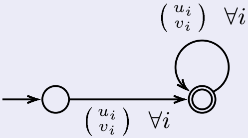
We reduce the Post's Correspondence Problem (**PCP**) to the model checking problem for FO logic over rational graphs.

A **PCP-instance** is a tuple $(u_1, \ldots, u_k, v_1, \ldots, v_k)$ of words.
A PCP-instance is **positive** iff there are indices $i_1, \ldots, i_n$ s.t.

$$w = u_{i_1} \cdot \ldots \cdot u_{i_n} \quad = \quad v_{i_1} \cdot \ldots \cdot v_{i_n} = w'$$

(checking if a given PCP-instance $(\bar{u}, \bar{v})$ is positive
is an *undecidable problem*)

Given $(\bar{u}, \bar{v})$, we define the transducer $\mathcal{A}_{\bar{u},\bar{v}}$:



The rational graph generated by $\mathcal{A}_{\bar{u},\bar{v}}$ satisfies
$\psi := \exists\, x.\ \delta(x, x)$ iff $(\bar{u}, \bar{v})$ is a positive PCP-instance.

We saw that graphs defined by *unrestricted* word transducers have an undecidable model checking problem for FO logic.

We saw that graphs defined by *unrestricted* word transducers have an undecidable model checking problem for FO logic.

$\Rightarrow$ let us consider restricted forms of word transducers:

Definition (Synchronized word transducer)

A **synchronized word transducer** is a word transducer $\mathcal{A} = (Q, \Gamma, \Delta, I, F)$ where $\Delta \subseteq Q \times \Gamma \times \Gamma \times Q$
(exactly one output symbol for each input symbol).

We saw that graphs defined by *unrestricted* word transducers have an undecidable model checking problem for FO logic.

$\Rightarrow$ let us consider restricted forms of word transducers:

Definition (Synchronized word transducer)

A **synchronized word transducer** is a word transducer $\mathcal{A} = (Q, \Gamma, \Delta, I, F)$ where $\Delta \subseteq Q \times \Gamma \times \Gamma \times Q$ *(exactly one output symbol for each input symbol)*.

Note: if $\mathcal{A}$ is a synchronized transducer and $(u, v) \in \mathscr{L}(\mathcal{A})$, then $|u| = |v|$

$\Rightarrow$ *every connected component in a graph generated by a synchronized transducer is finite.*

A more relaxed form of synchronization can be introduced by using a **padding symbol** $\#$ to fill up the words and achieve equal length:

Definition (Left-synchronized word transducer)

A **left-synchronized word transducer** is a word transducer $\mathcal{A} = (Q, \Gamma, \Delta, I, F)$ where $\Delta \subseteq Q \times (\Gamma \cup \{\#\}) \times (\Gamma \cup \{\#\}) \times Q$.

A more relaxed form of synchronization can be introduced by using a **padding symbol** $\#$ to fill up the words and achieve equal length:

Definition (Left-synchronized word transducer)

A **left-synchronized word transducer** is a word transducer $\mathcal{A} = (Q, \Gamma, \Delta, I, F)$ where $\Delta \subseteq Q \times (\Gamma \cup \{\#\}) \times (\Gamma \cup \{\#\}) \times Q$.

We say that the pair $(u, v) \in \Gamma^* \times \Gamma^*$ is accepted by $\mathcal{A}$ iff either $(u \cdot \#^{|v|-|u|}, v) \in \mathscr{L}(\mathcal{A})$ or $(u, v \cdot \#^{|u|-|v|}) \in \mathscr{L}(\mathcal{A})$.

A more relaxed form of synchronization can be introduced by using a **padding symbol** $\#$ to fill up the words and achieve equal length:

Definition (Left-synchronized word transducer)

A **left-synchronized word transducer** is a word transducer $\mathcal{A} = (Q, \Gamma, \Delta, I, F)$ where $\Delta \subseteq Q \times (\Gamma \cup \{\#\}) \times (\Gamma \cup \{\#\}) \times Q$.

We say that the pair $(u, v) \in \Gamma^* \times \Gamma^*$ is accepted by $\mathcal{A}$ iff either $(u \cdot \#^{|v|-|u|}, v) \in \mathscr{L}(\mathcal{A})$ or $(u, v \cdot \#^{|u|-|v|}) \in \mathscr{L}(\mathcal{A})$.

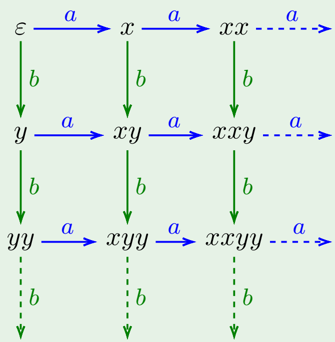
Definition (Automatic graph)

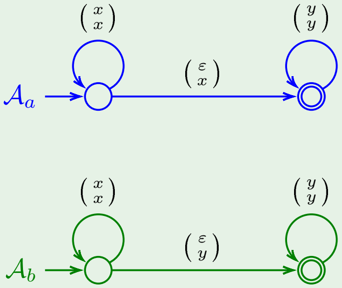
An **automatic graph** is a graph generated by a finite state automaton and a tuple of left-synchronized word transducers.

Example

The infinite grid is actually an *automatic graph*.

We simply need to put the transducers
$\mathcal{A}_a$ and $\mathcal{A}_b$ in a left-synchronized form:

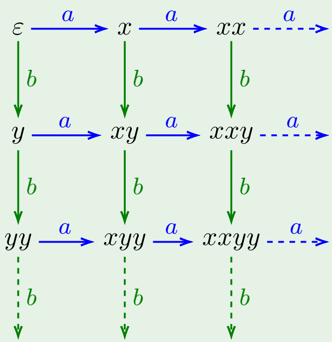


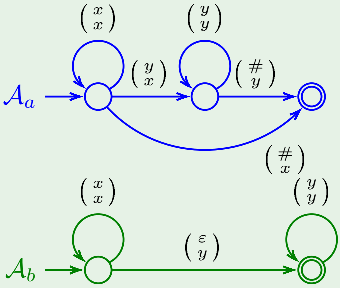


Example

The infinite grid is actually an *automatic graph*.

We simply need to put the transducers
$\mathcal{A}_a$ and $\mathcal{A}_b$ in a left-synchronized form:

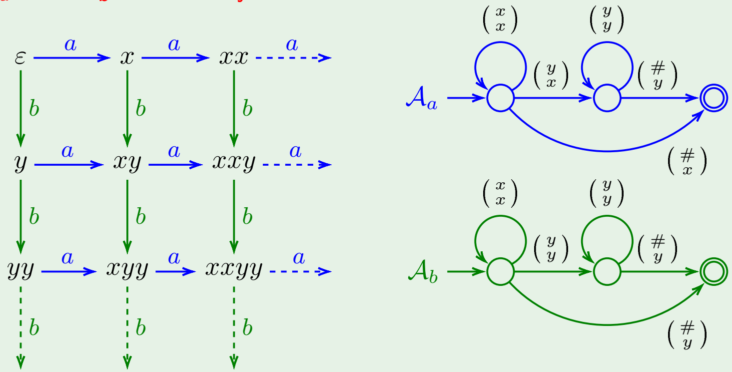




Example

The infinite grid is actually an *automatic graph*.

We simply need to put the transducers $\mathcal{A}_a$ and $\mathcal{A}_b$ in a left-synchronized form:



The previous example shows that the model checking problem for MSO logic over automatic graphs is undecidable.

The previous example shows that the model checking problem for MSO logic over automatic graphs is undecidable.

Fortunately, we achieve decidability when considering FO logic:

Theorem (Büchi '60, Hodgson '76, Khoussainov and Nerode '94)

The model checking problem for FO logic over automatic graphs is decidable.

Proof

Let $\mathcal{T} = \big(S, (\delta_a)_{a\in A}\big)$ be an automatic graph generated by a finite state automaton $\mathcal{A}_{dom}$ and by tuple of left-synchronized transducers $(\mathcal{A}_a)_{a\in A}$.

Proof

Let $\mathcal{T} = \left(S, (\delta_a)_{a \in A}\right)$ be an automatic graph generated by a finite state automaton $\mathcal{A}_{dom}$ and by tuple of left-synchronized transducers $(\mathcal{A}_a)_{a \in A}$.

Transducers can be generalized to recognize *k*-ary relations, for an arbitrary k: simply let $\Delta \subseteq Q \times (\Gamma \cup \{\#\})^k \times Q$.

Proof

Let $\mathcal{T} = \big(S, (\delta_a)_{a\in A}\big)$ be an automatic graph generated by a finite state automaton $\mathcal{A}_{dom}$ and by tuple of left-synchronized transducers $(\mathcal{A}_a)_{a\in A}$.

Transducers can be generalized to recognize k-ary relations, for an arbitrary k: simply let $\Delta \subseteq Q \times (\Gamma \cup \{\#\})^k \times Q$.

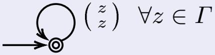
Let us prove that for any given FO-formula $\psi(x_1, \ldots, x_k)$ there is a left-synchronized transducer $\mathcal{A}_\psi$ such that, for all tuples $(u_1, \ldots, u_k) \in S^k$

$$\mathcal{T} \vDash \psi[u_1/x_1, \ldots, u_k/x_k] \quad \text{iff} \quad (u_1, \ldots, u_k) \in \mathscr{L}(\mathcal{A}_\psi)$$

Warning: here a transducer defines a relation $r \subseteq S^k$
($\Rightarrow$ it accepts tuples of vertices rather than vertex colorings)

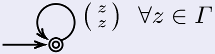
Proof (continued)

The proof goes by induction on the structure of $\psi(x_1, \ldots, x_k)$:

- if ψ is $x_1 = x_2$, then $\mathcal{A}_\psi :=$ 

Proof (continued)

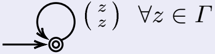
The proof goes by induction on the structure of $\psi(x_1, \ldots, x_k)$:

- if ψ is $x_1 = x_2$, then $\mathcal{A}_\psi :=$ 

- if ψ is $\delta_a(x_1, x_2)$, then $\mathcal{A}_\psi := \mathcal{A}_a$

Proof (continued)

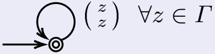
The proof goes by induction on the structure of $\psi(x_1, \dots, x_k)$:

- if ψ is $x_1 = x_2$, then $\mathcal{A}_\psi :=$ 

- if ψ is $\delta_a(x_1, x_2)$, then $\mathcal{A}_\psi := \mathcal{A}_a$
- if ψ is $\varphi_1 \wedge \varphi_2$, then $\mathcal{A}_\psi := \mathcal{A}_{\varphi_1} \cap \mathcal{A}_{\varphi_2}$

Proof (continued)

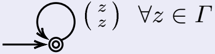
The proof goes by induction on the structure of $\psi(x_1, \ldots, x_k)$:

- if ψ is $x_1 = x_2$, then $\mathcal{A}_\psi :=$ 

- if ψ is $\delta_a(x_1, x_2)$, then $\mathcal{A}_\psi := \mathcal{A}_a$
- if ψ is $\varphi_1 \ \wedge \ \varphi_2$, then $\mathcal{A}_\psi := \mathcal{A}_{\varphi_1} \cap \mathcal{A}_{\varphi_2}$
- if ψ is $\varphi_1 \ \vee \ \varphi_2$, then $\mathcal{A}_\psi := \mathcal{A}_{\varphi_1} \cup \mathcal{A}_{\varphi_2}$

Proof (continued)

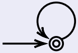
The proof goes by induction on the structure of $\psi(x_1, \ldots, x_k)$:

- if ψ is $x_1 = x_2$, then $\mathcal{A}_\psi :=$ 

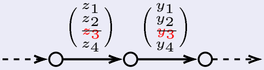
- if ψ is $\delta_a(x_1, x_2)$, then $\mathcal{A}_\psi := \mathcal{A}_a$
- if ψ is $\varphi_1 \wedge \varphi_2$, then $\mathcal{A}_\psi := \mathcal{A}_{\varphi_1} \cap \mathcal{A}_{\varphi_2}$
- if ψ is $\varphi_1 \vee \varphi_2$, then $\mathcal{A}_\psi := \mathcal{A}_{\varphi_1} \cup \mathcal{A}_{\varphi_2}$
- if ψ is $\neg\varphi$, then $\mathcal{A}_\psi$ is the complement automaton of $\mathcal{A}_\varphi$

Proof (continued)

The proof goes by induction on the structure of $\psi(x_1, \ldots, x_k)$:

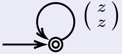
- if ψ is $x_1 = x_2$, then $\mathcal{A}_\psi :=$ 

- if ψ is $\delta_a(x_1, x_2)$, then $\mathcal{A}_\psi := \mathcal{A}_a$
- if ψ is $\varphi_1 \ \wedge \ \varphi_2$, then $\mathcal{A}_\psi := \mathcal{A}_{\varphi_1} \cap \mathcal{A}_{\varphi_2}$
- if ψ is $\varphi_1 \ \vee \ \varphi_2$, then $\mathcal{A}_\psi := \mathcal{A}_{\varphi_1} \cup \mathcal{A}_{\varphi_2}$
- if ψ is $\neg\varphi$, then $\mathcal{A}_\psi$ is the complement automaton of $\mathcal{A}_\varphi$
- if ψ is $\exists\, x_i .\ \varphi(x_1, \ldots, x_i, \ldots, x_m)$, then $\mathcal{A}_\psi$ is obtained from $\mathcal{A}_\varphi$ by first intersecting with $\mathcal{A}_{dom}$ and then removing the i-th symbol in each transition

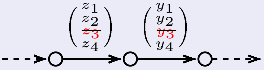


Proof (continued)

The proof goes by induction on the structure of $\psi(x_1, \ldots, x_k)$:

- if ψ is $x_1 = x_2$, then $\mathcal{A}_\psi :=$ 

- if ψ is $\delta_a(x_1, x_2)$, then $\mathcal{A}_\psi := \mathcal{A}_a$
- if ψ is $\varphi_1 \ \wedge \ \varphi_2$, then $\mathcal{A}_\psi := \mathcal{A}_{\varphi_1} \cap \mathcal{A}_{\varphi_2}$
- if ψ is $\varphi_1 \ \vee \ \varphi_2$, then $\mathcal{A}_\psi := \mathcal{A}_{\varphi_1} \cup \mathcal{A}_{\varphi_2}$
- if ψ is $\neg\varphi$, then $\mathcal{A}_\psi$ is the complement automaton of $\mathcal{A}_\varphi$
- if ψ is $\exists\, x_i .\ \varphi(x_1, \ldots, x_i, \ldots, x_m)$, then $\mathcal{A}_\psi$ is obtained from $\mathcal{A}_\varphi$ by first intersecting with $\mathcal{A}_{dom}$ and then removing the i-th symbol in each transition



$\Rightarrow$ The model checking problem for FO logic is reduced to the emptiness problem for word transducers.

Proposition

The reachability problem over automatic graphs is undecidable.

Proposition

The reachability problem over automatic graphs is undecidable.

Proof

Consider a generic Turing machine M where

- configurations are encoded by words $a_1 \dots a_{m-1} q a_m \dots a_n$
- transitions are of the following forms

$$\begin{array}{c} a_1 \dots a_{m-1} q a_m \dots a_n \\ \downarrow \\ a_1 \dots a_{m-1} q' a'_m \dots a_m \end{array} \qquad \begin{array}{c} a_1 \dots a_{m-1} q a_m \dots a_n \\ \downarrow \\ a_1 \dots a_{m-2} q' a_{m-1} a'_m \dots a_m \end{array} \qquad \begin{array}{c} a_1 \dots a_{m-1} q a_m \dots a_n \\ \downarrow \\ a_1 \dots a_{m-1} a'_m q' a_{m+1} \dots a_m \end{array}$$

Note that

- the words that encode a valid configuration of M can be recognized by a suitable *finite state automaton*
- the transition relations can be recognized by a suitable *left-synchronized word transducer*.

$\Rightarrow$ the transition graph of any Turing machine is automatic.

The previous definitions and results can be easily generalized to **relational structures** having relations of arbitrary arities:

Definition (Automatic structure)

An **automatic structure** is a relational structure $\mathcal{R} = \big(S, r_1, \ldots, r_m\big)$, where

- $S \subseteq \Gamma^*$ is a regular language of finite words
- $r_i \subseteq (\Gamma^*)_i^k$ is a k_i-ary relation over finite words recognized by a left-synchronized word transducer working with k_i-tuples of letters.

Theorem

The model checking problem for FO logic over automatic structures is decidable.

Example

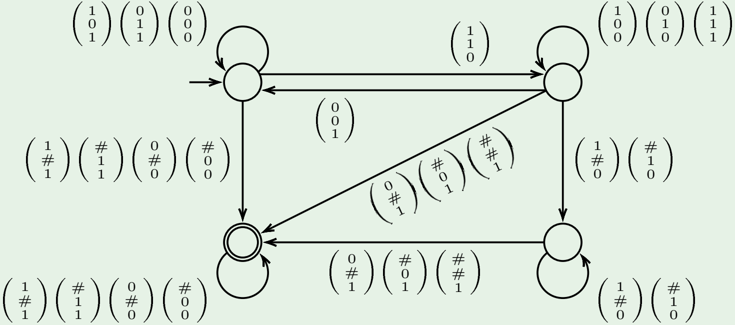
Building on previous ideas, one can show that the **Presburger arithmetic** (i.e., the FO-theory of $(\mathbb{N}, +)$) is decidable.

Example

Building on previous ideas, one can show that the **Presburger arithmetic** (i.e., the FO-theory of $(\mathbb{N}, +)$) is decidable.

It is sufficient to represent

- the natural numbers by their reversed binary expansions
- the *ternary* relation $+$ by the left-synchronized transducer



Alternative characterizations of automatic/rational graphs/structures have been given in the literature:

Theorem (Morvan '00)

A relational graph $\mathcal{T}$ is rational iff it can be obtained from the infinite binary tree via an inverse linear mapping and a rational restriction.

(an inverse mapping h^{-1} is **linear** *if, for every label b, $h(b)$ is a linear context-free language, namely, generated by a grammar with rules of the form $z \rightarrow \varepsilon$ and $z \rightarrow \bar{u} \cdot z' \cdot v$).*

$\Rightarrow$ Special forms of inverse linear mappings characterize the automatic graphs.

Theorem (Blumensath '99)

A relational structure $\mathcal{S}$ is automatic iff it can be obtained from $(\mathbb{B}^*, \delta_0, \delta_1, \sqsubseteq, L)$ via a FO-interpretation endowed with a congruence $\sim$ that defines the vertices of $\mathcal{S}$ as $\sim$-classes.

(the structure $(\mathbb{B}^*, \delta_0, \delta_1, \sqsubseteq, L)$ is the infinite binary tree expanded with the **ancestor relation** $\sqsubseteq$ *and the* **equi-level relation** L*).*

Theorem (Blumensath '99)

A relational structure $\mathcal{S}$ is automatic iff it can be obtained from $(\mathbb{B}^*, \delta_0, \delta_1, \sqsubseteq, L)$ via a FO-interpretation endowed with a congruence $\sim$ that defines the vertices of $\mathcal{S}$ as $\sim$-classes.

(the structure $(\mathbb{B}^*, \delta_0, \delta_1, \sqsubseteq, L)$ is the infinite binary tree expanded with the **ancestor relation** *$\sqsubseteq$ and the* **equi-level relation** *L).*

Theorem (Elgot and Rabin '66, Rubin '04)

A relational structure $\mathcal{S}$ is automatic iff it can be obtained from $\mathcal{L} := (\mathbb{N}, \delta)$ via a **WMSO-to-FO-interpretation**.

(a WMSO-to-FO-interpretation is an interpretation where free variables are instantiated by finite sets
$\Rightarrow$ the vertices of $\mathcal{S}$ are subsets of the domain of $\mathcal{L}$).

Richer (automatic-like) structures can be defined using:

- transducers over infinite words
  $\Rightarrow$ this leads to ω-**automatic structures** (e.g., $(\mathbb{R}, +)$)
- transducers over finite trees
  $\Rightarrow$ this leads to **tree-automatic structures**
- transducers over infinite trees
  $\Rightarrow$ this leads to ω-**tree-automatic structures**
- WMSO-to-FO-interpretations over the binary tree
  $\Rightarrow$ this leads to tree-automatic structures as well
- WMSO-to-FO-interpretations over Caucal graphs
  $\Rightarrow$ this leads to a (strictly increasing)
  hierarchy of automatic structures

Go