# Order 2 Tree-Automatic Graphs

*(work in progress)*

Antoine Meyer (UPEM)
Joint work with Arnaud Carayol (CNRS, UPEM)

Automatic Presentations of Graphs and Numbers
IMSc, 11/10/2016

# Summary

- The pushdown hierarchy of infinite graphs [KNU02, Cau02, CW03]
  - Level $n$ : transition graphs of $n$-pushdown automata
  - Can be obtained from a unique graph using logic-based transformations
- Using more expressive transformations, one gets strictly more graphs [CL07]
  - At level 1: tree-automatic graphs
  - Above: a *strict* hierarchy of tree-automatic-like graphs
- Our aim:
  - Characterize these graphs directly using automata
  - Characterize their traces
- This talk : spend time explaining levels 1 and 2

# Outline

# Outline

# Outline

# Word-automatic graphs

Let $u$, $v$ be finite words over alphabet $C$

- Padding of $u$ : $u^\diamond(i) = u(i)$ if $i \in \mathrm{dom}(u)$, $\diamond$ otherwise
- Overlap of $u$ and $v$ : $u \otimes v : \mathrm{dom}(u) \cup \mathrm{dom}(v) \to (C \cup \diamond)^2$ such that $u \otimes v(i) = (u^\diamond(i), v^\diamond(i))$

A relation $R$ is word-automatic if $\{u \otimes v \mid (u, v) \in R\}$ is regular (i.e. accepted by a finite automaton)

A graph $G$ is word-automatic if all its edge relations are

# Examples of automatic graphs

Infinite grid of dimension $k$

- Vertices : words of the form $a_1^{n_1} \ldots a_k^{n_k}$ representing coordinate vectors $(n_1, \ldots, n_k)$
- Edges :

$$L_i = \{a_1^{n_1} \ldots a_k^{n_k} \otimes a_1^{n_1} \ldots a_i^{n_i+1} \ldots a_k^{n_k} \mid n_1, \ldots, n_k \geq 0\}$$

# Examples of automatic graphs

Full binary tree with "equal length" predicate

- Vertices : $\{a, b\}^*$
- Edges :
  - $L_a = \{u \otimes ua \mid u \in \{a, b\}^*\}$
  - $L_b = \{u \otimes ub \mid u \in \{a, b\}^*\}$
  - $L_\sim = \{u \otimes v \mid u, v \in \{a, b\}^*, |u| = |v|\}$

# Tree-automatic graphs

Let $s$, $t$ be finite binary $C$-labelled trees

- Padding of $t$ : $t^\diamond(u) = t(u)$ if $u \in \mathrm{dom}(t)$, $\diamond$ otherwise
- Overlap of $s$ and $t$ : $s \otimes t : \mathrm{dom}(s) \cup \mathrm{dom}(t) \to (C \cup \diamond)^2$ such that $s \otimes t(u) = (s^\diamond(u), t^\diamond(u))$

A relation $R$ is tree-automatic if $\{s \otimes t \mid (s,t) \in R\}$ is regular (i.e. accepted by a finite tree automaton)

A graph $G$ is tree-automatic if all its edge relations are

# Finite tree automata

A finite (binary) tree automaton over alphabet $C$ consists in:

- A finite set of control states $Q$, some of which are root states, and some leaf states
- A finite set of transitions of the form $(p, c, q, r)$ or $(p, c)$ with $p, q, r \in Q$ and $c \in C$

# Finite tree automata

A $C$-labelled tree $t$ is accepted if it can be labelled by states in such a way that

- The root of $t$ is labelled by a root state
- For each leaf labelled $p$ and $c$ there exists a transition $(p, c)$
- For each internal node labelled $p$ and $c$, with children labelled $q$ and $r$, there exists a transition $(p, c, q, r)$

# Example of tree-automatic graph

Given $A \subseteq \{a, b\}^*$, write $t_A$ the smallest binary tree with all positions in $A$ marked

"Weak powerset" graph of the full binary tree

- Vertices : all $t_A$ for finite $A$
- Edges :
  - $L_a = \{t_{\{u\}} \otimes t_{\{ua\}} \mid u \in \{a, b\}^*\}$
  - $L_b = \{t_{\{u\}} \otimes t_{\{ub\}} \mid u \in \{a, b\}^*\}$
  - $L_\subseteq = \{t_A \otimes t_B \mid A \subseteq B\}$

# Outline

# Monadic second order logic (MSO)

Language consisting of :

- first-order variables $x, y, \ldots$ denoting elements
- second-order variables $X, Y, \ldots$ denoting sets
- atomic predicates $R(x, y)$, $x = y$, $x \in X$
- Boolean connectives $\wedge, \vee, \neg$
- first- and second-order quantification

Example (over binary relation $R$):

$$\mathrm{Reach}(s, t) \equiv \forall X \Big( s \in X \\ \wedge \forall x \forall y (x \in X \wedge xRy \Rightarrow y \in X) \Big) \Rightarrow t \in X$$

# MSO interpretations

Let:

- $G$ be a $\Sigma$-labelled graph, $\Gamma$ a finite set
- $\delta(x)$, $\phi_a(x, y)$ for all $a \in \Gamma$ be MSO-formulas over $\Sigma$-labelled graphs
- $J = \left(\delta(x), (\phi_a(x, y))_{a \in \Gamma}\right)$

$J$ is called an MSO-interpretation and

$$J(G) = \{u \xrightarrow{a} v \mid G \models \delta(u) \wedge \delta(v) \wedge \phi_a(u, v)\}$$

is the $\Gamma$-graph interpreted in $G$ via $J$

# Finite sets interpretations

Let:

- $G$ be a $\Sigma$-labelled graph, $\Gamma$ a finite set
- $\delta(X)$, $\phi_a(X, Y)$ for all $a \in \Gamma$ be WMSO formulas over $\Sigma$-labelled graphs
- $J = \Big(\delta(X), (\phi_a(X, Y))_{a\in\Gamma}\Big)$

$J$ is called an finite sets interpretation and

$$J(G) = \{U \xrightarrow{a} V \mid G \models \delta(U) \wedge \delta(V) \wedge \phi_a(U, V)\}$$

is the $\Gamma$-graph interpreted in $G$ via $J$

# Examples (revisited)

All previous examples can be finite-sets interpreted either in $(\mathbb{N}, \leq)$ or in the full binary tree $\Delta_2$

Grid :

- $\delta_X$ : set of distinct numbers $\{n_1, \ldots, n_k\}$ encodes tuple $(n_1 - 1, n_2 - n_1 - 1, \ldots, n_k - n_{k-1} - 1)$
- $\phi_i(X, Y)$ ensures that the smallest $i - 1$ elements of $X$ and $Y$ coincide, and all others are incremented by 1 in $Y$

# Examples (revisited)

All previous examples can be finite-sets interpreted either in $(\mathbb{N}, \leq)$ or in the full binary tree $\Delta_2$

Full binary tree with equal length predicate :

- Node $u$ is represented by $\{i \mid u(i) = b\} \cup \{|u|\}$
- $\phi_a(X, Y)$ checks that $Y = X \setminus \{\max(X)\} \cup \{\max(X) + 1\}$
- $\phi_b(X, Y)$ checks that $Y = X \cup \{\max(X) + 1\}$
- $\phi_\sim(X, Y)$ checks that $\max(X) = \max(Y)$

# Examples (revisited)

All previous examples can be finite-sets interpreted either in $(\mathbb{N}, \leq)$ or in the full binary tree $\Delta_2$

Weak powerset of $\Delta_2$ :

- $t_A$ (tree with all nodes in $A$ marked) represented by... $A$ !
- $\phi_a(X, Y)$ holds iff $X = \{u\}$ and $Y = \{ua\}$
- $\phi_b(X, Y)$ holds iff $X = \{u\}$ and $Y = \{ub\}$
- $\phi_\subseteq(X, Y)$ holds iff $X \subseteq Y$

# Outline

# Interpreting tree-automatic graphs

Proposition: [CL07] Tree-automatic graphs coincide with the finite-sets interpretations of $\Delta_2$

$\supseteq$: For each formula $\phi(X, Y)$ :

- From $\phi(X, Y)$, build as usual an equivalent parity automaton over $\Delta_2$ annotated by $\{0, 1\}^2$
- Convert into an automaton over finite trees containing all positions in $X$ and $Y$ (finite sets !)
- Below, it suffices to know from which states the parity automaton accepts $\Delta_2$ to crop the computation

# Interpreting tree-automatic graphs

Proposition: [CL07] Tree-automatic graphs coincide with the finite-sets interpretations of $\Delta_2$

$\subseteq$: For each tree automaton over $C^2$,

- Reduce $C$ to a singleton by coding (patterns in the tree's structure)
- Represent any (finite) tree by its domain (finite set $\subseteq \{a, b\}^*$)
- Build a WMSO formula satisfied in $\Delta_2$ by pairs of sets encoding accepted overlaps of pairs of trees

# Outline

# Outline

# A few words on the infinite binary tree ($\Delta_2$)

Close connection with pushdown automata :

- Set of paths of the full $\{a, b\}$ tree : $\{a, b\}^*$
- Positions may be used to represent stack contents over stack alphabet $\{a, b\}$
- MSO interpretations yield the transition graphs of pushdown automata (PDA)
- FS interpretations yield the tree-automatic graphs

Decidable MSO theory

Generalizations exist for more general pushdown automata accessing nested stacks of stacks

# Order 2 pushdown stacks (2-stacks)

Let $A$ be a finite stack alphabet

- a stack is a sequence $[a_1 \dots a_\ell]$ with $a_i \in A$
- a 2-stack is a sequence $[s_1 \dots s_\ell]$ with $s_i$ a stack

Allowed 2-stack operations:

- $push_1^a$: add $a$ at the top of the topmost stack
- $pop_1^a$: remove $a$ from the top of the topmost stack
- $push_2$: duplicate the topmost stack
- $pop_2$: destroy the topmost stack

# Order 2 pushdown automata

Definition: an order 2 pushdown automaton (2-PDA) is a finite-state automaton with an auxiliary 2-stack, with transitions of the form:

*from p, if top symbol is a, move to q and apply op, reading b*

$$p, a \xrightarrow{b} q, op$$

- All operations chosen in $\{push_1, pop_1, push_2, pop_2\}$
- Acceptance by final state
- If $b$ is $\varepsilon$, then all other $(p, a)$ transitions also labelled $\varepsilon$
- Deterministic or $\varepsilon$-free versions less expressive

# Pushdown graphs and trees

A 2-PDA $\mathcal{A}$ can be used to generate a language, or:

- A configuration graph (with $\varepsilon$-transitions):
  $(p, s) \xrightarrow{b} (q, s')$ if $(p, top(s) \xrightarrow{b} q, op) \in \mathcal{A},\ s' = op(s)$
- A transition graph ($\varepsilon$-closure of the configuration graph)
- A tree (unfolding of the transition graph)

Whenever $\mathcal{A}$ is deterministic, so are the above structures

# The "order 2 treegraph" ($\Delta_2^2$)

Definition: vertices corresponding to all 2-stacks, edges representing operations $push_1^a$, $push_1^b$ and $push_2$

Close connection with 2-PDA :

- Walks between $s$ and $t$ (allowing some backward edges) encode sequences of 2-stack operations yielding $t$ from $s$
- MSO interpretations of this graph yield the transition graphs of order 2 pushdown automata (2-PDA)

Decidable MSO theory

Definition: a graph $G$ is 2-(tree-)automatic if there exists a finite set interpretation $J$ such that $G = J(\Delta_2^2)$

# Outline

# An order 2 generator tree

Remarks:

- No pure automata-based characterization of 2-TA
- $\Delta_2^2$ is not a tree! $\rightarrow$ Look for another generator

Definition: Let $T_2^2$ be the unfolding of $\Delta_2$ with added backward edges (labelled $\bar{a}, \bar{b}$)

Properties:

- There is an MSO-int. $J$ such that $\Delta_2^2 = J(T_2^2)$
- Paths from the root of $T_2^2$ encode sequences of stack operations which are well-defined on $[\ ]_1$

# Finite tree automata over $T_2^2$

Idea:

- Use $T_2^2$ as a fixed enclosing domain to define binary relations over finite trees
- Define tree automata running on finite $C^2$-labelled prefixes of $T_2^2$

Problem:

- $T_2^2$ has infinitely many non-isomorphic subtrees
- Finite tree automata lack expressiveness w.r.t FSI

Solution: allow tree automata to test the stack content reached after a sequence of operations

# Tree automata with oracles

Definition: finite tree automata with transitions of the form $(p, c, O, q, r)$ with $O$ a regular language over $\{a, b\}$

A $C$-labelled tree $t$ is accepted if it can be labelled by states in such a way that

- The root of $t$ is labelled by a root state
- Each leaf is labelled by a leaf state
- For each internal node labelled $p$ and $c$ and reachable from the root by $w$, with children labelled $q$ and $r$, there exists a transition $(p, c, O, q, r)$ with $w([\ ]_1) \in O$

# 2-tree automatic relations

Definition: a relation $R$ is 2-tree-automatic if

- Its support are finite trees $t$ with $\text{dom}(t) \subset \text{dom}(T_2^2)$
- The set $\{s \otimes t \mid (s, t) \in R\}$ is accepted by a tree automaton with oracles

No change to the notion of padding

A graph is 2-tree-automatic if each of its edge relations is

# Outline

# Equivalence proof

Proposition: Given a graph $G$, the following statements are equivalent

1. There exists a FSI $J$ such that $G = J(\Delta_2^2)$
2. There exists a FSI $J$ such that $G = J(T_2^2)$
3. For each edge label $a$, the $a$-labelled edge relation in $G$ is accepted by a finite tree automaton with oracles

# Equivalence proof

$1 \iff 2$:

- For all FSI $J$ there exists a FSI $J'$ such that $J(\Delta_2^2) = J'(T_2^2)$
- The converse also holds

# Equivalence proof

$2 \implies 3$:

- From each $\phi(X, Y)$ in $J$, build an equivalent parity automaton $A$ over $T_2^2$ annotated by $\{0, 1\}^2$
- Lemma: for any state $p$ of $A$, there exists a regular language $O_p$ such that $A$ accepts $T_2^2$ from node $w$ and state $p$ iff $w([\,]_1) \in O_p$
- Convert $A$ into a tree automaton with oracles $O_p$ over finite prefixes of $T_2^2$ containing all positions in $X$ and $Y$

$3 \implies 2$: As previously, transforming tests into equivalent *WMSO* formulas

# Outline

# Traces of automatic graphs

Extension of previous results on traces:

- The traces of automatic graphs are the context-sensitive languages (linearly bounded Turing machines) [MS01, Ris02, CM06]
- The traces of tree-automatic graphs are the class DTIME($2^{O(m)}$) (alternating LBM, ASPACE($m$)) [Mey07]

Using similar techniques, show that the languages of 2-TA graphs form the class DTIME($2^{2^{O(m)}}$) accepted by ASPACE($m$)-P machines

# Towards a tree-automatic hierarchy

Similar classes of $n$-tree-automatic graphs are defined by finite-set interpretations from $\Delta_2^n$

Work in progress:

- Define corresponding trees $T_2^n$
- Define tree automata with oracles for level $n$ (difficult)
- Generalize the result on traces to all levels

Possible implications:

- New proof of strictness based on traces
- No known results about the classes obtained using collapsible stacks

Didier Caucal.
On infinite terms having a decidable monadic theory.
In *MFCS*, volume 2420 of *LNCS*, pages 165–176. Springer, 2002.

Thomas Colcombet and Christof Löding.
Transforming structures by set interpretations.
*Logical Methods in Computer Science (LMCS)*, 3(2), 2007.

Arnaud Carayol and Antoine Meyer.
Context-sensitive languages, rational graphs and determinism.
*Logical Methods in Computer Science*, 2(2), 2006.

Arnaud Carayol and Stefan Wöhrle.
The caucal hierarchy of infinite graphs in terms of logic and higher-order pushdown automata.
In *FSTTCS*, volume 2914 of *LNCS*, pages 112–123. Springer, 2003.

Teodor Knapik, Damian Niwinski, and Paweł Urzyczyn.
Higher-order pushdown trees are easy.
In *FoSSaCS*, volume 2303 of *LNCS*, pages 205–222. Springer, 2002.

Antoine Meyer.
Traces of term-automatic graphs.
In *MFCS*, volume 4708 of *LNCS*, pages 489–500. Springer, 2007.

Christophe Morvan and Colin Stirling.
Rational graphs trace context-sensitive languages.
In *MFCS*, volume 2136 of *LNCS*, pages 548–559. Springer, 2001.

Chloe Rispal.
The synchronized graphs trace the context-sensitive languages.
*Electr. Notes Theor. Comput. Sci.*, 68(6):55–70, 2002.