Finite Automata and Infinite Transition Systems

Wolfgang Thomas

RWTH Aachen University

thomas@informatik.rwth-aachen.de

CANT 2006, Liège

Classical view of automata

Finite automaton: model of finite-state discrete system

Behaviour given by set of finite or infinite words

$\longrightarrow$ **regular languages, regular ω-languages**

Applications in verification based on two roles of automata:

1. **finite system model (Kripke structure, labelled transition system)**
2. **representation of a specification**

The nontrivial side is item 2.

Basis for 2.: Transformation of logical formualas into automata

Illustration: LTL model-checking

Given a pointed finite transition system $\mathcal{M}$ and a temporal logic formula φ

$\mathcal{M} \models \varphi$?

Does every path from the origin through $\mathcal{M}$ satisfy the condition φ?

In other words:

Is there a path through $\mathcal{M}$ which violates φ (a bug)?

View $\mathcal{M}$ as a Büchi automaton $\mathcal{A}_{\mathcal{M}}$, and transform $\neg\varphi$ into a Büchi automaton $\mathcal{A}_{\neg\varphi}$

Check whether $L(\mathcal{A}_{\mathcal{M}}) \cap L(\mathcal{A}_{\neg\varphi})$ is nonempty (existence of a bug)

Infinite-state model-checking

We concentrate on the systems models as the nontrivial side

Infinite transition systems as models, and (for the start) quite simple specifications.

Emphasis on the reachability problem: Given two states, is the second reachable from the first by a finite path?

More general, we consider simple classical logics (first-order logic, monadic second-order logic)

The aim of these lectures: To give an overview of

- **finitely presented infinite transition systems**
- **fundamental results on the solvability of algorithmic problems over these systems**

The new role of automata

We use automata to represent discrete systems in a more indirect way than in the classical setting:

- **System states are words**
  State properties are (regular) sets of words
- **Transitions are pairs of words**
  Transition relations are automaton definable word relations

The unboundedness of the behaviour (words of unbounded length) is

- **no more used for the study of system evolution over time**
- **but is used for the description of an infinite state space**

Roadmap

Transition graphs:

- **Rational graphs**
- **Automatic graphs**
- **pushdown graphs, prefix rewriting graphs**
- **ground tree rewriting graphs**
- **higher-order pushdown graphs**

Algorithmic problems:

- **Checking truth of first-order formulas**
- **Solving the reachability problem**
- **Checking truth of monadic second-order formulas**

Transition Graphs

Format of edge-labelled and vertex-labelled transition graphs:

$$G = (V, (E_a)_{a\in\Sigma}, (P_b)_{b\in\Sigma'})$$

We write E for the union of the E_a.

Central examples :

- ***Kripke structures*** $G = (V, E, (P_b))$
- **the ordering** $(\mathbb{N}, <)$ **of the natural numbers,**
- **the *binary tree*** $T_2 = (\{0,1\}^*, S_0, S_1)$ **where** $S_i = \{(w, wi) \mid w \in \{0,1\}^*\}$
- **the n-ary tree** $T_n = (\{0,\ldots,n-1\}^*, S_0^n, \ldots, S_{n-1}^n\}$
- **finite automata** $\mathcal{A} = (Q, (E_a)_{A\in\Sigma}, I, F)$

First-order logic FO

Atomic formulas $x = y$, $E_a(x, y)$, $P_b(x)$

Connectives: $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$, quantifiers $\exists, \forall$

Expressibe: ”there is a cycle with three elements”

$$\exists x \exists y \exists z (\neg x = y \wedge \neg x = z \wedge \neg y = \\ z \wedge E(x, y) \wedge E(y, z) \wedge E(z, x))$$

Not expressible: Reachability relation E^*

$$E^*(u, v) \quad \Leftrightarrow \\ \exists v_0 \ldots v_k \in V : v_0 = u, (v_i, v_{i+1}) \in E \;\; (i < k), v_k = v$$

FO(R) = FO extended by symbol for E^*

by adjoining a symbol for the reachability relation E^* to the signature.

Monadic second-order logic MSO

obtained from FO by adjoining variables $X, Y, \ldots$ for sets of vertices

and atomic formulas $X(y)$ ("vertex y is in the set X")

$E^*(x, y)$ expressed as "each set which contains x and is closed under E must contain y"

MSO encompasses most standard temporal logics.

Rational Graphs

A relation $R \subseteq \Gamma^* \times \Gamma^*$ is *rational*

if it recognized by a nondeterministic automaton with two reading heads on an input (u, v)

Example: Suffix relation $\{(u, v) \mid u$ is a suffix of $v\}$.

Automaton progresses with its reading head on the second component v until it guesses that the suffix u starts

Then the u and the remaining prefix of v are checked for equality letter by letter.

Rational transition graph: $G = (V, (E_a)_{a \in \Sigma}, (P_b)_{b \in \Sigma'})$

where V and each $P_b \subseteq \Gamma^*$ is regular

and each $E_a \subseteq \Gamma^* \times \Gamma^*$ is rational.

An Undecidability Result

For each instance $(\overline{u}, \overline{v})$ of PCP (Post's Correspondence Problem) one can construct a rational graph $G_{(\overline{u}, \overline{v})}$ such that $(\overline{u}, \overline{v})$ has a solution iff $G_{(\overline{u}, \overline{v})}$ satifies $\exists x E(x, x)$.

Given a PCP-instance $(\overline{u}, \overline{v}) = ((u_1, \ldots, u_m), (v_1, \ldots, v_m))$

we specify a rational graph $G_{(\overline{u}, \overline{v})} = (V, E)$

$V = \Gamma^*$**.**

Edges: $(u_{i_1} \ldots u_{i_k}, v_{i_1} \ldots v_{i_k})$ where $i_j \in \{1, \ldots, m\}$ and $k \geq 1$.

Proof continued

$V = \Gamma^*$.

Edges: $(u_{i_1} \dots u_{i_k}, v_{i_1} \dots v_{i_k})$ **where** $i_j \in \{1, \dots, m\}$ **and** $k \geq 1$.

Edge relation is rational:

Automaton guesses indices $i_1, \dots, i_k$ **and checks the input word accordingly.**

There is an edge (w, w) **iff PCP-instance** $(\overline{u}, \overline{v})$ **has a solution (namely by the word** w**).**

Checking FO-properties

We have checked undecidability of uniform model-checking for rational graphs and FO-logic:

There is no algorithm which, given a presentation of a rational graph G and a first-order sentence φ, decides whether $G \models \varphi$.

We construct a single rational graph with an undecidable first-order theory.

Use a universal Turing machine M and its undecidable halting problem: Given input x, does M halt on x?

A Turing machine M with input word x is converted into a PCP-instance $((u_1, \ldots, u_m), (v_1, \ldots, v_m))$ over an alphabet A whose letters are the states and tape letters of M and a symbol #

TM halting problem and special PCP

$u_1 = c(x) := \# q_0 a_1 \dots a_n,\ v_1 = \#,$

$u_2, \dots, u_m, v_2, \dots, v_m$ **only depend on** M.

M **halts on input** x

iff the PCP-instance $((c(x), u_2, \dots, u_m), (\#, v_2, \dots, v_m))$ **has a special solution (starting with** $i_1 = 1$**)**

We modify the graph $G_{(\overline{u}, \overline{v})}$

E **contains** (w_1, w_2) **iff there are indices** $i_2, \dots, i_k$ **and a word** x **such that** $w_1 = c(x) u_{i_2} \dots u_{i_k}$ **and** $w_2 = \# v_{i_2} \dots v_{i_k}$.

G **is rational**

$G \models \exists z E(z, z)$ **iff some PCP instance** $((c(x), u_2, \dots, u_m), (\#, v_2, \dots, v_m))$ **has a solution.**

Refining the construction

We have to refine the formula for a reference to the TM input words x.

Add further vertices and edge relations E_a for $a \in A$.

Finishing the proof

A $\underline{c(x)}$-labelled path leads to each G-vertex W with prefix $c(x)$

w has has an edge back to itself

iff a special solution for the PCP-instance $((c(x), u_2, \ldots, u_m), (\#, v_2, \ldots, v_m))$ exists

The new vertices are words over a copy $\underline{A}$ of the alphabet A (consisting of the underlined versions of the A-letters).

G' is rational.

Finishing the construction

PCP-instance $((c(x), u_2, \ldots, u_m), (\#, v_2, \ldots, v_m))$ has a special solution iff

in G' there is a path, labelled with the word $c(x)$, from the vertex ε to a vertex which has an edge back to itself.

Turing machine M halts on input x iff $G' \models \varphi_x$.

Synchronous Processing of word tuples

Use a more restricted processing of an input (w_1, w_2) by an automaton:

Scanning of the two component words letter by letter

Represent (ab, abb) by the word $\binom{a}{b}\binom{b}{b}\binom{\$}{b}$

$[w_1, w_2] :=$ the word associated with (w_1, w_2)

$[w_1, w_2] \in ((\Gamma \times \Gamma) \cup ((\Gamma \cup \{\$\}) \times \Gamma) \cup (\Gamma \times (\Gamma \cup \{\$\})))^*$

For $R \subseteq \Gamma^* \times \Gamma^*$ define $L_R = \{[w_1, w_2] \mid (w_1, w_2) \in R\}$.

Automatic Graphs

R **is called** ***automatic*** **if** L_R **is regular.**

Example:

- **The prefix relation is automatic.**
- **The suffix relation is not automatic.**

A graph $(V, (E_a)_{a\in\Sigma}, (P_b)_{b\in\Sigma'})$ ***automatic***

if V **and each** $P_b \subseteq V$ **are regular languages (over some auxiliary alphabet** Γ**)**

and each edge relation $E_a \subseteq \Gamma^* \times \Gamma^*$ **is automatic.**

Examples

The infinite two-dimensional grid

$G_2 = (\mathbb{N} \times \mathbb{N}, E_a, E_b)$

(with E_a-edges $((i,j),(i,j+1))$ and E_b-edges $((i,j),(i+1,j))$)

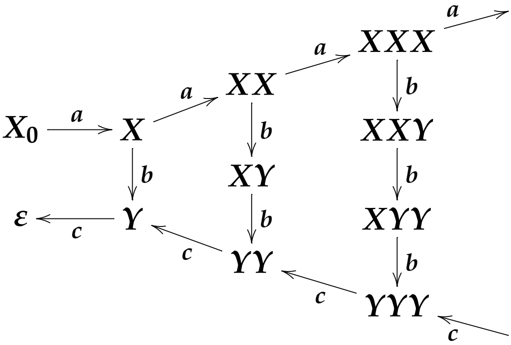
G_2 is automatic:

Use $\Gamma = \{a,b\}$, $V = a^*b^*$

$E_a = \{(X^iY^j, X^iY^{j+1}) \mid i,j \geq 0\}$

$E_b = \{(X^iY^j, X^{i+1}Y^j) \mid i,j \geq 0\}$

An infinite acceptor



This infinite automaton recognizes the context-sensitive language $\{a^i b^i c^i \mid i > 0\}$

More examples

- $T_2' = (\{0,1\}^*, S_0, S_1, \leq, \lambda)$
  with $\leq$ as the prefix relation
  $\lambda = \{(u, v \in \{0,1\}^* \mid \ |u| = |v|\}$.

- **For a TM M with state set Q and tape alphabet Γ:**
  $G_M = (\triangleleft\Gamma^* Q \Gamma^* \triangleright, E_M)$**, the configuration graph of M**

Reachability over automatic graphs

There is an automatic graph G such that over G the relation E^* is undecidable.

Let U be a universal Turing machine, modified such that termination is possible in a unique configuration c_{halt}

Consider G_U.

Then

TM M accepts the input word w

iff over G_U, from the configuration $q_0\text{code}(M)w$ the configuration c_{halt} can be reached.

FO over automatic graphs

The FO-theory of an automatic graph is decidable.

Corollary

The automatic graphs form a proper subclass of the class of rational graphs

$G = (V, (E_a)_{a \in \Sigma}, (P_b)_{b \in \Sigma'})$ **be automatic.**

Suppose we have an automatic presentation over Γ.

Show inductively over FO-formulas $\varphi(x_1, \ldots, x_n)$:

$R_\varphi = \{(w_1, \ldots, w_n) \mid (G, w_1, \ldots, w_n) \models \varphi(x_1, \ldots, x_n)\}$
is automatic

Atomic case: clear by presentation of G.

Induction step: Boolean connectives easy.

Existential quantifier

Typical case:

Given binary R recognized by the finite automaton $\mathcal{A}$ with final state set F.

Show that $S = \{w_1 \in \Gamma^* \mid \exists w_2 : (w_1, w_2) \in R\}$ is automatic.

Construct $\mathcal{A}'$ from $\mathcal{A}$ by

- **projection of the input letters to the first components**
- **extension of F to a set F'**
  $q \in F'$ iff some path of letters $(\$, a)$ leads to F

Note the blow-up of automaton size.

FO-theory of addition

The structure $(\mathbb{N}, +)$ is automatic.

Use reversed binary representation of numbers.

The addition realtion $x_1 + x_2 = x_3$ is automatic:

Check from lowest bit onwards that the addition is done correctly.

Consequence:

Presburger Arithmetic, the FO-theory of addition, is decidable.

FO - FO(R) - MSO

We have found an automatic graph where reachability is undecidable.

Consequence: Undecidability of MSO-theory.

We now look for restrictions where the MSO-theory is decidable.

An example separating FO(R) and MSO is the infinite grid.

The infinite grid

The FO(R)-theory of the infinite grid is decidable, but its MSO-theory undecidable.

Proof idea:

To each TM (working on a left-bounded tape) associate an MSO-sentence φ_M such that

M halts on the empty tape iff $G_2 \models \varphi_M$

Code the sequence of the M-configurations by a coloring of the grid, row by row

Assume that a halting computation keeps repeating the stop configuration and thus generates a coloring of the whole grid.

More details

For a TM with m states and n tape symbols need $m + n$ colors

φ_M says that the coloring codes a halting computation:

There are sets $X_1, \ldots, X_{m+n}$ which form a partition of the universe such that the induced coloring

- **represents the initial configuration (on the empty tape) in the first row,**
- **respects the transition table of M between any two successive rows,**
- **contains a vertex which is colored by a halting state.**

Pushdown Automata

Format: $\mathcal{P} = (P, \Sigma, \Gamma, p_0, Z_0, \Delta)$, **where**

- P **is the finite set of control states,**
- Σ **the input alphabet,** Γ **the stack alphabet,**
- p_0 **the initial control state,** $Z_0 \in \Gamma$ **the initial stack symbol,**
- $\Delta \subseteq P \times \Sigma \times \Gamma \times \Gamma^* \times P$ **the transition relation**

A PD-configuration is a pair (control state, stack content) written as a word from $P\Gamma^*$**.**

Pushdown Graphs

Notation for one-step transformation of PD-configuration using the transition (p, a, γ, v, q): $p\gamma w \vdash qvw$

A graph $G = (V, (E_a)_{a \in \Sigma})$ is called *pushdown graph* if it is the transition graph of the reachable global states of an ε-free pushdown automaton.

The graph $G = (V, (E_a)_{a \in \Sigma})$ is specified as follows:

- $V = \{qv \in P\Gamma^* \mid p_0 Z_0 \vdash^* qv\}$
- **E_a is the set of all pairs $(p\gamma w, qvw)$ from V^2 for which there is a transition (p, a, γ, v, q) in Δ.**

Edge relation E is the one-step derivation relation $p_1 w_1 \vdash p_2 w_2$ over $\mathcal{P}$

E^* is the derivability relation $\vdash^*$.

Prefix-recognizable Graphs

Discard control states generalize prefix rewrite rules

Generalized rewriting rules: $U_1 \xrightarrow{a} U_2$ with regular sets U_1, U_2 of words

as a definition of (generally infinitely many) rewrite triples (u_1, a, u_2) with $u_1 \in U_1$ and $u_2 \in U_2$.

$G = (V, (E_a)_{a \in \Sigma})$ is *prefix-recognizable* if for some finite system $\mathcal{S}$ of such generalized prefix rewriting rules $U_1 \xrightarrow{a} U_2$ over an alphabet Γ:

- **$V \subseteq \Gamma^*$ is a regular set,**
- **E_a consists of the pairs $(u_1 w, u_2 w)$ where $u_1 \in U_1$, $u_2 \in U_2$ for some rule $U_1 \xrightarrow{a} U_2$ from $\mathcal{S}$, and $w \in \Gamma^*$.**

Example

$(\mathbb{N}, \mathrm{Succ}, <)$ **is prefix recognizable.**

Represent numbers by stroke sequences

So $V = |^*$

Succ **is defined by** $\varepsilon \xrightarrow{\mathrm{Succ}} |$

$<$ **is defined by** $\varepsilon \xrightarrow{<} |^+$**.**

Comparing classes of infinite graphs

The pushdown graphs, prefix-recognizable graphs, automatic graphs, and rational graphs constitute, in this order, a strictly increasing inclusion chain of graph classes.

**Trivial: Each PD-graph is prefix-recognizable,
and each automatic graph is rational.**

Each prefix-recognizable graph is automatic:

**Consider a prefix-recognizable graph,
proceed to the isomorphic graph with reversed words and suffix rewriting.**

An edge (wu_1, wu_2) is generated by a suffix rewriting rule $U_1 \xrightarrow{a} U_2$, with regular U_1, U_2 and $u_1 \in U_1$, $u_2 \in U_2$.

This can be checked letter-to-letter

Strictness of the inclusions

- **Vertices of a PD-graph have bounded outdegree, which fails for prefix-recognizable graphs**
- **Later: Over a prefix-recognizable graph, reachability is decidable, which fails for automatic graphs**
- **FO-properties are decidable over automatic graphs, which fails for rational graphs**

Büchi's Theorem on Regular Canonical Systems

Given a pushdown automaton $\mathcal{P} = (P, \Sigma, \Gamma, p_0, Z_0, \Delta)$ and $T \subseteq P\Gamma^*$

$$\mathrm{pre}^*(T) := \{pv \in P\Gamma^* \mid \exists qw \in T : pv \vdash^* qw\}$$

Given a pushdown automaton $\mathcal{P} = (P, \Sigma, \Gamma, p_0, Z_0, \Delta)$ and a finite automaton recognizing a set $T \subseteq P\Gamma^*$, one can compute a finite automaton recognizing $\mathrm{pre}^*(T)$.

Deciding $p_1w_1 \vdash^* p_2w_2$:

Set $T = \{p_2w_2\}$ and check whether the automaton recognizing $\mathrm{pre}^*(T)$ accepts p_1w_1.

Büchi's motivation

Saturation Method: the Idea

Specify the set $T \subseteq P\Gamma^*$ by “P-automaton”

Use P as the set of initial states of $\mathcal{A}$.

No re-visits to initial state (*normalized* P-automata).

Saturation Algorithm

```
Input: $P$-automaton $\mathcal{A}$, pushdown system $\mathcal{P} = (P, \Gamma, \Delta)$
$\mathcal{A}_0 := \mathcal{A}$, $i := 0$
REPEAT:
    IF $pa \longrightarrow p'v \in \Delta$ and $\mathcal{A}_i : p' \xrightarrow{v} q$ THEN
        add $(p, a, q)$ to $\mathcal{A}_i$ and obtain $\mathcal{A}_{i+1}$
        $i := i + 1$
UNTIL no transition can be added
$\overline{\mathcal{A}} := \mathcal{A}_i$
Output: $\mathcal{A}'$
```

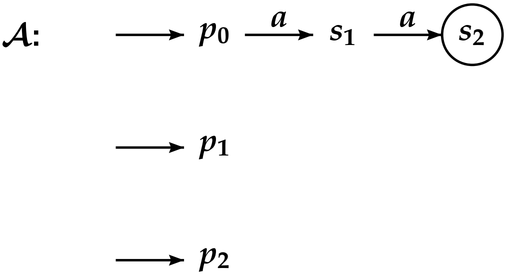
Example

$\mathcal{P} = (P, \Gamma, \Delta)$ **with** $P = \{p_0, p_1, p_2\}$, $\Gamma = \{a, b, c\}$,

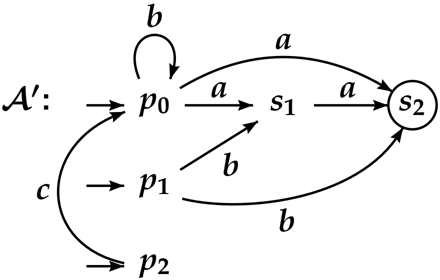
$\Delta = \{(p_0 a \rightarrow p_1 ba), (p_1 b \rightarrow p_2 ca), (p_2 c \rightarrow p_0 b), (p_0 b \rightarrow p_0)\}$

$T = \{p_0 aa\}$.

P-automaton for T:



Result



So for $T = \{p_0 aa\}$**:**

$$pre^*(T) = \; p_0 b^*(a + aa) \; + \; p_1 b \; + \; p_1 ba \; + \; p_2 c b^*(a + aa)$$

Correctness

$pw \in \mathbf{pre}^*(T) \Rightarrow \mathcal{A}' : p \xrightarrow{w} F$

Use induction over the number $n \geq 0$ of steps to get to T:

$pw \longrightarrow^n ru \in T \;\Rightarrow\; \mathcal{A}' : p \xrightarrow{w} F$

$n = 0$**: obvious.**

Assume $pw \longrightarrow^{n+1} ru$ and $ru \in T$.

$paw' \longrightarrow p'vw' \longrightarrow^n ru$ **with** $w = aw'$ **with transition** $pa \longrightarrow p'v$**.**

By induction assumption: $\mathcal{A}' : p' \xrightarrow{vw'} F$
say $\mathcal{A}' : p' \xrightarrow{v} q \xrightarrow{w'} F$

Aaturation algorithm produces the transition (p, a, q)**,**
so pw is accepted by $\mathcal{A}'$.

Reverse direction

Show $\mathcal{A}' : p \xrightarrow{w} q \implies \exists p'w' \in P\Gamma^*$ **:** $\mathcal{A}$ **such that** $p' \xrightarrow{w'} q \wedge pw \vdash^* p'w'$

With $q \in F_{\mathcal{A}}$ **the claim follows.**

Note that $\mathcal{A} : p' \xrightarrow{w'} q$ **says that** $p'w' \in T$**.**

$\mathcal{A}_i$ **is the** i**-th** P**-automaton produced by tha saturation algorithm.**

Show inductively over i**:**

If $\mathcal{A}_i : p \xrightarrow{w} q$, **then** $\exists p'w' \in P\Gamma^*$ **such that** $\mathcal{A} : p' \xrightarrow{w'} q \wedge pw \vdash^* p'w'$

The case $i = 0$ **obvious.**

Induction Step

Assume $\mathcal{A}_{i+1} : p \xrightarrow{w} q$

$j :=$ number of applications of the $(i+1)$-st transition.

Show inductively over j:

$\exists p'w' \in P\Gamma^*$ such that $\mathcal{A} : p' \xrightarrow{w'} q \wedge pw \vdash^* p'w'$

Case $j = 0$ is obvious (no use of the $(i+1)$-th transition).

For case $j+1$, decopose w into $w = uau'$ with

$$\mathcal{A}_i : p \xrightarrow{u} p_1, \;\; \mathcal{A}_{i+1} : \quad \underbrace{p_1 \xrightarrow{a} q_1}_{(i+1)\text{-st transition}} \quad \text{and } \mathcal{A}_{i+1} : q_1 \xrightarrow{u'} q$$

By induction (on i): $pu \vdash^* p_1'u_1$ with $\mathcal{A} : p_1' \xrightarrow{u_1} p_1$.

**$\mathcal{A}$ is normalized: p_1 has no ingoing transitions,
hence $u_1 = \varepsilon$ and $p_1 = p_1'$; thus $pu \vdash^* p_1$.**

Induction step completed

$\mathcal{A}_i : p \xrightarrow{u} p_1$, $\mathcal{A}_{i+1}$: $\underbrace{p_1 \xrightarrow{a} q_1}_{(i+1)\text{-st transition}}$ and $\mathcal{A}_{i+1} : q_1 \xrightarrow{u'} q$

Saturation algorithm adds (p_1, a, q_1) to $\mathcal{A}_i$:

$\exists p_2$ and a pushdown rule $p_1 a \longrightarrow p_2 v$ with $\mathcal{A}_i : p_2 \xrightarrow{v} q_1$.

In the run on u', the ($i + 1$)-st transition is used $\leq j$ times.

By induction assumption on j , from $\mathcal{A}_{i+1} : p_2 \xrightarrow{v} q_1 \xrightarrow{u'} q$

infer $p'w'$ with $\mathcal{A} : p' \xrightarrow{w'} q$ and $p_2 v u' \vdash^* p'w'$.

Altogether: $pw = puau' \vdash^* p_1 a u' \vdash^* p_2 v u' \vdash^* p'w' (\in T)$.

Looking back, looking forward

We have introduced an efficient algorithm for solving the reachability problem over pushdown graphs

Now we treat more ambitious specifications, using the whole expressive power of MSO.

It will turn out that also this more general model-checking problem can be solved over pushdown graphs

Method: Use an initial decidability result and methods of transfer to pass from one model to another.

Rabin's Tree Theorem

The MSO-theory of the infinite binary tree T_2 is decidable.

The proof is difficult.

MSO-interpretation, first example:

Show Rabin's Tree Theorem for $T_3 = (\{0,1,2\}^*, S_0^3, S_1^3, S_2^3)$.

Idea: Obtain a copy of T_3 in T_2:

Consider T_2-vertices in $T = (10 + 110 + 1110)^*$.

Interpretation: Details

The element $(i_1 - 1) \dots (i_m - 1)$ of T_3 is coded by

$1^{i_1}0 \dots 1^{i_m}0$ in T_2.

Define the set of codes by

$\varphi(x)$: $\forall Y[Y(x) \wedge \forall y((Y(y10) \vee Y(y110) \vee Y(y1110)) \rightarrow Y(y)) \ \rightarrow Y(\varepsilon)]$

"x is in the closure of ε under 10-, 110-, and 1110-successors"

Define the 0-th, 1-st 2-nd successors:

$\{(w, w10) | w \in \{0,1\}^*\}$ is defined by

$\psi_0(x, y) := \varphi(x) \wedge \exists z(S_1(x, z) \wedge S_0(z, y))$

The structure $(\varphi^{T_2}, (\psi_i^{T_2})_{i=0,1,2})$ restricted to φ^{T_2} is isomorphic to T_3.

Interpretations

An MSO-interpretation of a structure $\mathcal{A} = (A, R^{\mathcal{A}}, \ldots)$ in a structure $\mathcal{B}$ is given by

- **a "domain formula" $\varphi(x)$**
- **for each relation $R^{\mathcal{A}}$ of $\mathcal{A}$, say of arity m, an MSO-formula $\psi(x_1, \ldots, x_m)$**

such that $\mathcal{A}$ is isomorphic to $(\varphi^{\mathcal{B}}, \psi^{\mathcal{B}}, \ldots$

Then there is a transformation MSO-sentence χ (in the signature of $\mathcal{A}$) to χ' (in the signature of $\mathcal{B}$) such that $\mathcal{A} \models \chi$ iff $\mathcal{B} \models \chi'$.

Consequence:

If $\mathcal{A}$ is MSO-interpretable in $\mathcal{B}$ and the MSO-theory of $\mathcal{B}$ is decidable, then so is the MSO-theory of $\mathcal{A}$.

Interpretation: Second Example

A pushdown graph is MSO-interpretable in T_2

Given pushdown automaton $\mathcal{A}$ with stack alphabet $\{1, \ldots, k\}$ and states $q_1, \ldots, q_m$.

Let $G_\mathcal{A} = (V_\mathcal{A}, E_\mathcal{A})$ be the corresponding PD graph.
$n := \max\{k, m\}$

Find an MSO-interpretation of $G_\mathcal{A}$ in T_n.

Represent configuration $(q_j, i_1 \ldots i_r)$ by the vertex $i_r \ldots i_1 j$ of T_n.

$\mathcal{A}$-steps lead to local moves in T_n.

E.g. a push step from vertex $i_r \ldots i_1 j$ to $i_r \ldots i_1 i_0 j'$.

These moves are easily definable in MSO, and so is reachability (from the initial vertex 11).

Some more detail

Prefix-recognizable graphs

Instead of describing a move from one word u_0w to v_0w

describe all admissible moves from a word uw to vw for a rule $U \longrightarrow V$ with $u \in U, v \in V$.

This can be done by describing successful runs of the automata $\mathcal{A}_U, \mathcal{A}_V$ on the path segments from uw to w and from vw to w.

Interpretations in tree-like structures

A prefix-recognizable graph is MSO-interpretable in T_2; thus, the monadic second-order theory of a prefix-recognizable graph is decidable.

An automatic graph is FO-interpretable in T_2'.

Proof idea for the case that the the alphabet Γ for the automatic presentation of the given graph G is $\{0,1\}$.

Find an FO-description of a regular set $V \subseteq \{0,1\}^*$.

Assume $V = L(\mathcal{A})$

Develop an FO- formula $\varphi(x)$ which expresses whether the vertex $x \in \{0,1\}^*$ is accepted by $\mathcal{A}$.

Illustration

External characterizations

A graph is MSO-interpretable in T_2 iff its is prefix-recognizable

A graph is FO-interpretable in T_2' iff it is automatic

Unfoldings

Given a graph $(V, (E_a)_{a\in\Sigma}, (P_b)_{b\in\Sigma'})$

the unforlding of G **from a given vertex** v_0 **is the following tree** $T_G(v_0) = (V', (E'_a)_{a\in\Sigma}, (P'_b)_{b\in\Sigma'})$**:**

- V' **consists of the vertices** $v_0 a_1 v_1 \dots a_r v_r$ **with** $(v_{i-1}, v_i) \in E_{a_i}$,
- E'_a **contains the pairs** $(v_0 a_1 v_1 \dots a_r v_r, v_0 a_1 v_1 \dots a_r v_r a v)$ **with** $(v_r, v) \in E_a$,
- P'_b **the vertices** $v_0 a_1 v_1 \dots a_r v_r$ **with** $v_r \in P_b$.

Muchnik-Theorem ***If the MSO-theory of*** G ***is decidable and*** v_0 ***is an MSO-definable vertex of*** G***, then the MSO-theory of*** $T_G(v_0)$ ***is decidable.***

Examples

Caucal's Hierarchy

- $\mathcal{T}_0 =$ **the class of finite trees**
- $\mathcal{G}_n =$ **the class of graphs which are MSO-interpretable in a tree of** $\mathcal{T}_n$
- $\mathcal{T}_{n+1} =$ **the class of unfoldings of graphs in** G_n

Each structure in the Caucal hierarchy has a decidable MSO-theory.

Three Facts

- The hierarchy is strictly increasing.
- There is an internal desciption of the graphs in the Caucal Hierarchy, in terms of higher-order pushdown automata ("higher-order pushdown graphs")
- Instead of MSO-interpretations, one can use Caucal's "inverse rational substitutions"

Examples of graphs in the Caucal Hierarchy

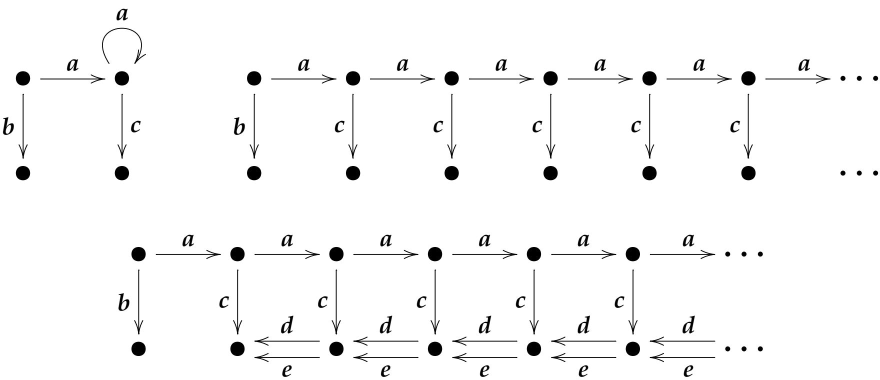
$\mathcal{G}_0$ is the class of finite graphs,

$\mathcal{T}_1$ contains the regular trees

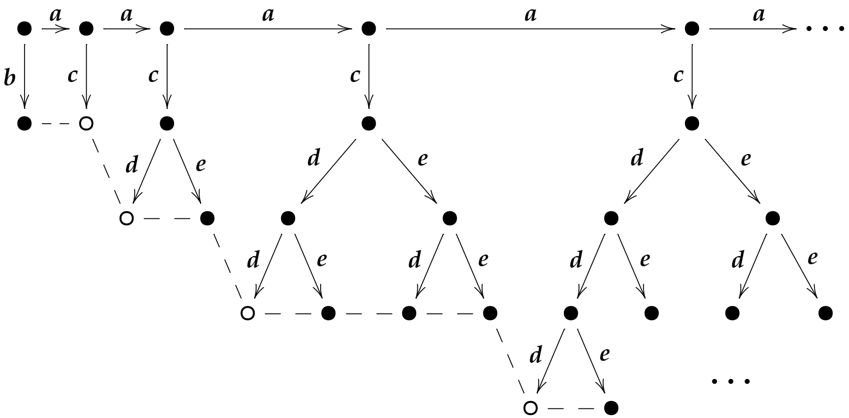
$\mathcal{G}_1$ contains the prefix-recognizable graphs

$\mathcal{T}_2$ contains the algrbaric trees

A finite graph, a regular tree, a PD graph



Unfolding again



Interpration in algebraic tree

We will produce a graph (V, E, P)

Use formulas such as $E_{d^*}(x, y)$.

V is defined by
$\varphi(x) = \exists z(E_b(z, x) \vee \exists y(E_c(z, y) \wedge E_{d^*}(y, x)))$.

E is defined by
$\psi(x, y) = \exists z \exists z'(\psi_1(x, y) \vee \psi_2(x, y) \vee \psi_3(x, y))$

where

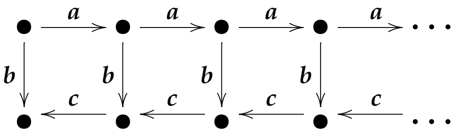
- $\psi_1(x, y) \;=\; E_a(z, z') \wedge E_b(z, x) \wedge E_c(z', y)$
- $\psi_2(x, y) \;=\; E_a(z, z') \wedge E_{ce^*}(z, x) \wedge E_{cd^*}(z', y)$
- $\psi_3(x, y) \;=\; E_{de^*}(z, x) \wedge E_{ed^*}(z, y)$

P is defined by $\chi(x) = \exists z \exists z'(E_c(z, z') \wedge E_{d^*}(z', x))$.

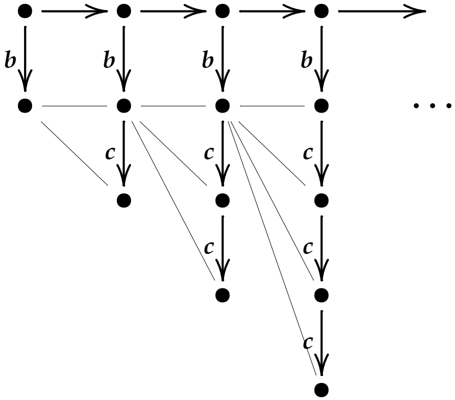
Factorial predicate

$(\mathbb{N}, \mathrm{Succ}, \mathrm{Fac})$

We start as follows:



Continuation



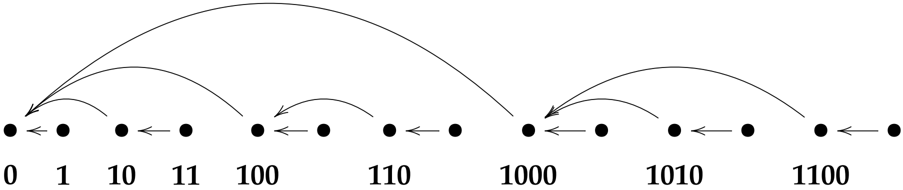
Another Unfolding

Beyond the Caucal Hierarchy

The Flip Function

The *flip function* maps 0 to 0 and

each nonzero n to the number which arises by switching the least significant 1-bit of n to 0.



$(\mathbb{N}, \mathrm{Succ}, \mathrm{Flip})$ **is in** G_2

The MSO-theory of $(\mathbb{N}, \mathrm{Succ}, \mathrm{Flip})$ **is decidable.**

Discussion of HOPD Graphs

very rich

but structurally connected to trees

too restricted for practical purposes

decidability result for a very strong logic

How to generalize the models in a good direction

and keeping interesting decidability results

(for logics weaker than MSO?

Tree Rewriting Graphs

Idea: Replace prefix rewriting of words by ground term rewriting of trees

Background on trees, tree languages, tree automata

Use ranked alphabet $\Sigma = \Sigma_0 \cup \ldots \cup \Sigma_k$

T_Σ

Tree automata

Regular tree languages

Ground tree rewriting system is a finite set of rules $t \longrightarrow t'$

Ground tree rewriting graphs

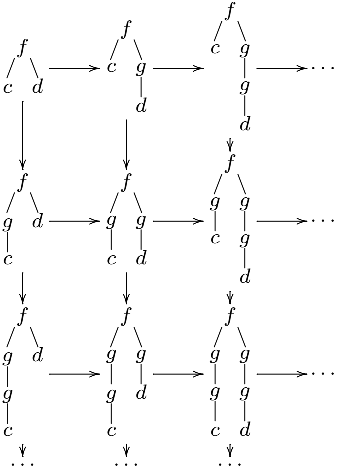
GTR system S induces derivability relations $\vdash$, $\vdash^*$

We say that $\vdash$ is defined by S.

A *ground term rewriting graph (GTRG)* is a structure $G = (V, (E_a), (P_b))$ where

- **V and the sets P_b are regular tree languages**
- **each edge relation E_a is defined by a GTR system**

Example



The infinite $\mathbb{N} \times \mathbb{N}$-grid is a GTRG. Hence the MSO-theory of a GTRG can be undecidable.

Saturation method reapplied

Let $G(V, \ldots)$ be a GTR graph, and $\mathcal{A}$ be a tree automaton recognizing a regular set $T \subseteq V$

Form T one can construct a tree automaton $\mathcal{A}'$ recognizing the set $\mathrm{pre}^*(T)$.

Moreover:

Over a ground tree rewriting graph, the model-checking problem is decidable for the logic FO(R) and for the branching-time logic with the following syntax:
T (regular) $\mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid EX_a\varphi \mid EF\varphi \mid EFG\varphi$.

An undecidability result

While the operators EF and EGF preserve decidability, this fails for the operator AF
(“on each path there is a vertex with a certain property”).

The following problem is undecidable:

> ***Given a ground tree rewriting graph G, a vertex v and a regular set T of vertices of G, does every path from v through G reach T?***

Reduction of the halting problem for Turing machines:

Represent we represent a Turing machine configuration $a_1 \dots a_k\ q\ b_l \dots b_1$ as a tree

The TM step simulation

Example instruction $(q\ a\ b\ L\ p)$.

TM halts iff in the GTR graph each path starting from the tree for the initial configuration meets either Err or Halt.

Internal vs. external representations

Two approaches of model construction:

- **the internal presentation in terms of automaton definable sets and relations of words, respectively trees,**
- **the external presentation by means of model transformations, starting from certain fundamental structures (in our case, the structures T_2, T_2').**

Corresponding results exist for all graph classes considered here.

Analogy in classical mathematics:

Specify a vector space by a basis, or as the space of linear maps over a given vector space

Structural characterization

Let $G = (V, (E_a)_{a \in \Sigma})$ be a graph of bounded degree and with designated "origin" vertex v_0.

$V_n =$ be the set of vertices whose distance to v_0 is at most n

$G_n =$ subgraph of G induced by the vertex set $V \setminus V_n$, with boundary vertices in $V_{n+1} \setminus V_n$.

The *ends* of G are the connected components (using edges in both directions) of the graphs G_n with $n \geq 0$.

Muller, Schupp 1985:
A transition graph G of bounded degree is a pushdown graph iff the number of distinct isomorphism types of its ends is finite.

Application to the grid

Recognized languages

Any graph $G = (V, (E_a)_{a \in \Sigma}, I, F)$ with unary predicates $I, F \subseteq V$ (of "initial" and "final" vertices) is an acceptor.

Muller-Schupp 1985, Caucal 1996:
A language L is context-free iff L is recognized by a pushdown graph (with regular sets of initial and final states) iff L is recognized by a prefix-recognizable graph (with regular sets of initial and final states).

Morvan-Stirling 2001, Rispal 2002
A language L is context-sensitive iff L is recognized by an automatic graph (with regular sets of initial and final states) iff L is recognized by a rational graph (with regular sets of initial and final states).

Retrospective and Outlook

Central ideas and Proofs

- PCP, TM halting problem, and simple questions about rational and automatic graphs
- the decidability of the FO-theory of an automatic graph
- the reachability analysis for pushdown systems by the saturation algorithm,
- the method of interpretations, with applications on pushdown graphs and the Caucal hierarchy,
- the role of the infinite grid (MSO model-checking undecidable, FO(R) model-checking decidable]
- the undecidability of properties over ground tree rewriting graphs that involve universal path quantification.

Perspectives

- Study systematically *all* automatic / prefix recognizable presentations
- Consider more transformations for the generation of models.
- Proceed from graphs to more general structures (e.g., hypergraphs).
- Fill the gap between FO and MSO (by interesting intermediate logics), and similarly between automatic and pushdown graphs (by interesting intermediate graphs).
- Merge the theory of infinite transition systems with arithmetical constraints (over $\mathbb{N}$ and $\mathbb{R}$).