Perspectives of Algorithmic Model Theory

Cachan, November 2005

Wolfgang Thomas
thomas@informatik.rwth-aachen.de

Informatik VII

RWTH AACHEN
Algorithmic model theory is a foundational discipline which

- combines automata theory, logic, algorithmics
- gives better understanding of complexity of decision problems
- contributes to automated verification and synthesis

Plan:

1. Some results and characteristics
2. On arithmetical constraints
3. Some perspectives
Part I

Some Results and Characteristics
Transition graphs $G = (V, (E_a)_{a \in \Sigma_1}, (P_b)_{b \in \Sigma_2})$

where $E_a \subseteq V \times V$, $P_b \subseteq V$

Set $E = \bigcup_{a \in \Sigma_1} E_a$

Logics:

- first-order logic FO
- first-order logic with reachability FO(R) (includes relation symbol for $E^*$)
- monadic second-order logic MSO (encompasses LTL, CTL, CTL*)
Finite Presentation of Infinite Graphs

A successful approach: “Regular presentations of infinite models” using

- regular languages as representations of state sets
- automaton-definable relations as representations of edge sets

Central examples:

- automatic graphs and pushdown graphs

Central facts:

- For an automatic graph, checking FO sentences is decidable.
- For a pushdown graph, checking MSO sentences is decidable.
Automatic Transition Graphs

\[ G = (V, (E_a), (P_b)) \] is automatic if for some alphabet \( \Gamma \)

- \( V \subseteq \Gamma^* \) and the \( P_b \subseteq \Gamma^* \) are regular
- the \( E_a \) are automatic relations (i.e., recognized by a dfa scanning the two words synchronously letter by letter)

Formally represent \((u, v) \in \Gamma^* \times \Gamma^*\) by a single word over \((\Gamma \cup \{\$\}) \times (\Gamma \cup \{\$\})\)

Illustration:

\((010, 11011)\) is represented by the word \(\begin{array}{cccc} 0 & 1 & 0 & \$ \\ 1 & 1 & 0 & 1 \end{array}\)

Standard closure and decidability properties are preserved.

Example in mathematics: Automatic groups
There is an automatic graph $G$ such that over $G$ the relation $E^*$ is undecidable.

**Proof:**

- Take configuration graph of universal Turing machine
- The one-step relation of pairs $(uqv, u'q'v')$ is automatic
- Then: Turing machine $M$ accepts word $w$ iff over $G$ from the configuration $q_0\text{code}(M)w$ a halting configuration can be reached.

**Alternative view:** Infix rewriting is too strong

This is the core reason for the difficulties in regular model-checking.
Pushdown Graphs

\[ G = (V, (E_a)) \text{ is a pushdown graph if there are} \]

alphabets \( Q, \Gamma \) with \( q_0 \in Q, Z_0 \in \Gamma \) and a finite system \( \Delta \) of rewrite rules \( p u_1 \xrightarrow{a} q u_2 \) such that

- \( E_a \) contains the pairs \( (p u_1 w, q u_2 w) \)

- \( V = q_0 Z_0 E^* \)

\[
\begin{array}{c}
q_0 Z_0 \xrightarrow{a} q_0 X Z_0 \xrightarrow{a} q_0 X X Z_0 \xrightarrow{a} q_0 X X X Z_0 \xrightarrow{a} \cdots \\
q_1 Z_0 \xleftarrow{b} q_1 X Z_0 \xleftarrow{b} q_1 X X Z_0 \xleftarrow{b} \cdots 
\end{array}
\]

More general situations: Prefix rewriting (where state alphabet \( Q \) is suppressed)

Prefix-recognizable graphs
We may use graphs $G = (V, (E_a)_{a \in \Sigma}, I, F)$ with $I, F \subseteq V$ as language acceptors.

Assumption: $V, I, F \subseteq \Gamma^*$, all regular.

We obtain a new characterization of fundamental Chomsky classes:

A language $L \subseteq \Sigma^*$ is

- regular iff it is recognizable by a finite graph
- context-free iff it is recognizable by a pushdown graph (equivalently: by a prefix recognizable graph) (Muller/Schupp 1985, Caucaul 2000)
- context-sensitive iff it is recognizable by an automatic graph (Morvan/Stirling 2001, Rispal 2002)
Definition of infinite structures using automata or rewriting systems gives “internal representations”.

- This is like defining a vector space by giving a basis (and the rule how to generate the elements from the basis).

Alternatively, one can consider transformations of given structures to generate new ones ("external representations").

- This is like forming a new vector space from a given one by taking all linear maps over it.

Natural operations over transition graphs:

- Unfoldings (from some vertex), transforming a graph to a tree
- Interpretations (transforming, for example, a graph into another graph)
- Products with different kinds of synchronization
The unfolding of a graph \( G = (V, (E_a), (P_b)) \) from \( v_0 \) is the tree \( T_{G,v_0} \) whose nodes have the form
\[ v_0 a_1 v_1 a_2 \ldots a_k v_k \] with \( (v_{i-1}, v_i) \in E_{a_i} \)
and \( v_0 a_1 v_1 a_2 \ldots a_k v_k \) \( \in P_b \) iff \( v_k \in P_b \)
Courcelle/Walukiewicz (1998):

If checking MSO-sentences over $G$ is decidable and $v$ is an MSO-definable vertex of $G$, then checking MSO-sentences over $T_{G,v}$ is decidable.

Rabin’s Tree Theorem is a special case.

More general construction: Tree model (Muchnik, Walukiewicz)
Interpretations

give description of a graph $H$ ”within” a given graph $G$

Example: Pushdown transition graph with stack symbols $a, b, \bot$
described in the ternary tree $T_3$:

Stack contents $b\bot$, $ba\bot$, $baa\bot$, $\ldots$ represented by

Fact: Suppose $H$ is MSO-interpretable in $G$. If checking MSO
sentences over $G$ is decidable, then so it is over $H$. 
Barthelmann, Blumensath, Grädel:

- A graph is prefix-recognizable iff it is MSO-interpretable in a tree $T_k$.
- A graph is automatic iff it is FO-interpretable in a tree $T_k$ (with successor relations, partial tree order, and equal level predicate).
A pointed graph (with designated vertex $v$) is finite in the infinite if after deletion of the $n$-neighborhoods of $v$ for increasing $n$ the remaining connected components have only finitely many different isomorphism types.

Example: Binary tree

Counterexample: Infinite $\mathbb{N} \times \mathbb{N}$ grid

Muller/Schupp 1985:
A pointed graph of bounded degree is a pushdown graph iff it is finite in the infinite.
Pushdown graphs can be described via

1. an internal representation (using prefix rewriting)
2. MSO-interpretation in a tree $T_k$
3. a structural property

Applications:

1. Efficient model-checking for special properties
2. Decidability of model-checking for a logic
3. Distinction from other graphs
Cauca$\mathrm{\text{s}}$’s Hierarchy

(in the version of Carayol/Wöhrle 2003)

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

Note:

- $\mathcal{G}_0$ is the class of finite graphs
- $\mathcal{T}_1$ contains the regular trees
- $\mathcal{G}_1$ is the class of prefix-recognizable graphs

For each graph in the hierarchy, checking MSO sentences is decidable.
Example

\[ \psi_d(x, y) = \psi_e(x, y) = \exists z \exists z'(E_a(z, z') \land E_c(z, y) \land E_c(z', x)) \]
Unfolding yields “algebraic tree”

\[
\begin{align*}
\bullet & \xrightarrow{a} \bullet \quad \bullet & \xrightarrow{a} \bullet & \quad \bullet & \xrightarrow{a} \bullet \\
\bullet & \quad \bullet & \quad \bullet & \quad \bullet & \quad \bullet \\
& \quad \bullet & \quad \bullet & \quad \bullet & \quad \bullet \\
& \quad \bullet & \quad \bullet & \quad \bullet & \quad \bullet \\
& \quad \bullet & \quad \bullet & \quad \bullet & \quad \bullet \\
& \quad \bullet & \quad \bullet & \quad \bullet & \quad \bullet \\
& \quad \bullet & \quad \bullet & \quad \bullet & \quad \bullet \\
& \quad \bullet & \quad \bullet & \quad \bullet & \quad \bullet \\
& \quad \bullet & \quad \bullet & \quad \bullet & \quad \bullet \\
& \quad \bullet & \quad \bullet & \quad \bullet & \quad \bullet \\
& \quad \bullet & \quad \bullet & \quad \bullet & \quad \bullet \\
& \quad \bullet & \quad \bullet & \quad \bullet & \quad \bullet \\
& \quad \bullet & \quad \bullet & \quad \bullet & \quad \bullet \\
& \quad \bullet & \quad \bullet & \quad \bullet & \quad \bullet \\
& \quad \bullet & \quad \bullet & \quad \bullet & \quad \bullet \\
& \quad \bullet & \quad \bullet & \quad \bullet & \quad \bullet \\
\end{align*}
\]

MSO-interpretation produces a graph isomorphic to \((\mathbb{N}, \text{Succ}, P_2)\) where \(P_2\) contains the powers of 2.
Higher-Order Pushdown Systems

are a generalization of pushdown systems where nested stacks are considered.

Level-2 stack is a stack where each entry is a stack.

Level-3 stack is a stack of level-2 stacks, etc.

For simplicity only consider level-2 stacks here.

Operations, depending on top symbol of top stack:

- on top level-1 stack as for standard pushdown systems
- deletion of top level-1 stack
- copy (and thus duplication) of top level-1 stack
Cauca 2002, Carayol/Wöhrle 2003:

A graph is the $\varepsilon$-closure of a level-$n$ pushdown graph with $\varepsilon$-transitions iff it belongs to the class $G_n$ of the Cauca hierarchy.

Consequence:

Checking MSO sentences over higher-order pushdown graphs is decidable.

Open problem: Which transition graphs occur in the Cauca hierarchy?
Algorithmic model theory:

- A jungle of basic models rather than canonical structures like $(\mathbb{N}, +, \cdot)$ and $(\mathbb{R}, +, \cdot)$.

- Focus is on relational structures rather than algebras.

- Fragments of second-order logic rather than first-order logic.

- Operations like unfolding and synchronized products rather than: substructures, reduced products, ultraproducts etc.

- Internal representations of elements are relevant.

- Results on (efficient) decidability are central.

Promising direction: Merge the two traditions

We discuss a small example on the algorithmic treatment of arithmetical constraints
Part II

Remarks on Arithmetical Constraints
Semi-Linear Sets

Well-known framework for algorithmic applications: Presburger arithmetic and semi-linear sets.

\[ A \subseteq \mathbb{N}^n \text{ linear } \iff \exists u_0 \ldots u_m \in \mathbb{N}^n : \]

\[ A = \{ u_0 + k_1 u_1 + \ldots + k_m u_m \mid k_1, \ldots, k_m \geq 0 \} \]

A semi-linear iff \( A \) is finite union of linear sets

We discuss two moderate ways to go beyond semi-linear sets
(ongoing work with Wong Karianto and Aloys Krieg, Aachen)

Platon, Politeia Book VII:

It is hard to find something which requires more effort from a man of learning and ambition than arithmetic.
A finite automaton, edge labels in $\Sigma \times \mathbb{N}^n$

$\varphi(x_1, \ldots, x_n)$ Presburger formula

$\varphi(x_1, x_2, x_3) : x_1 \geq 2(x_2 + x_3) \land x_2 = x_3$

“first $b$ is in second half, and from the first $b$ onwards the numbers of $b$ and $c$ coincide”

Call accepted language $L(\mathcal{A}, \varphi)$
Nonemptiness Problem for Parikh Automata

Given $A$ and $\varphi$

$A_A := \text{set of vectors in } \mathbb{N}^n \text{ generated by successful } A\text{-runs}$

$A_\varphi := \text{set of vectors in } \mathbb{N}^n \text{ defined by } \varphi$

Both are semi-linear.

$L(A, \varphi) \neq \emptyset \iff A_A \cap A_\varphi \neq \emptyset$

The set on the right is semi-linear.

Consequence:

The non-emptiness problem for Parikh automata is decidable.
1. Generalize semi-linear sets to “semi-polynomial” sets and solve intersection problem over $\mathbb{N}^n$ for semi-polynomial and componentwise semi-linear sets

2. Allow constraints $Q$ by quadratic equations and solve nonemptiness of intersection of such a $Q$ with a semi-linear set.
Simple Semi-Polynomial Sets

Example considered here: the quadratic case

\[ A \subseteq \mathbb{N}^n \text{ simple quadratic iff} \]

\[ \exists u_0, u_1, v_1 \ldots u_r, v_r \in \mathbb{N}^n \text{ such that} \]

\[ A = \{ u_0 + k_1 u_1 + k_1^2 v_1 + \ldots + k_r u_r + k_r^2 v_r \mid k_1, \ldots, k_r \geq 0 \} \]

A simple semi-quadratic iff A is finite union of simple quadratic sets

A is simple semi-polynomial (ssp) : analogous
Call $A \subseteq \mathbb{N}^n$ componentwise linear iff

there are linear sets $A_1, \ldots, A_n \subseteq \mathbb{N}$ with

$(m_1, \ldots, m_n) \in A \iff m_i \in A_i$ for $i = 1, \ldots, n$

A finite union of componentwise linear sets is componentwise semi-linear

Theorem

If $A$ is componentwise semi-linear and $B$ is simple semi-polynomial,

then $A \cap B$ is simple semi-polynomial

(and can be checked for non-emptiness)
Mixed Variables

\[ A_0 = \{ k_1 (1, 0, 0) + k_2 (0, 1, 0) + k_1 k_2 (0, 0, 1) \mid k_1, k_2 \geq 0 \} \]
is the product relation

Proposition: \( A_0 \) is not simple semi-polynomial.

Hilbert’s 10th Problem:

\[
\exists k_1 \ldots k_n \quad P(k_1, \ldots, k_n) = 0 \quad (P \in \mathbb{Z}[x_1, \ldots, x_n])
\]

iff \( \exists k_1 \ldots k_n \quad Q(k_1, \ldots, k_n) = R(k_1, \ldots, k_n) \)

\( (Q, R \in \mathbb{N}[x_1, \ldots, x_n]) \)

iff \( \{ (Q(k_1, \ldots, k_n)) \mid k_1, \ldots, k_n \geq 0 \} \cap \text{id}_\mathbb{N} \neq \emptyset \)

[Note: \( \text{id}_\mathbb{N} = \{ k \cdot (1) \mid k \geq 0 \} \)]
Consider **quadratic forms**

\[ Q(x_1, \ldots, x_n) = \sum_{i,j=1}^{n} a_{ij}x_i x_j + \sum_{j=1}^{n} b_j x_j + c \]

with \( a_{ij}, b_j, c \in \mathbb{Z} \)

**Linear forms** \( L(x_1, \ldots, x_n) \) are defined analogously.

Solvability in integers is decidable by Siegel (1972)

Solvability in natural numbers cannot be obtained via Lagrange’s Theorem:

\[ z \geq 0 \iff \exists y_1, \ldots, y_4 : y_1^2 + \ldots + y_4^2 = z \]

Obtain *system* of quadratic equations (where solvability is undecidable in general).
Solution of Quadratic Equations in $\mathbb{N}$

Grunewald/Segal (J. Reine Angew. Math. 2004)

Given quadratic form $Q(x_1, \ldots, x_n)$

linear forms $L_1(x_1, \ldots, x_n), \ldots, L_k(x_1, \ldots, x_n)$

Consider systems of the form

1. $Q(x_1, \ldots, x_n) = 0$
2. $(L_j(x_1, \ldots, x_n) \# c_j)_j$ where $c_j \in \mathbb{Z}$ and $\#$ is $<$ or $\leq$
3. $(x_1, \ldots, x_n) \equiv (h_1, \ldots, h_n) \pmod{m}$ with integers $h_j$

One can decide algorithmically whether a solution in $\mathbb{Z}^n$ exists.

Remark: Linear constraints $-x_i \leq 0$ restrict to solutions in natural numbers

Algorithmic analysis (upper complexity bound) is open
Corollary 1: Nonemptiness of intersection of a semi-linear set $A \subseteq \mathbb{N}^n$ with the solution set $S$ of a quadratic equation $Q(x_1, \ldots, x_n) = 0$ is decidable.

Call a set $A \subseteq \mathbb{N}^n$ 1-quadratic if

$$A = \{ (Q^{(k_1, \ldots, k_r)}_{L(k_1, \ldots, k_r)}) \mid k_1, \ldots, k_r \geq 0 \}$$

where $Q(x_1, \ldots, x_r)$ is a quadratic form

$$L(k_1, \ldots, k_r) = \{ u_0 + k_1 u_1 + \ldots + k_r u_r \mid k_1, \ldots, k_r \geq 0 \}$$

with $u_i \in \mathbb{N}^{n-1}$

Corollary 2:
Nonemptiness of the intersection of 1-quadratic sets is decidable.
Part III

Some Perspectives
Perspectives

There are several paradigmatic ideas to pursue.

An example: Büchi’s Research Program
(Chapter VI from *Finite Automata, Their Algebras and Grammars*)

“Proceed from words to trees”: Pass from unary algebras to $n$-ary algebras

Sometimes you may find yourself doing things which have not been done yet. Some of the generalizing, I hope, will require quite new ideas. And now I will say it again. Here is a promising program for research in automata theory, universal algebra, equational logic, and language theory:

Research Program: Work through all the ideas presented in chapters 1-7, extending...

Check out for places where the methods from the unary to the $n$-ary case. Watch out for places where the variables of equational logic come into play; these places are hidden in the unary case, as only one variable may occur (at only one place, in a unary term!)
- The idea of prefix rewriting can be exploited for trees (Dauchet, Heuillard, Lescanne, Tison 1990, Löding 2002, Colcombet 2002)

- Proceed from ranked to unranked trees (M. Bojanczyk, I. Walukiewicz) - this allows to preserve algebraic results (from the case of words)

- From trees proceed to tree-like structures, invoking structural properties of graphs (bounded tree-with, bounded clique-width, bounded dag-with)

Related idea: Well-structured transition systems to enforce solvability of the reachability problem
Internal and External Representations

- Study systematically *all* automatic / prefix recognizable presentations of a structure and choose an appropriate one for an algorithmic problem.
  
  Analogy: Base transformations in linear algebra.

- Study the interplay between internal and external representations in a more general context:
  
  ▶ Unfolding is a “sequence model”
  
  ▶ Consider also the “set model” and the “function model”
  
  ▶ Mixture of set model and sequence model:
    Latest appearance record \((i_1 \ldots i_m)(i_{m+1} \ldots i_n)\) where the set component is \(\{i_1, \ldots, i_m\}\)

- Proceed from graphs to hypergraphs / relational structures
Feferman/Vaught 1959 showed the paradigmatic result:
for ”direct” and ”reduced” products of structures and for
first-order logic.

In order to decide whether \( \Pi^n_{i=1} A_i \models \varphi \) one can proceed as follows:
Compute finitely many auxiliary formulas \( \psi_j \) and determine the sets
\[
I_j = \{ i \mid A_i \models \psi_j \}
\]
and check whether the sets \( I_j \) satisfy a Boolean condition \( \beta \).
The \( \psi_j \) and \( \beta \) are computable from \( \varphi \).
This describes a uniform procedure for a transfer of
FO-model-checking from the components to the product.
Given edge-labelled transition graphs $G_1, \ldots, G_n$

define a product via a set $C$ of synchronization constraints, each of them being a tuple $(c_1, \ldots, c_n)$ of labels.

$c_i = \varepsilon$ is allowed, meaning “$G_i$ stays in current state”

Wöhrle/Th. 2004: If $G$ is a finitely synchronized product of $G_1, \ldots, G_n$, and checking FO(R)-sentences is decidable over the $G_i$, then the same holds over $G$.

The $G_i$ are finitely synchronized via $C$ if for each constraint $(c_1, \ldots, c_n) \in C$ and any $c_i \neq \varepsilon$, only finitely many transitions in $G_i$ have label $c_i$. 
Further Tracks

- Find more applications of the composition method.

- More generally: Devise an “arithmetic of transition systems”, including quotients that are more complex than by simple equivalence relations.

- Fill the gap between FO and MSO, and similarly between automatic and pushdown graphs.

- Find more solutions on the combination of transition systems with arithmetical constraints.

Conclusion:

*Automata theory has a wonderful task in contributing to an algorithmic theory of models.*