Lecture 15

Automatic Structures

automata-based decision procedures for logical-theories

Print version of the lecture in *Logic and Proof*

presented on 3 June 2019

by Prof James Worrell

In the last lecture, we saw that quantifier-elimination is a technique that enables showing decidability of a logical theory. Today, we will see another such technique that enables deciding many logical theories and is based on regular languages, which is the class of languages accepted by finite-state automata which you may well-remember from your *Models of Computation* course.

The basic idea is easy to explain. We consider logical theories $\mathcal{T}$ of structures whose universe is isomorphic to a regular language and whose relations are also regular languages over a suitable alphabet. Given a formula $F(x_1, \ldots, x_n)$, we can then construct a finite-state automaton whose language L encodes all satisfying assignments of F. It follows that F is satisfiable if $L \neq \emptyset$, and if F is closed then F is a sentence of $\mathcal{T}$ whenever $L = \{\epsilon\}$.

In this lecture, we will exclusively consider *relational structures*. A σ-structure is **relational** if σ consists of only relation symbols. Fortunately, this restriction comes with no loss of generality. Suppose the signature σ of a σ-structure $\mathcal{A}$ contains an k-ary function symbol f. We can then turn $\mathcal{A}$ into a relational structure by introducing a fresh relation symbol R which is interpreted as the *graph* of f, i.e.,

$$R_{\mathcal{A}} = \left\{(a_1, \ldots, a_k, b) \in U_{\mathcal{A}}^{k+1} : f_{\mathcal{A}}(a_1, \ldots, a_k) = b\right\}.$$

Constant symbols can be dealt with in the same way, recalling that a constant symbol can be viewed as a 0-ary function symbol.

For a concrete example, consider the structure $\mathcal{A} = (\mathbb{N}, 0, 1, +, <)$ underlying Presburger arithmetic with constants 0, 1 and the binary addition function $+$. We introduce fresh relation symbols R_0, R_1 and R_+ such that

$$R_0 = \{0\} \qquad R_1 = \{1\} \qquad R_+ = \{(a,b,c) \in \mathbb{N}^3 : a + b = c\}.$$

Then $(\mathbb{N}, R_0, R_1, R_+, <)$ is a relational structure.

1 Automatic structures

We will now define what it means for a relational structure to be automatic and show how to decide the theory of an automatic structure. Let us fix some relational structure $\mathcal{A}$. The first requirement for $\mathcal{A}$ to be automatic is that its universe $U_{\mathcal{A}}$ is isomorphic to some regular language $L \subseteq \Sigma^*$.

For example, $U_{\mathcal{A}} = \mathbb{N}$ is isomorphic to the regular language

$$N := (\{0,1\}^* 1) \cup \{0\} \subseteq \{0,1\}^*.$$

To see this, think of a word $w = b_0 b_1 \cdots b_m \in N$ as encoding a natural number in binary in least-significant bit numbering, i.e., define $\mathrm{val}\colon N \to \mathbb{N}$ by

$$\mathrm{val}(w) := \sum_{i=0}^{m} 2^i \cdot b_i$$

Then val gives a bijection between $\mathbb{N}$ and N.

As stated above, relations of an automatic structure are also regular languages. Given that relations can have arity greater than one, this entails the question how to represent a k-ary relation $R \subseteq U_{\mathcal{A}}^k$, where $U_{\mathcal{A}} \subseteq \Sigma^*$ for some finite alphabet Σ. A key observation is that we can encode a tuple $(a_1, \ldots, a_k) \in U_{\mathcal{A}}^*$, where all a_i have the same length, as a word over the alphabet Σ^k. For example, $(10, 10, 01) \in (\{0,1\}^*)^3$ can be encoded as the word $\left[\begin{smallmatrix}1\\1\\0\end{smallmatrix}\right]\left[\begin{smallmatrix}0\\0\\1\end{smallmatrix}\right] \in (\Sigma^3)^*$. However, this encoding crucially relies on the a_i having the same length, which is rarely the case. For example, we cannot encode the tuple $(1, 1, 01) \in N^3$ in this way.

1.1 Word convolutions

A way out of this dilemma is to introduce an additional fresh padding symbol $\# \notin \Sigma$ serving as a blank symbol that allows us to make the length of different words equal in a unique way. We define $\Sigma_\# := \Sigma \cup \{\#\}$. Suppose we are given words $w_1, \ldots, w_k \in \Sigma^*$, each of the form $w_i = a_{i,1} a_{i,2} \cdots a_{i,\ell_i}$, and let $\ell = \max\{\ell_1, \ldots, \ell_k\}$. We define the **convolution operator** $\otimes$ that, informally speaking, glues the words of a tuple $(w_1, \ldots, w_k) \in (\Sigma^*)^k$ together and represents it as a unique word over $(\Sigma_\#^k)^\ell$. Define $a_{i,j} := \#$ for all $\ell_i < j \leq \ell$ and $1 \leq i \leq n$, the **convolution** of $w_1, \ldots, w_n$ is the word

$$w_1 \otimes w_2 \otimes \cdots \otimes w_k := \begin{bmatrix} a_{1,1} \\ \vdots \\ a_{n,1} \end{bmatrix} \begin{bmatrix} a_{1,2} \\ \vdots \\ a_{n,2} \end{bmatrix} \cdots \begin{bmatrix} a_{1,\ell} \\ \vdots \\ a_{n,\ell} \end{bmatrix} \in (\Sigma_\#^k)^*$$

The definition of convolution is probably best understood with the help of an example.

Example 1. Let $\Sigma = \{a, b\}$, then

$$abba \otimes abaabba = \left[\begin{smallmatrix}a\\a\end{smallmatrix}\right]\left[\begin{smallmatrix}b\\b\end{smallmatrix}\right]\left[\begin{smallmatrix}b\\a\end{smallmatrix}\right]\left[\begin{smallmatrix}a\\a\end{smallmatrix}\right]\left[\begin{smallmatrix}\#\\b\end{smallmatrix}\right]\left[\begin{smallmatrix}\#\\b\end{smallmatrix}\right]\left[\begin{smallmatrix}\#\\a\end{smallmatrix}\right].$$

Recall that the universe of automatic structures is a regular language. The next proposition enables us to obtain tuples of convoluted words of those universes as regular languages.

Proposition 2. Let $U \subseteq \Sigma^*$ be a regular language and $k > 0$. Then

$$L_{U,k} := \{w_1 \otimes w_2 \otimes \cdots \otimes w_k \in (\Sigma_\#^k)^* : w_1, \ldots, w_k \in U\}$$

is regular.

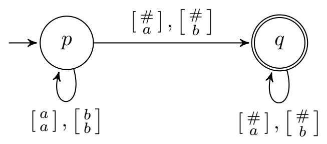
Proof. We leave the proof of this proposition as an exercise to the reader. □

1.2 Automatic relations

With word convolutions at hand, we can now formally define what it means for a relation to be automatic: A relation $R \subseteq U_{\mathcal{A}}^k$ is **automatic** if the language

$$L_R := \{w_1 \otimes w_2 \otimes \cdots \otimes w_k : (w_1, \ldots, w_k) \in R\}$$

is regular. For example, $R = \{(u, v) \in (\Sigma^*)^2 : u \text{ is a prefix of } v\}$ with $\Sigma = \{a, b\}$ is automatic, since L_R is the language of the following non-deterministic finite state automaton:



p
q
$[\begin{smallmatrix}\#\\a\end{smallmatrix}], [\begin{smallmatrix}\#\\b\end{smallmatrix}]$
$[\begin{smallmatrix}a\\a\end{smallmatrix}], [\begin{smallmatrix}b\\b\end{smallmatrix}]$
$[\begin{smallmatrix}\#\\a\end{smallmatrix}], [\begin{smallmatrix}\#\\b\end{smallmatrix}]$

Likewise, the equals-relation "=" is clearly automatic.

With those definitions at hand, we can now give a formal definition of automatic structures.

Definition 3. A relational structure $\mathcal{A} = (U_{\mathcal{A}}, R_1, \ldots, R_m)$ is **automatic** if there are a finite alphabet Σ and regular languages $L \subseteq \Sigma^*, L_1, \ldots, L_m$ such that

- $L = U_{\mathcal{A}}$
- $L_i = L_{R_i}$ for all $1 \leq i \leq m$

A structure $\mathcal{A}$ has an **automatic presentation** if $\mathcal{A}$ is isomorphic to an automatic structure.

1.3 Examples of automatic structures

We will now explore some examples of structures that have an automatic presentation. Recall the theory of unbounded dense linear orders $\mathcal{T}_{\text{UDLO}}$ from the previous lecture.

Proposition 4. Any structure $\mathcal{A} = (Q, <)$ that is a model of the unbounded dense linear order axioms has an automatic presentation.

Proof. To show the statement, we (i) prove that any unbounded dense linear orders are isomorphic to each other, and (ii) exhibit an automatic structure that is a model of the unbounded dense linear order axioms. In fact, we defer part (i) to the next lecture.

For part (ii), we define a structure $\mathcal{A} = (L, <)$ such that $L := \{0,1\}^* \cdot 1$ and $x < y$ iff either

- $y = xu$ for some $u \in \{0,1\}^+$, or
- $x = z0u$ and $y = z1v$ for some $u, v, z \in \{0,1\}^*$.

Thinking of a word $w \in \{0,1\}^*$ as encoding a node in a binary tree, we have $x < y$ if and only if the node identified by y is below or to the right of the node identified by x. Clearly, L is regular, and it is not difficult to construct an NFA accepting $L_<$. Hence, $\mathcal{A}$ is an automatic structure.

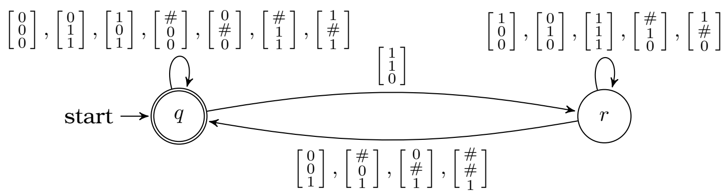
It remains to show that $\mathcal{A}$ is a model of the axioms of unbounded dense linear orders. But that is not difficult to see:

- There is no smallest element: for any $u1 \in L$, we have $u01 \in L$ and $u01 < u1$.
- There is no largest element: for any $u1 \in L$, we have $u11 \in L$ and $u1 < u11$.
- L is dense with respect to $<$. Given $x, y \in L$ such that $x < y$, we distinguish two cases.
  1. If $x = u1$ and $y = u1v1$, then $x < u10^{|v|+1}1 < y$.
  2. If $x = u0v1$ and $y = u1w$, then $x < u01^{|v|+2} < y$. □

Another example of a structure with an automatic presentation is the (relational) structure underlying Presburger arithmetic.

Proposition 5. The structure $\mathcal{P} = (\mathbb{N}, 0, 1, +, <)$ has an automatic presentation.

Proof. We established above that there is a bijection between $\mathbb{N}$ and N. It thus remains to find regular languages L_0, L_1, L_+ and $L_<$ "implementing" the relations of $\mathcal{P}$. Obviously, $L_0 := \{0\}$ and $L_1 := \{1\}$ are regular. For L_+, consider the following NFA A:


$\begin{bmatrix}0\\0\\0\end{bmatrix}, \begin{bmatrix}0\\1\\1\end{bmatrix}, \begin{bmatrix}1\\0\\1\end{bmatrix}, \begin{bmatrix}\#\\0\\0\end{bmatrix}, \begin{bmatrix}0\\\#\\0\end{bmatrix}, \begin{bmatrix}\#\\1\\1\end{bmatrix}, \begin{bmatrix}1\\\#\\1\end{bmatrix}$
$\begin{bmatrix}1\\1\\0\end{bmatrix}$
$\begin{bmatrix}1\\0\\0\end{bmatrix}, \begin{bmatrix}0\\1\\0\end{bmatrix}, \begin{bmatrix}1\\1\\1\end{bmatrix}, \begin{bmatrix}\#\\1\\0\end{bmatrix}, \begin{bmatrix}1\\\#\\0\end{bmatrix}$
start
q
r
$\begin{bmatrix}0\\0\\1\end{bmatrix}, \begin{bmatrix}\#\\0\\1\end{bmatrix}, \begin{bmatrix}0\\\#\\1\end{bmatrix}, \begin{bmatrix}\#\\\#\\1\end{bmatrix}$

This automaton implements the ternary addition relation via "written addition." The control state q is accepting, and the automaton moves to r if a carry occurs which needs to be resolved before the automaton can move back to q. Identifying the padding symbol # with zero, it follows that any word in the language $L(A) \subseteq (\{0,1,\#\}^3)^*$ of A has a unique corresponding tuple $(u,v,w) \in N^3$ such that $\mathrm{val}(u) + \mathrm{val}(v) = \mathrm{val}(w)$. All that remains to be done for obtaining L_+ is to make sure that once a padding symbol # occurs in some component of a word in $L(A)$, it remains there forever. It follows that $L_+ = L(A) \cap L_{N,3}$, where $L_{N,3}$ is defined as in Proposition 2, is regular since regular languages are closed under intersection.

We leave proving regularity of $L_<$ as an exercise to the reader, which can be shown analogously to the regularity of L_+. □

1.4 Decidability of the theories of automatic structures

What is striking about a structure $\mathcal{A}$ with an automatic presentation is that we immediately obtain decidability of $\mathrm{Th}(\mathcal{A})$ thanks to the rich closure properties of regular languages.

Theorem 6 (Khoussainov, Nerode)**.** *For every first-order structure $\mathcal{A}$ with an automatic presentation, the first-order theory $\mathrm{Th}(\mathcal{A})$ is decidable.*

To prove the statement, for a given formula F, we show how to decide $F \in \mathrm{Th}(\mathcal{A})$ by structural induction on F. Subformulas of F have free variables, and we proceed by showing how to inductively construct regular languages that encode all satisfying assignments of the subformulas of F.

Proposition 7. Let $\mathcal{A} = (U, R_1, \ldots, R_m)$ be an automatic σ-structure such that $U \subseteq \Sigma^*$, and let F be a σ-formula with at most free variables $x_1, \ldots, x_n$. There is an effectively constructible regular language $L_F \subseteq (\Sigma_\#^n)^*$ such that

$$L_F = \left\{ w_1 \otimes \cdots \otimes w_n \in (\Sigma_\#^n)^* : w_1, \ldots, w_n \in U,\ \mathcal{A}_{[x_1 \mapsto w_1] \cdots [x_n \mapsto w_n]} \models F \right\}.$$

Proof. The proof is by induction on the structure of F. In the base case, F is some relation $R_i(x_{i_1}, \ldots, x_{i_k})$. Since $\mathcal{A}$ is an automatic structure, $L_{R_i} \subseteq (\Sigma_\#^k)^*$ is regular by assumption. However, in general $k \neq n$, so we need to "lift" the alphabet of L_{R_i} to $(\Sigma_\#^n)^*$. To this end, define a homomorphism $h\colon (\Sigma_\#^n)^* \to (\Sigma_\#^k)^*$ such that for $a_1, \ldots, a_n \in \Sigma_\#$:

$$h(a_1, \ldots, a_n) = \begin{cases} \epsilon & \text{if } a_{i_1} = \cdots = a_{i_k} = \# \\ (a_{i_1}, \ldots, a_{i_k}) & \text{otherwise} \end{cases}$$

Since $L_{R_i} \subseteq (\Sigma_\#^k)^*$ is regular, using the fact that regular languages are closed under inverse homomorphisms, we obtain

$$L_F = h^{-1}(L_{R_i}) \cap L_{U,n}.$$

For the induction step, the cases $F = G \wedge H$, $F = G \vee H$ and $F = \neg G$ are easily dealt with, since the induction hypothesis gives regular languages $L_G, L_H \subseteq (\Sigma_\#^n)^*$ and regular languages are closed under intersection, union and complement. Thus, for instance, for $F = \neg G$ we have

$$L_F = L_{U,n} \setminus L_G.$$

The case $\exists x_{n+1} G$ with $x_1, \ldots, x_n, x_{n+1}$ being free in G is also easily dealt with. By the induction hypothesis, there is a regular language $L_G \subseteq (\Sigma_{\#}^{n+1})^*$ whose language encodes all satisfying assignments of G. To define L_F, all we have to do is to project onto the first n components of L_G. To this end, define the homomorphism $h\colon (\Sigma_{\#}^{n+1})^* \to (\Sigma_{\#}^{n})^*$ such that for $a_1, \ldots, a_n \in \Sigma_{\#}$

$$h(a_1, \ldots, a_n, a_{n+1}) = \begin{cases} \epsilon & \text{if } a_1 = \cdots = a_n = \# \\ (a_1, \ldots, a_n) & \text{otherwise} \end{cases}$$

We then obviously have $L_F = h(L_G)$, which is regular due to regular languages being closed under homomorphism.

Finally, for the case $F = \forall x_{n+1}\, G$, we have $F \equiv \neg \exists x_{n+1}\, \neg G$, which gives the desired regular language L_F. □

Thus, in order to decide for a given sentence F whether $F \in \mathrm{Th}(\mathcal{A})$, the proof of Proposition 7 gives an algorithm for constructing a representation of a regular language L_F (e.g. as an NFA) such that $F \in \mathrm{Th}(\mathcal{A})$ if and only if $L \neq \emptyset$. Since emptiness is decidable for NFA, decidability of $\mathrm{Th}(\mathcal{A})$ follows.

Naturally, the question arises how efficient the obtained decision procedure is. When processing universal quantifiers, we need to construct the complement of a regular language twice. Each complementation step can cause an exponential blow-up, and unfortunately this blow-up cannot be avoided.

Proposition 8. There exists an automatic structure $\mathcal{A}$ with non-elementary complexity, i.e., no algorithm decides $F \in \mathrm{Th}(\mathcal{A})$ in time $2^{2^{\cdot^{\cdot^{\cdot 2^n}}}}$.

Proof. This can be shown for the structure $\mathcal{A} = (\{0,1\}^*, S_0, S_1, \leq)$, where

$$S_i := \{(w, wi) : w \in \{0,1\}^*\},\ i \in \{0,1\} \qquad \leq\ := \{(w, wu) : w, u \in \{0,1\}^*\}.$$

The idea is to take a Turing machine M whose runtime for a given input w is bounded by a tower of exponentials whose height grows with $|w|$, and then to construct a sentence F such that $F \in \mathrm{Th}(\mathcal{A})$ if and only if M halts on w. Giving full details of this construction would go beyond the scope of these lecture notes. □

On the positive side, for many important first-order structures one can show that the intermediate automata obtained in the algorithmic approach presented in the proof of Proposition 7 only grow elementary and, for instance, are "only" of size $2^{2^{2^{O(n)}}}$ when deciding formulas of Presburger arithmetic.

1.5 Proving non-automaticity

Obviously, not every structure is automatic, since not every theory of a given structure is decidable. But there are also structures with a decidable theory which do not have an automatic presentation. An example is the first-order theory of the reals with addition $\mathrm{Th}(\mathbb{R}, 0, 1, +)$. Since there are only countably many words in a regular language, we cannot find a regular language U that is isomorphic to $\mathbb{R}$. Thus, to check whether a given structure has an automatic presentation, a first step is to examine whether its universe is countable.

Automatic structures also impose strong restrictions on the functions they can include (as relations, of course).

Lemma 9 (Constant Growth Lemma). *If $f\colon U^n \to U$ is a function whose graph is an automatic relation, then there is a constant $c > 0$ such that for all $a_1, \ldots, a_n \in U$,*

$$|f(a_1, \ldots, a_n)| \leq \max\{|a_1|, \ldots, |a_n|\} + c$$

Proof. We consider a "reversed" version of the Pumping Lemma for regular languages that states that for any regular language L, there exists a constant $p > 0$ such that for any $w \in L$ such that $|w| > p$, we have $w = xyz$ such that $|yz| \leq p$, $|y| > 0$, and $xy^i z \in L$ for all $i \geq 0$.

Let $R \subseteq U^{n+1}$ be the graph of f, by assumption L_R is regular. We choose c to be the constant for L_R obtained from the above Pumping Lemma. For a contradiction, assume that there exist $a_1, \ldots, a_n \in U$ such that $|f(a_1, \ldots, a_n)| - \max\{|a_1|, \ldots, |a_n|\} > c$. Then the Pumping Lemma implies that there two possible cases:

1. We have $a_1 \otimes \cdots \otimes a_n \otimes b \in L_R$ for infinitely many $b \neq f(a_1, \ldots, a_n)$.
2. L_R contains a word that is not a convolution of a tuple of words from U.

The first case contradicts R being the graph of f, and the second case contradicts R being automatic. □

Observe that the ternary R_+ relation we have seen in the context of Presburger arithmetic obeys the Constant Growth Lemma, since for tuples $(u, v, w) \in R_+$, we have $|w| \leq \max\{|u|, |v|\} + 1$. On the other hand, thanks to the Constant Growth Lemma, we can immediately see that any structure including a relation $\{(u, v) : \mathrm{val}(v) = 2^{\mathrm{val}(u)}\}$ cannot be automatic.

Further structures with decidable first-order theories which have no automatic presentation include $(\mathbb{N}, \cdot)$, $(\mathbb{N}, |)$ and $(\mathbb{Q}, +)$.

2 Beyond regular languages

We close this lecture by remarking that one can make alternative definitions of automaticity. The automatic structures we have seen in this lecture rely on regular languages, but the approach we have taken here works for any class of languages that has the required closure properties.

Instead of using regular languages, we can switch to ω-regular languages which are an analogue to regular languages for *infinite* words. In this setting, so-called *Büchi automata* correspond to finite-state automata and share their nice closure properties. For example, it is then possible to show that the structure $(\mathbb{R}, 0, 1, +, \leq)$ is *ω-automatic*.

Likewise, one can define analogues of automatic structures using so-called *tree automata*, which generalise finite-state automata and operate on trees instead of words (a word can be viewed as a tree that has only a single branch). It is then possible to show that $(\mathbb{N}, \cdot)$ is *tree-automatic*.

Deciding logical theories using some generalisation of automatic structures has given spectacular results in recent years. Call a number $n \in \mathbb{N}$ a *binary palindrome* if the binary representation of n equals its reverse. For instance, $27 = \mathrm{val}(11011)$ is a binary palindrome. The following number-theoretic statement, an analogue of the classical Lagrange's theorem, has recently been fully automatically proved using (extensions of) the techniques we have seen in this lecture.

Theorem 10 (Rajasekaran, Shallit, Smith)**.** *Every natural number can be written as the sum of four binary palindromes.*