Theories of automatic structures and their complexity

Dietrich Kuske

Institut für Informatik, Universität Leipzig, Germany

Abstract. For automatic structures, several logics have been shown decidable: first-order logic, its extension by the infinity quantifier, by modulo-counting quantifiers, and even by a restricted form of second-order quantification. We review these decidability proofs. As a new result, we determine the data, the expression, and the combined complexity of quantifier-classes for first-order logic. Finally, we also recall that first-order logic becomes elementary decidable for automatic structures of bounded degree.

1 Introduction

The idea of an automatic structure goes back to Büchi and Elgot who used finite automata to decide, e.g., Presburger arithmetic [9]. In essence, a structure is automatic if the elements of the universe can be represented as strings from a regular language (an element can be represented by several strings) and every relation of the structure can be recognized by a finite automaton with several heads that proceed synchronously. Automaton decidable theories [13] and automatic groups [10] are similar concepts. A systematic study was initiated by Khoussainov and Nerode [15] who also coined the name "*automatic structure*". They received increasing interest over the last years [3, 4, 17, 18, 1, 16, 20, 2, 22]; Rubin's survey [25] gives an excellent overview of the results in this area in particular regarding the structure and decidability issues. One of the main motivations for investigating automatic structures is that their first-order theories are decidable. This decidability holds even if one extends first-order logic by quantifiers "there exist infinitely many", "the number of elements satisfying φ is finite and equals (modulo q) p", and "there exists an infinite relation satisfying φ" (provided φ mentions the infinite relation only negatively).

But there exist automatic structures whose first-order theory is non-elementary (i.e., does not belong to n-EXPSPACE for any $n \in \mathbb{N}$). An inspection of the decidability proof (that we indicate in this article) shows that validity of a formula in Σ_{n+1} (i.e., with at most $n+1$ nested negations) can be decided in n-EXPSPACE. We prove this to be optimal in two very strict senses. First, we construct (for every $n \in \mathbb{N}$) a fixed formula $\varphi_n \in \Sigma_{n+1}$ such that validity in an automatic structure is complete for n-EXPSPACE (the input to this problem is a presentation of the structure by automata). Second, we also construct one automatic structure such that validity of a sentence from Σ_{n+1} is complete for

n-EXPSPACE. In other words, both the data and the expression complexity (and therefore the combined complexity) are complete for n-EXPSPACE.

In the final part, we present a class of automatic structures that allow elementary decision procedures, namely the class of structures of bounded degree [23, 21].

2 Preliminaries

Let Γ be a finite alphabet and $w \in \Gamma^*$ be a finite word over Γ. The length of w is denoted by $|w|$.

2.1 Structures

A *signature* is a finite set τ of relational symbols, where every symbol $r \in \tau$ has some fixed arity m_r. Then a *τ-structure* $\mathcal{A}$ consists of a non-empty universe A and, for every $r \in \tau$, an m_r-ary relation $r^{\mathcal{A}} \subseteq A^{m_r}$. Note that we only consider relational structures. Sometimes, we will also use constants, but in our context, a constant c can be always replaced by the unary relation $\{c\}$. Let us fix a τ-structure $\mathcal{A} = (A, (r^{\mathcal{A}})_{r\in\tau})$, where $r^{\mathcal{A}} \subseteq A^{m_r}$. To simplify notation, we will write $a \in \mathcal{A}$ for $a \in A$. For $B \subseteq A$ we define the restriction $\mathcal{A}{\restriction}B = (B, (r^{\mathcal{A}} \cap B^{m_r})_{r\in\tau})$. Given further constants $a_1, \ldots, a_n \in \mathcal{A}$, we write $(\mathcal{A}, a_1, \ldots, a_k)$ for the structure $(A, (r^{\mathcal{A}})_{r\in\tau}, a_1, \ldots, a_k)$. In the rest of the paper, we will often identify a symbol $r \in \tau$ with its interpretation $r^{\mathcal{A}}$.

A *congruence* on the structure $\mathcal{A} = (A, (r)_{r\in\tau})$ is an equivalence relation $\equiv$ on A such that for every $r \in \tau$ and all $a_1, b_1, \ldots, a_{m_r}, b_{m_r} \in A$ we have: If $(a_1, \ldots, a_{m_r}) \in r$ and $a_1 \equiv b_1, \ldots, a_{m_r} \equiv b_{m_r}$, then also $(b_1, \ldots, b_{m_r}) \in r$. As usual, the equivalence class of $a \in A$ w.r.t. $\equiv$ is denoted by $[a]_\equiv$ or just $[a]$ and $A/_\equiv$ denotes the set of all equivalence classes. We define the *quotient structure* $\mathcal{A}/_\equiv = (A/_\equiv, (r/_\equiv)_{r\in\tau})$, where $r/_\equiv = \{([a_1], \ldots, [a_{m_r}]) \mid (a_1, \ldots, a_{m_r}) \in r\}$.

2.2 Automatic structures

Let us fix $n \in \mathbb{N}$ and a finite alphabet Γ. Let $\# \notin \Gamma$ be an additional padding symbol. For words $w_1, w_2, \ldots, w_n \in \Gamma^*$ we define the *convolution* $w_1 \otimes w_2 \otimes \cdots \otimes w_n$, which is a word over the alphabet $(\Gamma \cup \{\#\})^n$, as follows: Let $w_i = a_{i,1}a_{i,2}\cdots a_{i,k_i}$ with $a_{i,j} \in \Gamma$ and $k = \max\{k_1, \ldots, k_n\}$. For $k_i < j \leq k$ define $a_{i,j} = \#$. Then $w_1 \otimes \cdots \otimes w_n = (a_{1,1}, \ldots, a_{n,1}) \cdots (a_{1,k}, \ldots, a_{n,k})$. Thus, for instance $aba \otimes bbabb = (a,b)(b,b)(a,a)(\#,b)(\#,b)$. An n-ary relation $R \subseteq (\Gamma^*)^n$ is called *automatic* if the language $\{w_1 \otimes \cdots \otimes w_n \mid (w_1, \ldots, w_n) \in R\}$ is a regular language.

An *m-dimensional (synchronous) automaton* over Γ is just a finite automaton A over the alphabet $(\Gamma \cup \{\#\})^m$ such that $L(A) \subseteq \{w_1 \otimes \cdots \otimes w_m \mid w_1, \ldots, w_m \in \Gamma^*\}$. Such an automaton defines an m-ary relation

$$R(A) = \{(w_1, \ldots, w_m) \mid w_1 \otimes \cdots \otimes w_m \in L(A)\} .$$

An *automatic presentation* is a tuple $P = (\Gamma, A_0, A_=, (A_r)_{r\in\tau})$, where:

- Γ is a finite alphabet.
- τ is a signature (the signature of P), as before m_r is the arity of the symbol $r \in \tau$.
- A_0 is a finite automaton over the alphabet Γ.
- For every $r \in \tau$, A_r is an m_r-dimensional automaton over the alphabet $\Gamma \cup \{\#\}$ such that $R(A_r) \subseteq L(A_0)^{m_r}$.
- $A_=$ is a 2-dimensional automaton over the alphabet $\Gamma \cup \{\#\}$ such that the relation $R(A_=) \subseteq L(A_0) \times L(A_0)$ is a congruence on the structure $(L(A_0), (R(A_r))_{r\in\tau})$.

The structure presented by P is the quotient

$$\mathcal{A}(P) = (L(A_0), (R(A_r))_{r\in\tau})/_{R(A_=)} .$$

A structure $\mathcal{A}$ is called *automatic* if there exists an automatic presentation P such that $\mathcal{A} \cong \mathcal{A}(P)$. We will write $[u]$ for the element $[u]_{R(A_=)}$ $(u \in L(A_0))$ of the structure $\mathcal{A}(P)$.

By SA, we denote the set of all automatic presentations (here S indicates that we work with *string*-automata). A presentation $P = (\Gamma, A_0, A_=, (A_r)_{r\in\tau})$ is called *injective* if $R(A_=)$ is the identity relation on $L(A_0)$. In this case, we can omit the automaton $A_=$ and identify P with the tuple $(\Gamma, A_0, (A_r)_{r\in\tau})$. Then iSA denotes the set of injective automatic presentations.

Examples

- All finite structures $\mathcal{A}$ are automatic with alphabet the universe of $\mathcal{A}$. While there are many infinite automatic structures (see below), there are no infinite automatic fields [16].
- The complete binary tree with universe $\{0,1\}^*$, together with the binary relations "first son" S_0, "second son" S_1, "prefix" $\leq$, and "equal length" is automatic.
- Presburger arithmetic $(\mathbb{N}, +)$ is automatic: the alphabet is $\{0,1\}$, the language of A_0 is $\{0,1\}$ where the word $a_0a_1\dots a_n$ represents the number $\sum_{0\leq i\leq n} a_i 2^i$. Differently Skolem arithmetic $(\mathbb{N}, \cdot)$ is not automatic [3]. But there is an extended notion of tree-automaticity based on tree-automata instead of finite automata. Blumensath also showed that Skolem arithmetic is tree-automatic.
- The linear order $(\mathbb{Q}, \leq)$ is automatic: The universe is $\{0,1\}^*$ with $u < v$ iff $(u \wedge v)0$ is a prefix of u or $(u \wedge v)1$ is a prefix of v (where $u \wedge v$ is the longest common prefix of u and v). This presentation is even "automatic-homogeneous": Let $u_1, \dots, u_n$ and $v_1, \dots, v_n$ be increasing sequences of equal length. Then there is an automatic automorphism f of $(\{0,1\}^*, \leq)$ mapping u_i to v_i [19].
- The rewrite graph $(\Sigma^*, \to)$ of every semi-Thue system and therefore the configuration graph of every Turing machine are automatic.
- The extension of this configuration graph by the binary relation of reachability is in general not automatic. But for pushdown automata, the configuration graph with reachability $(Q\Gamma^*, \to, \to^*)$ is automatic: a configuration is represented the control state followed by the stack content.

– The theory of automatic structures was preceeded by that of automatic groups [10] and semigroups [5]. In terms of automatic structures, a semigroup is automatic (in the original sense) if its Cayley-graph has an injective automatic presentation such that $L(A_0)$ forms a rational cross-section of the (semi-)group. Many natural groups and semigroups were shown to be automatic and therefore to have automatic Cayley-graphs:
  - rational monoids [26],
  - virtually free finitely generated, virtually free Abelian finitely generated, and hyperbolic groups [10],
  - singular Artin monoids of finite type [7], and
  - graph products of such monoids [11].

  In contrast, it seems that not many infinite groups are automatic in the sense of this article. For instance, a finitely generated group is automatic iff it is virtually Abelian [24].
– An automatic structure can be at most countably infinite. Hence, the ordinal ω_1 is certainly not automatic. Delhommé, Goranko, and Knapik proved that an ordinal α is automatic iff $\alpha < \omega^\omega$ [8].
– Let $\mathcal{B}$ denote the Boolean algebra of all finite and co-finite subsets of $\mathbb{N}$. Then an infinite Boolean algebra is automatic iff it is a finite power of $\mathcal{B}$ [16].

3 Model checking

3.1 The logic FSO

Fix a signature τ. Then let $V_0 = \{x_i \mid i \in \mathbb{N}\}$ be a countably infinite set of individual variables and, for $k \geq 1$, let $V_k = \{X_i^k \mid i \in \mathbb{N}\}$ be a set of k-ary relation variables. Formulas of FSO are then built according to the following formation rules (where α and β are formulas, $x, y, y_1, \ldots, y_k \in V_0$ are individual variables, R is a k-ary relation symbol, and $X \in V_k$ is a k-ary relation variable):

(L1) $x = y$
(L2) $R(y_1, \ldots, y_k)$
(L3) $X(y_1, \ldots, y_k)$
(L4) $\alpha \vee \beta$ and $\alpha \wedge \beta$
(L5) $\neg\alpha$
(L6) $\exists x : \alpha$
(L7) $\exists^\infty x : \alpha$
(L8) $\exists^{(p,q)} x : \alpha$ for $0 \leq p < q$
(L9) $\exists X$ infinite $: \alpha$ provided $X \in V_k \setminus \mathrm{pos}(\alpha)$

To complete this definition, we have to explain what $\mathrm{pos}(\alpha)$, the set of positively occuring relation variables, is. This is achieved by induction as follows:

(1) $\mathrm{pos}(x = y) = \mathrm{neg}(x = y) = \emptyset$
(2) $\mathrm{pos}(R(y_1, \ldots, y_k)) = \mathrm{neg}(R(y_1, \ldots, y_k)) = \emptyset$
(3) $\mathrm{pos}(X(y_1, \ldots, y_k)) = \{X\}$ and $\mathrm{neg}(X(y_1, \ldots, y_k)) = \emptyset$

(4) $\mathrm{pos}(\alpha \vee \beta) = \mathrm{pos}(\alpha) \cup \mathrm{pos}(\beta)$, $\mathrm{neg}(\alpha \vee \beta) = \mathrm{neg}(\alpha) \cup \mathrm{neg}(\beta)$, and similarly for $\alpha \wedge \beta$
(5) $\mathrm{pos}(\neg\alpha) = \mathrm{neg}(\alpha)$ and $\mathrm{neg}(\neg\alpha) = \mathrm{pos}(\alpha)$
(6) $\mathrm{pos}(\exists x : \alpha) = \mathrm{pos}(\exists x : \alpha)$ and $\mathrm{neg}(\exists x : \alpha) = \mathrm{neg}(\exists x : \alpha)$
(7) $\mathrm{pos}(\exists^{\infty} x : \alpha) = \mathrm{pos}(\alpha)$ and $\mathrm{neg}(\exists^{\infty} x : \alpha) = \mathrm{neg}(\alpha)$
(8) $\mathrm{pos}(\exists^{(p,q)} x : \alpha) = \mathrm{pos}(\alpha)$ and $\mathrm{neg}(\exists^{(p,q)} x : \alpha) = \mathrm{neg}(\alpha)$
(9) $\mathrm{pos}(\exists X \text{ infinite} : \alpha) = \mathrm{pos}(\alpha)$ and $\mathrm{neg}(\exists X \text{ infinite} : \alpha) = \mathrm{neg}(\alpha) \setminus \{X\}$

Before we define the semantics, we observe that FSO contains several interesting fragments:

- If we only allow the formation rules (L1,2,4,5,6), we obtain first-order logic FO.
- If, in addition, (L7) is allowed, we obtain FO[$\exists^{\infty}$].
- Similarly, FO[$\exists^{\infty}, \exists^{\mathrm{mod}}$] is obtained by allowing all the rules except (L3) and (L9).

We next define the semantics of these formulas. To this aim, let $\mathcal{A}$ be a τ-structure with universe A. An *interpretation in* $\mathcal{A}$ is a family $f = (f_k)_{k \geq 0}$ of functions with $f_0 : V_0 \to \mathcal{A}$ and $f_k : V_k \to 2^{\mathcal{A}^k}$ for all $k \geq 1$. Given such an interpretation, we set $\mathcal{A} \models_f \varphi$ (read as "φ holds in $\mathcal{A}$ under the interpretation f") iff one of the following hold

(S1) $\varphi = (x = y)$ and $f_0(x) = f_0(y)$.
(S2) $\varphi = (R(y_1, \ldots, y_k))$ and $(f_0(y_1), \ldots, f_0(y_k)) \in R^{\mathcal{A}}$.
(S3) $\varphi = (X(y_1, \ldots, y_k))$ and $(f_0(y_1), \ldots, f_0(y_k)) \in f_k(X)$.
(S4) $\varphi = (\alpha \vee \beta)$ and $\mathcal{A} \models_f \alpha$ or $\mathcal{A} \models_f \beta$, or
$\varphi = (\alpha \wedge \beta)$, $\mathcal{A} \models_f \alpha$, and $\mathcal{A} \models_f \beta$.
(S5) $\varphi = \neg\alpha$ and not $\mathcal{A} \models_f \alpha$.
(S6) $\varphi = \exists x : \alpha$ and there exists $a \in \mathcal{A}$ with $\mathcal{A} \models_{f[^a_x]} \alpha$ where $f[^a_x] = g$ is a family of functions $(g_k)_{k \geq 0}$ with $g_k = f_k$ for all $k \geq 1$, $g_0(x) = a$, and $g_0(y) = f_0(y)$ for all $y \in V_0 \setminus \{x\}$.
(S7) $\varphi = \exists^{\infty} x : \alpha$ and there exist infinitely many $a \in \mathcal{A}$ with $\mathcal{A} \models_{f[^a_x]} \alpha$.
(S8) $\varphi = \exists^{(p,q)} x : \alpha$ and the number of elements $a \in \mathcal{A}$ with $\mathcal{A} \models_{f[^a_x]} \alpha$ is finite and congruent p modulo q.
(S9) $\varphi = \exists X$ infinite $: \alpha$ and there exists an infinite set $B \subseteq \mathcal{A}^k$ with $\mathcal{A} \models_{f[^B_X]} \alpha$ (where we assume $X \in V_k$).

Note that the formulas $\exists^{\infty} x : \varphi$ and $\neg\exists^{(0,1)} x : \varphi$ are equivalent. Therefore, above, we did not define the fragment FO[$\exists^{\mathrm{mod}}$] since it would be equivalent with the more liberal FO[$\exists^{\infty}, \exists^{\mathrm{mod}}$].

It is an easy exercise to show the following: let $\mathcal{A}$ be a τ-structure, φ a formula, and suppose $f(y) = g(y)$ for all $y \in \mathrm{free}(\varphi)$, the set of variables occurring freely in φ. Then $\mathcal{A} \models_f \varphi$ iff $\mathcal{A} \models_g \varphi$. Assuming a fixed tuple of variables $(y_1, \ldots, y_n)$ with $y_i \in \mathrm{free}(\varphi)$ for all $1 \leq i \leq n$, we can therefore simply write $\mathcal{A} \models \varphi(f(y_1), \ldots, f(y_n))$ for $\mathcal{A} \models_f \varphi$. For *sentences* (i.e., formulas without free variables), it makes in particular sense to write $\mathcal{A} \models \varphi$.

3.2 The model checking problem

Let $\mathsf{C} \subseteq \mathsf{SA}$ be a class of automatic presentations and $L \subseteq \mathrm{FSO}$ a set of formulas. Then

$$\{(\varphi, P) \mid \varphi \in L \text{ sentence}, P \in \mathsf{C}, \mathcal{A}(P) \models \varphi\}$$

is the *model checking problem* $\mathrm{MC}(L, \mathsf{C})$ *for* L *and* C, i.e., it is the following problem:
INPUT: a sentence φ from L and an automatic presentation P from C.
OUTPUT: Does $\mathcal{A}(P) \models \varphi$ hold?

To solve the most general model checking problem $\mathrm{MC}(\mathrm{FSO}, \mathsf{SA})$, one proceeds as follows:

Proposition 3.1 (cf. [15, 3, 17, 22]). *Let P be an automatic presentation and $\varphi \in \mathrm{FSO}$ a formula with $\mathrm{free}(\varphi) \subseteq \{y_1, \ldots, y_m\} \subseteq V_0$. Then the relation $R = \{(u_1, \ldots, u_m) \in L(A_0)^m \mid \mathcal{A}(P) \models \varphi([u_1], \ldots, [u_m])\}$ is regular. Even more, an m-dimensional automaton for this relation can be computed.*

First assume P to be injective and therefore $\mathcal{A}(P) = (L(A_0), (R(A_r))_{r \in \tau})$. If φ is a first-order formula, then the automaton is constructed by induction on the structure of the formula φ: disjunction corresponds to the disjoint union of automata, existential quantification to projection, and negation to complementation [15]. The quantifier $\exists^\infty$ can be reduced to the first-order case as follows [3]: Let $\mathcal{B}$ be the extension of $\mathcal{A}(P)$ by the length-lexicographic order $\leq_{\mathrm{ll}}$ on $L(A_0)$. Then $\mathcal{B}$ is still automatic and a formula of the form $\exists^\infty x : \varphi$ is equivalent with $\forall y \exists x (y \leq_{\mathrm{ll}} x \wedge \varphi)$. This allows to apply the result for first-order logic. Such a reduction is not possible for the quantifiers $\exists^{(p,q)}$, but explicite automata-constructions provide the solution [17] (I recommend the presentation in [25, Proof of Thm. 3.19]). The basic idea is that a finite $(k+1)$-dimensional automaton A can be transformed into a k-dimensional that, on input of k words $(u_1, \ldots, u_k)$ determines, modulo q, the number of words u_{k+1} such that $(u_1, \ldots, u_k, u_{k+1})$ is accepted by A.

The idea for handling the remaining quantifier $\exists X$ infinite is as follows (cf. [22] for the details): Let σ be the extension of the signature τ by unary relation symbols L and C_k and $(k+1)$-ary relation symbols el_k for $k \geq 1$. Then let $\mathcal{B}$ be the structure obtained from $\mathcal{A}(P) = (L(A_0), (R(A_r))_{r \in \tau})$ as follows

- $L^{\mathcal{B}} = L(A_0)$ is the universe of $\mathcal{A}(P)$
- $C_k^{\mathcal{B}} \subseteq 2^{L(A_0)^k}$ is the set of all infinite k-ary relations on $L(A_0)$
- the universe of $\mathcal{B}$ consists of the language $L(A_0)$ and all infinite relations, i.e., $B = L(A_0) \cup \bigcup_{k \geq 1} C_k^{\mathcal{B}}$,
- the relations $r^{\mathcal{B}}$ and $r^{\mathcal{A}(P)} = R(A_r)$ coincide for $r \in \tau$, and
- $\mathrm{el}_k^{\mathcal{B}} \subseteq L(A_0)^k \times C_k^{\mathcal{B}}$ is the set of all $(k+1)$-tuples $(u_1, \ldots, u_k, X)$ with $(u_1, \ldots, u_k) \in X$.

Now the formula $\varphi \in \mathrm{FSO}$ can be easily translated into a formula ψ from $\mathrm{FO}[\exists^\infty, \exists^{\mathrm{mod}}]$ with

$$\mathcal{A}(P) \models \varphi(u_1, \ldots, u_n) \iff \mathcal{B} \models \psi(u_1, \ldots, u_n) \ .$$

The article [18] (see also [25]) provides an encoding of certain infinite sets of words by infinite words: A *word comb* is an infinite set $\{s_0 s_1 \dots s_{i-1} t_i \mid i \in \mathbb{N}\} \subseteq \Sigma^*$ such that $0 < |s_i| < |t_i|$ for all $i \in \mathbb{N}$. A relation $R \subseteq (\Gamma^*)^k$ can be encoded if the set of convolutions $\{w_1 \otimes w_2 \cdots \otimes w_k \mid (w_1, w_2, \dots, w_k) \in R\} \subseteq (\Gamma \cup \{\#\})^*$ is a word comb. Then a compactness argument shows that any infinite k-ary relation on finite words contains an infinite subset that can be encoded in this particular way. In a second step, we restrict the sets $C_k^{\mathcal{B}}$ to these "encodeable" relations and denote the resulting structure by $\mathcal{B}'$. The restriction in formation rule (L9) then ensures

$$\mathcal{B} \models \psi(u_1, \dots, u_n) \iff \mathcal{B}' \models \psi(u_1, \dots, u_n) \ .$$

The particular coding from [18] ensures that the structure $\mathcal{B}'$ has an injective "ω-automatic presentation" P'. These ω-automatic presentations are defined in the same way as automatic presentations, but using Büchi- instead of finite automata. Techniques similar to the above for FO$[\exists^\infty, \exists^{\bmod}]$ provide a k-dimensional Büchi-automaton for the relation defined by ψ in $\mathcal{B}'$ [20] that can be transformed into a k-dimensional finite automaton for the relation defined by φ in $\mathcal{A}(P)$.

Now let P be non-injective. Then the set of all words from $L(A_0)$ that are length-lexicographically minimal in their equivalence class (as determined by the automaton $A_=$) is effectively regular [15]. This allows to compute an equivalent injective presentation $P' = (B_0, (B_r)_{r \in \tau})$ with $L(B_0) \subseteq L(A_0)$ and $R(B_r) = R(A_r) \cap L(B_0)^k$ for all k-ary relation symbols $r \in \tau$. Then by the above, one can compute an m-dimensional automaton A with $R(A) = \{(v_1, \dots, v_m) \in L(B_0)^m \mid \mathcal{A}(P') \models \varphi(v_1, \dots, v_m)\}$. Hence $P_h = (A_0, A, A_=)$ is an injective automatic presentation and R is the set of tuples $(u_1, \dots, u_m) \in L(A_0)^m$ with

$$\mathcal{A}(P_h) \models \exists v_1, \dots, v_m : (v_1, \dots, v_m) \in R \wedge \bigwedge_{1 \le i \le m} (u_i, v_i) \in R(A_=) \ .$$

This finishes the proof of Prop. 3.1.

Now we come to a direct consequence of Prop. 3.1: to decide whether the FSO-sentence φ holds, one adds a dummy variable $x \in V_0$ and then computes an automaton A with $L(A) = \{u \in L(A_0) \mid \mathcal{A} \models \varphi([u])\}$. Then $\mathcal{A} \models \varphi$ iff $L(A) \neq \emptyset$ which is decidable.

Theorem 3.2 (cf. [15, 3, 17, 22]). *The model checking problem* MC(FSO, SA) *for all automatic presentations is decidable. In particular, the* FSO*-theory* $\{\varphi \in \text{FSO} : \mathcal{A}(P) \models \varphi\}$ *(that corresponds to* MC(FSO, $\{P\}$)*) of every automatic structure* $\mathcal{A}(P)$ *is decidable.*

Since the first-order theory of the binary tree $(\{0,1\}, S_0, S_1, \le)$ is non-elementary (cf. [6, Example 8.1]), there cannot be an elementary algorithm for deciding the model checking problem MC(FSO, SA). The following two sections analyse this situation a bit further for first-order logic.

Since the number of nested negations will be crucial in this analysis, we define the following classes of formulas of FO:

– Σ_0 is the set of quantifier-free formulas, i.e., those build according to the formation rules (L1,2,4,5).
– $B\Sigma_n$ is the set of Boolean combinations of formulas from Σ_n, i.e., we close the set Σ_n with respect to the formation rules (L4,5).
– Σ_{n+1} is the closure of the set $B\Sigma_n$ with respect to the formation rules (L4,6).

By de Morgan's rules, we can eliminate from every Σ_0-formula any nesting of negations. Since $\Sigma_0 = B\Sigma_0$ and since the formation of Σ_1 does not involve additional negations, the same applies to this set. By induction, we can rewrite every $B\Sigma_n$-formula (for $n > 0$) such that it has at most $n + 1$ nested negations and the same applies to Σ_{n+1}-formulas. Note that this process does not increase the size of the formula.

So let $\varphi \in \Sigma_{n+1}$ have at most $n+1$ nested negations and consider the proofs of Prop. 3.1 and Theorem 3.2. Since each complementation increases the size of the automaton exponentially, the automaton A from Prop. 3.1 has $(n + 1)$-fold exponential size (in the formula φ and the presentation P). Since non-emptiness of the language of a finite automaton is in NL, the decision procedure indicated above requires n-fold exponential space. Thus, we have the following more precise statement of Theorem 3.2 for first-order logic:

Proposition 3.3. *For all $n \geq 0$, the model checking problem* $\mathrm{MC}(\Sigma_{n+1}, \mathsf{SA})$ *belongs to n-*EXPSPACE *(where* 0-EXPSPACE = PSPACE*).*

From the result of the following sections, it will follow that this upper bound is optimal.

3.3 Data complexity

In this section, we want to construct, for every $n \in \mathbb{N}$, a first-order sentence $\varphi_n \in \Sigma_{n+1}$ such that the question "Does φ_n hold in the structure $\mathcal{A}(P)$?" is hard for n-fold exponential space. To this aim, we define the following family of functions $F_n : \mathbb{N} \to \mathbb{N}$ by induction:

$$F_0(m) = m \text{ and } F_{n+1}(m) = F_n(m) \cdot 2^{F_n(m)} \ .$$

Note that $F_n(m)$ is an n-fold exponential function. Hence there is a deterministic Turing machine M with an n-EXPSPACE-complete language that runs in space $F_n(|w|) - 2$ for every sufficiently long input w. Let Q be the set of states of M, $q_0 \in Q$ the initial state, $q_f \in Q$ the accepting state, and Γ_{tape} the tape alphabet. For $m \in \mathbb{N}$, an m-configuration of M is a word from $\Gamma_{\text{tape}}^* Q \Gamma_{\text{tape}}^+$ of length $F_n(m) - 1$ ($u\,q\,v$ denotes the configuration with tape content uv, control state q, and head position the first symbol of v). By $\vdash$, we denote the one-step relation of M. An *m-computation* of M is a word $\$c_0\$c_1 \ldots \$c_k\$$ over $\Gamma = Q \cup \Gamma_{\text{tape}} \cup \{\$\}$ where $k \in \mathbb{N}$ is arbitrary, $c_0, c_1, \ldots, c_k$ are m-configurations of equal length with $c_i \vdash c_{i+1}$ for all $0 \leq i < k$, and \$ is an additional delimiter. An m-computation is *successful* if $c_k \in \Gamma_{\text{tape}}^* q_f \Gamma_{\text{tape}}^+$, it is *with input* $w \in \Gamma_{\text{tape}}^*$ if $c_0 \in q_0 w \square^*$ where $\square$ is the blank symbol of the machine M.

Now let w be some input word and let m be its length. We construct an injective automatic presentation P_w such that the acceptance of w by M is equivalent to validity of a formula $\varphi_n \in \Sigma_{n+1}$ that does not depend on the word w. The structure $\mathcal{A}(P_w)$ consists of two parts that both depend on the input word w: the alphabet of the first is Γ, that of the second is $\{0,1\}$. Later, we will present formulas $\lambda_i(s) \in \Sigma_i$ such that $\mathcal{A}(P_w) \models \lambda_i(s)$ for $s \in \{0,1\}^*$ iff $s \in L_i = 0^*10^{F_i(m)-1}10^*$, i.e., iff $s \in \{0,1\}^*$ contains precisely two occurrences of 1 and these two occurrences are $F_i(m)$ apart. But first, we describe the first part of the structure $\mathcal{A}(P_w)$: Its universe is the set Γ^* and its core is the binary relation StepInBlocks. A pair of words (c, c') belongs to StepInBlocks if and only if

- $c = \$c_0\$c_1\$\dots\$c_k\$$ and $c' = \$c'_0\$c'_1\$\dots\$c'_k\$$ for some configurations c_i and c'_i,
- $c_0 = c'_0$,
- $|c_i| = |c'_i|$ and $c_i \vdash c'_i$ for all $1 \leq i \leq k$, and
- $c_k \in \Gamma^*_{\text{tape}} q_f \Gamma^+_{\text{tape}}$ is some accepting configuration.

Then w is accepted by M iff there exists a word $c = \$c_0\$c_1\$\dots\$c_k\$$ with $c_i \in (\Gamma \setminus \{\$\})^*$ such that

(A1) $c_0 \in q_0 w \,\square^*$ is of length $F_n(m) - 1$ and
(A2) $(\$c_0\$c_0\$c_1\$\dots\$c_{k-1}\$, \$c_0\$c_1\$\dots\$c_k\$) \in$ StepInBlocks.

To express (A1), we use the following two automatic relations:

- W^w is unary and consists of all words from $\$q_0 w\,\square^*\$$.
- Prefix is binary and consists of all pairs $(u, v) \in \Gamma^* \times \Gamma^*$ where u is a prefix of v.

Let $x \in W^w$ be some prefix of c from W^w (i.e., $x = \$c_0\$$). To express that c_0 is of length $F_n(m) - 1$, we will actually express that the delimiter \$ occurs at positions in x that are $F_n(m)$ apart. To this and later purposes, we will use the following ternary relation:

- EqLet $\subseteq \{0,1\}^* \times [(\Gamma^* \times \Gamma^*) \cup (\{0,1\}^* \times \{0,1\}^*)]$ is the set of triples $(0^{k_0-1}10^{k_1-1}10^{k_2}, x, y)$ such that the letter at position k_0 in x equals the letter at position $k_0 + k_1$ in y.

Recall that $\lambda_n(s)$ is a formula that defines the set $L_n = 0^*10^{F_n(m)-1}10^*$. Then (A1) is equivalent to

$$\begin{aligned} \mathcal{A}(P_w) \models \exists x : \quad & W^w(x) \wedge \text{Prefix}(x, c) \\ & \wedge \exists x' : x' \in 1\{0,1\}^* \wedge \lambda_n(x') \wedge \text{EqLet}(x', x, x)\ . \end{aligned} \tag{1}$$

Here, the second line ensures that the letters number 1 and $F_i(m) + 1$ of x are equal. Hence $x \in W^w = \$q_0 w 0^*\$$ equals $\$q_0 w 0^{F_n(m)-m-2}\$$. Since it is a prefix of $c = \$c_0\$c_1 \dots c_k\$$, Eq. 1 is equivalent to $c_0 = q_0 w 0^{F_n(m)-m-2}$ and therefore to (A1).

Next we argue that, given (A1) (and therefore in particular $|c| > F_n(m)$), (A2) is equivalent to

$$\mathcal{A}(P_w) \models \exists c' : \quad \text{StepInBlocks}(c', c) \\ \wedge \forall x' : (\lambda_n(x') \wedge |x'| \leq |c'| \rightarrow \text{EqLet}(x', c, c')) \ . \tag{2}$$

By StepInBlocks(c', c), the words c and c' have the same length, are sequences of configurations, and c' starts with $\$c_0\$$. The second conjunct expresses that, for all $0 < i \leq |c'| - F_n(m)$, the i^{th} letter of c and the letter number $i + F_n(m) + 1$ of c' coincide, i.e., c' is the prefix of $\$c_0c$ of length $|c|$. But this is equivalent with $c' = \$c_0\$c_0\$c_1 \ldots \$c_{k-1}\$$ as required by (A2).

It remains to present the formula λ_n that is build by induction and uses not-yet-defined relations on the second part of $\mathcal{A}(P_w)$. Before we present these relations, we need the following auxiliary definitions:

- For $x = x_0x_1 \ldots x_k$ with $x_i \in \{0, 1\}$, let $\text{val}(x) = \sum_{0 \leq i \leq k} x_i 2^i$, i.e., the word x is considered as binary number written with the least significant bit first.
- For $x = x_1x_2 \ldots x_k$ with $x_i \in \{0, 1\}^*$ and $1 \leq i \leq j \leq k$, let $x[i, j) = x_ix_{i+1} \ldots x_{j-1}$ be the factor of x from position i to position $j - 1$.

The ternary relation DecBlocks is the core of the second part of the automatic structure $\mathcal{A}(P_w)$, it is very similar to the relation StepInBlocks from the first part:

- Let $x \in \{0, 1\}^*$ such that $0 < k_0 < k_1 \cdots < k_\ell$ are the positions of 1 in x. Then the triple (x, y, z) of words of equal length belongs to DecBlocks iff
  1. $\text{val}(y[k_0, k_1)) = 0$ (i.e., $y[k_0, k_1) = 0^{k_1 - k_0}$),
  2. $\text{val}(y[k_j, k_{j+1})) - 1 = \text{val}(z[k_j, k_{j+1}))$ for all $1 \leq j < \ell$, and
  3. $\text{val}(y[k_{\ell-1}, k_\ell)) = 2^{k_\ell - k_{\ell-1}} - 1$ (i.e., $y[k_{\ell-1}, k_\ell) = 1^{k_\ell - k_{\ell-1}}$).

The idea is that the first word divides the second and third into blocks that are decremented separately. In addition, the first and last blocks of the second word have the minimal and maximal possible value. Note in particular that y has only one block of value 0 (since all the other can be decremented).

Further relations on the second part of $\mathcal{A}(P_w)$ are the following that all can be accepted by automata whose size is polynomial in the size m of the input word w:

- $L^w = 0^*10^{m-1}10^*$.
- $S^w = 0^*(10^{m-1})^+10^*$ is the set of words with at least two occurrences of 1 such that consecutive occurrences are m positions apart.
- S_1 is the set of pairs $(x, 0^{k_0-1}10^{k_1-1}10^{k_2-1})$ of words of equal length such that k_0 and $k_0 + k_1$ are the first and last occurrences of the letter 1 in x.
- T^w is the set of pairs (x, y) of words of equal length such that, for some $1 \leq k_0 \leq |y| - m$, the letter at position k_0 in x is different from the letter at position $k_0 + m$ in y.

– S_2 is the set of pairs $(0^{k_0-1}10^{k_1-1}10^{k_2}, y)$ of words of equal length such that the positions k_0 and $k_0 + k_1$ are either consecutive occurrences of 1 in y, or they both are occurrences of 0 in y.

Next we define formulas $\lambda_i \in \Sigma_i$ that define the sets L_i for $0 \leq i \leq n$: The formula $\lambda_0(s) = (L^w(s))$ is the trivial starting point. Let $\lambda_1(s)$ denote the following formula:

$$\exists x_1, x_2, x_3 : S^w(x_1) \wedge S_1(x_1, s) \wedge \neg T^w(x_2, x_3) \wedge \mathrm{DecBlocks}(x_1, x_2, x_3)$$

The formula $S^w(x_1)$ ensures $x_1 = 0^a(10^{m-1})^\ell 10^b$ for some $a, b, \ell \in \mathbb{N}$, $\ell \geq 1$. Then $S_1(x_1, s)$ expresses $s = 0^a 10^{\ell m-1} 10^b$. By $\mathrm{DecBlocks}(x_1, x_2, x_3)$, we know $|x_1| = |x_2| = |x_3|$ and

1. $\mathrm{val}(x_2[a, a + m)) = 0$,
2. $\mathrm{val}(x_2[a + jm, a + (j + 1)m)) - 1 = \mathrm{val}(x_3[a + jm, a + (j + 1)m))$ for all $1 \leq j < \ell$, and
3. $\mathrm{val}(x_2[a + (\ell - 1)m, a + \ell m)) = 2^m - 1$.

Finally, the formula $\neg T^w(x_2, x_3)$ expresses that for all $0 < k \leq |x_2| - m$, the letter at position k in x_2 equals that at position $k + m$ in x_3. Hence we have

$$\begin{aligned}\mathrm{val}(x_2[a + jm, a + (j + 1)m)) - 1 &= \mathrm{val}(x_3[a + jm, a + (j + 1)m)) \\ &= \mathrm{val}(x_2[a + (j - 1)m, a + jm))\end{aligned}$$

for all $1 \leq j < \ell$, i.e., the blocks (as determined by x_1) in x_2 carry consecutive numbers from 0 to $2^m - 1$. Since $\mathrm{DecBlocks}(x_1, x_2, x_3)$ also implies that no other than the first block of x_2 carries 0, we have precisely 2^m blocks of length m each. Hence $1 + (\ell m - 1) = m \cdot 2^m = F_1(m)$ which is equivalent with $s \in L_1$. Thus, indeed, $\lambda_1(s) \in \Sigma_1$ defines the set L_1.

Now we proceed by induction on i and let λ_{i+1} denote the following formula:

$$\begin{aligned}\exists x_1, x_2, x_3 : {} & S_1(x_1, s) \wedge \mathrm{DecBlocks}(x_1, x_2, x_3) \wedge \\ & x_2 \notin 0^* \cup 1^* \wedge \exists x_4 : \lambda_i(x_4) \wedge |x_4| \leq |x_2| \wedge \\ & \forall x_4 : \lambda_i(x_4) \wedge |x_4| \leq |x_2| \rightarrow S_2(x_4, x_2) \wedge \\ & \forall x_4 : \lambda_i(x_4) \wedge |x_4| = |x_3| \rightarrow \mathrm{EqLet}(x_4, x_2, x_3)\end{aligned}$$

By the quantified formula in the second line and the induction hypothesis, $|x_2| > F_i(m) \geq m$. Hence the third line ensures that x_2 is of the form $0^a(10^{F_i(m)-1})^\ell 10^b$ for some $a, b, \ell \in \mathbb{N}$ with $\ell \geq 1$. Then the last line expresses that, for all $0 < k \leq |x_3| - F_i(m)$, the letter at position k in x_2 equals the letter at position $k + F_i(m)$ in x_3. In particular,

$$\begin{aligned}x_2[a + kF_i(m), & a + (k + 1)F_i(m)) \\ & = x_3[a + (k + 1)F_i(m), a + (k + 2)F_i(m))\end{aligned} \tag{3}$$

for all $0 \leq k < \ell$. Now, by the first line, $s = 0^a 10^{\ell F_i(m)-1} 10^b$ and

– $\mathrm{val}(x_2[a, a + F_i(m))) = 0$,
– $\mathrm{val}(x_2[a+kF_i(m), a+(k+1)F_i(m))-1 = \mathrm{val}(x_3[a+kF_i(m), a+(k+1)F_i(m))$ for all $0 < k < \ell$, and
– $\mathrm{val}(x_3[a + (\ell - 1)F_i(m), a + \ell F_i(m))) = 2^{F_i(m)} - 1$.

Now, together with Eq. 3, we obtain as above

$$\begin{aligned}\mathrm{val}(x_2[a + kF_i(m),&a + (k + 1)F_i(m))) - 1\\ &= \mathrm{val}(x_3[a + kF_i(m), a + (k + 1)F_i(m)))\\ &= \mathrm{val}(x_2[a + (k - 1)F_i(m), a + kF_i(m)))\end{aligned}$$

for all $1 \leq k < \ell$. As above, this ensures $1 + (\ell m - 1) = F_i(m) \cdot 2^{F_i(m)} = F_{i+1}(m)$ which is equivalent with $s \in L_{i+1}$.

Assuming by induction $\lambda_i \in \Sigma_i$, we obtain $\lambda_{i+1} \in \Sigma_{i+1}$ as required.

Now we can complete the definition of the automatic structure $\mathcal{A}(P_w)$: Its universe is the set $\Gamma^* \cup \{0, 1\}^*$ and it has the following automatic relations:

– W^w, L^w, S^w, and T^w (these relations depend on the word w).
– StepInBlocks, Prefix, EqLet, DecBlocks, S_1, S_2, $1\{0, 1\}^*$, and $0^* \cup 1^*$ (these relations are independent from the word w).

Summarizing, we have the following: an input word w of length m is accepted by the machine M if and only if $\mathcal{A}(P_w) \models \exists c : c \in C \wedge \alpha_1 \wedge \alpha_2$ where α_1 and α_2 are the Σ_{n+1}-formulas from Eq. 1 and Eq. 2, resp. Note that these formulas are independent from the word w and that the automata from P_w can be computed from w in polynomial time. Since the language of the machine M is complete for n-EXPSPACE, we therefore proved

Proposition 3.4. *For $n \geq 0$, there exists a sentence $\varphi_n \in \Sigma_{n+1}$ such that the model checking problems* $\mathrm{MC}(\{\varphi\}, \mathsf{SA})$ *and* $\mathrm{MC}(\{\varphi\}, \mathsf{iSA})$ *are complete for* n-EXPSPACE.

From Propositions 3.3 and 3.4, we obtain immediately

Corollary 3.5. *For all $n \in \mathbb{N}$, the model checking problems* $\mathrm{MC}(\Sigma_{n+1}, \mathsf{SA})$ *and* $\mathrm{MC}(\Sigma_{n+1}, \mathsf{iSA})$ *are complete for* n-EXPSPACE.

3.4 Expression complexity

In the previous section, we constructed a fixed sentence $\varphi_n \in \Sigma_{n+1}$ and, from an input word w, an automatic presentation P_w such that acceptence of w is equivalent to validity of φ_n in $\mathcal{A}(P_w)$. In this section, we proceed complementary: we construct a fixed automatic presentation P and, from an input word w, a sentence $\varphi_n^w \in \Sigma_{n+1}$ such that acceptence of w is equivalent to validity of φ_n^w in $\mathcal{A}(P)$. Consequently, we will prove that the Σ_{n+1}-theory of $\mathcal{A}(P)$ is hard for n-EXPSPACE. Note the subtlety that the presentation P is even independent from n, i.e., the hardness holds for all $n \in \mathbb{N}$. Therefore, we now assume the Turing machine M to be universal. Furthermore, we define the following functions

$G_n : \mathbb{N} \to \mathbb{N}$ that are a slight variation of the functions F_n from the previous section:

$$G_0(m) = m, G_1(m) = 2^m, \text{ and } G_{n+1}(m) = G_n(m) \cdot 2^{G_n(m)}$$

Note that $G_n(m)$ is an n-fold exponential function. Hence, for every $n \in \mathbb{N}$, the following language is complete for n-EXPSPACE:

$$M_n = \{w \in L(M) \mid M \text{ accepts } w \text{ in space } G_n(|w|) - 2\}$$

Now let $w = a_1 a_2 \dots a_m$ be some input word of length m. We construct an injective automatic presentation P (that does not depend on w) such that the acceptance of w by M in space $G_n(|w|)$ is equivalent to validity of a formula $\varphi_n^w \in \Sigma_{n+1}$ of polynomial size. As before, the structure $\mathcal{A}(P)$ consists of two parts: the alphabet of the first is Γ, that of the second is $\{0,1\}$. Later, we will present formulas $\lambda_i'(s) \in \Sigma_i$ such that $\mathcal{A}(P) \models \lambda_i'(s)$ for $s \in \{0,1\}^*$ iff $s \in L_i' = 0^*10^{G_i(m)-1}10^*$, i.e., iff $s \in \{0,1\}^*$ contains precisely two occurrences of 1 and these two occurrences are $G_i(m)$ apart.

But first, we describe the first part of the structure $\mathcal{A}(P)$. Recall that the relation W^w is the only one in the first part of $\mathcal{A}(P_w)$ that depends on the input word w. It is therefore our task to replace it by relations independent from w, and then express membership in W^w by a small and simple formula. To this aim, we use the following relations:

- Succ_a for $a \in \Gamma \cup \{0,1\}$ consists of all pairs (u, ua) with $u \in \Gamma^*$ if $a \in \Gamma$ and $u \in \{0,1\}^*$ if $a \in \{0,1\}$.
- $\mathrm{Succ}_{\square^*\$}$ consists of all pairs (u, uv) with $u \in \Gamma^*$ and $v \in \square^*\$$.

Now consider the following formula:

$$\begin{aligned} \exists x, x_0, \dots, x_m : \quad & \mathrm{Succ}_{q_0}(\varepsilon, x_0) \wedge \bigwedge_{0 \le i < m} \mathrm{Succ}_{a_{i+1}}(x_i, x_{i+1}) \\ & \wedge \mathrm{Succ}_{\square^*\$}(x_m, x) \wedge \mathrm{Prefix}(x, c) \\ & \wedge \exists x' : x' \in 1\{0,1\}^* \wedge \lambda_n'(x') \wedge \mathrm{EqLet}(x', x, x) \end{aligned} \tag{4}$$

Given the relations Succ_x, it is equivalent to the formula from Eq. 1, but this time, it depends on the word w. This completes the changes regarding the first part of the automatic structure.

The second part of the structure $\mathcal{A}(P_w)$ contains the relations L^w, S^w, and T^w that all depend on the word w and therefore have to be replaced. The following formula $\lambda_0'(s)$

$$s \in 0^*10^*10^* \wedge \exists x_0, x_1, \dots, x_m : \begin{pmatrix} \mathrm{Succ}_1(x_0, x_1) \wedge \bigwedge_{1 \le i < m} \mathrm{Succ}_0(x_i, x_{i+1}) \\ \wedge \mathrm{Succ}_1(x_{m-1}, x_m) \wedge \mathrm{Prefix}(x_m, s) \end{pmatrix}$$

is equivalent with $\lambda_0(s)$, i.e., $\lambda_0'(s)$ holds iff $s \in L_0' = 0^*10^{m-1}10^*$. But differently from $\lambda_0(s)$, it belongs to Σ_1.

Next we deal with the formula λ_1' that defines the set $L_1' = 0^*10^{G_1(m)-1}10^*$. Consider the two automata A_0 and A_1 from Fig. 1 that accept the relations

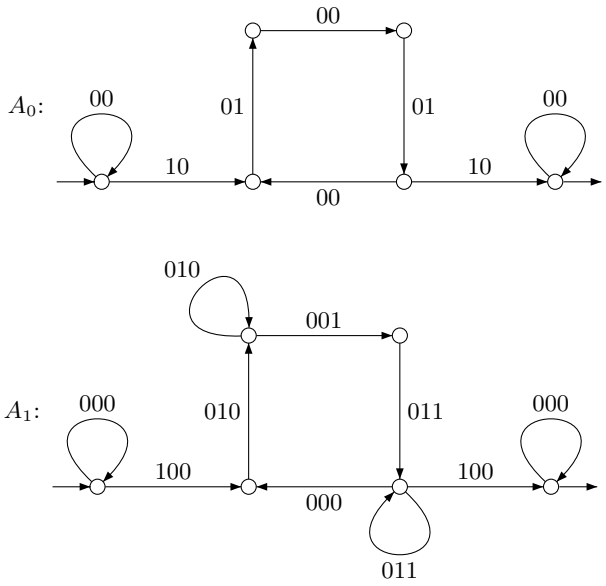

Fig. 1. The automata for R_0 and R_1

$R(A_0) = R_0 \subseteq (\{0,1\}^*)^2$ and $R(A_1) = R_1 \subseteq (\{0,1\}^*)^3$. Now $\lambda_1'(s)$ is the following formula from Σ_1:

$$\exists x_1, \ldots, x_m : R_0(s, x_1) \wedge \bigwedge_{1 \le i < m} R_1(s, x_i, x_{i+1}) \wedge x_m \in 0^*1^*0^*$$

Then $R_0(s, x_1)$ ensures $s = 0^a 10^{b-1} 10^c$ for some $a, b, c \in \mathbb{N}$ and all the words $x_1, \ldots, x_m$ are of the form $0^a 0\{0,1\}^{b-1}00^c$ by $R_0(s, x_1)$ and $R_1(s, x_i, x_{i+1})$, resp. From $R_0(s, x_1)$, we also obtain $x_1 = 0^a(01)^{\frac{1}{2}b}00^c$ (in particular, b is even). Now consider $R_1(s, x_1, x_2)$: it expresses that, at any position between the two occurrences of 1 in s, the digit in the word x_1 drops from 1 to 0 iff the digit in the word x_2 changes. Hence $R_1(s, x_1, x_2)$ ensures $x_2 = 0^a(0011)^{\frac{1}{4}b}00^c$. By induction, we obtain $x_i = 0^a(0^{2^{n-1}}1^{2^{n-1}})^{\frac{1}{2^n}b}00^c$ and therefore in particular $x_m = 0^a(0^{2^{m-1}}1^{2^{m-1}})^{\frac{1}{2^m}b}00^c$. Since x_m is from $0^*1^*0^*$, this implies $b = 2^m = G_1(m)$. Conversely, only these words x_i make the above formula true so that, indeed, $\lambda_1'(s)$ expresses $s \in 0^*10^{G_1(m)-1}10^* = L_1'$ as required.

To define the formulas $\lambda_i'(s) \in \Sigma_i$ for $i > 1$, we can proceed as in the previous section.

Now we can complete the definition of the automatic structure $\mathcal{A}(P_w)$: Its universe is the set $\Gamma^* \cup \{0,1\}^*$ and it has the following automatic relations that are all independent from the word w:

- Succ_a, $\mathrm{Succ}_{\square^*\$}$, R_0, R_1, StepInBlocks, Prefix, EqLet, DecBlocks, S_1, S_2, $1\{0,1\}^*$, $0^* \cup 1^*$, $0^*10^*10^*$, and $0^*1^*0^*$.

Summarizing, we have the following: an input word w of length m is accepted by the machine M in space $G_n(m)$ if and only if $\mathcal{A}(P) \models \exists c : c \in C \wedge \alpha_4 \wedge \alpha_2'$ where α_4 is the Σ_{n+1}-formula from Eq. 4 and α_2' arises from α_2 (cf. Eq. 2) by replacing λ_n by λ_n'. Note that this formula can be computed from the input word w in polynomial time. Since the language M_n is complete for n-EXPSPACE, we therefore proved the following analog of Prop. 3.4 which is, at the same time, a strengthening of Cor. 3.5:

Proposition 3.6. *There exists an injective automatic presentation P such that, for all $n \geq 0$, the model checking problems* $\mathrm{MC}(\Sigma_{n+1}, \{P\})$ *and* $\mathrm{MC}(\Sigma_{n+1}, \{P\})$ *are complete for n-*EXPSPACE*.*

3.5 Bounded degree

From Cor. 3.5, it follows immediately that $\mathrm{MC}(\mathrm{FO}, \mathsf{SA})$ cannot be decided in elementary space (although it is decidable by Theorem 3.2). In this section, we present a class of automatic presentations SAb such that $\mathrm{MC}(\mathrm{FO}, \mathsf{SAb})$ becomes elementary.

In the following, let $P = (\Gamma, A_0, A_=, (A_r)_{r \in \tau})$ be an automatic presentation and let $\mathcal{A} = \mathcal{A}(P)$. The decision procedure will be based on the naïve algorithm from Fig. 2 for deciding whether $\mathcal{A} \models \varphi$. The problem with this naïve approach

```
1   check(φ(x̄), ū) : {0,1}
          (φ(x̄) formula of quantifier rank ≤ d
          with k = |ū| = |x̄| many free variables,
          ū tuple of words from L(A_0))
2     case φ = R(x̄)
3       if ū ∈ L(A_R) then return(1) else return(0) endif
4     case φ = α ∨ β
5       return(max(check(α, ū), check(β, ū)))
6     case φ = ¬α
7       return(1 − check(α, ū))
8     case φ = ∃y : α(x̄, y)
9       return(max{check(φ_1, (ū, v)) | v ∈ L(A_0)})
```

Fig. 2. The naïve algorithm

is that the computation in line 9 requires infinitely many recursive calls if the language $L(A_0)$ is infinite. Hence our task is to find properties of the automatic presentation P that allow to effectively reduce the number of recursive calls in line 9. To describe such a condition, we need the following model theoretic definitions.

The *Gaifman-graph* $G(\mathcal{A})$ of the τ-structure $\mathcal{A}$ is the following symmetric graph:

$$G(\mathcal{A}) = (A, \{(a,b) \in A \times A \mid \bigvee_{r \in \tau} \exists (a_1, \ldots, a_{m_r}) \in r \; \exists j,k : a_j = a, a_k = b\}) \; .$$

Thus, the set of nodes is the universe of $\mathcal{A}$ and there is an edge between two elements if and only if they are contained in some tuple belonging to one of the relations of $\mathcal{A}$. The structure $\mathcal{A}$ has *bounded degree* if the Gaifman graph $G(\mathcal{A})$ has bounded degree, i.e., there exists a constant δ such that every $a \in A$ is adjacent to at most δ many other nodes in $G(\mathcal{A})$. For convenience, we say that the automatic presentation P has bounded degree if the structure $\mathcal{A}(P)$ has bounded degree.

The Gaifman-graph is also the basis for the definition of spheres: For $a, b \in \mathcal{A}$, let $d(a,b)$ be the distance between a and b in the Gaifman-graph $G(\mathcal{A})$. For $a \in \mathcal{A}$ and $r \in \mathbb{N}$, let $S(r,a)$ denote the set of elements b of $\mathcal{A}$ with $d(a,b) \leq r$. Then, for $\bar{a} = (a_1, \ldots, a_k) \in \mathcal{A}^k$ and $d \geq 0, k > 0$, we denote with $\mathcal{A}[d+k, \bar{a}]$ the induced substructure $\mathcal{A}\restriction \bigcup_{i=1}^{k} S(2^{d+k-i}, a_i)$. Given these notions, the following locality principle can be formulated.

Theorem 3.7 ([14]). *Let $\mathcal{A}$ be a structure, $\bar{a}, \bar{b} \in \mathcal{A}^k$ and $d \geq 0$ such that $(\mathcal{A}[d+k, \bar{a}],\ \bar{a}) \cong (\mathcal{A}[d+k, \bar{b}],\ \bar{b})$. Then, for every formula $\varphi(x_1, \ldots, x_k) \in \mathrm{FO}[\exists^{\infty}, \exists^{\bmod}]$ of quantifier depth at most d, we have: $\mathcal{A} \models \varphi(\bar{a}) \iff \mathcal{A} \models \varphi(\bar{b})$.*

A short remark on the history of this result: for first-order logic, it was first shown by Gaifman [12] where, in the definition of $\mathcal{A}[d+k, \bar{a}]$, he used the radius 7^d instead of 2^{d+k-i}. His result has been improved in two directions: the radius 7^d was reduced and the logic has been extended. The final result of this line of research was provided by Keisler and Lotfallah in [14] proving the above theorem in an even stronger version: instead of $\mathcal{A}[d+k, \bar{a}]$, it suffices to consider the restriction of $\mathcal{A}$ to $\bigcup_{i=1}^{k} S(2^d, a_i)$ and, secondly, the result holds for even stronger extensions of FO.

This theorem provides the first step in the restriction of the search space in line 9 of the naïve algorithm: If $(\mathcal{A}[d+k, (\bar{u}, v)], (\bar{u}, v)) \cong (\mathcal{A}[d+k, (\bar{u}, v')], (\bar{u}, v'))$, then it suffices to consider the word v. But still, there could be infinitely many isomorphism types of the form $\mathcal{A}[d+k, (\bar{u}, v)]$ – which can be excluded by requiring the structure $\mathcal{A}$ to be of bounded degree. Then pumping arguments show that it suffices to consider words of triply exponential length in line 9 of the naïve algorithm:

Theorem 3.8 ([20]). *Let P be an automatic presentation of bounded degree. Then the model checking problem* $\mathrm{MC}(\mathrm{FO}[\exists^{\infty}, \exists^{\bmod}], \{P\})$ *is in* 3-EXPSPACE.

A further improvement of the naïve algorithm is obtained if, instead of the argument $\bar{u}$, one passes the isomorphism type of the finite structure $(\mathcal{A}[d+k, \bar{u}], \bar{u})$. This approach yields the following result.

Theorem 3.9 ([21]).

(1) The model checking problem $\mathrm{MC}(\mathrm{FO}, \mathsf{iSAb})$ *belongs to* 2-EXPSPACE *where* $\mathsf{iSAb} = \mathsf{iSA} \cap \mathsf{SAb}$ *is the set of injective automatic presentations of bounded degree.*
(2) The model checking problem $\mathrm{MC}(\mathrm{FO}, \mathsf{SAb})$ *belongs to* 3-EXPSPACE.
(3) The model checking problem $\mathrm{MC}(\mathrm{FO}, \{P\})$ *belongs to* 2-EXPSPACE *for every automatic presentation* P *of bounded degree.*

Note that (1) and (2) give upper bounds for the combined complexities while (3) bounds the expression complexity of FO-model checking. This latter upper bound is shown to be tight in [21] since we constructed an injective automatic presentation of bounded degree P such that $\mathrm{MC}(\mathrm{FO}, \{P\})$ is hard for 2-EXPSPACE. Hence also the upper bound in (1) is tight. So far, nothing is known about the data complexity, i.e., the complexity of the problems $\mathrm{MC}(\{\varphi\}, \mathsf{iSAb})$ and $\mathrm{MC}(\{\varphi\}, \mathsf{SAb})$ for $\varphi \in \mathrm{FO}$. Furthermore, it is not known whether Theorem 3.8 is optimal nor what the combined or data complexity of $\mathrm{FO}[\exists^{\infty}, \exists^{\mathrm{mod}}]$ is.

References

1. V. Bárány. Invariants of automatic presentations and semi-synchronous transductions. In *STACS'06*, Lecture Notes in Comp. Science vol. 3884, pages 289–300. Springer, 2006.
2. V. Bárány, Ł. Kaiser, and S. Rubin. Cardinality and counting quantifiers on omega-automatic structures. In *STACS'08*, pages 385–396. IFIB Schloss Dagstuhl, 2008.
3. A. Blumensath. Automatic structures. Technical report, RWTH Aachen, 1999.
4. A. Blumensath and E. Grädel. Automatic Structures. In *LICS'00*, pages 51–62. IEEE Computer Society Press, 2000.
5. C.M. Campbell, E.F. Robertson, N. Ruškuc, and R.M. Thomas. Automatic semigroups. *Theoretical Computer Science*, 250:365–391, 2001.
6. K.J. Compton and C.W. Henson. A uniform method for proving lower bounds on the computational complexity of logical theories. *Annals of Pure and Applied Logic*, 48:1–79, 1990.
7. R. Corran, M. Hoffmann, D. Kuske, and R.M. Thomas. Singular Artin monoids of finite type are automatic. Submitted.
8. Ch. Delhommé, V. Goranko, and T. Knapik. Automatic linear orderings. Manuscript, 2003.
9. C.C. Elgot. Decision problems of finite automata design and related arithmetics. *Trans. Am. Math. Soc.*, 98:21–51, 1961.
10. D.B.A. Epstein, J.W. Cannon, D.F. Holt, S.V.F. Levy, M.S. Paterson, and W.P. Thurston. *Word Processing In Groups*. Jones and Bartlett Publishers, Boston, 1992.
11. E. Fohry and D. Kuske. On graph products of automatic and biautomatic monoids. *Semigroup forum*, 72:337–352, 2006.
12. H. Gaifman. On local and nonlocal properties. In J. Stern, editor, *Logic Colloquium '81*, pages 105–135. North-Holland, 1982.
13. B.R. Hodgson. On direct products of automaton decidable theories. *Theoretical Computer Science*, 19:331–335, 1982.
14. H.J. Keisler and W.B. Lotfallah. A local normal form theorem for infinitary logic with unary quantifiers. *Mathematical Logic Quarterly*, 51(2):137–144, 2005.

15. B. Khoussainov and A. Nerode. Automatic presentations of structures. In *Logic and Computational Complexity*, Lecture Notes in Comp. Science vol. 960, pages 367–392. Springer, 1995.
16. B. Khoussainov, A. Nies, S. Rubin, and F. Stephan. Automatic structures: richness and limitations. *Log. Methods in Comput. Sci.*, 3(2), 2007.
17. B. Khoussainov, S. Rubin, and F. Stephan. Definability and regularity in automatic structures. In *STACS'04*, Lecture Notes in Comp. Science vol. 2996, pages 440–451. Springer, 2004.
18. B. Khoussainov, S. Rubin, and F. Stephan. Automatic linear orders and trees. *ACM Transactions on Computational Logic*, 6(4):675–700, 2005.
19. D. Kuske. Is Cantor's theorem automatic? In *LPAR'03*, Lecture Notes in Comp. Science vol. 2850, pages 332–345. Springer, 2003.
20. D. Kuske and M. Lohrey. First-order and counting theories of ω-automatic structures. *Journal of Symbolic Logic*, 73:129–150, 2008.
21. D. Kuske and M. Lohrey. Automatic structures of bounded degree revisited. In *CSL'99*, Lecture Notes in Comp. Science. Springer, 2009. To appear.
22. D. Kuske and M. Lohrey. Some natural problems in automatic graphs. *Journal of Symbolic Logic*, 2009. Accepted.
23. M. Lohrey. Automatic structures of bounded degree. In *LPAR'03*, Lecture Notes in Comp. Science vol. 2850, pages 344–358. Springer, 2003.
24. G. Oliver and R. Thomas. Automatic presentations for finitely generated groups. In *STACS'05*, Lecture Notes in Computer Science vol. 3404, pages 693–704. Springer, 2005.
25. S. Rubin. Automata presenting structures: A survey of the finite string case. *Bulletin of Symbolic Logic*, 14:169–209, 2008.
26. J. Sakarovitch. Easy multiplications. I. The realm of Kleene's Theorem. *Information and Computation*, 74:173–197, 1987.