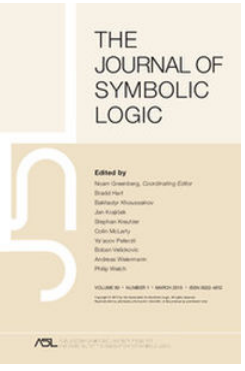

Automatic structures of bounded degree revisited

Dietrich Kuske and Markus Lohrey

The Journal of Symbolic Logic / Volume 76 / Issue 04 / December 2011, pp 1352 - 1380
DOI: 10.2178/jsl/1318338854, Published online: 12 March 2014

Link to this article: http://journals.cambridge.org/abstract_S0022481200001420

How to cite this article:
Dietrich Kuske and Markus Lohrey (2011). Automatic structures of bounded degree revisited . The Journal of Symbolic Logic, 76, pp 1352-1380 doi:10.2178/jsl/1318338854

THE JOURNAL OF SYMBOLIC LOGIC
Volume 76, Number 4, Dec. 2011

AUTOMATIC STRUCTURES OF BOUNDED DEGREE REVISITED

DIETRICH KUSKE AND MARKUS LOHREY

Abstract. The first-order theory of a string automatic structure is known to be decidable, but there are examples of string automatic structures with nonelementary first-order theories. We prove that the first-order theory of a string automatic structure of bounded degree is decidable in doubly exponential space (for injective automatic presentations, this holds even uniformly). This result is shown to be optimal since we also present a string automatic structure of bounded degree whose first-order theory is hard for 2EXPSPACE. We prove similar results also for tree automatic structures. These findings close the gaps left open in [28] by improving both the lower and the upper bounds.

§1. Introduction. The idea of an automatic structure goes back to Büchi and Elgot who used finite automata to decide, e.g., Presburger arithmetic [14]. Automaton decidable theories [18] and automatic groups [15] are similar concepts. A systematic study was initiated by Khoussainov and Nerode [20] who also coined the name "*automatic structure*" (we prefer the term "*string automatic structure*" in this paper). In essence, a structure is string 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 state automaton with several heads that proceed synchronously. String automatic structures received increasing interest over the last years [1, 3, 4, 5, 7, 17, 19, 21, 22, 24, 27, 31]. One of the main motivations for investigating string automatic structures is that their first-order theories can be decided uniformly (i.e., the input is a string automatic presentation and a first-order sentence). But even the non-uniform first-order theory is far from efficient since there exist string automatic structures with a nonelementary first-order theory. This motivates the search for subclasses of string automatic structures whose first-order theories are elementary. The first such class was identified by the second author in [28] who showed that the first-order theory of every string automatic structure of *bounded degree* can be decided in triply exponential alternating time with linearly many alternations. A structure has bounded degree, if in its Gaifman graph, the number of neighbors of a node is bounded by some fixed constant. The paper [28] also presents a specific example of a string automatic structure of bounded degree, where the first-order theory is hard for doubly exponential alternating time with linearly many alternations. Hence, an exponential gap between the upper and lower

Received May 31, 2010.
The second author acknowledges support from the DFG-project GELO. These results were obtained when Dietrich Kuske was affiliated with the Institut für Informatik, Universität Leipzig.

0022-4812/11/7604-0014/$3.90

bound remained. An upper bound of 4-fold exponential alternating time with linearly many alternations was shown for *tree automatic structures* (which are defined analogously to automatic structures using tree automata) of bounded degree. Our paper [26] proves a triply exponential space bound for the first-order theory of an injective ω-string automatic structure (that is defined via Büchi-automata) of bounded degree. Here, injectivity means that every element of the structure is represented by a *unique* ω-string from the underlying regular language. By [17], the class of injective ω-string automatic structures is a strict subclass of the class of all ω-string automatic structures, whereas for string and tree automatic structures injectivity is not a restriction [9, 20, 35].

In this paper, we achieve three goals:

- We close the complexity gaps from [28] for string/tree automatic structures of bounded degree.
- We investigate, for the first time, the complexity of the *uniform* first-order theory (where the automatic presentation is part of the input) of string/tree automatic structures of bounded degree.
- We refine our complexity analysis using the growth function of a structure. This function measures the size of a sphere in the Gaifman graph depending on the radius of the sphere. The growth function of a structure of bounded degree can be at most exponential.

Our main results are the following:

- The uniform first-order theory for injective string automatic presentations of bounded degree is 2EXPSPACE-complete. The lower bound already holds in the non-uniform setting, i.e., there exists a string automatic structure of bounded degree with a 2EXPSPACE-complete first-order theory.
- For every string automatic structure of bounded degree, where the growth function is polynomially bounded, the first-order theory is in EXPSPACE, and there exists an example with an EXPSPACE-complete first-order theory.
- The uniform first-order theory for injective tree automatic presentations of bounded degree belongs to 4EXPTIME. For every fixed tree automatic structure of bounded degree, the first-order theory belongs to 3EXPTIME, and to 2EXPTIME if the growth function is polynomial. Our bounds for the non-uniform problem are sharp, i.e., there are tree automatic structures of bounded degree (and polynomial growth) with a 3EXPTIME-complete (2EXPTIME-complete, resp.) first-order theory.

For the uniform first-order theory for injective tree automatic presentations of bounded degree, the precise complexity remains open (it is in 4EXPTIME and 3EXPTIME-hard). If the input presentations are not necessarily injective, the upper bounds for the uniform theories are one exponent higher: 3EXPSPACE in the string case and 5EXPTIME in the tree case. We conclude this paper with some results on the complexity of first-order fragments with fixed quantifier alternation depth one or two on string/tree automatic structures of bounded degree.

Further related work. In [12] the blow-up in formula size inherent in Gaifman's locality theorem has been investigated. That work reveals that in the worst case a non-elementary blow-up is unavoidable already on locally finite structures (in fact,

forests). In the same paper it is also remarked that on (not necessarily automatic) structures of bounded degree every first-order formula is equivalent to a Boolean combination of basic local sentences of at most 4-fold exponential total size. In some sense, our results refine this result for the case of automatic structures of bounded degree.

In [2], Bárány considers p-automatic structures, which are string automatic structures having an automatic presentation with a domain language of polynomial growth (i.e., the number of words of length at most n grows polynomially with n). For each of these structures, the first-order theory belongs to PSPACE. A p-automatic structure of bounded degree must have polynomial growth in the sense used in this paper. On the other hand, p-automatic structures are not required to be of bounded degree. Moreover, our example of an automatic structure of bounded degree with polynomial growth and an EXPSPACE-complete first-order theory is not p-automatic (since PSPACE $\subsetneq$ EXPSPACE by the space hierarchy theorem). Hence, the class of p-automatic structures and the class of bounded degree automatic structures of polynomial growth are incomparable.

§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 $|w|$. Let $\Gamma^n = \{w \in \Gamma^* \mid n = |w|\}$.

Let us define $\exp(0, x) = x$ and $\exp(n+1, x) = 2^{\exp(n,x)}$ for $x \in \mathbb{N}$. We assume that the reader has some basic knowledge in complexity theory, see e.g., [30]. By Savitch's theorem, $\mathsf{NSPACE}(s(n)) \subseteq \mathsf{DSPACE}(s(n)^2)$ if $s(n) \geq \log(n)$. Hence, we can just write $\mathsf{SPACE}(s(n)^{O(1)})$ for either $\mathsf{NSPACE}(s(n)^{O(1)})$ or $\mathsf{DSPACE}(s(n)^{O(1)})$. For $k \geq 1$, we denote with kEXPSPACE (resp. kEXPTIME) the class of all problems that can be accepted in space (resp. time) $\exp(k, n^{O(1)})$ on a deterministic Turing machine. For 1EXPSPACE we write just EXPSPACE and 0EXPSPACE stands for PSPACE. A computational problem is called *elementary* if it belongs to kEXPTIME for some $k \in \mathbb{N}$.

2.1. Tree and string automata. For our purpose it suffices to consider only tree automata on binary trees. Let Γ be a finite alphabet. A *finite binary tree* over Γ is a mapping $t\colon \mathsf{dom}(t) \to \Gamma$, where $\mathsf{dom}(t) \subseteq \{0,1\}^*$ is finite, nonempty, and satisfies the following closure condition for all $w \in \{0,1\}^*$: if $\{w0, w1\} \cap \mathsf{dom}(t) \neq \emptyset$, then also $w, w0 \in \mathsf{dom}(t)$. With T_Γ we denote the set of all finite binary trees over Γ. A (top-down) *tree automaton over* Γ is a tuple $A = (Q, \Delta, q_0)$, where Q is the finite set of states, $q_0 \in Q$ is the initial state, and

$$\Delta \subseteq (Q \times \Gamma \times Q \times Q) \cup (Q \times \Gamma \times Q) \cup (Q \times \Gamma) \tag{1}$$

is the non-empty transition relation. A *successful run* of A on a tree t is a mapping $\rho\colon \mathsf{dom}(t) \to Q$ such that (i) $\rho(\varepsilon) = q_0$ and (ii) for every $w \in \mathsf{dom}(t)$ with children $w0, \ldots, wi$ (thus $-1 \leq i \leq 1$) we have $(\rho(w), t(w), \rho(w0), \ldots, \rho(wi)) \in \Delta$. With $L(A)$ we denote the set of all finite binary trees t such that there exists a successful run of A on t. A set $L \subseteq T_\Gamma$ is called *regular* if there exists a finite tree automaton A with $L = L(A)$.

A tree t with $\mathsf{dom}(t) \subseteq 0^*$ and $n = |\mathsf{dom}(t)|$ can be identified with the nonempty string $t(\varepsilon)t(0)t(00)\ldots t(0^{n-1})$. In the same spirit, a finite *string* automaton can be defined as a tree automaton, where the transition relation Δ in (1) satisfies $\Delta \subseteq (Q \times \Gamma \times Q) \cup (Q \times \Gamma)$.

We will need the following well known facts on string/tree automata: Emptiness (resp. inclusion) of the languages of string automata can be decided in nondeterministic logarithmic space (resp. polynomial space), whereas emptiness (resp. inclusion) of the languages of tree automata can be decided in polynomial time (resp. exponential time), see e.g., [10]. In all four cases completeness holds.

2.2. Structures and first-order logic. A *signature* is a finite set $\mathcal{S}$ of relational symbols, where every symbol $r \in \mathcal{S}$ has some fixed arity m_r. The notion of an $\mathcal{S}$-structure (or model) is defined as usual in logic. 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 an $\mathcal{S}$-structure $\mathscr{A} = (A, (r^{\mathscr{A}})_{r\in\mathcal{S}})$, where $r^{\mathscr{A}} \subseteq A^{m_r}$. To simplify notation, we will write $a \in \mathscr{A}$ for $a \in A$. For $B \subseteq A$ we define the restriction $\mathscr{A} \restriction B = (B, (r^{\mathscr{A}} \cap B^{m_r})_{r\in\mathcal{S}})$. Given further constants $a_1, \ldots, a_n \in \mathscr{A}$, we write $(\mathscr{A}, a_1, \ldots, a_k)$ for the structure $(A, (r^{\mathscr{A}})_{r\in\mathcal{S}}, a_1, \ldots, a_k)$. In the rest of the paper, we will always identify a symbol $r \in \mathcal{S}$ with its interpretation $r^{\mathscr{A}}$.

A *congruence* on the structure $\mathscr{A} = (A, (r)_{r\in\mathcal{S}})$ is an equivalence relation $\equiv$ on A such that for every $r \in \mathcal{S}$ 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* $\mathscr{A}/_\equiv = (A/_\equiv, (r/_\equiv)_{r\in\mathcal{S}})$, where $r/_\equiv = \{([a_1], \ldots, [a_{m_r}]) \mid (a_1, \ldots, a_{m_r}) \in r\}$.

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

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

Thus, the set of nodes is the universe of $\mathscr{A}$ 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 $\mathscr{A}$. With $d_{\mathscr{A}}(a,b)$, where $a, b \in \mathscr{A}$, we denote the distance between a and b in $G(\mathscr{A})$, i.e., it is the length of a shortest path connecting a and b in $G(\mathscr{A})$. For $a \in \mathscr{A}$ and $d \geq 0$ we denote with $S_{\mathscr{A}}(d,a) = \{b \in A \mid d_{\mathscr{A}}(a,b) \leq d\}$ the d-sphere around a. If $\mathscr{A}$ is clear from the context, then we will omit the subscript $\mathscr{A}$. We say that the structure $\mathscr{A}$ is *locally finite* if its Gaifman graph $G(\mathscr{A})$ is locally finite (i.e., every node has finitely many neighbors). Similarly, the structure $\mathscr{A}$ has *bounded degree*, if $G(\mathscr{A})$ 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 $G(\mathscr{A})$; the minimal such δ is called the *degree of* $\mathscr{A}$. For a structure $\mathscr{A}$ of bounded degree we can define its *growth function* as the mapping $g_{\mathscr{A}}\colon \mathbb{N} \to \mathbb{N}$ with $g_{\mathscr{A}}(n) = \max\{|S_{\mathscr{A}}(n,a)| \mid a \in \mathscr{A}\}$. Note that if the function $g_{\mathscr{A}}$ is not bounded then $g_{\mathscr{A}}(n) \geq n$ for all $n \geq 1$. For us, it is more convenient to not have a bounded function describing the growth. Therefore, we define the *normalized growth function* $g'_{\mathscr{A}}$ by $g'_{\mathscr{A}}(n) = \max\{n, g_{\mathscr{A}}(n)\}$. Note that $g_{\mathscr{A}}$ and $g'_{\mathscr{A}}$ are different only in the pathological case that all connected components of $\mathscr{A}$ contain at most m elements (for some fixed m). Clearly, $g'_{\mathscr{A}}(n)$ can grow at most exponentially (since $\mathscr{A}$ is assumed to have bounded degree). We say that $\mathscr{A}$ has *exponential growth* if $g'_{\mathscr{A}}(n) \in 2^{\Omega(n)}$; if $g'_{\mathscr{A}}(n) \in n^{O(1)}$, then $\mathscr{A}$ has *polynomial growth*.

To define logical formulas, we fix a countably infinite set V of variables, which evaluate to elements of structures. *Formulas over the signature* $\mathcal{S}$ (or *formulas* if the

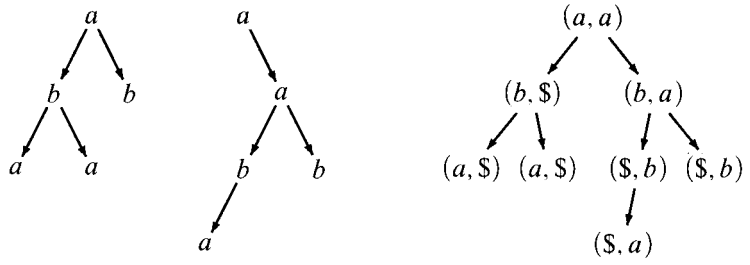

FIGURE 1. The convolution of two trees.

signature is clear from the context) are constructed from the atomic formulas $x = y$ and $r(x_1, \ldots, x_{m_r})$, where $r \in \mathcal{S}$ and $x, y, x_1, \ldots, x_{m_r} \in V$, using the Boolean connectives $\vee$ and $\neg$ and existential quantification over variables from V. The Boolean connective $\wedge$ and universal quantification can be derived from these operators in the usual way. The *quantifier depth* of a formula φ is the maximal nesting depth of quantifiers in φ. The notion of a free variable is defined as usual. A formula without free variables is called *closed*. If $\varphi(x_1, \ldots, x_m)$ is a formula with free variables among $x_1, \ldots, x_m$ and $a_1, \ldots, a_m \in \mathscr{A}$, then $\mathscr{A} \models \varphi(a_1, \ldots, a_m)$ means that φ evaluates to true in $\mathscr{A}$ when the free variable x_i evaluates to a_i. The *first-order theory* of $\mathscr{A}$, denoted by $\mathsf{FOTh}(\mathscr{A})$, is the set of all closed formulas φ such that $\mathscr{A} \models \varphi$. For $n \geq 0$, Σ_n-formulas and Π_n-formulas are inductively defined as follows:

- A quantifier-free first-order formula is a Σ_0-formula as well as a Π_0-formula.
- If $\varphi(x_1, \ldots, x_n, \overline{y})$ is a Σ_n-formula, then $\forall x_1 \ldots \forall x_n \colon \varphi(x_1, \ldots, x_n, \overline{y})$ is a Π_{n+1}-formula.
- If $\varphi(x_1, \ldots, x_n, \overline{y})$ is a Π_n-formula, then $\exists x_1 \ldots \exists x_n \colon \varphi(x_1, \ldots, x_n, \overline{y})$ is a Σ_{n+1}-formula.

The Σ_n-theory Σ_n-$\mathsf{FOTh}(\mathscr{A})$ of a structure $\mathscr{A}$ is the set of all Σ_n-formulas in $\mathsf{FOTh}(\mathscr{A})$; the Π_n-theory is defined analogously.

2.3. Structures from automata. This section recalls string automatic and tree automatic structures and basic results about them. Details can be found in the surveys [31, 3].

2.3.1. *Tree and string automatic structures.* String automatic structures were introduced in [18], their systematic study was later initiated by [20]. Tree automatic structures were introduced in [6], they generalize string automatic structures. Here, we will first introduce tree automatic structures. String automatic structures can be considered as a special case of tree automatic structures.

Let Γ be a finite alphabet and let $\$ \notin \Gamma$ be an additional padding symbol. Let $t_1, \ldots, t_m \in T_\Gamma$. We define the *convolution* $t = t_1 \otimes \cdots \otimes t_m$, which is a finite binary tree over the alphabet $(\Gamma \cup \{\$\})^m$, as follows: $\mathsf{dom}(t) = \bigcup_{i=1}^m \mathsf{dom}(t_i)$ and for all $w \in \bigcup_{i=1}^m \mathsf{dom}(t_i)$ we define $t(w) = (a_1, \ldots, a_m)$, where $a_i = t_i(w)$ if $w \in \mathsf{dom}(t_i)$ and $a_i = \$$ otherwise. In Fig. 1, the third tree is the convolution of the first two trees.

An *m-dimensional* (*synchronous*) *tree automaton* over Γ is just a tree automaton A over the alphabet $(\Gamma \cup \{\$\})^m$ such that $L(A) \subseteq \{t_1 \otimes \cdots \otimes t_n \mid t_1, \ldots, t_m \in T_\Gamma\}$.

Such an automaton defines an m-ary relation

$$R(A) = \{(t_1, \ldots, t_m) \mid t_1 \otimes \cdots \otimes t_m \in L(A)\}.$$

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

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

This presentation P 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 $(\Gamma, A_0, (A_r)_{r\in\mathcal{S}})$. The structure presented by P is the quotient

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

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

A *string automatic presentation* is a tree automatic presentation, where all tree automata are in fact string automata (as explained in Section 2.1), and a structure $\mathcal{A}$ is called *string automatic* if there exists a string automatic presentation P such that $\mathcal{A} \simeq \mathcal{A}(P)$. Typical examples of string automatic structures are $(\mathbb{N}, +)$ (Presburger's arithmetic), $(\mathbb{Q}, \leq)$, and all ordinals below ω^ω [13, 20]. An example of a tree automatic structure, which is not string automatic is $(\mathbb{N}, \cdot)$ (the natural numbers with multiplication) [6], or the ordinal ω^ω [13]. Examples of string automatic structures of bounded degree are transition graphs of Turing machines and Cayley-graphs of automatic groups [15] (or even right-cancellative monoids [33]).

REMARK 2.1. Usually a *tree automatic presentation* for an $\mathcal{S}$-structure $\mathcal{A} = (A, (r)_{r\in\mathcal{S}})$ is defined as a tuple (Γ, L, h) such that

- Γ is a finite alphabet,
- $L \subseteq T_\Gamma$ is a regular set of trees,
- $h\colon L \to A$ is a surjective function,
- the relation $\{(u, v) \in L \times L \mid h(u) = h(v)\}$ can be recognized by a 2-dimensional tree automaton, and
- for all $r \in \mathcal{S}$, the relation $\{(u_1, \ldots, u_{m_r}) \in L^{m_r} \mid (h(u_1), \ldots, h(u_{m_r})) \in r\}$ can be recognized by an m_r-dimensional tree automaton.

Since for our considerations, tree automatic presentations are part of the input for algorithms, we prefer our definition, where a tree automatic presentation is a finite object (a tuple of finite tree automata), whereas in the standard definition, the presentation also contains the presentation map h.

Let SA be the class of all string automatic presentations and let TA be the class of all tree automatic presentations. Moreover, for $\mathsf{X} \in \{\mathsf{SA}, \mathsf{TA}\}$ let

$$\mathsf{Xb} = \{P \in \mathsf{X} \mid \mathscr{A}(P) \text{ has bounded degree}\},$$
$$\mathsf{iX} = \{P \in \mathsf{X} \mid P \text{ is injective}\},$$
$$\mathsf{iXb} = \mathsf{Xb} \cap \mathsf{iX}.$$

2.3.2. *The model checking problem.* For the above classes of tree automatic presentations, we will be interested in the following decision problems.

DEFINITION 2.2. Let C be a class of tree automatic presentations.

- The *first-order model checking problem* FOMC(C) *for* C denotes the set of all pairs (P, φ), where $P \in \mathsf{C}$ and $\varphi \in \mathsf{FOTh}(\mathscr{A}(P))$.
- For $n \geq 1$, the Σ_n*-model checking problem* Σ_n-FOMC(C) *for* C denotes the set of all pairs (P, φ), where $P \in \mathsf{C}$ and $\varphi \in \Sigma_n\text{-}\mathsf{FOTh}(\mathscr{A}(P))$.

If $\mathsf{C} = \{P\}$ is a singleton, then the model checking problem FOMC(C) for C can be identified with the first-order theory of the structure $\mathscr{A}(P)$. An algorithm deciding the model checking problem for a nontrivial class C decides the first-order theories of each element of C uniformly.

The following two results are the main motivations for investigating tree automatic structures.

PROPOSITION 2.3. [6, 20] *There is an algorithm that computes from a tree automatic presentation* $P = (\Gamma, A_0, A_=, (A_r)_{r \in \mathcal{S}})$ *and a formula* $\varphi(x_1, \ldots, x_m)$ *an* m*-dimensional tree automaton* A *over* Γ *with* $R(A) = \{(u_1, \ldots, u_m) \in L(A_0)^m \mid \mathscr{A}(P) \models \varphi([u_1], \ldots, [u_m])\}$.

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. The following result is a direct consequence.

THEOREM 2.4. [6, 20] *The model checking problem* FOMC(TA) *for all tree automatic presentations is decidable. In particular, for every tree automatic structure* $\mathscr{A}$ *the first-order theory* $\mathsf{FOTh}(\mathscr{A})$ *is decidable.*

REMARK 2.5. Strictly speaking, [6, 20] devise algorithms that, given a tree automatic presentation and a closed formula, decide whether the formula holds in the presented structure. But a priori, it is not clear whether it is decidable, whether a given tuple $(\Gamma, A_0, A_=, (A_r)_{r \in \mathcal{S}})$ is a tree automatic presentation. Lemma 2.12 below shows that TA is indeed decidable, which then completes the proof of Theorem 2.4.

Theorem 2.4 holds even if we add quantifiers for "there are infinitely many x such that $\varphi(x)$" [6, 7] and "the number of elements satisfying $\varphi(x)$ is divisible by k" (for $k \in \mathbb{N}$) [23].[1] This implies in particular that it is decidable whether a tree automatic presentation describes a locally finite structure. But the decidability of the first-order theory is far from efficient, since there are even string automatic structures with a nonelementary first-order theory [7]. For instance the structure $(\{0,1\}^*, s_0, s_1, \preceq)$,

[1] [23] only provides the proofs for string automatic structures. These proofs are easily extended to tree automatic structures once the presentation is injective. But every tree automatic presentation can be transformed into an equivalent injective one [9, 35].

where $s_i = \{(w, wi) \mid w \in \{0,1\}^*\}$ for $i \in \{0,1\}$ and $\preceq$ is the prefix order on finite words, has a nonelementary first-order theory, see e.g., [11, Example 8.3]. In fact, even locally finite examples exist:

PROPOSITION 2.6. *There exists a locally finite string automatic structure with a nonelementary first-order theory.*

PROOF. The theory of all finite binary labeled linear orders is nonelementary [29]. Since this theory can be reduced to the first-order theory of the structure consisting of the disjoint union of all finite binary labeled linear orders, the latter structure has a nonelementary first-order theory too. But this structure is automatic: The universe is the set $L = \{u \otimes v \mid u \in \{0,1\}^+, v \in 0^*, |v| < |u|\}$. In addition, we have a partial order $\{(u \otimes v, u \otimes v') \in L \times L \mid |v| \leq |v'|\}$ that encodes the union of all the linear order relations, and a unary relation $\{u \otimes v \in L \mid \text{position } |v| \text{ in } u \text{ carries } 1\}$ that encodes the labeling. ⊣

The following two results refine Theorem 2.4:

THEOREM 2.7. [25] *The following holds for all $n \geq 0$:*

(1) *The Σ_{n+1}-model checking problem Σ_{n+1}-FOMC(SA) for all string automatic presentations is in nEXPSPACE.*
(2) *There is a fixed string automatic structure with an nEXPSPACE-complete Σ_{n+1}-theory.*
(3) *There is a closed formula $\varphi_n \in \Sigma_{n+1}$ for which $\{P \in \mathsf{SA} \mid \mathscr{A}(P) \models \varphi_n\}$ is nEXPSPACE-complete.*

REMARK 2.8. For $n = 0$ and $n = 1$, the above statement (2) can be found in [3]. Regarding (1) and (3), an exponentially better bound holds for automatic presentations that consist of deterministic automata, only: for $n \leq 1$, this can be found in [3], the general case can be shown using the methods from [25].

THEOREM 2.9. *For all $n \geq 1$, the Σ_n-model checking problem Σ_n-FOMC(TA) for all tree automatic presentations is in nEXPTIME.*

In [25] only Theorem 2.7 is shown. But the proof for Theorem 2.9 is almost the same as for the first statement of Theorem 2.7. The only difference comes from the fact that emptiness for string automata is NL-complete, whereas emptiness for tree automata is P-complete.

2.3.3. *First complexity results: the classes* TA *etc and boundedness.* This paper is concerned with the uniform and non-uniform complexity of the first-order theory of (some subclass of) tree automatic structures of bounded degree. Thus, we will consider algorithms that take as input tree automatic presentations (together with closed formulas). For complexity considerations, we have to define the size $|P|$ of a tree automatic presentation $P = (\Gamma, A_0, A_=, (A_r)_{r \in \mathcal{S}})$. First, let us define the size $|A|$ of an m-dimensional tree automaton $A = (Q, \Delta, q_0)$ over Γ. A transition tuple from Δ (see (1)) can be stored with at most $3\log(|Q|) + m\log(|\Gamma|)$ many bits. Hence, up to constant factors, Δ can be stored in space $|\Delta| \cdot (\log(|Q|) + m\log(|\Gamma|))$. We can assume that every state is the first component of some transition tuple, i.e., $|Q| \leq |\Delta|$. Furthermore, the size of the basic alphabet Γ can be bounded by $|\Delta|$ as well, but the dimension m is independent from the size of Δ. Since our complexity measures will be up to polynomial time reductions, it therefore makes sense to define the size of the tree automaton A to be $|A| = |\Delta| \cdot m$. We assume Δ to be

nonempty, hence $|A| \geq 1$. The size of the presentation $P = (\Gamma, A_0, A_=, (A_r)_{r \in \mathcal{S}})$ is $|P| = |A_0| + |A_=| + \sum_{r \in \mathcal{S}} |A_r|$. Note that $|\mathcal{S}| \leq |P|$ and $m \leq |P|$, when m is the maximal arity in $\mathcal{S}$.

It will be convenient to work with injective string (resp. tree) automatic presentations. The following lemma says that this is no restriction, at least if we do not consider complexity aspects.

LEMMA 2.10 ([20, Corollary 4.3] and [35]). *From a given $P \in$ TA (resp. $P \in$ SA) one can compute in time $2^{O(|P|)}$ a presentation $P' \in$ iTA (resp. $P' \in$ iSA) with $\mathscr{A}(P) \simeq \mathscr{A}(P')$.*

REMARK 2.11. For string automatic presentations, the statement of Lemma 2.10 was shown in [20]. Although the exponential time bound on the construction of $P' \in$ iSA is not stated explicitly in [20], it can be easily extracted from the construction. In [9, Corollary 4.2], it is stated that for every $P \in$ TA there exists $P' \in$ iTA with $\mathscr{A}(P) \simeq \mathscr{A}(P')$. Although the construction of P' is effective, the complexity is difficult to extract from [9]. An exponential construction of $P' \in$ iTA was presented in [35].

The following lemma shows that the classes of all tree and string automatic presentations are decidable and gives complexity bounds. While these two results are not surprising, it is not clear how to determine whether $\mathscr{A}(P)$ has bounded degree – this will be solved by Proposition 2.14 below.

LEMMA 2.12. *The class* TA *is* EXPTIME*-complete and the class* SA *is* PSPACE*-complete.*

PROOF. We start with a proof of the first statement. Suppose we are given a tuple of tree automata $(A_0, A_=, (A_r)_{r \in \mathcal{S}})$ over an alphabet Γ. In a first step, we check in polynomial time, whether $A_=$ is a 2-dimensional tree automaton over Γ and that every A_r ($r \in \mathcal{S}$) is an m_r-dimensional tree automaton over Γ according to Section 2.3.1. We proceed as follows for every $r \in \mathcal{S}$ (for $A_=$ the same algorithm works):

First, we check that no tree from $L(A_r)$ contains the label $(\$, \ldots, \$)$. To this aim, replace in all transitions of A_r the letters from $(\Gamma \cup \{\$\})^{m_r} \setminus \{(\$, \ldots, \$)\}$ by $\top$ and the letter $(\$, \ldots, \$)$ by $\bot$ and check whether the language of the resulting automaton is contained in $T_{\{\top\}}$ (the set of all $\top$-labeled binary trees). Since the set $T_{\{\top\}}$ can be accepted by a fixed automaton, this inclusion can be decided in polynomial time. Hence, we can assume that no tree from $L(A_r)$ contains the label $(\$, \ldots, \$)$. Next, let $H \subseteq T_{\{\top,\$\}}$ denote the set of those trees t whose $\top$-labeled nodes form an initial segment of t. Again this set can be accepted by a fixed automaton. For all $1 \leq i \leq m_r$ we construct (in polynomial time) an automaton $A_{r,i}$ as follows: First we project A_r onto its i^{th} component. Then, we replace in every transition of the resulting automaton all occurrences of symbols from Γ by $\top$. It remains to check that $L(A_{r,i}) \subseteq H$ for all $1 \leq i \leq m_r$, which can be done in polynomial time.

For the rest of the proof, let us assume that $A_=$ is a 2-dimensional tree automaton over Γ and that every A_r ($r \in \mathcal{S}$) is an m_r-dimensional tree automaton over Γ. This implies that the tuple $P = (\Gamma, B, A_0, A_=, (A_r)_{r \in \mathcal{S}})$, where $L(B) = T_\Gamma$, is an injective tree automatic presentation. Here, A_0 defines a unary relation on the domain T_Γ, and $A_=$ defines a binary relation. Then, $(\Gamma, A_0, A_=, (A_r)_{r \in \mathcal{S}})$ is a tree automatic

presentation if and only if the following closed first-order formulas are true in $\mathcal{S}(P')$ for all $r \in \mathcal{S}$:

$$\forall x, y\colon (x, y) \in R(A_=) \to x, y \in L(A_0),$$
$$\forall x \in L(A_0)\colon (x, x) \in R(A_=),$$
$$\forall x, y \in L(A_0)\colon (x, y) \in R(A_=) \to (y, x) \in R(A_=),$$
$$\forall x, y, z \in L(A_0)\colon ((x, y) \in R(A_=) \wedge (y, z) \in R(A_=)) \to (x, z) \in R(A_=),$$
$$\forall x_1, \ldots, x_{m_r}\colon (x_1, \ldots, x_{m_r}) \in R(A_r) \to x_1, \ldots, x_{m_r} \in L(A_0),$$
$$\forall (x_1, y_1), \ldots, (x_{m_r}, y_{m_r}) \in R(A_=)\colon \begin{pmatrix} (x_1, \ldots, x_{m_r}) \in R(A_r) \\ \to (y_1, \ldots, y_{m_r}) \in R(A_r) \end{pmatrix}.$$

These are Π_1-formulas. Hence, by Theorem 2.9, we can check in EXPTIME whether they hold in $\mathcal{S}(P')$.

Completeness follows since the inclusion $L(A) \subseteq L(B)$ is EXPTIME-complete for tree automata A and B.

This finishes the proof of the first statement. To prove the second, one can proceed analogously using Theorem 2.7. ⊣

Recall that $G(\mathscr{A})$ denotes the Gaifman graph of a structure $\mathscr{A}$. The following lemma says that the Gaifman graph of a string (resp. tree) automatic structure is effectively string (resp. tree) automatic. This is an immediate consequence of Proposition 2.3, so the novelty lies in the estimation of the complexity.

LEMMA 2.13. *From a given tree (string) automatic presentation*

$$P = (\Gamma, A_0, A_=, (A_r)_{r \in \mathcal{S}})$$

one can construct a 2-dimensional tree (string) automaton A such that

$$R(A) = \{(u, v) \in L(A_0) \times L(A_0) \mid ([u], [v]) \text{ is an edge in } G(\mathscr{A}(P))\}. \quad (2)$$

If m is the maximal arity in $\mathcal{S}$, then A has $m^2 \cdot |P|^2$ many states and can be computed in time $O(m^2 \cdot |P|^2) \leq |P|^{O(1)}$.

PROOF. We only give the proof for string automatic presentations, the tree automatic case can be shown verbatim. Let E be the edge relation of the Gaifman graph $G(\mathscr{A}(P))$. Note that for all $u, v \in L(A_0)$ we have $([u], [v]) \in E$ if and only if for some $r \in \mathcal{S}$ of arity $m_r \leq m$ and $1 \leq i, j \leq m_r$, there exist $u_1, \ldots, u_{m_r} \in L(A_0)$ with $(u_1, \ldots, u_{m_r}) \in R(A_r)$, $u = u_i$, and $v = u_j$. Let $r \in \mathcal{S}$ and $1 \leq i, j \leq m_r$. Projecting the automaton A_r onto the tracks i and j, one obtains a 2-dimensional automaton accepting all pairs $(u, v) \in \Gamma^* \times \Gamma^*$ such that there exists $(u_1, \ldots, u_{m_r}) \in R(A_r)$ with $u = u_i$ and $v = u_j$. Then the disjoint union of all these automata (for $r \in \mathcal{S}$ and $1 \leq i, j \leq m_r$) satisfies (2). Since $|\mathcal{S}| \leq |P|$, the construction can be performed in time $O(m^2 \cdot |P|^2)$. ⊣

Lemma 2.13 allows to show that also the bounded class TAb is decidable in exponential time:

PROPOSITION 2.14. *The class* TAb (*and hence also* SAb) *belongs to* EXPTIME.

PROOF. Let $P \in$ TA (which is decidable by Lemma 2.12 in exponential time). By Lemma 2.10, we can construct in exponential time an injective presentation $P' \in$ iTA with $\mathscr{A}(P) \cong \mathscr{A}(P')$. Hence, $|P'|$ is exponentially bounded in $|P|$. By Lemma 2.13 we can compute an automaton A with (2), i.e., A defines the edge

relation of the Gaifman-graph of $\mathscr{A}(P)$. The size of A is again exponentially bounded in the size of P. Since P' is injective (i.e., every equivalence class $[u]$ is the singleton $\{u\}$), $\mathscr{A}(P)$ is of bounded degree if and only if A (seen as a transducer) is finite-valued. But this is decidable in polynomial time [32, 34] in the size of A and hence in exponential time in the size of P. ⊣

Finally, since we deal with structures of bounded degree, it will be important to estimate the degree of such a structure given its presentation. Such estimates are provided by the following result.

PROPOSITION 2.15. *The following hold*:

(a) *If* $P \in$ iSAb, *then the degree of* $\mathscr{A}(P)$ *is bounded by* $\exp(1, |P|^{O(1)})$.
(b) *If* $P \in$ iTAb, *then the degree of* $\mathscr{A}(P)$ *is bounded by* $\exp(2, |P|^{O(1)})$.
(c) *If* $P \in$ SAb, *then the degree of* $\mathscr{A}(P)$ *is bounded by* $\exp(2, |P|^{O(1)})$.
(d) *If* $P \in$ TAb, *then the degree of* $\mathscr{A}(P)$ *is bounded by* $\exp(3, |P|^{O(1)})$.

PROOF. For statement (a) let $P \in$ iSAb. From Lemma 2.13, we can construct a string automaton A of size $|P|^{O(1)}$ that accepts the edge relation of the Gaifman graph of $\mathscr{A}(P)$. Then the degree of $\mathscr{A}(P)$ equals the maximal outdegree of the relation $R(A)$. For string transducers, this number is exponential in the size of A, i.e., it is in $\exp(1, |P|^{O(1)})$ [34].

For (b) we can use a similar argument. But since the maximal outdegree of the relation recognized by a tree transducer A is doubly exponential in the size of A [32], we obtain the bound $\exp(2, |P|^{O(1)})$ for the degree of $\mathscr{A}(P)$.

Finally statement (c) (resp. (d)) follows immediately from Lemma 2.10 and (a) (resp. (b)). ⊣

REMARK 2.16. All bounds in Proposition 2.15 are sharp:

(a) Let $\mathscr{A}_n$ be the complete graph on $\{a, b\}^n$; it has degree $2^{\Omega(n)}$. Moreover, $\mathscr{A}_n$ has an injective string automatic presentation of size $O(n)$.
(b) Let $\mathscr{A}_n$ be the complete graph on the set of all trees from $T_{\{a,b\}}$ that have height n. This graph has degree $\exp(2, \Omega(n))$ and it has an injective tree automatic presentation of size $O(n)$.
(c) In [35], it was shown that for every n there exists a finite (non-deterministic) string automaton A_n (over the alphabet $\{a, b\}$) of size $n^{O(1)}$ such that the complement $\{a, b\}^* \setminus L(A_n)$ is finite and has size $\exp(2, \Omega(n))$. Let us now consider the (non-injective) string automatic presentation $(A_0, A_=, A_E)$ over the signature $\{E\}$, where $L(A_0) = \{a, b\}^*$, $R(A_E) = \{a, b\}^* \times \{a, b\}^*$, and $R(A_=) = L(A_n) \times L(A_n)$. This presentation has size $n^{O(1)}$ and the structure it defines is a complete graph of size $\exp(2, \Omega(n))$ and therefore has degree $\exp(2, \Omega(n))$.
(d) The analogous result for tree automatic structures follows from [35] as well: in the previous paragraph, replace "string" by "tree", "$\{a, b\}^*$" by "$T_{\{a,b\}}$", and "$\exp(2, \Omega(n))$" by "$\exp(3, \Omega(n))$".

REMARK 2.17. The additional exponents in (b), (c), and (d), are the reason for the remaining complexity gaps for FOMC(iTAb), FOMC(SAb), and FOMC(TAb). For instance, the double exponential bound in (c) will result in a 3EXPSPACE bound for FOMC(SAb), whereas we only can prove a 2EXPSPACE lower bound (which already holds for the non-uniform theory).

Note that the example presentations in point (c) and (d) from Remark 2.16 contain non-deterministic finite automata. It is not clear, whether these examples can be adapted so that the presentations are deterministic. On the other hand, if the degree bounds in (c) and (d) from Proposition 2.15 can be improved for deterministic presentations, then this would give better upper bounds for FOMC(SAb) and FOMC(TAb), when restricted to deterministic automata.

REMARK 2.18. In the proofs of Lemma 2.14 and 2.15 we used the main results from [32, 34]. These results are proved for general (asynchronous) transducers, and are quite difficult to obtain. Here, we need these results only for synchronous transducers, and for these one can provide simpler proofs. On the other hand, using the general results from [32, 34] has no drawback for our upper bounds.

§3. Upper bounds. It is the aim of this section to give an algorithm that decides the theory of a string/tree automatic structure of bounded degree. The algorithm from Theorem 2.4 (that in particular solves this problem) is based on Proposition 2.3, i.e., the inductive construction of an automaton accepting all satisfying assignments. Differently, we base our algorithm on Gaifman's Theorem 3.1, i.e., on the combinatorics of spheres. We therefore start with some model theory.

3.1. Model-theoretic background. The following locality principle of Gaifman implies that super-exponential distances cannot be handled in first-order logic:

THEOREM 3.1. [16] *Let $\mathscr{A}$ be a structure, $(a_1, \ldots, a_k), (b_1, \ldots, b_k) \in \mathscr{A}^k$, $d \geq 0$, and $D_1, \ldots, D_k \geq 2^d$ such that*

$$(\mathscr{A} \restriction (\bigcup_{i=1}^{k} S(D_i, a_i)),\ a_1, \ldots, a_k) \simeq (\mathscr{A} \restriction (\bigcup_{i=1}^{k} S(D_i, b_i)),\ b_1, \ldots, b_k). \tag{3}$$

Then, for every formula $\varphi(x_1, \ldots, x_k)$ of quantifier depth at most d, we have:

$$\mathscr{A} \models \varphi(a_1, \ldots, a_k) \iff \mathscr{A} \models \varphi(b_1, \ldots, b_k).$$

Note that (3) says that there is an isomorphism between the two induced substructures $\mathscr{A} \restriction (\bigcup_{i=1}^{k} S(D_i, a_i))$ and $\mathscr{A} \restriction (\bigcup_{i=1}^{k} S(D_i, b_i))$ that maps a_i to b_i for all $1 \leq i \leq k$.

Let $\mathcal{S}$ be a signature and let $k, d \in \mathbb{N}$ with $0 \leq k \leq d$. A (d, k)-*sphere* is a tuple $(\mathscr{B}, b_1, \ldots, b_k)$ such that the following holds:

- $\mathscr{B}$ is an $\mathcal{S}$-structure with $b_1, \ldots, b_k \in \mathscr{B}$.
- For all $b \in \mathscr{B}$ there exists $1 \leq i \leq k$ such that $d_{\mathscr{B}}(b_i, b) \leq 2^{d-i}$.

There is only one $(d, 0)$-sphere namely the empty sphere $\emptyset$. For our later applications, $\mathscr{B}$ will be always a finite structure, but in this subsection finiteness is not needed. The parameters $b_1, \ldots, b_k$ will be the values for quantified variables $y_1, \ldots, y_k$, where y_1 is the variable from the outermost quantifier. This explains the shrinking radiuses $2^{d-1}, \ldots, 2^{d-k}$ in the definition of a (d, k)-sphere. For each additional quantifier, the distance of two vertices that can be related with a formula doubles, see also Lemma 4.1 below.

The (d,k)-sphere $(\mathscr{B}, b_1, \ldots, b_k)$ is *realizable in the structure* $\mathscr{A}$ if there exist $a_1, \ldots, a_k \in \mathscr{A}$ such that

$$(\mathscr{A} \restriction (\bigcup_{i=1}^{k} S(2^{d-i}, a_i)), a_1, \ldots, a_k) \simeq (\mathscr{B}, b_1, \ldots, b_k).$$

Take a (d,k)-sphere $\sigma = (\mathscr{B}, b_1, \ldots, b_k)$ and a $(d, k+1)$-sphere $(k+1 \leq d)$ $\sigma' = (\mathscr{B}', b_1', \ldots, b_k', b_{k+1}')$. Then σ' *extends* σ (abbreviated $\sigma \preceq \sigma'$) if

$$(\mathscr{B}' \restriction (\bigcup_{i=1}^{k} S(2^{d-i}, b_i')), b_1', \ldots, b_k') \simeq (\mathscr{B}, b_1, \ldots, b_k).$$

The following definition is the basis for our decision procedure.

DEFINITION 3.2. Let $\mathscr{A}$ be an $\mathcal{S}$-structure, $\psi(y_1, \ldots, y_k)$ a formula of quantifier depth at most d, and let $\sigma = (\mathscr{B}, b_1, \ldots, b_k)$ be a $(d+k, k)$-sphere. The Boolean value $\psi_\sigma \in \{0, 1\}$ is defined inductively as follows:

- If $\psi(y_1, \ldots, y_k)$ is an atomic formula, then
$$\psi_\sigma = \begin{cases} 1 & \text{if } \mathscr{B} \models \psi(b_1, \ldots, b_k), \\ 0 & \text{if } \mathscr{B} \not\models \psi(b_1, \ldots, b_k). \end{cases} \tag{4}$$
- If $\psi = \neg\theta$, then $\psi_\sigma = 1 - \theta_\sigma$.
- If $\psi = \alpha \vee \beta$, then $\psi_\sigma = \max(\alpha_\sigma, \beta_\sigma)$.
- If $\psi(y_1, \ldots, y_k) = \exists y_{k+1} \theta(y_1, \ldots, y_k, y_{k+1})$ then
$$\psi_\sigma = \max\{\theta_{\sigma'} \mid \sigma' \text{ is a realizable } (d+k, k+1)\text{-sphere with } \sigma \preceq \sigma'\}. \tag{5}$$

The following result ensures for every closed formula ψ that $\psi_\emptyset = 1$ if and only if $\mathscr{A} \models \psi$. Hence the above definition can possibly be used to decide validity of the formula φ in the structure $\mathscr{A}$.

PROPOSITION 3.3. *Let $\mathcal{S}$ be a signature, $\mathscr{A}$ an $\mathcal{S}$-structure with $a_1, \ldots, a_k \in \mathscr{A}$, $\psi(y_1, \ldots, y_k)$ a formula of quantifier depth at most d, and $\sigma = (\mathscr{B}, b_1, \ldots, b_k)$ a $(d+k, k)$-sphere with*

$$(\mathscr{A} \restriction (\bigcup_{i=1}^{k} S(2^{d+k-i}, a_i)),\ a_1, \ldots, a_k) \simeq (\mathscr{B}, b_1, \ldots, b_k). \tag{6}$$

Then $\mathscr{A} \models \psi(a_1, \ldots, a_k) \iff \psi_\sigma = 1$.

PROOF. We prove the lemma by induction on the structure of the formula ψ. First assume that ψ is atomic, i.e., $d = 0$. Then we have:

$$\begin{aligned} \psi_\sigma = 1 &\overset{(4)}{\iff} \mathscr{B} \models \psi(b_1, \ldots, b_k) \\ &\overset{(6)}{\iff} \mathscr{A} \restriction (\bigcup_{i=1}^{k} S(2^{k-i}, a_i)) \models \psi(a_1, \ldots, a_k) \\ &\iff \mathscr{A} \models \psi(a_1, \ldots, a_k), \end{aligned}$$

where the last equivalence holds since ψ is atomic.

The cases $\psi = \neg\theta$ and $\psi = \alpha \vee \beta$ are straightforward and therefore omitted.

We finally consider the case $\psi(y_1, \ldots, y_k) = \exists y_{k+1} \theta(y_1, \ldots, y_k, y_{k+1})$.

First assume that $\psi_\sigma = 1$. By (5), there exists a realizable $(d+k, k+1)$-sphere σ' with $\sigma \preceq \sigma'$ and $\theta_{\sigma'} = 1$. Since σ' is realizable, there exist $a'_1, \ldots, a'_k, a'_{k+1} \in \mathscr{A}$ with

$$(\mathscr{A} \restriction (\bigcup_{i=1}^{k+1} S(2^{d+k-i}, a'_i)),\ a'_1, \ldots, a'_k, a'_{k+1}) \simeq (\mathscr{B}', b'_1, \ldots, b'_k, b'_{k+1}) = \sigma'. \tag{7}$$

By induction, we have $\mathscr{A} \models \theta(a'_1, \ldots, a'_k, a'_{k+1})$ and therefore $\mathscr{A} \models \psi(a'_1, \ldots, a'_k)$. From (6), (7), and $\sigma \preceq \sigma'$, we also obtain

$$(\mathscr{A} \restriction (\bigcup_{i=1}^{k} S(2^{d+k-i}, a'_i)),\ a'_1, \ldots, a'_k) \simeq (\mathscr{A} \restriction (\bigcup_{i=1}^{k} S(2^{d+k-i}, a_i)),\ a_1, \ldots, a_k)$$

and therefore by Gaifman's Theorem 3.1 $\mathscr{A} \models \psi(a_1, \ldots, a_k)$.

Conversely, let $a_{k+1} \in \mathscr{A}$ such that $\mathscr{A} \models \theta(a_1, \ldots, a_k, a_{k+1})$. Let $\sigma' = (\mathscr{B}', b'_1, \ldots, b'_k, b'_{k+1})$ be the unique (up to isomorphism) $(d+k, k+1)$-sphere such that

$$(\mathscr{A} \restriction (\bigcup_{i=1}^{k+1} S(2^{d+k-i}, a_i)),\ a_1, \ldots, a_k, a_{k+1}) \simeq (\mathscr{B}', b'_1, \ldots, b'_k, b'_{k+1}). \tag{8}$$

Then (6) implies $\sigma \preceq \sigma'$. Moreover, by (8), σ' is realizable in $\mathscr{A}$, and $\mathscr{A} \models \theta(a_1, \ldots, a_k, a_{k+1})$ implies by induction $\theta_{\sigma'} = 1$. Hence, by (5), we get $\psi_\sigma = 1$ which finishes the proof of the lemma. ⊣

3.2. The decision procedure. Now suppose we want to decide whether the closed formula φ holds in a tree automatic structure $\mathscr{A}$ of *bounded degree*. By Proposition 3.3 it suffices to compute the Boolean value $\varphi_\emptyset$. This computation will follow the inductive definition of φ_σ from Definition 3.2. Since every (d,k)-sphere that is realizable in $\mathscr{A}$ is finite, we only have to deal with finite spheres. The crucial part of our algorithm is to determine whether a finite (d,k)-sphere is realizable in $\mathscr{A}$. In the following, for a finite (d,k)-sphere $\sigma = (\mathscr{B}, b_1, \ldots, b_k)$, we denote with $|\sigma|$ the number of elements of $\mathscr{B}$ and with $\delta(\sigma)$ we denote the degree of the finite structure $\mathscr{B}$. We have to solve the following realizability problem:

DEFINITION 3.4. Let C be a class of tree automatic presentations. Then the *realizability problem* REAL(C) *for* C denotes the set of all pairs (P, σ) where $P \in \mathsf{C}$ and σ is a *finite* (d,k)-sphere over the signature of P for some $0 \leq k \leq d$ such that σ can be realized in $\mathscr{A}(P)$.

In this definition, we assume that the finite (d,k)-sphere σ is represented by enumerating all elements as well as all tuples for each relation symbol from the signature of P.

Note that the following lemma is not restricted to string/tree automatic presentations of *bounded degree*. Moreover, one could prove an analogous statement for non-injective presentations as well. We do not do so, because (i) the proof becomes more technical and (ii) a version for non-injective presentations would not improve our upper bounds for the decision problems FOMC(SAb) and FOMC(TAb), respectively. As already stated in Remark 2.17, the main bottlenecks in our algorithms for these problems are the (unavoidable) multiply exponential degree bounds in Proposition 2.15.

LEMMA 3.5. *The problems* REAL(iSA) *and* REAL(iTA) *are decidable. More precisely*:

- *Let* $P \in$ iSA *and let* m *be the maximal arity of a relation in* $\mathscr{A}(P)$. *Let* σ *be a finite* (d,k)*-sphere over the signature of* P. *Then it can be checked in space* $|\sigma|^{O(m)} \cdot |P|^2 \cdot 2^{O(\delta(\sigma))}$, *whether* σ *is realizable in* $\mathscr{A}(P)$.
- *If* $P \in$ iTA, *then realizability can be checked in time* $\exp(1, |\sigma|^{O(m)} \cdot |P|^2 \cdot 2^{O(\delta(\sigma))})$.

PROOF. We first prove the statement on injective string automatic presentations. Let $P = (\Gamma, A_0, (A_r)_{r \in \mathcal{S}}) \in$ iSA. Let $\sigma = (\mathscr{B}, b_1, \ldots, b_k)$ and let $c_1, \ldots, c_{|\sigma|}$ be a list of all elements of $\mathscr{B}$. Note that every b_i occurs in this list. Let $E_{\mathscr{A}(P)}$ be the edge relation of the Gaifman graph $G(\mathscr{A}(P))$ and $E_{\mathscr{B}}$ that of the Gaifman graph $G(\mathscr{B})$. Then σ is realizable in $\mathscr{A}(P)$ if and only if there are words $u_1, \ldots, u_{|\sigma|} \in \Gamma^*$ such that

(a) $u_i \in L(A_0)$ for all $1 \leq i \leq |\sigma|$,
(b) $u_i \neq u_j$ for all $1 \leq i < j \leq |\sigma|$,
(c) $(u_{i_1}, \ldots, u_{i_{m_r}}) \in R(A_r)$ for all $r \in \mathcal{S}$ and all $(c_{i_1}, \ldots, c_{i_{m_r}}) \in r^{\mathscr{B}}$,
(d) $(u_{i_1}, \ldots, u_{i_{m_r}}) \notin R(A_r)$ for all $r \in \mathcal{S}$ and all $(c_{i_1}, \ldots, c_{i_{m_r}}) \in \mathscr{B}^{m_r} \setminus r^{\mathscr{B}}$, and
(e) there is no $v \in L(A_0)$ such that, for some $1 \leq j \leq |\sigma|$ and $1 \leq i \leq k$ with $d(c_j, b_i) < 2^{d-i}$, we have
  (e.1) $(u_j, v) \in E_{\mathscr{A}(P)}$ and
  (e.2) $v \notin \{u_p \mid (c_j, c_p) \in E_{\mathscr{B}}\}$.

Then (a)–(d) express that the mapping $f\colon c_i \mapsto u_i$ $(1 \leq i \leq |\sigma|)$ is well-defined and an embedding of $\mathscr{B}$ into $\mathscr{A}(P)$. In (e), $(u_j, v) \in E_{\mathscr{A}(P)}$ implies that v belongs to $\bigcup_{1 \leq i \leq k} S(2^{d-i}, f(b_i))$. Hence (e) expresses that all elements of $\bigcup_{1 \leq i \leq k} S(2^{d-i}, f(b_i))$ belong to the image of f.

We now construct a $|\sigma|$-dimensional automaton A over the alphabet Γ that checks (a)–(e). More precisely, this automaton will accept all convolutions $u_1 \otimes u_2 \otimes \cdots \otimes u_{|\sigma|}$ with $u_1, \ldots, u_{|\sigma|} \in \Gamma^*$ such that (a)–(e) hold. At the end, we have to check the language of this automaton for non-emptiness. Our actual algorithm for checking realizability will not construct the automaton A explicitly (it would not fit into the space bound) but will check its non-emptiness on the fly. The automaton A is the direct product of automata A_a, A_b, $A_{c,d}$, and A_e that check the conditions separately ($A_{c,d}$ checks both (c) and (d)). The automaton A_a is the direct product of $|\sigma|$ many copies of the automaton A_0, hence A_a has at most $|P|^{|\sigma|}$ many states.

Next, the automaton for (b) is the direct product of $O(|\sigma|^2)$ many copies of an automaton of fixed size (that checks whether two tracks are different). Hence, this automaton has $2^{O(|\sigma|^2)}$ many states.

The automaton $A_{c,d}$ checks for every relation symbol $r \in \mathcal{S}$ of arity $m_r \leq m$ and every tuple $(i_1, \ldots, i_{m_r}) \in \{1, \ldots, |\sigma|\}^{m_r}$ whether the input words on tracks $i_1, \ldots, i_{m_r}$ are accepted by the automaton A_r (in case $(c_{i_1}, \ldots, c_{i_{m_r}}) \in r^{\mathscr{B}}$) or by an automaton for the complement of $L(A_r)$ (in case $(c_{i_1}, \ldots, c_{i_{m_r}}) \notin r^{\mathscr{B}}$). Using the powerset construction, we obtain an automaton for the complement of $L(A_r)$ with at most $2^{|P|}$ many states. Hence, since the number of relation symbols in $\mathcal{S}$ is bounded by $|P|$, the automaton $A_{c,d}$ is the direct product of at most $|P| \cdot |\sigma|^m$ many automata of size at most $2^{|P|}$. Hence, the number of states of $A_{d,e}$ is bounded by $(2^{|P|})^{|P| \cdot |\sigma|^m} = \exp(1, |P|^2 \cdot |\sigma|^m)$.

It remains to construct the automaton A_e. For this, we first construct its complement, i.e., an automaton A'_e that accepts all convolutions $u_1 \otimes u_2 \otimes \cdots \otimes u_{|\sigma|}$, for which there exists $v \in L(A_0)$ with the desired properties. This automaton A'_e is the disjoint union of at most $|\sigma|$ many automata $A'_{e,j}$, one for each $1 \leq j \leq |\sigma|$ such that there exists $1 \leq i \leq k$ with $d(c_j, b_i) < 2^{d-i}$. Each of these components $A'_{e,j}$ is the projection onto the first $|\sigma|$ many tracks of an automaton $A''_{e,j}$ that accepts all convolutions $u_1 \otimes u_2 \otimes \cdots \otimes u_{|\sigma|} \otimes v$ such that (e.1) and (e.2) hold. Hence, $A''_{e,j}$ is the direct product of automata $A''_{e.1,j}$ and $A''_{e.2,j}$ checking (e.1) and (e.2), respectively. By Lemma 2.13, $A''_{e.1,j}$ has at most $m^2 \cdot |P|^2$ many states. Recall that the degree of $\mathscr{B}$ is $\delta(\sigma)$. Hence, the set $\{u_p \mid (c_j, c_p) \in E_{\mathscr{B}}\}$ contains at most $\delta(\sigma)$ many elements, and $A''_{e.2,j}$ is the direct product of at most $\delta(\sigma)$ many automata of constant size (checking whether two tracks are different). Thus, $A''_{e.2,j}$ has $2^{O(\delta(\sigma))}$ many states. Hence, A'_e is the disjoint union of at most $|\sigma|$ many automata of size $|P|^2 \cdot m^2 \cdot 2^{O(\delta(\sigma))}$ and therefore has at most $|\sigma| \cdot |P|^2 \cdot m^2 \cdot 2^{O(\delta(\sigma))}$ many states. Since A_e results from complementing the nondeterministic automaton A'_e, the number of states of A_e can be bound by $\exp(1, |\sigma| \cdot |P|^2 \cdot m^2 \cdot 2^{O(\delta(\sigma))})$.

In summary, the automaton A has at most

$$|P|^{|\sigma|} \cdot 2^{O(|\sigma|^2)} \cdot 2^{|P|^2 \cdot |\sigma|^m} \cdot 2^{|\sigma| \cdot |P|^2 \cdot m^2 \cdot 2^{O(\delta(\sigma))}} \leq \exp(1, |\sigma|^{O(m)} \cdot |P|^2 \cdot 2^{O(\delta(\sigma))})$$

many states. Hence checking emptiness of its language (and therefore realizability of σ in $\mathscr{A}(P)$) can be done in space logarithmic to the number of states, i.e., in space $|\sigma|^{O(m)} \cdot |P|^2 \cdot 2^{O(\delta(\sigma))}$. For this the algorithm does not have to construct A but only has to store two states of A, for which space $|\sigma|^{O(m)} \cdot |P|^2 \cdot 2^{O(\delta(\sigma))}$ is sufficient. This proves the statement for string automatic presentations.

For injective tree automatic presentations, the construction and size estimate for A are the same as above. But emptiness of tree automata can only be checked in deterministic polynomial time (and not in logspace unless NL = P). Hence, emptiness of A can be checked in time $\exp(1, |\sigma|^{O(m)} \cdot |P|^2 \cdot 2^{O(\delta(\sigma))})$. ⊣

REMARK 3.6. Realizability of a given (d,k)-sphere σ can be expressed as a Σ_2-formula of the form $\exists x_1 \ldots \exists x_{|\sigma|} \forall y \colon \theta(x_1, \ldots, x_{|\sigma|}, y)$, where θ is a quantifier free formula. Using the standard automata construction (which underlies the proof of Theorem 2.7), one can translate the formula $\forall y \colon \theta(x_1, \ldots, x_{|\sigma|}, y)$ into an equivalent automaton of size $\exp(2, |P| \cdot |\theta|)$, which can be checked for non-emptiness in space $\exp(1, |P| \cdot |\theta|)$. Our finer analysis has the advantage of yielding a space bound, which is only exponential in the maximal arity m and the degree $\delta(\sigma)$ of the sphere σ; this is crucial in order to obtain our upper bounds in Theorem 3.7 below. Basically, this finer analysis is possible, since the universal quantifier $\forall y$ is used in a very restricted way in the formula for realizability (in some sense, it is a guarded quantifier).

In the following, for a tree automatic presentation P of bounded degree, we denote with $g'_P \doteq g'_{\mathscr{A}(P)}$ the normalized growth function of the structure $\mathscr{A}(P)$.

THEOREM 3.7. *The model checking problem* FOMC(TAb) *is decidable, i.e., on input of a tree automatic presentation P of bounded degree and a closed formula φ over the signature of P, one can effectively determine whether $\mathscr{A}(P) \models \varphi$ holds. More precisely* (*where m is the maximal arity of a relation from the signature of P*):

(1) FOMC(iSAb) *can be decided in space*

$$g'_P(2^{|\varphi|})^{O(m)} \cdot \exp(2, |P|^{O(1)}) \leq \exp(2, |P|^{O(1)} + |\varphi|).$$

(2) FOMC(SAb) *can be decided in space*

$$\exp(3, O(|P|) + \log(|\varphi|)).$$

(3) FOMC(iTAb) *can be decided in time*

$$\exp\Big(1,\ g'_P(2^{|\varphi|})^{O(m)} \cdot \exp(3, |P|^{O(1)})\Big) \leq \exp(4, |P|^{O(1)} + \log(|\varphi|)).$$

(4) FOMC(TAb) *can be decided in time*

$$\exp(4, 2^{O(|P|)} + \log(|\varphi|)) \leq \exp(5, O(|P|) + \log(\log(|\varphi|))).$$

PROOF. The decidability follows immediately from Theorem 2.4 and Proposition 2.14(a).

We first give the proof for injective string automatic presentations. So, let $P \in$ iSAb and let φ be a closed first-order formula of quantifier rank d. Let δ be the degree of $\mathscr{A}(P)$. By Proposition 2.15, it is bounded by $\exp(1, |P|^{O(1)})$. By Proposition 3.3 it suffices to compute the Boolean value $\varphi_\emptyset$. Recall the inductive definition of φ_σ from Definition 3.2 that we now translate into an algorithm for computing $\varphi_\emptyset$.

First note that such an algorithm has to handle (d,k)-spheres for $1 \leq k \leq d \leq |\varphi|$ that are realizable in $\mathscr{A}(P)$. The number of nodes of a (d,k)-sphere realizable in $\mathscr{A}(P)$ is bounded by $k \cdot g'_P(2^d) \leq g'_P(2^d)^{O(1)}$ since $k \leq d < 2^d \leq g'_P(2^d)$. The number of relations of $\mathscr{A}(P)$ is bounded by $|P|$. Hence, any (d,k)-sphere that is realizable in $\mathscr{A}(P)$ can be described by $|P| \cdot g'_P(2^d)^{O(m)}$ many bits. Moreover, only (d,k)-spheres of degree at most $\delta \leq \exp(1, |P|^{O(1)})$ can be realizable.

Note that the set of (d,k)-spheres with $0 \leq k \leq d$ (ordered by the extension relation $\preceq$) forms a tree of depth $d+1$. The algorithm visits the nodes of this tree (restricted to spheres with at most $g'_P(2^d)^{O(1)}$ many nodes) in a depth-first manner and descends when unraveling an existential quantifier. Hence we have to store $d+1$ many spheres. For this, the algorithm needs space $(d+1) \cdot |P| \cdot g'_P(2^d)^{O(m)} \leq |P| \cdot g'_P(2^{|\varphi|})^{O(m)}$.

Moreover, during the unraveling of a quantifier, the algorithm has to check realizability of a (d,k)-sphere for $1 \leq k \leq d \leq |\varphi|$. Any such sphere has at most $g'_P(2^d)^{O(1)}$ many elements. If the current sphere has degree larger than $\exp(1, |P|^{O(1)})$ (which is an upper bound for the degree of $\mathscr{A}(P)$) then it is clearly not realizable. Otherwise, we can check realizability by Lemma 3.5 in space $g'_P(2^d)^{O(m)} \cdot |P|^2 \cdot \exp(2, |P|^{O(1)}) \leq g'_P(2^{|\varphi|})^{O(m)} \cdot \exp(2, |P|^{O(1)})$.

At the end, we have to check whether a tuple $\overline{b}$ satisfies an atomic formula $\psi(\overline{y})$, which is trivial. In total, the algorithm runs in space

$$|P| \cdot g'_P(2^{|\varphi|})^{O(m)} + g'_P(2^{|\varphi|})^{O(m)} \cdot \exp(2, |P|^{O(1)}) = g'_P(2^{|\varphi|})^{O(m)} \cdot \exp(2, |P|^{O(1)}).$$

Recall that $g'_{\mathscr{A}}(2^{|\varphi|}) \leq \delta^{2^{|\varphi|}}$ and $\delta \leq 2^{|P|^{O(1)}}$ by Proposition 2.15. Since also $m \leq |P|$, we obtain

$$g'_P(2^{|\varphi|})^{O(m)} \cdot \exp(2, |P|^{O(1)}) \leq \exp(1, |P|^{O(1)} \cdot 2^{|\varphi|} \cdot O(m)) \cdot \exp(2, |P|^{O(1)}) \leq \exp(2, |P|^{O(1)} + |\varphi|).$$

This completes the consideration for injective string automatic presentations.

If P is just string automatic, we can transform it into an equivalent injective string automatic presentation which increases the size exponentially by Lemma 2.10. Hence, replacing $|P|$ by $2^{O(|P|)}$ yields the space bound.

Next, we consider injective tree automatic presentations. The algorithm is the same, i.e., it parses the tree of all (d,k)-spheres and checks them for realizability. Note that the number of (d,k)-spheres that are realizable is bounded by $\exp(1, |P| \cdot g'_P(2^d)^{O(m)})$. By Proposition 2.15, the degree δ of $\mathscr{A}(P)$ is bounded by $\exp(2, |P|^{O(1)})$. By Lemma 3.5, the realizability of any (d,k)-sphere of degree $\exp(2, |P|^{O(1)})$ can be checked in time

$$\exp\Big(1,\ g'_P(2^d)^{O(m)} \cdot |P|^2 \cdot \exp(3, |P|^{O(1)})\Big) \leq \exp\Big(1,\ g'_P(2^{|\varphi|})^{O(m)} \cdot \exp(3, |P|^{O(1)})\Big).$$

Recall that $g'_P(2^{|\varphi|}) \leq \delta^{2^{|\varphi|}}$ and $\delta \leq \exp(2, |P|^{O(1)})$ by Proposition 2.15. Since also $m \leq |P|$, we obtain

$$g'_P(2^{|\varphi|})^{O(m)} \cdot \exp(3, |P|^{O(1)}) \leq \exp(2, |P|^{O(1)})^{2^{|\varphi|} \cdot O(|P|)} \cdot \exp(3, |P|^{O(1)}) = \exp(2, |P|^{O(1)} + |\varphi|) \cdot \exp(3, |P|^{O(1)}) \leq \exp(3, |P|^{O(1)} + \log(|\varphi|)).$$

Finally, the last statement for FOMC(TAb) follows from the time bound for FOMC(iTAb) and Lemma 2.10. ⊣

We derive a number of consequences on the uniform and non-uniform complexity of the first-order theories of string/tree automatic structures of bounded degree. The first one concerns the uniform model checking problems and is a direct consequence of the above theorem.

COROLLARY 3.8. *The following holds*:

(a) *The model checking problem* FOMC(iSAb) *belongs to* 2EXPSPACE.
(b) *The model checking problem* FOMC(SAb) *belongs to* 3EXPSPACE.
(c) *The model checking problem* FOMC(iTAb) *belongs to* 4EXPTIME.
(d) *The model checking problem* FOMC(TAb) *belongs to* 5EXPTIME.

Next we concentrate on the non-uniform complexity, where the structure is fixed. For string automatic structures, we do not get a better upper bound in this case (statement (i) below) except in case of polynomial growth (statement (ii) below).

COROLLARY 3.9. *Let $\mathscr{A}$ be a string automatic structure of bounded degree.*

(i) *Then* FOTh($\mathscr{A}$) *belongs to* 2EXPSPACE.
(ii) *If $\mathscr{A}$ has polynomial growth then* FOTh($\mathscr{A}$) *belongs to* EXPSPACE.

PROOF. Since $\mathscr{A}$ is string automatic, it has a fixed injective string automatic presentation P, i.e., $|P|$ and m are fixed constants. Hence the first result follows immediately from (1) in Theorem 3.7.

Now suppose that $\mathscr{A}$ has polynomial growth, i.e., $g'_{\mathscr{A}}(x) \in x^{O(1)}$. Then, again, the second claim follows immediately from (1) in Theorem 3.7, since $g'_{\mathscr{A}}(2^{|\varphi|})^{O(m)} \leq 2^{O(|\varphi|)}$. ⊣

The last consequence of Theorem 3.7 concerns tree automatic structures. Here, we can improve the upper bound from Theorem 3.7 for the non-uniform case by one exponent. In case of polynomial growth, we can save yet another exponent:

COROLLARY 3.10. *Let $\mathscr{A}$ be a tree automatic structure of bounded degree.*

(i) *Then* FOTh($\mathscr{A}$) *belongs to* 3EXPTIME.
(ii) *If $\mathscr{A}$ has polynomial growth then* FOTh($\mathscr{A}$) *belongs to* 2EXPTIME.

PROOF. Since $\mathscr{A}$ is tree automatic, it has a fixed injective tree automatic presentation P. Hence, again, the first claim follows immediately from (3) in Theorem 3.7.

Now suppose that $\mathscr{A}$ has polynomial growth, i.e., $g'_{\mathscr{A}}(x) \in x^{O(1)}$. Then the second claim follows since

$$\exp(1, g'_{\mathscr{A}}(2^{|\varphi|})^{O(m)}) \leq \exp(1, 2^{O(|\varphi|)}) = \exp(2, O(|\varphi|)),$$

implying that the problem belongs to 2EXPTIME. ⊣

3.2.1. *Two observations on the growth function.* We complement this section with a short excursion into the field of growth functions of automatic structures. The two results to be reported indicate that these growth functions do not behave as nicely as one would wish. Fortunately, these negative findings are of no importance to our main concerns.

Recall that the growth rate of a regular language is either bounded by a polynomial from above or by an exponential function from below and that it is decidable which of these cases applies. The next lemmas show that the analogous statements for growth functions of string automatic structures are false.

LEMMA 3.11. *There is a string automatic graph of intermediate growth (i.e., the growth is neither exponential nor polynomial).*

PROOF. Let $L = \{0,1\}^*\$\{0,1\}^*$ and let E be

$$\{(u\$bv, ub\$v) \mid u, v \in \{0,1\}^*, b \in \{0,1\}\} \cup \{(u\$, \$ub) \mid u \in \{0,1\}^*, b \in \{0,1\}\}.$$

Then $T = (L, E)$ is a string automatic tree obtained from the complete binary tree $\$\{0,1\}^*$ by adding a path of length n between u and ub for $u \in \{0,1\}^n$ and $b \in \{0,1\}$. Hence, a path of length n starting in the root $\$$ of T branches at distance $0, 2, 5, 10, \ldots, i^2 + 1, \ldots, \lfloor\sqrt{n-1}\rfloor^2 + 1$ from the root. Hence, for the growth function g_T we obtain the following estimate:

$$g_T(n) \in \sum_{i=0}^{\Theta(\sqrt{n})} (i+1) \cdot 2^i = \Theta(\sqrt{n}) \cdot 2^{\Theta(\sqrt{n})} = 2^{\Theta(\sqrt{n})}.$$

⊣

LEMMA 3.12. *It is undecidable whether a string automatic graph of bounded degree has polynomial growth.*

PROOF. We show the undecidability by a reduction of the halting problem (with empty input) for Turing machines. So let N be a Turing machine. We can transform N into a deterministic reversible Turing machine M such that:

(i) N halts on empty input if and only if M does so.
(ii) M does not allow infinite sequences of backwards steps (i.e., there are no configurations c_i with $c_{i+1} \vdash_M c_i$ for all $i \in \mathbb{N}$), see also [27] for a similar construction.

Let C be the set of configurations of M (a regular set) and c_0 the initial configuration with empty input. Now define $L = (\{0,1\}C)^+$ (we assume that 0 and 1 do not belong to the alphabet of C) and

$$E = \{(uac, uac') \mid u \in L \cup \{\varepsilon\}, a \in \{0,1\}, c, c' \in C, c \vdash_M c'\}$$
$$\cup \{(uac, uacbc_0) \mid u \in L \cup \{\varepsilon\}, a, b \in \{0,1\}, c \in C \text{ is halting}\}.$$

Then (L, E) is an automatic directed graph. Since M is reversible, it is a forest of rooted trees (by (ii)).

First suppose there are configurations $c_1, c_2, \ldots, c_n$ with $c_{i-1} \vdash_M c_i$ for $1 \leq i \leq n$ such that c_n is halting. Then the set $0(c_n\{0,1\})^*\{c_0, c_1, \ldots, c_n\}$ forms an infinite tree in (L, E). Any branch in this tree branches every n steps. Hence (L, E) has exponential growth.

Now assume that c_0 is the starting point of an infinite computation. Let T be any tree in the forest (L, E). Then its root is of the form $uac \in L$ with $u \in L \cup \{\varepsilon\}$, $a \in \{0,1\}$, and $c \in C$ such that c is no successor configuration of any other configuration. There are two possibilities:

1. The configuration c is the starting configuration of an infinite computation of M. Then T is an infinite path.
2. There is a halting configuration c' and $n \in \mathbb{N}$ with $c \vdash_M^n c'$. Then T starts with a path of length n. The final node of this path has two children, namely $uac'0c_0$ and $uac'1c_0$. But, since M does not halt on the empty input, each of these nodes is the root of an infinite path.

Thus, in this case (L, E) has polynomial (even linear) growth. ⊣

§4. Lower bounds. In this section, we will prove that the upper complexity bounds for the non-uniform problems (Corollary 3.9 and Corollary 3.10) are sharp. This will imply that the complexity of the uniform problem for injective string automatic presentations from Corollary 3.8 is sharp as well.

For a binary relation r and $m \in \mathbb{N}$ we denote with r^m the m-fold composition of r. Then the following lemma is folklore.

LEMMA 4.1. *Let the signature $\mathcal{S}$ contain a binary symbol r. From a given number n (encoded unary), we can construct in linear time a formula $\varphi_n(x, y)$ such that for every $\mathcal{S}$-structure $\mathcal{A}$ and all elements $a, b \in \mathcal{A}$ we have: $(a, b) \in r^{2^n}$ if and only if $\mathcal{A} \models \varphi_n(a, b)$.*

PROOF. Let $\varphi_0(x, y) = r(x, y)$ and, for $n > 0$ define

$$\varphi_n(x, y) = \exists z \, \forall x', y' (((x' = x \wedge y' = z) \vee (x' = z \wedge y' = y)) \rightarrow \varphi_{n-1}(x', y')).$$ ⊣

For a bit string $u = a_1 \dots a_m$ ($a_i \in \{0,1\}$) let $\mathsf{val}(u) = \sum_{i=0}^{m-1} a_{i+1} 2^i$ be the integer value represented by u. Vice versa, for $0 \leq i \leq 2^m - 1$ let $\mathsf{bin}_m(i) \in \{0,1\}^m$ be the unique string with $\mathsf{val}(\mathsf{bin}_m(i)) = i$.

THEOREM 4.2. *There is a fixed string automatic structure $\mathscr{A}$ of bounded degree such that* FOTh($\mathscr{A}$) *is* 2EXPSPACE*-hard.*

PROOF. Let M be a fixed Turing machine with a space bound of $\exp(2, n)$ such that M accepts a 2EXPSPACE-complete language; such a machine exists by standard arguments. Let Γ be the tape alphabet, $\Sigma \subseteq \Gamma$ be the input alphabet, and Q be the set of states. The initial (resp. accepting) state is $q_0 \in Q$ (resp. $q_f \in Q$), the blank symbol is $\Box \in \Gamma \setminus \Sigma$. Let $\Omega = Q \cup \Gamma$. A configuration of M is described by a string from $\Gamma^* Q \Gamma^+ \subseteq \Omega^+$ (later, symbols of configurations will be preceded with additional counters). For two configurations u and v, we write $u \vdash_M v$ if $|u| = |v|$ and u can evolve with a single M-transition into v. Note that there exists a relation $\alpha_M \subseteq \Omega^3 \times \Omega^3$ such that for all configurations $u = a_1 \dots a_m$ and $v = b_1 \dots b_m$ ($a_i, b_i \in \Omega$) we have

$$u \vdash_M v \iff \forall i \in \{1, \dots, m-2\} \colon (a_i a_{i+1} a_{i+2}, b_i b_{i+1} b_{i+2}) \in \alpha_M. \tag{9}$$

Let $\Delta = \{0, 1, \#\} \cup \Omega$, and let $\pi \colon \Delta \to \Omega \cup \{\#\}$ be the projection morphism with $\pi(a) = a$ for $a \in \Omega \cup \{\#\}$ and $\pi(0) = \pi(1) = \varepsilon$. For $m \in \mathbb{N}$, a string $x \in \Delta^*$ is an *accepting 2^m-computation* if x can be factorized as $x = x_1 \# x_2 \# \dots x_n \#$ for some $n \geq 1$ such that the following holds:

- For every $1 \leq i \leq n$ there exist $a_{i,0}, \dots, a_{i,2^m-1} \in \Omega$ such that $x_i = \prod_{j=0}^{2^m-1} \mathsf{bin}_m(j) a_{i,j}$.
- For every $1 \leq i \leq n$, $\pi(x_i) \in \Gamma^* Q \Gamma^+$.
- $\pi(x_1) \in q_0 \Sigma^* \Box^*$ and $\pi(x_n) \in \Gamma^* q_f \Gamma^+$.
- For every $1 \leq i < n$, $\pi(x_i) \vdash_M \pi(x_{i+1})$.

From M we now construct a fixed string automatic structure $\mathscr{A}$ of bounded degree. We start with the following language U_0:

$$U_0 = \pi^{-1}((\Gamma^* Q \Gamma^+ \#)^*) \tag{10}$$

$$\cap\, (0^+ \Omega (\{0,1\}^+ \Omega)^* 1^+ \Omega \#)^+ \tag{11}$$

$$\cap\, 0^+ q_0 (\{0,1\}^+ \Sigma)^* (\{0,1\}^+ \Box)^* \# \Delta^* \tag{12}$$

$$\cap\, \Delta^* q_f (\Delta \setminus \{\#\})^* \#. \tag{13}$$

A string $x \in U_0$ is a candidate for an accepting 2^m-computation of M. With (10) we describe the basic structure of such a computation; it consists of a list of configurations separated by #. Moreover, every symbol in a configuration is preceded by a bit string, which represents a *counter*. By (11) every counter is non-empty, the first symbol in a configuration is preceded by a counter from 0^+, the last symbol is preceded by a counter from 1^+. Moreover, by (12), the first configuration is an initial configuration, whereas by (13), the last configuration is accepting (i.e., the current state is q_f).

For the further considerations, let us fix some $x \in U_0$. Hence, we can factorize x as $x = x_1 \# x_2 \# \dots x_n \#$ such that:

- For every $1 \leq i \leq n$, there exist $m_i \geq 1$, $a_{i,0}, \dots, a_{i,m_i} \in \Omega$ and counters $u_{i,0}, \dots, u_{i,m_i} \in \{0,1\}^+$ such that $x_i = \prod_{j=0}^{m_i} u_{i,j} a_{i,j}$.

- For every $1 \leq i \leq n$, $u_{i,0} \in 0^+$, $u_{i,m_i} \in 1^+$, and $\pi(x_i) \in \Gamma^* Q \Gamma^+$.
- $\pi(x_1) \in q_0 \Sigma^* \square^*$ and $\pi(x_n) \in \Gamma^* q_f \Gamma^+$.

We next want to construct, from $m \in \mathbb{N}$, a small formula expressing that x is an accepting 2^m-computation. To achieve this, we add some structure around strings from U_0. Then the formula we are seeking has to ensure two facts:

(a) The counters behave correctly, i.e., for all $1 \leq i \leq n$ and $0 \leq j \leq m_i$, we have $|u_{i,j}| = m$ and if $j < m_i$, then $\mathsf{val}(u_{i,j+1}) = \mathsf{val}(u_{i,j}) + 1$. Note that this enforces $m_i = 2^m - 1$ for all $1 \leq i \leq n$.

(b) For two successive configurations, the second one is the successor configuration of the first one with respect to the machine M, i.e., $\pi(x_i) \vdash_M \pi(x_{i+1})$ for all $1 \leq i < n$.

In order to achieve (a), we introduce the following three binary relations; it is straightforward to exhibit 2-dimensional automata for these relations:

$$\delta = \{(w, w \otimes w) \mid w \in U_0\},$$

$$\sigma_0 = \{\big((0v_1\#0v_2\# \dots 0v_n\#) \otimes w, (v_1 0\# v_2 0\# \dots v_n 0\#) \otimes w\big) \mid w \in U_0, v_1, \dots, v_n \in (\Delta \setminus \{\#\})^*\},$$

$$\sigma_\Omega = \{\big((a_1 v_1\# a_2 v_2\# \dots a_n v_n\#) \otimes w, (v_1 a_1\# v_2 a_2\# \dots v_n a_n\#) \otimes w\big) \mid w \in U_0, a_1, \dots, a_n \in \Omega, v_1, \dots, v_n \in (\Delta \setminus \{\#\})^*\}.$$

Hence, δ just duplicates a string from U_0 and σ_0 cyclically rotates every configuration to the left for one symbol, provided the first symbol is 0, whereas σ_Ω rotates symbols from Ω. Moreover, let U_1 be the following language over $\Delta^* \otimes \Delta^*$:

$$U_1 = \left(\left\{ ua \otimes vb \;\middle|\; \begin{array}{l} u, v \in \{0,1\}^+, a, b \in \Omega, \\ |u| = |v|, \mathsf{val}(u) = \mathsf{val}(v) + 1 \bmod 2^{|u|} \end{array} \right\}^+ (\#, \#) \right)^+.$$

Clearly, U_1 is a regular language. The crucial fact is the following, whose proof is straightforward:

FACT 1. For every $m \in \mathbb{N}$, the following two properties are equivalent (recall that $x \in U_0$):

- There exist $y_1, y_2, y_3 \in \Delta^* \otimes \Delta^*$ such that $\delta(x, y_1)$, $\sigma_0^m(y_1, y_2)$, $\sigma_\Omega(y_2, y_3)$, and $y_3 \in U_1$.
- For all $1 \leq i \leq n$ and $0 \leq j \leq m_i$, we have $|u_{i,j}| = m$ and if $j < m_i$, then $\mathsf{val}(u_{i,j+1}) = \mathsf{val}(u_{i,j}) + 1$.

Assume now that $x \in U_0$ satisfies one (and hence both) of the two properties from Fact 1 for some m. It follows that $m_i = 2^m - 1$ for all $1 \leq i \leq n$ and

$$x = x_1\#x_2\# \dots x_n\#, \text{ where } x_i = \prod_{j=0}^{2^m-1} \mathsf{bin}_m(j) a_{i,j} \text{ for every } 1 \leq i \leq n. \tag{14}$$

In order to establish (b) we need additional structure. The idea is, for every counter value $0 \leq j < 2^m$, to have a word y_j that coincides with x, but has all the occurrences of $\mathsf{bin}_m(j)$ marked. Then an automaton can check that successive occurrences of the counter $\mathsf{bin}_m(j)$ obey the transition condition of the Turing machine. There are two problems with this approach: first, in order to relate x and y_j, we would

need a binary relation of degree 2^m (for arbitrary m) and, secondly, an automaton cannot mark all the occurrences of $\mathrm{bin}_m(j)$ at once (for some j). In order to solve these problems, we introduce a binary relation μ, which for every $x \in U_0$ as in (14) generates a binary tree of depth m with root x; this will be the only relation in our string automatic structure that causes exponential growth. This relation will mark in x every occurrence of an arbitrary counter. For this, we need two copies $\overline{0}$ and $\underline{0}$ of 0 as well as two copies $\overline{1}$ and $\underline{1}$ of 1. For $b \in \{0,1\}$, define the mapping

$$f_b\colon \{\underline{0},\overline{0},\underline{1},\overline{1}\}^*\{0,1\}^+ \to \{\underline{0},\overline{0},\underline{1},\overline{1}\}^+\{0,1\}^*$$

as follows (where $u \in \{\underline{0},\overline{0},\underline{1},\overline{1}\}^*$, $c \in \{0,1\}$, and $v \in \{0,1\}^*$):

$$f_b(ucv) = \begin{cases} u\underline{c}v & \text{if } b \neq c, \\ u\overline{c}v & \text{if } b = c. \end{cases}$$

We extend f_b to a function on $((\{\underline{0},\overline{0},\underline{1},\overline{1}\}^*\{0,1\}^+\Omega)^+\#)^*$ as follows: Let $w = w_1a_1 \ldots w_\ell a_\ell$ with $w_i \in \{\underline{0},\overline{0},\underline{1},\overline{1}\}^*\{0,1\}^+$ and $a_i \in \Omega \cup \Omega\#$. Then $f_b(w) = f_b(w_1)a_1 \ldots f_b(w_\ell)a_\ell$; this mapping can be computed with a synchronized transducer. Hence, the relation

$$\mu = f_0 \cup f_1 = \{(u, f_b(u)) \mid u \in ((\{\underline{0},\overline{0},\underline{1},\overline{1}\}^*\{0,1\}^+\Omega)^+\#)^*, b \in \{0,1\}\}$$

can be recognized by a 2-dimensional automaton.

Let $x \in U_0$ as in (14), let the word y be obtained from x by overlining or underlining each bit in x, and let $u \in \{0,1\}^m$ be some counter. We say *the counter u is marked in y* if every occurrence of the counter u is marked by overlining each bit, whereas all other counters contain at least one underlined bit $\underline{0}$ or $\underline{1}$. The following fact follows immediately from the definition of the relation μ.

FACT 2. Let $x \in U_0$ be as in (14).

- For all counters $u \in \{0,1\}^m$, there exists a unique word y with $(x,y) \in \mu^m$ such that the counter u is marked in y.
- If $(x,y) \in \mu^m$, then there exists a unique counter $u \in \{0,1\}^m$ such that u is marked in y.

Now, we can achieve our final goal, namely checking whether two successive configurations in $x \in U_0$ represent a transition of the machine M. Let the counter $u \in \{0,1\}^m$ be marked in y. We describe a finite automaton A_2 that checks on the string y, whether at position $\mathrm{val}(u)$ successive configurations in x are "locally consistent". The automaton A_2 searches for the first marked counter in y. Then it stores the next three symbols a_1, a_2, a_3 from Ω (only if the separator symbol does not occur in between a_1 and a_3), walks right until it finds the next marked counter, reads the next three symbols b_1, b_2, b_3 from Ω, and checks whether $(a_1a_2a_3, b_1b_2b_3) \in \alpha_M$, where α_M is from (9). If this is not the case, the automaton will reject, otherwise it will store $b_1b_2b_3$ and repeat the procedure described above. Let $U_2 = L(A_2)$. Together with Fact 1 and 2, the behavior of A_2 implies that for all $x \in U_0$ and all $m \in \mathbb{N}$, x represents an accepting 2^m-computation of M if and

only if

$$\exists y_1, y_2, y_3 \Big(\delta(x, y_1) \wedge \sigma_0^m(y_1, y_2) \wedge \sigma_\Omega(y_2, y_3) \wedge y_3 \in U_1 \Big)$$
$$\wedge \forall y \Big(\mu^m(x, y) \to y \in U_2 \Big).$$

Let us now fix some input $w = a_1 a_2 \dots a_n \in \Sigma^*$ with $|w| = n$, and let $a_{n+1} = \square$ and $m = 2^n$. Thus, w is accepted by M if and only if there exists an accepting 2^m-computation x such that in the first configuration of x, the tape content is of the form $w\square^+$. It remains to add some structure that allows us to express the latter by a formula. But this is straightforward: Let $\rhd$ be a new symbol and let $\Pi = \Delta \cup \{\underline{0}, \overline{0}, \underline{1}, \overline{1}, \rhd\}$; this is our final alphabet. Define the binary relations $\iota_{0,1}$ and ι_a ($a \in \Omega$) as follows:

$$\begin{aligned} \iota_{0,1} &= \{(u \rhd av, ua \rhd v) \mid a \in \{0,1\}, u, v \in \Delta^*, uav \in U_0\} \\ &\quad \cup \{(0v, 0 \rhd v) \mid v \in \Delta^*, 0v \in U_0\}, \\ \iota_a &= \{(u \rhd av, ua \rhd v) \mid u, v \in \Delta^*, uav \in U_0\}. \end{aligned}$$

Then, $\mathscr{A} = (\Pi^* \cup (\Pi^* \otimes \Pi^*), \delta, \sigma_0, \sigma_\Omega, \mu, \iota_{0,1}, (\iota_a)_{a \in \Omega}, U_0, U_1, U_2)$ is a string automatic structure of bounded degree such that w is accepted by M if and only if the following formula is true in $\mathscr{A}$:

$$\exists x \in U_0 \left(\begin{array}{ll} & \exists y_1, y_2, y_3 \Big(\delta(x, y_1) \wedge \sigma_0^m(y_1, y_2) \wedge \sigma_\Omega(y_2, y_3) \wedge y_3 \in U_1 \Big) \\ \wedge & \forall y \Big(\mu^m(x, y) \to y \in U_2 \Big) \\ \wedge & \exists y_0, z_0, \dots, y_{n+1}, z_{n+1} \left(\begin{array}{l} \iota_{0,1}^m(x, y_0) \wedge \iota_{q_0}(y_0, z_0) \wedge \\ \bigwedge_{i=1}^{n+1} \iota_{0,1}^m(z_{i-1}, y_i) \wedge \iota_{a_i}(y_i, z_i) \end{array} \right) \end{array} \right).$$

By Lemma 4.1 we can compute in time $O(\log(m)) = O(n)$ an equivalent formula over the signature of $\mathscr{A}$. This concludes the proof. ⊣

The following theorem, which proves an analogous result for tree automatic structures, uses alternating Turing machines, see [8, 30] for more details. Roughly speaking, an *alternating Turing machine* is a nondeterministic Turing machine, where the set of states is partitioned into accepting, existential, and universal states. A configuration is accepting, if either (i) the current state is accepting, or (ii) the current state is existential and at least one successor configuration is accepting, or (iii) the current state is universal and every successor configuration is accepting. By [8], kEXPTIME is the set of all problems that can be accepted in space $\exp(k - 1, n^{O(1)})$ on an alternating Turing machine (for all $k \geq 1$).

THEOREM 4.3. *There is a fixed tree automatic structure $\mathscr{A}$ of bounded degree such that* FOTh($\mathscr{A}$) *is* 3EXPTIME*-hard.*

PROOF. Let M be a fixed *alternating* Turing machine with a space bound of $\exp(2, n)$ such that M accepts a 3EXPTIME-complete language. W.l.o.g. every configuration, where the current state is either existential or universal has exactly

two successor configurations. Let Σ, Γ, Q, and Ω have the same meaning as in the previous proof. Moreover, let $\Delta = \Omega \cup \{0, 1, \#_\exists, \#_\forall\}$.

The idea is that a binary tree x over the alphabet Δ can encode a computation tree for some input. Configurations can be encoded by linear chains over the alphabet $\Omega \cup \{0, 1\}$ as in the previous proof. The separator symbol $\#_\exists$ is used to separate an existential configuration from a successor configuration, whereas the separator symbol $\#_\forall$ is used to separate a universal configuration from its two successor configurations. Hence, a $\#_\exists$-labeled node has exactly one child, whereas a $\#_\forall$-labeled node has exactly two children. Checking whether the counters behave correctly can be done similarly to the previous proof by introducing binary relations σ_0 and σ_Ω, which rotate symbols within configurations. Remember that in our tree encoding, configurations are just long chains. Also the marking of some specific counter can be done in the same way as before. Finally, having marked some specific counter allows to check with a top-down tree automaton, whether the tree x represents indeed a valid computation tree. Of course, the tree automaton has to check whether the current configuration is existential or universal. In case of a universal configuration, the automaton branches at the next separator symbol $\#_\forall$. If e.g., the current configuration is universal but the next separator symbol is $\#_\exists$, then the automaton rejects the tree. ⊣

The proof of the next result is in fact a simplification of the proof of Theorem 4.2, since we do not need counters.

THEOREM 4.4. *There is a fixed string automatic structure $\mathscr{A}$ of bounded degree and polynomial growth (in fact linear growth) such that* FOTh($\mathscr{A}$) *is* EXPSPACE*-hard.*

PROOF. Let M be a fixed Turing machine with a space bound of 2^n such that M accepts an EXPSPACE-complete language. Let Σ, Γ, Q, q_0, q_f, $\Box$, and Ω have the usual meaning. Let $\Delta = \{\#\} \cup \Omega$. This time, for $m \in \mathbb{N}$, an *accepting m-computation* is a string $x_1 \# x_2 \# \dots x_n \#$, where $x_1, \dots, x_n \in \Gamma^* Q \Gamma^+$ are configurations with $|x_i| = m$ $(1 \le i \le n)$, $x_i \vdash_M x_{i+1}$ $(1 \le i < n)$, $x_1 \in q_0 \Sigma^* \Box^*$, and $x_n \in \Gamma^* q_f \Gamma^+$. Let U_0 be the fixed regular language

$$U_0 = (\Gamma^* Q \Gamma^+ \#)^+ \cap q_0 \Sigma^* \Box^* \# \Delta^* \cap \Delta^* q_f (\Delta \setminus \{\#\})^* \#.$$

The following binary relations δ and σ_Ω can be easily recognized by 2-dimensional automata:

$$\delta = \{(w, w \otimes w) \mid w \in U_0\},$$
$$\sigma_\Omega = \{(av \otimes w, va \otimes w) \mid w \in U_0, a \in \Omega, v \in \Delta^*\}.$$

Moreover, let U_1 be the following regular language over $\Delta^* \otimes \Delta^*$:

$$U_1 = \{\#u \otimes v\# \mid u, v \in \Omega^+, |u| = |v|, v \vdash_M u\}^+ \{\#u \otimes v\# \mid u, v \in \Omega^+, |u| = |v|\}.$$

Then, for every $x \in U_0$ and $m \in \mathbb{N}$ we have: x is an accepting m-computation if and only if there exist $y_1, y_2 \in \Delta^* \otimes \Delta^*$ such that $\delta(x, y_1)$, $\sigma_\Omega^m(y_1, y_2)$, and $y_2 \in U_1$.

Let us now fix some input $w = a_1 \dots a_n \in \Sigma^*$ with $|w| = n$, let $a_{n+1} = \Box$, and let $m = 2^n$. Thus, w is accepted by M if and only if there exists an accepting m-computation x such that in the first configuration of x, the tape content is of the form $w\Box^+$. It remains to add some structure that allows us to express the latter by a

formula. This can be done similarly to the proof of Theorem 4.2: Let $\Pi = \Delta \cup \{\triangleright\}$, where $\triangleright$ is a new symbol and define the binary relations ι_a $(a \in \Sigma \cup \{\square\})$ as follows:

$$\begin{aligned}\iota_a = &\{(q_0 av, q_0 a \triangleright v) \mid v \in \Delta^*, q_0 av \in U_0\} \\ &\cup \{(u \triangleright av, ua \triangleright v) \mid u, v \in \Delta^*, uav \in U_0\}.\end{aligned}$$

Then, $\mathscr{A} = (\Pi^* \cup (\Delta^* \otimes \Delta^*), \delta, \sigma_\Omega, (\iota_a)_{a \in \Sigma \cup \{\square\}}, U_0, U_1)$ is a fixed string automatic structure of bounded degree and linear growth. For the latter note that the Gaifman graph of $\mathscr{A}$ is just a disjoint union of cycles and finite paths (in fact, every node has degree at most 2). Moreover, w is accepted by M if and only if the following statement is true in $\mathscr{A}$:

$$\exists x \in U_0 \left(\begin{array}{l} \exists y_1, y_2 \left(\delta(x, y_1) \wedge \sigma_\Omega^m(y_1, y_2) \wedge y_2 \in U_1 \right) \\ \wedge \exists y_0, \ldots, y_n \left(\iota_{a_1}(x, y_0) \wedge \bigwedge_{i=1}^{n} \iota_{a_i}(y_{i-1}, y_i) \right) \end{array} \right). \tag{15}$$

By Lemma 4.1 this concludes the proof. ⊣

The next result can be easily shown by combining the techniques from the proofs of Theorems 4.3 and 4.4. We leave the details for the reader.

THEOREM 4.5. *There is a fixed tree automatic structure $\mathscr{A}$ of bounded degree and polynomial growth (in fact linear growth) such that* FOTh($\mathscr{A}$) *is* 2EXPTIME*-hard.*

§5. Bounded quantifier alternation depth. In this section we prove some facts about first-order fragments of fixed quantifier alternation depth. These results will follow easily from the constructions in the preceding section. Recall Theorem 2.7 and 2.9 on the complexity of Σ_n-FOMC(SA) and Σ_n-FOMC(TA), respectively. These results are not restricted to structures of bounded degree. From our construction in the proof of Theorem 4.4, we can slightly sharpen the lower bound from Theorem 2.7 for $n = 0$.

THEOREM 5.1. *There exists a fixed string automatic structure of bounded degree and linear growth with a* PSPACE*-complete Σ_1-theory.*

PROOF. Let M be a fixed linear bounded automaton with a PSPACE-complete acceptance problem and consider the structure $\mathscr{A}$ from the proof of Theorem 4.4. If we replace the number m in the formula (15) by $n+1$, where n is the input length, then (15) is equivalent to the following formula, which is equivalent to a Σ_1-formula:

$$\exists x \in U_0 \left(\begin{array}{l} \exists y_0, \ldots, y_{n+2} \left(\delta(x, y_0) \wedge \bigwedge_{i=0}^{n+1} \sigma_\Omega(y_i, y_{i+1}) \wedge y_{n+2} \in U_1 \right) \\ \wedge \exists y_1, \ldots, y_n \left(\iota_{a_1}(x, y_1) \wedge \bigwedge_{i=2}^{n} \iota_{a_i}(y_{i-1}, y_i) \right) \end{array} \right).$$

This formula is true in $\mathscr{A}$ if and only if the linear bounded automaton accepts the input $w = a_1 \ldots a_n$. ⊣

Let us now move on to Σ_2-formulas and structures of arbitrary growth:

THEOREM 5.2. *There is a fixed string automatic structure of bounded degree with an* EXPSPACE*-complete Σ_2-theory.*

PROOF. We reuse our construction from the proof of Theorem 4.2. We start with an $\exp(1,n)$-space-bounded machine M that accepts an EXPSPACE-complete language. We carry out the same construction as in the proof of Theorem 4.2, but replace 2^m (resp. m) everywhere by m (resp. the input length n). In addition, we need the following (trivial) analogue of Lemma 4.1: Let the signature $\mathcal{S}$ contain a binary symbol r. From a given number n (encoded unary), we can construct in linear time a Σ_1-formula $r^{(n)}(x,y)$ such that for every $\mathcal{S}$-structure $\mathcal{A}$ and all elements $a,b \in \mathcal{A}$ we have: $(a,b) \in r^n$ if and only if $\mathcal{A} \models r^{(n)}(a,b)$.

Then, the final formula from the proof of Theorem 4.2 can be written as

$$\exists x \in U_0 \left(\begin{array}{ll} & \exists y_1,y_2,y_3 \left(\delta(x,y_1) \wedge \sigma_0^{(n)}(y_1,y_2) \wedge \sigma_\Omega(y_2,y_3) \wedge y_3 \in U_1 \right) \\ \wedge & \forall y \left(\neg \mu^{(n)}(x,y) \vee y \in U_2 \right) \\ \wedge & \exists y_0,z_0,\ldots,y_{n+1},z_{n+1} \left(\begin{array}{l} \iota_{0,1}^{(n)}(x,y_0) \wedge \iota_{q_0}(y_0,z_0) \wedge \\ \bigwedge_{i=1}^{n+1} \iota_{0,1}^{(n)}(z_{i-1},y_i) \wedge \iota_{a_i}(y_i,z_i) \end{array} \right) \end{array} \right).$$

This formula is equivalent to a Σ_2-formula. Moreover, this formula is true in the string automatic structure $\mathcal{A}$ (of bounded degree) from the proof of Theorem 4.2, if and only if the input $w = a_1 a_2 \ldots a_n$ is accepted by the machine M. ⊣

As before, Theorems 5.1 and 5.2 can be extended to tree automatic structures as follows:

THEOREM 5.3. *The following holds*:

- *There is a fixed tree automatic structure of bounded degree and linear growth with an* EXPTIME*-complete* Σ_1*-theory.*
- *There is a fixed tree automatic structure of bounded degree with a* 2EXPTIME*-complete* Σ_2*-theory.*

§6. Open problems. The most obvious open question regards the uniform first-order theory for (injective) string and tree automatic structures: we do not know whether the upper bounds in Corollary 3.8(b–d) are sharp.

In [7, 23], it is shown that not only the first-order theory of every string automatic structure is (uniformly) decidable, but even its extension by the quantifiers "there are infinitely many x with ... " and "the number of x satisfying ... is divisible by p". In [26], we proved that this extended theory can be decided in triply exponential time for ω-string automatic structures of bounded degree. It is not clear whether the doubly-exponential upper bound proved in this paper extends to this more expressive theory.

Recall that there are tree automatic structures which are not string automatic. Provided 2EXPSPACE $\neq$ 3EXPTIME, our results on the non-uniform first-order theories imply the existence of such a structure of bounded degree (namely the tree automatic structure constructed in the proof of Theorem 4.3). But no example is known that does not rest on the complexity theoretic assumption 2EXPSPACE $\neq$ 3EXPTIME.

For $n \geq 3$, the precise complexity of the Σ_n-theory of a string/tree automatic structure of bounded degree remains open. We know that these theories belong to 2EXPSPACE for string automatic structures and to 3EXPTIME for tree automatic structures. Moreover, from our results for the Σ_2-fragment we obtain lower bounds of EXPSPACE and 2EXPTIME, respectively.

CONJECTURE 6.1. For every $n \geq 3$, the problems Σ_n-FOMC(SAb) and Σ_n-FOMC(TAb) belong to EXPSPACE and 2EXPTIME, respectively.

A possible attack to this conjecture would follow the line of argument in the proof of Theorem 3.7 and would therefore be based on Gaifman's theorem. To make this work, the exponential bound in Gaifman's theorem would have to be reduced which leads to the following conjecture.

CONJECTURE 6.2. Let $\mathscr{A}$ be a structure, $(a_1, \ldots, a_k), (b_1, \ldots, b_k) \in \mathscr{A}^k$, $d \geq n \geq 0$, and $D_1, \ldots, D_k \geq d \cdot 2^n$ such that

$$(\mathscr{A} \restriction (\bigcup_{i=1}^{k} S(D_i, a_i)),\ a_1, \ldots, a_k) \simeq (\mathscr{A} \restriction (\bigcup_{i=1}^{k} S(D_i, b_i)),\ b_1, \ldots, b_k).$$

Then, for every Σ_n-formula $\varphi(x_1, \ldots, x_k)$ of quantifier depth at most d, we have:

$$\mathscr{A} \models \varphi(a_1, \ldots, a_k) \iff \mathscr{A} \models \varphi(b_1, \ldots, b_k). \tag{16}$$

Gaifman's Theorem 3.1 implies that the conclusion (16) holds, if the lower bound $d \cdot 2^n$ for the radius is replaced by 2^d. The intuition behind this conjecture is that quantifier alternation (and not only nesting of quantifiers of the same type) seems to be essential in order to express exponential distances in the Gaifman-graph (see also the proof of Lemma 4.1).

REFERENCES

[1] V. BÁRÁNY, *Invariants of automatic presentations and semi-synchronous transductions*, ***Proceedings of STACS'06***, Lecture Notes in Computer Science, vol. 3884, Springer, 2006, pp. 289–300.

[2] ———, ***Automatic presentations of infinite structure***, Ph.D. thesis, RWTH Aachen, 2007.

[3] V. BÁRÁNY, E. GRÄDEL, and S. RUBIN, *Automata-based presentations of infinite structures*, ***Finite and algorithmic model theory*** (J. Esparza, C. Michaux, and C. Steinhorn, editors), London Mathematical Society Lecture Note Series, no. 379, Cambridge University Press, 2011, pp. 1–76.

[4] V. BÁRÁNY, Ł. KAISER, and S. RUBIN, *Cardinality and counting quantifiers on omega-automatic structures*, ***Proceedings of STACS'08*** (S. Albers and P. Weil, editors), IFIB Schloss Dagstuhl, 2008, pp. 385–396.

[5] M. BENEDIKT, L. LIBKIN, TH. SCHWENTICK, and L. SEGOUFIN, *Definable relations and first-order query languages over strings*, ***Journal of the ACM***, vol. 50 (2003), no. 5, pp. 694–751.

[6] A. BLUMENSATH, ***Automatic structures***, Diploma Thesis, RWTH Aachen, 1999.

[7] A. BLUMENSATH and E. GRÄDEL, *Automatic structures*, ***Proceedings of LICS'00***, IEEE Computer Society Press, 2000, pp. 51–62.

[8] A. K. CHANDRA, D. C. KOZEN, and L. J. STOCKMEYER, *Alternation*, ***Journal of the ACM***, vol. 28 (1981), no. 1, pp. 114–133.

[9] TH. COLCOMBET and CH. LÖDING, *Transforming structures by set interpretations*, ***Logical Methods in Computer Science***, vol. 3 (2007), pp. 1–36.

[10] H. COMON, M. DAUCHET, R. GILLERON, C. LÖDING, F. JACQUEMARD, D. LUGIEZ, S. TISON, and M. TOMMASI, *Tree automata techniques and applications*, available on `http://www.grappa.univ-lille3.fr/tata`, release October, 12th 2007, 2007.

[11] 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***, vol. 48 (1990), pp. 1–79.

[12] A. DAWAR, M. GROHE, ST. KREUTZER, and N. SCHWEIKARDT, *Model theory makes formulas large*, ***Proceedings of ICALP 2007*** (L. Arge, C. Cachin, T. Jurdzinski, and A. Tarlecki, editors), Lecture Notes in Computer Science, vol. 4596, Springer, 2007, pp. 913–924.

[13] CH. DELHOMMÉ, V. GORANKO, and T. KNAPIK, *Automatic linear orderings*, manuscript, 2003.

[14] C. C. ELGOT, *Decision problems of finite automata design and related arithmetics*, ***Transactions of the American Mathematical Society***, vol. 98 (1961), pp. 21–51.

[15] 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.

[16] H. GAIFMAN, *On local and nonlocal properties*, ***Logic colloquium '81*** (J. Stern, editor), North-Holland, 1982, pp. 105–135.

[17] G. HJORTH, B. KHOUSSAINOV, A. MONTALBÁN, and A. NIES, *From automatic structures to Borel structures*, ***Proceedings of LICS'08***, IEEE Computer Society, 2008, pp. 431–441.

[18] B. R. HODGSON, *On direct products of automaton decidable theories*, ***Theoretical Computer Science***, vol. 19 (1982), pp. 331–335.

[19] I. ISHIHARA, B. KHOUSSAINOV, and S. RUBIN, *Some results on automatic structures*, ***Proceedings of LICS'02***, IEEE Computer Society Press, 2002, pp. 235–244.

[20] B. KHOUSSAINOV and A. NERODE, *Automatic presentations of structures*, ***Logic and computational complexity*** (D. Leivant, editor), Lecture Notes in Computer Science, vol. 960, Springer, 1995, pp. 367–392.

[21] B. KHOUSSAINOV and S. RUBIN, *Graphs with automatic presentations over a unary alphabet*, ***Journal of Automata, Languages and Combinatorics***, vol. 6 (2001), pp. 467–480.

[22] B. KHOUSSAINOV, S. RUBIN, and F. STEPHAN, *On automatic partial orders*, ***Proceedings of LICS'03***, IEEE Computer Society Press, 2003, pp. 168–177.

[23] ———, *Definability and regularity in automatic structures*, ***Proceedings of STACS'04*** (V. Diekert and M. Habib, editors), Lecture Notes in Computer Science, vol. 2996, Springer, 2004, pp. 440–451.

[24] D. KUSKE, *Is Cantor's theorem automatic?*, ***Proceedings of LPAR'03*** (M. Y. Vardi and A. Voronkov, editors), Lecture Notes in Computer Science, vol. 2850, Springer, 2003, pp. 332–345.

[25] ———, *Theories of automatic structures and their complexity*, ***Proceedings of CAI'09*** (S. Bozapalidis and G. Rahonis, editors), Lecture Notes in Computer Science, vol. 5725, 2009, pp. 81–98.

[26] D. KUSKE and M. LOHREY, *First-order and counting theories of ω-automatic structures*, this JOURNAL, vol. 73 (2008), pp. 129–150.

[27] ———, *Some natural decision problems in automatic graphs*, this JOURNAL, vol. 75 (2010), no. 2, pp. 678–710.

[28] M. LOHREY, *Automatic structures of bounded degree*, ***Proceedings of LPAR'03*** (M. Y. Vardi and A. Voronkov, editors), Lecture Notes in Computer Science, vol. 2850, Springer, 2003, pp. 344–358.

[29] A. R. MEYER, *Weak monadic second order theory of one successor is not elementary recursive*, ***Logic colloquium*** (R. Parikh, editor), Lecture Notes in Mathematics, vol. 453, Springer, 1975, pp. 132–154.

[30] C. H. PAPADIMITRIOU, ***Computational complexity***, Addison Wesley, 1994.

[31] S. RUBIN, *Automata presenting structures: A survey of the finite string case*, ***The Bulletin of Symbolic Logic***, vol. 14 (2008), pp. 169–209.

[32] H. SEIDL, *Single-valuedness of tree transducers is decidable in polynomial time*, ***Theoretical Computer Science***, vol. 106 (1992), pp. 135–181.

[33] P. V. SILVA and B. STEINBERG, *A geometric characterization of automatic monoids*, ***Quarterly Journal of Mathematics***, vol. 55 (2004), pp. 333–356.

[34] A. WEBER, *On the valuedness of finite transducers*, ***Acta Informatica***, vol. 27 (1990), pp. 749–780.

[35] TH. WEIDNER, ***Die Größe injektiver baumautomatischer Darstellungen und die Komplexität ihrer Berechnung***, Diploma thesis, University of Leipzig, 2010, in German. English version in preparation.

TECHNISCHE UNIVERSITÄT ILMENAU, INSTITUT FÜR THEORETISCHE INFORMATIK
ILMENAU, GERMANY
E-mail: dietrich.kuske@tu-ilmenau.de

UNIVERSITÄT LEIPZIG, INSTITUT FÜR INFORMATIK
LEIPZIG, GERMANY
E-mail: lohrey@informatik.uni-leipzig.de