Automatic Structures: Richness and Limitations

Bakhadyr Khoussainov
Department of Computer Science
University of Auckland, New Zealand
bmk@cs.auckland.ac.nz

Andre Nies
Department of Computer Science
University of Auckland, New Zealand
andre@math.auckland.ac.nz

Sasha Rubin
Department of Mathematics
University of Auckland, New Zealand
rubin@math.auckland.ac.nz

Frank Stephan
National ICT Australia
Sydney Research Laboratory at Kensington
fstephan@cse.unsw.edu.au

Abstract

This paper studies the existence of automatic presentations for various algebraic structures. The automatic Boolean algebras are characterised, and it is proven that the free Abelian group of infinite rank and many Fraïssé limits do not have automatic presentations. In particular, the countably infinite random graph and the universal partial order do not have automatic presentations. Furthermore, no infinite integral domain is automatic. The second topic of the paper is the isomorphism problem. We prove that the complexity of the isomorphism problem for the class of all automatic structures is Σ_1^1-complete.

1 Introduction

Classes of infinite structures with nice algorithmic properties (such as decidable model checking) are of increasing interest in a variety of fields of computer science. One is in the theory of infinite state transition systems and the questions of their symbolic representations, model checking, specification and verification. Another is that of extending the framework of finite model theory to infinite models that have finite presentations. Also, string query languages in databases may be captured by (decidable) infinite string structures. Automatic structures are (usually) infinite relational structures whose domain and atomic relations can be recognised by finite automata operating synchronously on their input. Consequently, automatic structures have finite presentations and are closed under first order interpretability (as well as some of extensions of first order). Moreover, the model checking problem for automatic structures is decidable. Hence automatic structures and tools developed for their study are well suited to these fields of computer science, see for instance [1].

From a computability and logical point of view, automatic structures are used to provide generic examples of structures with decidable theories, to investigate the relationship between automata and definability, and to refine the ideas and approaches in the theory of computable structures. This paper investigates the problem of characterising automatic structures in algebraic, model theoretic or logical terms.

This paper addresses two foundational problems in the theory of automatic structures. The first is that of providing *structure theorems* for classes of automatic structures. Fix a class $\mathcal{C}$ of structures, closed under isomorphism. For instance, $\mathcal{C}$ may be the class of groups or linear orders. A structure theorem should be able to distinguish whether a given member of $\mathcal{C}$ has an automatic presentation or not; a special case of this is to know whether a given structure is automatically presentable or not. This usually concerns the interactions between the combinatorics of finite automata presenting structures and properties of the structures themselves. The second problem, which is related to the first, is the complexity of the *isomorphism problem* for classes of automatic structures. Namely fix a class of automatic structures $\mathcal{C}$. Given automatic presentations of two structures from $\mathcal{C}$, are the structures isomorphic ?

With regard to the first problem, we provide new techniques for proving that some foundational structures in computer science and mathematics do not have automatic presentations. For example, we show that the

Fraïssé limits of many classes of finite structures such as finite partial orders or graphs do not have automatic presentations. This shows that the infinite random graph and universal partial order do not have automatic presentations. The idea is that the finite amount of memory intrinsic to finite automata presenting the structure can be used to extract algebraic and model theoretic properties (invariants) of the structure, and so used to classify such structures up to isomorphism. This line of research has indeed been successful in investigating automatic ordinals, linear orders, trees and Boolean algebras. For example we know a full structure theorem for the automatically presentable ordinals; namely, they are those strictly less than ω^ω [5]. We have partial structure theorems saying that automatic linear orders and automatic trees have finite Cantor-Bendixson rank [12]. In this paper we provide a structure theorem for the (infinite) automatic Boolean algebras; namely, they are those isomorphic to finite products of the Boolean algebra of finite and co-finite subsets of $\mathbb{N}$.

With regard to the second problem, it is not surprising that the isomorphism problem for the class of all automatic structures in undecidable [4]. The reason for the undecidability is that the configuration space of a Turing machine considered as a graph is an automatic structure, and the reachability problem in the configuration space is undecidable. Thus with some extra work as in [3] or [10] one can reduce the reachability problem to the isomorphism problem for automatic structures. In addition, the isomorphism problem for automatic ordinals [12] and Boolean algebras (Corollary 3.5) is decidable, for equivalence structures is Π^0_1, and for configuration spaces of Turing machines is Π^0_3-complete [13].

Hence it is somewhat unexpected that the complexity of the isomorphism problem for the class of automatic structures is Σ^1_1-complete. The Σ^1_1-completeness is proved by reducing the isomorphism problem for computable trees, known to be Σ^1_1-complete [8], to the isomorphism problem for automatic structures.

The two problems are related in the following way. If one has a 'nice' structure theorem for a class $\mathcal{C}$ of automatic structures, then one expects that the isomorphism problem for $\mathcal{C}$ be computationally 'reasonable'. For instance, as corollaries of the structure theorems for automatic ordinals and for automatic Boolean algebras, one obtains that their corresponding isomorphism problems are decidable. In contrast, the Σ^1_1-completeness of the isomorphism problem of the class of all automatic structures tells us that the language of first order arithmetic is not powerful enough to give a structure theorem for the class of all automatic structures. In other words we should not expect a 'nice' structure theorem for the class of all automatic structures.

Consequently the following approaches to studying automatic structures suggest themselves. Classify classes $\mathcal{C}$ of automatic structures in terms of the complexity of the isomorphism problem for $\mathcal{C}$. Find structure theorems for automatic structures with restricted presentations that guarantee the decidability (or more generally Σ^0_n-completeness) of the isomorphism problem. For instance, one may restrict the class of automatic graphs to those for which the reachability relation is regular, or decidable. These are not pursued in this paper.

Here is an outline of the rest of the paper. The next section is a brief introduction to the basic definitions. Section 3 provides some counting techniques sufficient to prove non-automaticity of many classical structures such as fields, integral domains and Boolean algebras. Section 4 provides a technique that is used to show non-automaticity of several structures such as the infinite random graph and the universal partial order. The last section is devoted to proving that the isomorphism problem for automatic structures is Σ^1_1-complete. All unproved results are proved in the complete version of this paper.

Finally, we note that this paper does not deal with the tree automatic structures, a natural generalisation of structures presented by finite automata. Some of the techniques and results of this paper can be applied or extended to tree automatic structures in a natural way. In addition, techniques developed and results obtained in the study of structures presented by finite automata give rise to better understanding and deeper development of tree automatic structures.

2 Preliminaries

A thorough introduction to automatic structures can be found in [3] and [11]. We assume familiarity with the basics of finite automata theory though to fix notation the necessary definitions are included. A *finite automaton* $\mathcal{A}$ over an alphabet Σ is a tuple (S, ι, Δ, F), where S is a finite set of *states*, $\iota \in S$ is the *initial state*, $\Delta \subset S \times \Sigma \times S$ is the *transition table* and $F \subset S$ is the set of *final states.* A *computation* of $\mathcal{A}$ on a word $\sigma_1\sigma_2 \dots \sigma_n$ ($\sigma_i \in \Sigma$) is a sequence of states, say $q_0, q_1, \dots, q_n$, such that $q_0 = \iota$ and $(q_i, \sigma_{i+1}, q_{i+1}) \in \Delta$ for all $i \in \{0, 1, \dots, n-1\}$. If $q_n \in F$, then the computation is *successful* and we say that automaton $\mathcal{A}$ *accepts* the word. The *language* accepted by the automaton $\mathcal{A}$ is the set of all words accepted by $\mathcal{A}$. In general, $D \subset \Sigma^\star$ is *finite automaton recognisable*, or *regular*, if D is the language accepted by a finite automaton $\mathcal{A}$.

The following definitions extends recognisability to relations of arity n, called *synchronous n–tape automata*. A synchronous n–tape automaton can be thought of as a one-way Turing machine with n input tapes [7]. Each tape is regarded as semi-infinite having written on it a word in the alphabet Σ followed by an infinite succession of blanks, $\diamond$ symbols. The automaton starts in the initial state, reads simultaneously the first symbol of each tape, changes state, reads simultaneously the second symbol of each tape, changes state, etc., until it reads a blank on each tape. The automaton then stops and accepts the n–tuple of words if it is in a final state. The set of all n–tuples accepted by the automaton is the relation recognised by the automaton. Here is a definition.

Definition 2.1 *Write $\Sigma_\diamond$ for $\Sigma \cup \{\diamond\}$ where $\diamond$ is a symbol not in Σ. The* convolution *of a tuple $(w_1, \cdots, w_n) \in \Sigma^{\star n}$ is the string $\otimes(w_1, \cdots, w_n)$ of length $\max_i |w_i|$ over alphabet $(\Sigma_\diamond)^n$ defined as follows. Its k'th symbol is $(\sigma_1, \ldots, \sigma_n)$ where σ_i is the k'th symbol of w_i if $k \le |w_i|$ and $\diamond$ otherwise.*

The convolution *of a relation $R \subset \Sigma^{\star n}$ is the relation $\otimes R \subset (\Sigma_\diamond)^{n\star}$ formed as the set of convolutions of all the tuples in R. That is $\otimes R = \{\otimes w \mid w \in R\}$.*

Definition 2.2 An n–tape automaton *on Σ is a finite automaton over the alphabet $(\Sigma_\diamond)^n$. An n–ary relation $R \subset \Sigma^{\star n}$ is* finite automaton recognisable *(in short FA recognisable) or* regular *if its convolution $\otimes R$ is recognisable by an n–tape automaton.*

We now relate n–tape automata to structures. A *structure $\mathcal{A}$* consists of a countable set A called the *domain* and some relations and operations on A. We may assume that $\mathcal{A}$ only contains relational predicates as the operations can be replaced with their graphs. We write $\mathcal{A} = (A, R_1^A, \ldots, R_k^A, \ldots)$ where R_i^A is an n_i–ary relation on $\mathcal{A}$. The relation R_i are sometimes called basic or atomic relations. We assume that the function $i \to n_i$ is always a computable one.

Definition 2.3 *A structure $\mathcal{A}$ is* automatic *over Σ if its domain $A \subset \Sigma^\star$ is finite automata recognisable, and there is an algorithm that for each i produces a finite automaton recognising the relation $R_i^A \subset (\Sigma^\star)^{n_i}$. A structure is called* automatic *if it is automatic over some alphabet. If $\mathcal{B}$ is isomorphic to an automatic structure $\mathcal{A}$, then call $\mathcal{A}$ an* automatic presentation *of $\mathcal{B}$ and say that $\mathcal{B}$ is called* automatically presentable *(over Σ).*

An example of an automatic structure is the word structure $(\{0,1\}^\star, L, R, E, \preceq)$, where for all $x, y \in \{0,1\}^\star$, $L(x) = x0$, $R(x) = x1$, $E(x,y)$ iff $|x| = |y|$, and $\preceq$ is the lexicographical order. The configuration graph of any Turing machine is another example of an automatic structure. Examples of automatically presentable structures are $(\mathbb{N}, +)$, $(\mathbb{N}, \le)$, $(\mathbb{N}, S)$, the group $(\mathbb{Z}, +)$, the order on the rationals $(Q, \le)$, and the Boolean algebra of finite or co-finite subsets of $\mathbb{N}$. Note that every finite structure is automatically presentable. We use the following important theorem without reference.

Theorem 2.4 [11] *Let $\mathcal{A}$ be an automatic structure. There exists an algorithm that from a first order definition ϕ in $\mathcal{A}$ of a relation R produces an automaton recognising R.*

3 Proving Non-Automaticity via Counting

The first technique for proving non-automaticity was presented in [11] and later generalised in [3]. The technique is based on a pumping argument and exhibits the interplay between finitely generated (sub) algebras and finite automata. We briefly recall the technique for completeness.

A relation $R \subset A^{k+l}$ is called *locally finite* if for every $\bar{a}$ (of size k) there are at most a finite number of $\bar{b}$ (of size l) such that $(\bar{a}, \bar{b}) \in R$. For $\bar{b} = (b_1, \cdots, b_m)$, write $b \in \bar{b}$ to mean $b = b_i$ for some i.

We start with the following elementary but important lemma that will later be used without reference to it.

Lemma 3.1 *Suppose that a relation $R \subset A^{k+l}$ is locally finite and FA recognisable. There exists a constant p such that for every $(\overline{x}, \overline{y}) \in R$ it is the case that*

$$\max\{|y| \mid y \in \overline{y}\} - \max\{|x| \mid x \in \overline{x}\} \le p.$$

Assume $\mathcal{A}$ is an automatic structure in which each atomic relation R_i is a graph of a function f_i, $i = 1, \ldots, n$. Let $a_1, a_2, \ldots$ be a sequence of some elements of A such that the relation $\{(a_i, a_j) \mid i \le j\}$ is regular. Consider the sequence $G_1 = \{a_1\}$, $G_{n+1} = G_n \cup \{a_{n+1}\} \cup \{f_i(\bar{a}) \mid \bar{a} \in G_n, i = 1, \ldots, n\}$. By the lemma above there is a constant C such that the length of all elements in G_n is bounded by $C \cdot n$. Therefore the number of elements in G_n is bounded by $2^{O(n)}$. Some combinatorial reasoning combined with this observation can now be applied to provide examples of structures with no automatic presentations, see [3] and

[11]. For example, these include: (a) The free group on $k > 1$ generators; (b) The structure $(\mathbb{N}, |)$; (c) The structure $(\mathbb{N}, p)$, where $p : \mathbb{N}^2 \to \mathbb{N}$ is a pairing function; (d) The term algebra generated by finitely many constants with at least one non-unary atomic operation[1]. Note all these structures have decidable first order theory.

In the next sections we provide other more intricate techniques for showing that particular structures do not have automatic presentations. We then apply those techniques to give a characterisation of Boolean algebras that have automatic presentations. We also prove that $(\mathbb{Q}^+, \times)$ has no automatic presentation and show that no infinite integral domain (in particular no infinite field) has an automatic presentation. We also study automaticity of some Fraïssé limits.

3.1 Automatic Boolean algebras

All finite Boolean algebras are automatic. Thus, in this section we deal with infinite countable Boolean algebras only. Our goal in this section is to give a full characterisation of infinite automatic Boolean algebras. Our characterisation can then be applied to show that the isomorphism problem for automatic Boolean algebras is decidable. Compare this with the result from computable algebra that the isomorphism problem for computable Boolean algebras is Σ_1^1-complete [8].

Recall that a Boolean algebra $\mathcal{B} = (B, \cup, \cap, \setminus, \mathbf{0}, \mathbf{1})$ is a structure, where $\cap$ and $\cup$ and $\setminus$ operations satisfy all the basic properties of the set-theoretic intersection, union, and complementation operations; In $\mathcal{B}$ the relation $a \subseteq b \iff a \cap b = a$ is a partial order in which $\mathbf{0}$ is the smallest element, and $\mathbf{1}$ is the greatest element. The complement of an element $b \in B$ is $\mathbf{1} \setminus b$ and is denoted by $\bar{b}$.

The following lemma is useful.

Lemma 3.2 *Suppose $\mathcal{B}$ is an automatic boolean algebra and let $n \in \mathbb{N}$. There exists a constant $e \in \mathbb{N}$ such that for every finite set S of n elements of B, the length of the element $\cup S$ is at most*

$$\max\{|s|\} + e \log n$$

where the maximum varies over $s \in S$.

[1]Thus, elements of the term algebra are all the ground terms, and the operations are defined in a natural way: the value of a function f of arity n from the language on ground terms $g_1, \dots, g_n$ is $f(g_1, \dots, g_n)$.

Proof We calculate a bound on $|\cup S|$ by considering a 'computation tree' computing $\cup S$. Let h be the least integer greater than or equal to $\log n$. Denote by B_h the full binary tree of height h. It has at least n leaves. Label the nodes of B_h as follows. The leaves are labelled by elements of S in such a way that every element of S is the label of at least one leaf. Suppose we have labelled all the elements on i'th level of the tree, $x_1, \cdots, x_k$, identifying the nodes of the trees with their labels. Label the $i-1$'st level as follows. For every odd $j < k$, label the parent of x_j and x_{j+1} by $x_j \cup x_{j+1}$. This completes the labelling of the tree B_h. Note that the label of the root of the tree is $\cup S$. By Lemma 3.1 there is a constant d such that for every $a, b \in B$ it holds that $|a \cup b| \le \max |a|, |b| + d$. Then by induction if $x \in B$ is a label of a node on the i'th level of the tree B_n then $|x| \le \max |s| + d(h - i)$. Setting $i = 0$ and noting that $h \le 1 + \log n$ it holds that $|\cup S| \le \max |s| + (d+1) \log n$, as required. □

A linearly ordered set determines a Boolean algebra in a natural way described as follows. Let $\mathcal{L} = (L, \le)$ be a linearly ordered set. An interval is a subset of L of the form $[a, b) = \{x \mid a \le x < b\}$, where $a, b \in L \cup \{\infty\}$. The **interval algebra** denoted by $\mathcal{B}_{\mathcal{L}}$ is the collection of all finite unions of intervals of $\mathcal{L}$, with the usual set-theoretic operations of intersection, union and complementation. Every interval algebra is a Boolean algebra. Moreover for every countable Boolean algebra $\mathcal{A}$ there exists an interval algebra $\mathcal{B}_{\mathcal{L}}$ isomorphic to $\mathcal{A}$. We write $\mathcal{L}_1 \times \mathcal{L}_2$ for the ordered sum $\sum_{l \in \mathcal{L}_1} \mathcal{L}_2$. The proof of the following lemma is straightforward.

Lemma 3.3 *The interval Boolean algebras $\mathcal{B}_{i \times \omega}$, where i is positive integer, all have automatic presentations.*

An **atom** in a Boolean algebra is a non-zero element a such that for every $b \le a$ we have $a = b$ or $b = \mathbf{0}$. Assume that $\mathcal{B}$ is an automatic Boolean algebra not isomorphic to any of the algebras $\mathcal{B}_{i \times \omega}$. Call two elements $a, b \in B$ F**-equivalent** if the element $(a \cap \bar{b}) \cup (b \cap \bar{a})$ is a union of finitely many atoms. Factorise $\mathcal{B}$ with respect to the equivalence relation. Denote the factor algebra by $\mathcal{B}/F$. Due to the assumption on $\mathcal{B}$ the algebra $\mathcal{B}/F$ is not finite. Call x in $\mathcal{B}$ **large** if its image in $\mathcal{B}/F$ is not a finite union of atoms or $\mathbf{0}$. For example the element $\mathbf{1}$ is large in $\mathcal{B}$ because $\mathcal{B}$ is not isomorphic to $\mathcal{B}_{i \times \omega}$. Call an element x in $\mathcal{B}$ **infinite** if there are infinitely many elements below it. Say that x **splits** y, for $x, y \in B$, if $x \cap y \ne \mathbf{0}$ and $\bar{x} \cap y \ne \mathbf{0}$. For every large element $l \in B$ there exists an element $x \in B$ that splits l such that $x \cap l$ is large and $\bar{x} \cap l$ is infinite. Also for every infinite element $i \in B$ there exists an element $x \in B$ that splits i such that either $x \cap i$ or $\bar{x} \cap i$ is infinite.

Now we construct a sequence T_n of trees and elements $a_\sigma \in B$ corresponding to elements $\sigma \in T_n$ as follows. The tree T_n will be a set of binary strings closed under prefixes. Therefore it suffices to define leaves of T_n. Initially, we set $T_0 = \{\lambda\}$ and $a_\lambda = \mathbf{1}$. Assume that T_n has been constructed. By induction hypothesis we may assume that the leaves of T_n satisfy the following properties: (1) There exists at least one leaf σ such that a_σ is large in $\mathcal{B}$. Call the element a_σ leading in T_n. (2) There exist n leaves $\sigma_1, \ldots, \sigma_n$ such that each a_{σ_i} is an infinite element in $\mathcal{B}$. Call these elements sub-leading elements. (3) The number of leaves in T_n is greater than or equal to $n \cdot (n+1)/2$. (4) For every pair of leaves x, y of T_n it holds that $x \cap y = \mathbf{0}$.

For each leaf $\sigma \in T_n$ proceed as follows:

1. If σ is leading then find the first length lexicographical b that splits a_σ such that both $a_\sigma \cap b$ and $a_\sigma \cap \bar{b}$ are infinite and one of them is large. Put $\sigma 0$ and $\sigma 1$ into T_{n+1}. Set $a_{\sigma 0} = a_\sigma \cap b$ and $a_{\sigma 1} = a_\sigma \cap \bar{b}$.

2. If σ is a sub-leading then find the first length lexicographical b that splits a_σ such that one of $a_\sigma \cap b$ or $a_\sigma \cap \bar{b}$ is infinite. Put $\sigma 0$ and $\sigma 1$ into T_{n+1}. Set $a_{\sigma 0} = a_\sigma \cap b$ and $a_{\sigma 1} = a_\sigma \cap \bar{b}$.

Thus, we have constructed the tree T_{n+1} and elements a_σ corresponding to the leaves of the tree. Note that the inductive hypothesis holds for T_{n+1}. This completes the definition of the trees.

There exists a constant C_1 such that $|a_{\sigma\epsilon}| \leq |a_\sigma| + C_1$ for all defined elements a_σ . Now for every n consider the set $X_n = \{a_\sigma \mid \sigma \text{ is a leaf of } T_n\}$. There exists a constant C_2 such that for all $x \in X_n$ we have $|x| \leq C_2 \cdot n$. Therefore $X_n \subset \Sigma^{C_2 \cdot n}$ and the number of leaves in T_n is greater than or equal to $n \cdot (n+1)/2$. Now for every pair of elements a, b in X_n we have $a \cap b = \mathbf{0}$. Therefore the number of elements of the Boolean algebra generated by the elements in X_n is $2^{|X_n|}$. Now let $Y = \{b_1, \ldots, b_k\} \subset X_n$. Consider the element $\cup Y = b_1 \cup \ldots \cup b_k$. By Lemma 3.2 there exists a constant C_3 such that $|\cup Y| \leq C_3 \cdot n$. This gives us a contradiction because the number of elements generated by elements of X_n clearly exceeds the cardinality of $\Sigma^{O(n)}$.

Thus, we have proved the following theorem characterising infinite automatic Boolean algebras.

Theorem 3.4 *A Boolean algebra has an automatic presentation if and only if it is isomorphic to $\mathcal{B}_\alpha$ for some ordinal $\alpha < \omega^2$.*

The proof of the following is straightforward.

Corollary 3.5 *It is decidable whether two automatic Boolean algebras are isomorphic.*

3.2 Commutative monoids and Abelian groups

We prove that $(\mathbb{Q}^+, \times)$, or equivalently, the free Abelian group of rank ω, is not automatic. The following generalises Lemma 3.2 and is proved the same way.

Lemma 3.6 *Suppose $(M, \times)$ is an automatic monoid. There exists a constant $k \in \mathbb{N}$ such that for every $n > 0$ and every sequence $s_1, \ldots, s_n \in M$, the length of the element $\prod_i s_i$ is at most*

$$\max_i\{|s_i|\} + k \log n.$$

Theorem 3.7 *Let $(M, \times)$ be a monoid containing $(\mathbb{N}, \times)$ as a submonoid. Then $(M, \times)$ is not automatic.*

Proof Assume for a contradiction that an automatic presentation of M is given. Let $a_0, a_1, \ldots$ be the prime numbers, viewed as elements of M, and listed in length-lexicographical order (with respect to this presentation of M). Let r_n be such that $a_0, \ldots, a_{r_n - 1}$ are the primes of length at most n. Let

$$F_n = \{\textstyle\prod_{i:0 \leq i < r_n} a_i^{\beta_i} : 0 \leq \beta_i < 2^n\}.$$

By Lemma 3.6, each term $a_i^{\beta_i}$ has length at most $n + k \log \beta_i \leq n(1+k)$. Again by the lemma, each product has length at most $n(1+k) + k \log r_n$. Since all the products are distinct,

$$2^{nr_n} \leq |F_n| \leq |\Sigma|^{(1+k)n + k \log r_n}.$$

Thus $nr_n \leq \log|\Sigma|[(1+k)n + k \log r_n]$ and $r_n \in O(\log r_n)/n$, a contradiction because r_n goes to infinity. □

Corollary 3.8 $(\mathbb{Q}^+, \times)$ *is not automatically presentable.*

Corollary 3.9 *The monoid $(M_k(\mathbb{N}), \cdot)$, where $M_k(\mathbb{N})$ is the set of $k \times k$ matrices with entries from $\mathbb{N}$, is not automatically presentable.*

For a prime p, let $\mathbb{Q}_p$ be the additive group of rationals of the form z/p^m, z an integer, $m \in \mathbb{N}$, and let C_p be the Prüfer group $\mathbb{Q}_p/\mathbb{Z}$. Using representations to base p, it is easy to give automatic presentations of these groups. Hence finite direct sums of those groups are also automatic. The proof of the following uses similar methods.

Theorem 3.10 *Let $A^{(\omega)}$ denote the direct sum of infinitely many copies of the group A. The infinite direct sums $\mathbb{Q}_p^{(\omega)}$ and $C_p^{(\omega)}$ are not automatically presentable.*

3.3 Integral domains

In our next result we prove that no infinite integral domain is automatically presentable. The following definition and lemma will be used in the next section as well.

Definition 3.11 *Suppose $\mathcal{D}$ is a structure over alphabet Σ. Write D_n for $D \cap \Sigma^{\leq n}$; that is the elements of D of length at most n. Write S_n for $\{x \in \Sigma^n \mid \exists z \in \Sigma^\star \wedge xz \in D\}$; that is prefixes of length n of words in the domain.*

Lemma 3.12 *If $D \subset \Sigma^\star$ is a regular language then*

1. *$|S_n| \in O(|D_n|)$ and*
2. *$|D_{n+k}| \in \Theta(|D_n|)$ for every constant $k \in \mathbb{N}$.*

Proof Suppose the automaton recognising D has c states. Then for $x \in S_n$ there exists $z \in \Sigma^\star$ with $|z| \leq c$ such that $xz \in D$ (†). If $n \geq c$ then $|S_n| \leq |\Sigma|^c \times |S_{n-c}|$ since the map associating $x \in S_n$ with the word consisting of the first $n-c$ letters of x, is $|\Sigma|^c$-to-one. But by using (†) we see that $|S_{n-c}| \leq |D_n|$. So $|S_n| \leq |\Sigma|^c \times |D_n|$. This completes the first part.

Fix $k \in \mathbb{N}$. The mapping associating $x \in D_{n+k}$ to the prefix of x of length n is $|\Sigma|^k$-to-one. Hence $|D_{n+k}| \leq |\Sigma|^k \times |S_n| \leq |\Sigma|^k \times |\Sigma|^c \times |D_n|$. Since $D_n \subset D_{n+k}$ one has that $|D_n| \leq |D_{n+k}|$ and $|D_{n+k}| \in O(|D_n|)$. This completes the second part. □

Recall that an integral domain $(D, +, \cdot, 0, 1)$ is a commutative ring with identity such that $x \cdot y = 0$ only if $x = 0$ or $y = 0$. For example every field is an integral domain.

Theorem 3.13 *No infinite integral domain is automatically presentable.*

Proof Suppose that $(D, +, \cdot, 0, 1)$ is an infinite automatic integral domain. For each $n \in \mathbb{N}$ recall that $D_n = \{u \in D \mid |u| \leq n\}$. We claim that there exists an x in D such that for all $a, b, a', b' \in D_n$ the condition $a \cdot x + b = a' \cdot x + b'$ implies that $a = a'$ and $b = b'$. We say such x **separates** D_n. Indeed, assume that such an x does not exist. Then for each $x \in D$ there exist $a, b, a', b' \in D_n$ such that $a \cdot x + b = a' \cdot x + b'$ but $(a, b) \neq (a', b')$. Hence, since D_n is finite, there exist $a, b, a', b' \in D_n$ such that $a \cdot x + b = a' \cdot x + b'$ but $(a, b) \neq (a', b')$ for infinitely many x. Thus, for infinitely many x we have $(a - a') \cdot x = b' - b$. But $a \neq a'$ for otherwise also $b = b'$, contrary to assumption. Also there exist distinct x_1 and x_2 such that $(a - a') \cdot x_1 = (a - a') \cdot x_2$. Since D is an integral domain we conclude that $x_1 = x_2$ which is a contradiction.

For each D_n we can select the length-lexicographically first x_n separating D_n. Now the set $E_n = \{y \mid \exists a, b \in D_n\, [y = ax_n + b]\}$ has at least $|D_n|^2$ many elements. However by Lemma 3.1 there exists a constant C such that $E_n \subset D_{n+C}$, and by Lemma 3.12 the number of elements in D_{n+C} is in $O(|D_n|)$. Thus, we have a contradiction. The theorem is proved. □

Corollary 3.14 *No infinite field is automatically presentable.*

4 Non-automaticity of some Fraïssé Limits

In this section we provide some methods for proving non-automaticity of structures. These methods are then applied to prove that some Fraïssé limits do not have automatic presentation. We already know that the atomless Boolean algebra does not have an automatic copy. Below are examples of Fraïssé limits that have automatic presentations.

Example 4.1 *The linear order of rational numbers has an automatic presentation. In fact it is straightforward to check that $(\{0,1\}^\star \cdot 1, \preceq_{lex})$ is an automatic presentation of $(Q, \leq)$.*

Example 4.2 *Let $U = \{u \mid u \in \{0,1\}^\star \cdot 1,\ |u| \text{ is even}\}$. The structure $(\{0,1\}^\star \cdot 1; \preceq_{lex}, U)$ is the Fraïssé limit for the class K of all finite linear orders with one unary predicate.*

Let $\mathcal{A}$ be an automatic structure over the alphabet Σ. The **standard approximation** of this structure is the sequence $A_0 \subseteq A_1 \subseteq A_2 \subseteq \ldots$ such that $A_n = \{v \in A \mid |v| \leq n\}$. Let $\Phi(x, y)$ be a two variable formula in the language of this structure. We do not exclude that $\Phi(x, y)$ has a finite number of parameters from the domain of the structure. Now for each $y \in A$ and $n \in \mathbb{N}$ we define the following function $c_{n,y} : A_n \to \{0, 1\}$:

$$c_{n,y}(x) = \begin{cases} 1 & \text{if } \mathcal{A} \models \Phi(x, y); \\ 0 & \text{if } \mathcal{A} \models \neg\Phi(x, y). \end{cases}$$

In the next theorem we count the number of functions $c_{n,y}$ (which clearly depends on the given formula Φ) using the fact that $\mathcal{A}$ is an automatic structure.

Theorem 4.3 *If $\mathcal{A}$ is automatic, then the number of functions $c_{n,y}$ is in $O(|A_n|)$.*

Proof Let (Q, ι, ρ, F) be a deterministic automaton recognising the relation $\otimes\Phi = \{\otimes(x, y) \mid \mathcal{A} \models \Phi(x, y)\}$. Fix $n \in \mathbb{N}$. We need to find a constant k such that for all $y \in A$, the number of functions $c_{n,y}$ is at most $k|A_n|$. For $y \in A$ define the function $T_y : A_n \to Q$ and the subset Q_y of Q as follows. Let $\otimes(x, y)$ be $\sigma_1 \ldots \sigma_k$. Then:

$$T_y(x) = \begin{cases} \rho(\iota, \otimes(x, y)) & \text{if} \quad k \le n; \\ \rho(\iota, \sigma_1 \cdots \sigma_n) & \text{if} \quad k > n. \end{cases}$$

Thus, $T_y(x)$ is the state resulted by reading the whole input (x, y) if $|y| \le n$, and otherwise $T_y(x)$ is the state obtained by reading the first n symbols of the input $\otimes(x, y)$. The set Q_y is defined in a similar manner as follows. If $k \le n$ then $Q_y = \{\rho(\iota, \otimes(x, y)) \mid x \in A_n \ \& \ \rho(\iota, \otimes(x, y)) \in F\}$; If $k > n$ then $Q_y = \{s \mid \rho(s, \sigma_{n+1} \cdots \sigma_k) \in F\}$.

Note that $\sigma_{n+1} \cdots \sigma_k = \otimes(\epsilon, z)$ for some $z \in \Sigma^*$ in case $k > n$. We claim that if $(T_{y_1}, Q_{y_1}) = (T_{y_2}, Q_{y_2})$ then $c_{n,y_1} = c_{n,y_2}$. Assume that $c_{n,y_1}(x) \ne c_{n,y_2}(x)$ for some $x \in A_n$. So without loss of generality we may assume that $\mathcal{A} \models \Phi(x, y_1)$ and $\mathcal{A} \models \neg\Phi(x, y_2)$. If $T_{y_1}(x) \ne T_{y_2}(x)$ then we are done. Otherwise there exists a state q which is in Q_{y_1} but not in Q_{y_2}. This contradicts the assumption that $(T_{y_1}, Q_{y_1}) = (T_{y_2}, Q_{y_2})$. Thus, the number of functions of type $c_{n,y}$ is bounded by the number of pairs of type (T_y, Q_y).

Recall the set $S_n = \{v \in \Sigma^n \mid \exists z(z \in \Sigma^* \wedge xz \in A)\}$. If $y \in A$ and $|y| > n$ then there exist v and z such that v is a prefix of y, $v \in S_n$ and $y = vz$. Moreover, $T_y = T_v$. Therefore, the number of pairs of type (T_y, Q_y) is bounded by $(|A_n| + |S_n|) \times 2^{|Q|}$. However by Lemma 3.12, $|S_n| \in O(|A_n|)$. So, finally we get that for $n \ge m$, the number of functions of type $c_{n,y}$ is bounded by the number of pairs of type (T_y, Q_y) which is bounded by

$$(|A_n| + |S_n|) \times 2^{|Q|} \in O(|A_n|) \times 2^{|Q|}.$$

We have proved the theorem. □

Now we give several applications of this theorem. First we apply the theorem to random graphs. An explicit description of a random graph is the following. The set V of vertices is $\mathbb{N}$. The set of edges is defined as follows. Put an edge between n and m if in the binary representation of n, the term 2^m has coefficient 1.

Corollary 4.4 independently [6] *The random graph has no automatic presentation.*

Proof The random graph has the following algebraic property: For every disjoint partition X_1, X_2 of every finite set Y of vertices there exists a vertex y such that $(y, x_1) \in E$ and $(y, x_2) \notin E$ for all $x_1 \in X_1$ and $x_2 \in X_2$.

Let (A, E) be an automatic presentation of the random graph. Consider the standard approximation $A_0, A_1, A_2, \ldots$ of it. For every partition X_1, X_2 of set A_n of vertices there exists a vertex y such that $(x_1, y) \in E$ and $(x_2, y) \notin E$ for all $x_1 \in X_1$ and $x_2 \in X_2$. Hence, for every n, we have $2^{|A_n|}$ number of functions of type $c_{n,y}$, contradicting Theorem 4.3. Hence the random graph has no automatic presentation. □

Corollary 4.5 *Let $\mathcal{A}$ be a random structure of a signature L, where L contains at least one non-unary symbol. Then $\mathcal{A}$ does not have an automatic presentation.*

Proof Let R be a non-unary predicate of L of arity k. Consider the following formula $E(x, y) = R(x, y, a_1, \ldots, a_{k-2})$, where $a_1, \ldots, a_{k-2}$ are fixed constants from the domain of $\mathcal{A}$. It is not hard to prove that (A, E) is isomorphic to the random graph. But if $\mathcal{A}$ is automatically presentable then so is the random graph (A, E), hence contradicting the previous corollary. □

The next goal is to show that the K_p-free random graph has no automatic presentation. For this one needs to have a finer analysis than the one for the random graph. Let $\mathcal{G} = (V, E)$ be a finite graph. Denote by $\Delta(\mathcal{G})$ the maximum degree over all the vertices of $\mathcal{G}$. Call a subgraph $\mathcal{G}$ with no edges an *independent* graph. Let $\alpha(\mathcal{G})$ be the number of vertices of a largest independent subgraph of $\mathcal{G}$. We need the following combinatorial lemma.

Lemma 4.6 *Let $\mathcal{G} = (V, E)$ be a finite graph. Then each of the following is true:*

1. $\alpha(\mathcal{G}) \ge |V|/(\Delta(\mathcal{G}) + 1)$.
2. *For all $p \ge 3$, if $\mathcal{G}$ is K_p-free then* $\alpha(\mathcal{G}) \ge \sqrt[p-1]{|V|} - 1$.

Corollary 4.7 *For $p \ge 3$, the K_p-free random graph is not automatically presentable.*

Proof Fix $p \ge 3$ and let (A, E) be an automatic copy of the K_p- free random graph over alphabet Σ. Recall that $A_n = A \cap \Sigma^{\le n}$. Let $\mathcal{G}$ be an independent subgraph of A_n of size at least $\sqrt[p-1]{|A_n|} - 1$ as in the lemma. It has $2^{|\mathcal{G}|}$ subsets. By the property of the K_p-free random graph, for every K_{p-1}-free subset $K \subset A_n$ there exists an $x \in A$ that is connected to every vertex in K and no vertex in $A_n \setminus K$. So for a fixed $n \in \mathbb{N}$, the number of functions $c_{n,y}$ as y varies over A, is at least $2^{\sqrt[p-1]{|A_n|}-1}$, contradicting Theorem 4.3. □

As the fourth application we prove that the random partial order $\mathcal{U}$ does not have an automatic presentation. For the proof we need the following combinatorial result that connects the size of a finite partial order $(B, \leq)$ with the cardinalities of its chains and anti-chains.

Lemma 4.8 (Dilworth) *Let $(B, \leq)$ be a* finite *partial order of cardinality n. Let a be the size of largest anti-chain in $(B, \leq)$ and let c be the size of the largest chain in $(B, \leq)$. Then $n \leq ac$.*

Corollary 4.9 *The random partial order $\mathcal{U} = (U, \leq)$ has no automatic presentation.*

Proof The universal partial order has the following property (see [9]):

1. If Z is a *finite* anti-chain of $\mathcal{U}$ and X and Y partition Z then there exists an element $z \in U$ such that for every $x \in X$, $z > x$ and for every $y \in Y$, element z is not comparable with y.
2. If Z is a *finite* chain of $\mathcal{U}$ with least element x and largest element y then there exists an element $z \in U$ such that $z > x$ and $z < y$ and z is not comparable with every $v \in X \setminus \{x, y\}$.

Assume that $\mathcal{U}$ has an automatic presentation $(A, \leq)$. Consider the standard approximation $A_0, A_1, A_2, \ldots$ of $(A, \leq)$. The formula $\Phi(x, y)$ is $x \leq y \vee y \leq x$. Now let us take A_n.

Let Z be an anti-chain of A_n. Consider a subset X of Z. There exists an element $y \in U$ such that for every $x \in X$, $y > x$ and for every $x' \notin A_n \setminus X$, element y is not comparable with x'. From this we conclude that

$$(\star) \qquad \#(\text{functions of type } c_{n,y}) \geq 2^{|Z|}.$$

Let Z is a *finite* chain of $\mathcal{P}$ with least element v and largest element w then there exists an element $y \in U$ such that $y > v$ and $v < w$ and y is not comparable with x for every $x \in X$. From this we conclude that

$$(\star\star) \qquad \#(\text{functions of type } c_{n,y}) \geq |Z|^2.$$

Let X be the largest anti-chain and Y be the largest chain of A_n with cardinalities a and c, respectively. By Dilworth Lemma above, $(\star)$ and $(\star\star)$ we derive that $|A_n|^2 < c^2 \cdot a^2$ which in turn is less than

$$\#(\text{functions of type } c_{n,y}) \quad \times \quad \log^2[\#(\text{functions of type } c_{n,y})].$$

This bound clearly contradicts the statement of Theorem 4.3. The corollary is proved. □

The following is another application of Theorem 4.3. It was observed by Leonid Libkin, though originally proven using growth level of string lengths in [3].

Corollary 4.10 *The pairing algebra (A, p) has no automatic presentation.*

5 The Isomorphism Problem

The results in the previous sections give us hope that one can characterise automatic structures for certain classes of structures, e.g. Boolean algebras. However, in this section we prove that the isomorphism problem is Σ_1^1-complete, thus showing that the problem is as hard as possible when considering the class of *all* automatic structures. Then **complexity of the isomorphism problem** for automatic structures consists in establishing the complexity of the set $\{(\mathcal{A}, \mathcal{B}) \mid \mathcal{A}$ and $\mathcal{B}$ are automatic structures and $\mathcal{A}$ is isomorphic to $\mathcal{B}\}$.

Let $\mathcal{M}$ be a Turing machine over input alphabet Σ. Its configuration graph $C(\mathcal{M})$ is the set of all configurations of $\mathcal{M}$, with an edge from c to d if T can move from c to d in a single transition. The following is an easy lemma:

Lemma 5.1 *For every Turing machine T the configuration graph $C(T)$ is automatic. Further, the set of all vertices with outdegree (indegree) 0 is FA-recognisable.*

Definition 5.2 *A Turing machine $\mathcal{R}$ is* **reversible** *if every vertex in $C(\mathcal{R})$ has both indegree and outdegree at most one.*

Now let $\mathcal{R}$ be a reversible Turing machine. Consider its configuration space $C(\mathcal{R})$. The machine $\mathcal{R}$ can be modified so that it only halts in an accepting state; so, instead of halting in a rejecting state, it loops forever. Let x be a configuration of $\mathcal{R}$. Consider the sequence: $x = x_0, x_1, x_2, \ldots$ such that $(x_i, x_{i+1}) \in E$, where E is the edge relation of the configuration space. Call this sequence the **chain defined by** x. We say that x is the **base** of chain X. If x does not have a predecessor then the chain defined by x is maximal. Since $\mathcal{R}$ is reversible, the configuration space $C(\mathcal{R})$ is a disjoint union of maximal chains such that each chain is either finite, or isomorphic to $(\mathbb{N}, S)$, or isomorphic to $(\mathbb{Z}, S)$, where $(x, y) \in S$ iff $y = x + 1$. It is known that every Turing machine can be converted into an equivalent reversible Turing machine (see for example [2]). Our next lemma states this fact and provides some additional structural information about the configuration space of reversible Turing machines:

Lemma 5.3 *A deterministic Turing machine can be converted into an equivalent reversible Turing machine $\mathcal{R}$ such that every maximal chain in $C(\mathcal{R})$ is either finite or isomorphic to $(\mathbb{N}, S)$.*

Denote by $\mathbb{N}^\star$ the set of all finite strings from $\mathbb{N}$. A set $T \subset \mathbb{N}^\star$ is a *special tree* if T is closed downward, namely $xy \in T$ implies $x \in T$, for $x, y \in \mathbb{N}^\star$. We view these special trees as structures of the signature E, where $E(x,y)$ if and only if $y = xz$ for some $z \in \mathbb{N}$. Thus, for every $x \in \mathbb{N}^\star$ the set $\{y \mid E(x,y)\}$ can be thought as the set of all *immediate successors of x*.

A special tree T is *recursively enumerable* if the set T is the domain of the function computed by a Turing machine. We will use the following fact from computable model theory [8, Thm 3.2].

Lemma 5.4 *The isomorphism problem for recursively enumerable special trees is Σ^1_1-complete. In fact, the trees can be chosen to be subtrees of $\{2n : n \in \mathbb{N}\}^\star$.*

It is clear from the proof in [8] that the trees obtained are special (namely, subtrees of $\mathbb{N}^\star$). By a mere change of notation one obtains subtrees of $\{2n : n \in \mathbb{N}\}^\star$.

Lemma 5.5 *The special tree $\mathbb{N}^\star$ has an automatic presentation $\mathcal{A}_1$.*

Proof. Consider the prefix relation $\leq_p$ on the set of all binary strings. The tree $\mathbb{N}^\star$ is isomorphic to the automatic tree $\mathcal{A}_1 = (\{0,1\}^\star 1 \cup \{\lambda\}; E_1)$, where E_1 is the set of all pairs (x,y) such that y is the immediate $<_p$-successor of x. The isomorphism is established via the computable mapping sending $n_1 \ldots n_k$ to $0^{n_1}1 \ldots 0^{n_k}1$ and the root to λ.

Define $\mathcal{S} = \{0,1\}^\star 1 \cup \{\lambda\}$. From now on we make the following conventions:

1. All special trees we consider will be viewed as subsets of $(\mathcal{S}, \leq_p)$.
2. The set of inputs for all Turing machines considered are strings from $\mathcal{S}$. Each $w \in \mathcal{S}$ is identified with the initial configuration starting from w.
3. All Turing machines considered are reversible.
4. The domains of all Turing machines considered are assumed to be downward closed. Thus, the domains of all Turing machines form recursively enumerable special trees.

Our goal is to transform every recursively enumerable tree $T \subset \mathcal{S}$ into an automatic structure $\mathcal{A}_\mathcal{R}$, where $\mathcal{R}$ is the reversible machine with domain T. The idea is to attach computations of $\mathcal{R}$ to the initial configurations in $\mathcal{S}$. For all recursively enumerable trees T_1 and T_2, T_1 and T_2 are isomorphic if and only if the automatic structures $\mathcal{A}_{\mathcal{R}_1}$ and $\mathcal{A}_{\mathcal{R}_2}$ are isomorphic. To ensure this we also have to attach infinitely many chains of each finite length to the initial configurations, for in that case the length of a halting computation does not make a difference.

Let $\mathcal{R}$ be a reversible Turing machine. A **chain** is an initial segment of $(\mathbb{N}, S)$. Let $\mathcal{J}$ be the graph consisting of infinitely many finite chains of every finite length. We denote by J the set of vertices of $\mathcal{J}$, and the bases of the chains in $\mathcal{J}$ by $b_1, b_2, \ldots$. The following is not hard to prove:

Lemma 5.6 *The graph $\mathcal{J}$ has an automatic presentation.*

To construct $\mathcal{A}_\mathcal{R}$, with every $w \in A_1$ associate a graph $\mathcal{J}_w$ defined as follows. The vertex set of $\mathcal{J}_w$ consists of all $\{(w,j) \mid j \in J\}$ and edges between (w,j) and (w,j') if and only if (j,j') is an edge in $\mathcal{J}$. Put connecting edges from w into each (w, b_i). Let $\mathcal{A}_2$ be the graph consisting of $\mathcal{A}_1$ and every $\mathcal{J}_w$ and the connecting edges. Clearly, the graph $\mathcal{A}_2$ is automatic.

To the set A_2 add all the configurations of the Turing machine $\mathcal{R}$ and all the edges of the configuration space of $\mathcal{R}$. Note that each $w \in \mathcal{S}$, which is an initial configuration of $\mathcal{R}$ is by Convention 2 already in A_2. In addition, add a disjoint automatic copy of the graph $\mathcal{J}$ and an automatic copy of infinitely many infinite chains. Call all chains added at this stage **isolated chains**. The resulting graph is denoted by $\mathcal{A}_\mathcal{R}$.

The proof of the following is straightforward and is omitted.

Lemma 5.7 *The graph $\mathcal{A}_\mathcal{R}$ is automatic.*

We summarise the structure of the graph $\mathcal{A}_\mathcal{R}$. Firstly, in $\mathcal{A}_\mathcal{R}$ there are infinitely many isolated chains of every (finite and infinite) length. With each element w in $\mathcal{A}_1$ there is an associated structure $\mathcal{J}_w$, which consists of infinitely many chains of every finite length, and no infinite chain. Finally with every initial configuration w (which is an element of $\mathcal{A}_\mathcal{R}$ and belongs to the infinitely branching tree $\mathcal{A}_1$) there is a chain with base w that is the computation path from the configuration space of Turing machine T. These observations prove the following.

Lemma 5.8 *Let w be an initial configuration of $\mathcal{A}_\mathcal{R}$. Then w is the base of infinitely many chains of every finite length. Also the Turing machine T halts on w if and only if there is no infinite chain with base w. In case T does not halt on w there is exactly one infinite chain with base w.*

Lemma 5.9 *Let T_1 and T_2 be computable trees which are domains of reversible Turing machines $\mathcal{R}_1$ and $\mathcal{R}_2$ respectively. Then the trees T_1 and T_2 are isomorphic if and only if the automatic graphs $\mathcal{A}_{\mathcal{R}_1}$ and $\mathcal{A}_{\mathcal{R}_2}$ are isomorphic.*

Proof First note that the domain T of $\mathcal{R}$ consists of all w in $\mathcal{A}_{\mathcal{R}}$ that are initial configurations and are not the base of an infinite chain. Hence T may be thought of as a subgraph of $\mathcal{A}_{\mathcal{R}}$. Now suppose $\mathcal{A}_{\mathcal{R}_1}$ is isomorphic to $\mathcal{A}_{\mathcal{R}_2}$ via the map ϕ. Note that w is the base of an infinite chain if and only if $\phi(w)$ is the base of an infinite chain. Hence ϕ is an isomorphism between the trees T_1 and T_2 viewed as subgraphs of $\mathcal{A}_{\mathcal{R}_1}$ and $\mathcal{A}_{\mathcal{R}_2}$ respectively.

Conversely assume T_1 and T_2 are isomorphic via ψ (as before we may assume that T_i is a subgraph of $\mathcal{A}_{\mathcal{R}_i}$). Consider the sets I_1 and I_2 of all isolated chains in $\mathcal{A}_{\mathcal{R}_1}$ and $\mathcal{A}_{\mathcal{R}_2}$. There is an isomorphism between I_1 and I_2, since both consist of infinitely many chains of every length (finite and infinite), independently of the chains of the configuration graphs that are *not* defined by initial configurations. Therefore it suffices to establish an isomorphism extending ψ on the rest of the structure.

If $w \in T_1$ then there is an isomorphism between all the chains with base w and all the chains with base $\psi(w)$. Indeed each is the base of infinitely many chains of every finite length and no chain of infinite length, independently of the (possibly different) lengths of the finite computations of R_1 on w and R_2 on $\psi(w)$. Hence extend ψ from the chains defined by elements of T_1 to the chains defined by elements of T_2.

Suppose $w \in T_1$ and consider the set S_1 of immediate successors of w that are initial configurations but not in T_1. Similarly write S_2 for the set of immediate successors of $\psi(w)$ that are initial configurations but not in T_2. By the extra condition in Lemma 5.4 that the trees be subtrees of $\{2n : n \in \mathbb{N}\}^*$, both S_1 and S_2 are infinite. So we may extend ψ to those immediate successors by adding a bijection between S_1 and S_2.

Finally, for any initial configurations $v_1 \in \mathcal{S} - T_1$ and $v_2 \in \mathcal{S} - T_2$, there is an isomorphism between the nodes in $\mathcal{S}$ extending v_1 and the ones extending v_2. This isomorphism extends to the attached chains (as there is one infinite chain and infinitely many of each finite length). So we may extend ψ to an isomorphism of $\mathcal{A}_{\mathcal{R}_1}$ and $\mathcal{A}_{\mathcal{R}_2}$. $\square$

Hence we have reduced the isomorphism problem for recursively enumerable trees to the isomorphism problem for automatic graphs. The main result now follows.

Theorem 5.10 *The isomorphism problem for automatic structures is Σ^1_1-complete.*

References

[1] M. Benedikt, L. Libkin, T. Schwentick, and L. Segoufin. A Model-Theoretic approach to regular string relations. In *Proceedings of the 16th Annual IEEE Symposium on Logic in Computer Science* (LICS 2001), pages 431–440, June 16–19 2001.

[2] C. H. Bennett. Logical reversibility of computation. *IBM Journal of Research and Development*, 17(6):525–532, November 1973.

[3] A. Blumensath. *Automatic Structures.* Diploma thesis, RWTH Aachen, 1999.

[4] A. Blumensath and E. Grädel. Automatic structures. In *15th Symposium on Logic in Computer Science* (LICS 2000), pages 51–62, June 2000.

[5] C. Delhomme. Non-automaticity of ω^ω, 2001. manuscript.

[6] C. Delhomme. The rado graph is not automatic, 2001. manuscript.

[7] S. Eilenberg, C.C. Elgot, and J.C. Shepherdson. Sets recognised by n–tape automata. *Journal of Algebra*, 13(4):447–464, 1969.

[8] S.S. Goncharov and J.F. Knight. Computable structure and non-structure theorems. *Algebra and Logic*, 41(6):351–373, 2002.

[9] W. A. Hodges. *Model Theory.* Cambridge University Press, 1993.

[10] H. Ishihara, B. Khoussainov, and S. Rubin. Some results on automatic structures. In *17th Symposium on Logic in Computer Science* (LICS 2002), pages 235–242, July 2002.

[11] B. Khoussainov and A. Nerode. Automatic presentations of structures. *Lecture Notes in Computer Science*, 960:367–392, 1995.

[12] B. Khoussainov, S. Rubin, and F. Stephan. Automatic linear orders and trees, 2003. CDMTCS technical report 208, Department of Computer Science, University of Auckland.

[13] S. Rubin. *Automatic Structures.* Phd thesis, University of Auckland, 2004. In preparation.