Model Theoretic Complexity of Automatic Structures (Extended Abstract)

Bakhadyr Khoussainov[1] and Mia Minnes[2]

[1] Department of Computer Science
University of Auckland
Auckland, New Zealand
bmk@cs.auckland.ac.nz
[2] Mathematics Department
Cornell University
Ithaca, New York 14853
minnes@math.cornell.edu

Abstract. We study the complexity of automatic structures via well-established concepts from both logic and model theory, including ordinal heights (of well-founded relations), Scott ranks of structures, and Cantor-Bendixson ranks (of trees). We prove the following results: 1) The ordinal height of any automatic well-founded partial order is bounded by ω^ω; 2) The ordinal heights of automatic well-founded relations are unbounded below ω_1^{CK}; 3) For any infinite computable ordinal α, there is an automatic structure of Scott rank at least α. Moreover, there are automatic structures of Scott rank $\omega_1^{CK}, \omega_1^{CK}+1$; 4) For any ordinal $\alpha < \omega_1^{CK}$, there is an automatic successor tree of Cantor-Bendixson rank α.

1 Introduction

In recent years, there has been increasing interest in the study of structures that can be presented by automata. The underlying idea is to apply techniques of automata theory to decision problems that arise in logic and applications such as databases and verification. A typical decision problem is the model checking problem: for a structure $\mathcal{A}$ (e.g. a graph), design an algorithm that, given a formula $\phi(\bar{x})$ in a formal system and a tuple $\bar{a}$ from the structure, decides if $\phi(\bar{a})$ is true in $\mathcal{A}$. In particular, when the formal system is the first order predicate logic or the monadic second order logic, we would like to know if the theory of the structure is decidable. Fundamental early results in this direction by Büchi ([4], [5]) and Rabin ([20]) proved the decidability of the monadic second order theories of the successor on the natural numbers and of the binary tree. There have been numerous applications and extensions of these results in logic, algebra, verification, model checking, and databases (see, for example, [9] [23] [24] and [25]). Moreover, automatic structures provide a theoretical framework for constraint databases over discrete domains such as strings and trees [1].

A structure $\mathcal{A} = (A; R_0, \ldots, R_m)$ is **automatic** if the domain A and all the relations $R_0, \ldots, R_m$ of the structure are recognised by finite automata (precise definitions are in the next section). Independently, Hodgson [13] and later

Khoussainov and Nerode [14] proved that for any given automatic structure there is an algorithm that solves the model checking problem for the first order logic. In particular, the first order theory of the structure is decidable. There is a body of work devoted to the study of resource-bounded complexity of the model checking problem for automatic structures. Most current results demonstrate that automatic structures are not complex in various concrete senses. However, in this paper we use well-established concepts from both logic and model theory to prove results in the opposite direction. We now briefly describe the measures of complexity we use (ordinal heights of well-founded relations, Scott ranks of structures, and Cantor-Bendixson ranks of trees) and connect them with the results of this paper.

A relation R is called **well-founded** if there is no infinite sequence $x_1, x_2, x_3, \ldots$ such that $(x_{i+1}, x_i) \in R$ for $i \in \omega$. In computer science, well-founded relations are of interest due to a natural connection between well-founded sets and terminating programs. We say that a program is **terminating** if every computation from an initial state is finite. This is equivalent to well-foundedness of the collection of states reachable from the initial state, under the reachability relation [3]. The **ordinal height** is a measure of the depth of well-founded relations. Since all automatic structures are computable, the obvious bound for ordinal heights of automatic well-founded relations is ω_1^{CK} (the first non-computable ordinal). Sections 3 and 4 study the sharpness of this bound. Theorem 1 characterizes automatic well-founded partial orders in terms of their ordinal heights, whereas Theorem 2 shows that ω_1^{CK} is the sharp bound in the general case.

Theorem 1. *For each ordinal α, α is the ordinal height of an automatic well-founded partial order if and only if $\alpha < \omega^\omega$.*

Theorem 2. *For each (computable) ordinal $\alpha < \omega_1^{CK}$, there is an automatic well-founded relation $\mathcal{A}$ whose ordinal height is greater than α.*

Section 5 is devoted to building automatic structures with high Scott ranks. The concept of Scott rank comes from a well-known theorem of Scott stating that for every countable structure $\mathcal{A}$ there exists a sentence ϕ in $L_{\omega_1,\omega}$-logic which characterizes $\mathcal{A}$ up to isomorphism [22]. The minimal quantifier rank of such a formula is called the Scott rank of $\mathcal{A}$. A known upper bound on the Scott rank of computable structures implies that the Scott rank of automatic structures is at most $\omega_1^{CK} + 1$. But, until now, all the known examples of automatic structures had low Scott ranks. Results in [19], [7], [17] suggest that the Scott ranks of automatic structures could be bounded by small ordinals. This intuition is falsified in Section 5 with the theorem:

Theorem 3. *For each infinite computable ordinal α there is an automatic structure of Scott rank at least α.*

In the last section, we investigate the Cantor-Bendixson ranks of automatic trees. A **partial order tree** is a partially ordered set $(T, \leq)$ such that there is a $\leq$-minimal element of T, and each subset $\{x \in T : x \leq y\}$ is finite and

is linearly ordered under $\leq$. A **successor tree** is a pair (T, S) such that the reflexive and transitive closure $\leq_S$ of S produces a partial order tree $(T, \leq_S)$. The **derivative** of a tree $\mathcal{T}$ is obtained by removing all the nonbranching paths of the tree. One applies the derivative operation to $\mathcal{T}$ successively until a fixed point is reached. The minimal ordinal that is needed to reach the fixed point is called the **Cantor-Bendixson (CB) rank** of the tree. The CB rank plays an important role in logic, algebra, and topology. Informally, the CB rank tells us how far the structure is from algorithmically (or algebraically) simple structures. Again, the obvious bound on CB ranks of automatic successor trees is ω_1^{CK}. In [16], it is proved that the CB rank of any automatic partial order tree is finite and can be computed from the automaton for the $\leq$ relation on the tree. It has been an open question whether the CB ranks of automatic successor trees can be bounded by small ordinals. We answer this question in the following theorem.

Theorem 4. *For $\alpha < \omega_1^{CK}$ there is an automatic successor tree of CB rank α.*

The main tool we use to prove results about high ranks is the configuration spaces of Turing machines, considered as automatic graphs. It is important to note that graphs which arise as configuration spaces have very low model-theoretic complexity: their Scott ranks are at most 3, and if they are well-founded then their ordinal heights are at most ω (see Propositions 1 and 2). Hence, the configuration spaces serve merely as building blocks in the construction of automatic structures with high complexity, rather than contributing materially to the high complexity themselves.

2 Preliminaries

A (relational) **vocabulary** is a finite sequence $(P_1^{m_1}, \ldots, P_t^{m_t}, c_1, \ldots, c_s)$, where each $P_j^{m_j}$ is a predicate symbol of arity $m_j > 0$, and each c_k is a constant symbol. A **structure** with this vocabulary is a tuple $\mathcal{A} = (A; P_1^{\mathcal{A}}, \ldots, P_t^{\mathcal{A}}, c_1^{\mathcal{A}}, \ldots, c_s^{\mathcal{A}})$, where $P_j^{\mathcal{A}}$ and $c_k^{\mathcal{A}}$ are interpretations of the symbols of the vocabulary. When convenient, we may omit the superscripts $\mathcal{A}$. We only consider infinite structures.

A **finite automaton** $\mathcal{M}$ 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 $\sigma_1\sigma_2 \ldots \sigma_n$ ($\sigma_i \in \Sigma$) is a sequence of states, say $q_0, q_1, \ldots, q_n$, such that $q_0 = \iota$ and $(q_i, \sigma_{i+1}, q_{i+1}) \in \Delta$ for all $i \in \{0, \ldots, n-1\}$. If $q_n \in F$, then the computation is **successful** and we say that automaton $\mathcal{M}$ **accepts** the word $\sigma_1\sigma_2 \ldots \sigma_n$. The **language** accepted by the automaton $\mathcal{M}$ is the set of all words accepted by $\mathcal{M}$. In general, $D \subset \Sigma^\star$ is **finite automaton recognisable**, or **regular**, if D is the language accepted by some finite automaton $\mathcal{M}$.

To define automaton recognisable relations, we use n-variable (or n-tape) automata. An n–**tape automaton** can be thought of as a one-way Turing machine with n input tapes [8]. Each tape is regarded as semi-infinite, having written on it a word over the alphabet Σ followed by an infinite succession of blanks (denoted by $\diamond$ 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. For a formal definition see, for example, [14].

Definition 1. *A structure $\mathcal{A} = (A; R_0, R_1, \ldots, R_m)$ is* **automatic** *over Σ if its domain A and all relations R_0, R_1, …, R_m are regular over Σ.*

The configuration graph of any Turing machine is an example of an automatic structure. The graph is defined by letting the configurations of the Turing machine be the vertices, and putting an edge from configuration c_1 to configuration c_2 if the machine can make an instantaneous move from c_1 to c_2. Many examples of automatic structures can be formed using the ω-fold disjoint union of a structure $\mathcal{A}$ (the disjoint union of ω many copies of $\mathcal{A}$).

Lemma 1. *[21] If $\mathcal{A}$ is automatic then its ω-fold disjoint union is isomorphic to an automatic structure.* □

The class of automatic structures is a proper subclass of the computable structures. In this paper, we will be coding computable structures into automatic ones. Good references for the theory of computable structures include [11], [15].

Definition 2. *A* **computable structure** *is a structure $\mathcal{A} = (A; R_1, \ldots, R_m)$ whose domain and relations are all computable.*

The domains of computable structures can always be identified with the set ω of natural numbers. Under this assumption, we introduce new constant symbols c_n for each $n \in \omega$ and interpret c_n as n. In this context, $\mathcal{A}$ is computable iff the **atomic diagram** of $\mathcal{A}$ (the set of Gödel numbers of all quantifier-free sentences in the extended vocabulary that are true in $\mathcal{A}$) is computable.

3 Ranks of automatic well-founded partial orders

In this section we consider structures $\mathcal{A} = (A; R)$ with a single binary relation. An element x is said to be **R-minimal for a set** X if for each $y \in X$, $(y, x) \notin R$. The relation R is said to be **well-founded** if every non-empty subset of A has an R-minimal element. This is equivalent to saying that $(A; R)$ has no infinite chains $x_1, x_2, x_3, \ldots$ where $(x_{i+1}, x_i) \in R$ for all i.

A **ranking function** for $\mathcal{A}$ is an ordinal-valued function f such that $f(y) < f(x)$ whenever $(y, x) \in R$. For f a ranking function on $\mathcal{A}$, let $ord(f) = \sup\{f(x) : x \in A\}$. The structure $\mathcal{A}$ is well-founded if and only if $\mathcal{A}$ admits a ranking function. The **ordinal height** of $\mathcal{A}$, denoted $r(\mathcal{A})$, is the least ordinal α which is $ord(g)$ for some ranking function g on $\mathcal{A}$. For $B \subseteq A$, we write $r(B)$ for the ordinal height of the structure obtained by restricting R to B. Recall that if $\alpha < \omega_1^{CK}$ then α is a computable ordinal.

Lemma 2. *If $\alpha < \omega_1^{CK}$, there is a computable well-founded relation of ordinal height α.*

Lemma 2 amounts to taking a computable copy of any linear order of type α. The next lemma follows easily from well-foundedness of ordinals and of R.

Lemma 3. *For a structure $\mathcal{A} = (A; R)$ where R is well-founded, if $r(\mathcal{A}) = \alpha$ and $\beta < \alpha$ then there is an $x \in A$ such that $r_{\mathcal{A}}(x) = \beta$.* □

For the remainder of this section, we assume further that R is a partial order. For convenience, we write $\leq$ instead of R. Thus, we consider automatic well-founded partial orders $\mathcal{A} = (A, \leq)$. We will use the notion of **natural sum of ordinals**. The natural sum of ordinals α, β (denoted $\alpha +' \beta$) is defined recursively: $\alpha +' \beta$ is the least ordinal strictly greater than $\gamma +' \beta$ for all $\gamma < \alpha$ and strictly greater than $\alpha +' \gamma$ for all $\gamma < \beta$.

Lemma 4. *Let A_1 and A_2 be disjoint subsets of A such that $A = A_1 \cup A_2$. Consider the partially ordered sets $\mathcal{A}_1 = (A_1, \leq_1)$ and $\mathcal{A}_2 = (A_2, \leq_2)$ obtained by restricting $\leq$ to A_1 and A_2 respectively. Then, $r(\mathcal{A}) \leq \alpha_1 +' \alpha_2$, where $\alpha_i = r(\mathcal{A}_i)$.*

Proof. We will show that there is a ranking function on A whose range is contained in the ordinal $\alpha_1 +' \alpha_2$. For each $x \in A$ consider the partially ordered sets $\mathcal{A}_{1,x}$ and $\mathcal{A}_{2,x}$ obtained by restricting $\leq$ to $\{z \in A_1 \mid z < x\}$ and $\{z \in A_2 \mid z < x\}$, respectively. Define $f(x) = r(\mathcal{A}_{1,x}) +' r(\mathcal{A}_{2,x})$. It is not hard to see that f is the desired ranking function. □

Corollary 1. *If $r(\mathcal{A}) = \omega^n$ and $A = A_1 \cup A_2$, where $A_1 \cap A_2 = \emptyset$, then either $r(\mathcal{A}_1) = \omega^n$ or $r(\mathcal{A}_2) = \omega^n$.* □

Khoussainov and Nerode [14] show that, for each n, there is an automatic presentation of the ordinal ω^n. It is clear that such a presentation has ordinal height ω^n. The next theorem shows that ω^ω is the sharp bound on ranks of all automatic well-founded partial orders. Now that Corollary 1 has been established, the proof of Theorem 1 follows Delhommé [7] and Rubin [21].

Theorem 1. *For each ordinal α, α is the ordinal height of an automatic well-founded partial order if and only if $\alpha < \omega^\omega$.*

Proof. One direction of the proof is clear. For the other, assume for a contradiction that there is an automatic well-founded partial order $\mathcal{A} = (A, \leq)$ with $r(\mathcal{A}) = \alpha \geq \omega^\omega$. Let $(S_A, \iota_A, \Delta_A, F_A)$ and $(S_\leq, \iota_\leq, \Delta_\leq, F_\leq)$ be finite automata over Σ recognizing A and $\leq$ (respectively). By Lemma 3, for each $n > 0$ there is $u_n \in A$ such that $r_{\mathcal{A}}(u_n) = \omega^n$. For each $u \in A$ we define the set

$$u \downarrow = \{x \in A : x < u\}.$$

Note that if $r_{\mathcal{A}}(u)$ is a limit ordinal then $r_{\mathcal{A}}(u) = r(u \downarrow)$. We define a finite partition of $u \downarrow$ in order to apply Corollary 1. To do so, for $u, v \in \Sigma^\star$, define

$X_v^u = \{vw \in A : w \in \Sigma^\star \ \&\ vw < u\}$. Each set of the form $u\downarrow$ can then be partitioned based on the prefixes of words as follows:

$$u\downarrow = \{x \in A : |x| < |u| \ \&\ x < u\} \cup \bigcup_{v \in \Sigma^\star : |v| = |u|} X_v^u.$$

(All the unions above are finite and disjoint.) Hence, applying Corollary 1, for each u_n there exists a v_n such that $|u_n| = |v_n|$ and $r(X_{v_n}^{u_n}) = r(u_n\downarrow) = \omega^n$.

On the other hand, we use the automata to define the following equivalence relation on pairs of words of equal lengths:

$$(u,v) \sim (u',v') \iff \Delta_A(\iota_A, v) = \Delta_A(\iota_A, v') \ \&\ \Delta_\leq(\iota_\leq, \binom{v}{u}) = \Delta_\leq(\iota_\leq, \binom{v'}{u'})$$

There are at most $|S_A| \times |S_\leq|$ equivalence classes. Thus, the infinite sequence (u_1, v_1), (u_2, v_2), ... contains m, n such that $m \neq n$ and $(u_m, v_m) \sim (u_n, v_n)$.

Lemma 5. *For any $u, v, u', v' \in \Sigma^\star$, if $(u,v) \sim (u',v')$ then $r(X_v^u) = r(X_{v'}^{u'})$.*

To prove the lemma, consider $g : X_v^u \to X_{v'}^{u'}$ defined as $g(vw) = v'w$. From the equivalence relation, we see that g is well-defined, bijective, and order preserving. Hence $X_v^u \cong X_{v'}^{u'}$ (as partial orders). Therefore, $r(X_v^u) = r(X_{v'}^{u'})$.

By Lemma 5, $\omega^m = r(X_{v_m}^{u_m}) = r(X_{v_n}^{u_n}) = \omega^n$, a contradiction with the assumption that $m \neq n$. Therefore, there is no automatic well-founded partial order of ordinal height greater than or equal to ω^ω. □

4 Ranks of automatic well-founded relations

4.1 Configuration spaces of Turing machines

In the following, we embed computable structures into automatic ones via configuration spaces of Turing machines. Let $\mathcal{M}$ be an n-tape deterministic Turing machine. The **configuration space** of $\mathcal{M}$, denoted by $Conf(\mathcal{M})$, is a directed graph whose nodes are configurations of $\mathcal{M}$. The nodes are n-tuples, each of whose coordinates represents the contents of a tape. Each tape is encoded as $(w\ q\ w')$, where $w, w' \in \Sigma^\star$ are the symbols on the tape before and after the location of the read/write head, and q is one of the states of $\mathcal{M}$. The edges of the graph are all the pairs of the form (c_1, c_2) such that there is an instruction of $\mathcal{M}$ that transforms c_1 to c_2. The configuration space is an automatic graph. The out-degree of every vertex in $Conf(\mathcal{M})$ is 1; the in-degree need not be 1.

Definition 3. *A deterministic Turing machine $\mathcal{M}$ is* **reversible** *if $Conf(\mathcal{M})$ consists only of finite chains and chains of type ω.*

Lemma 6. *[2] For any deterministic 1-tape Turing machine there is a reversible 3-tape Turing machine which accepts the same language.*

Proof. (Sketch) Given a deterministic Turing machine, define a 3-tape Turing machine with a modified set of instructions. The modified instructions have the property that neither the domains nor the ranges overlap. The first tape performs the computation exactly as the original machine would have done. As the new machine executes each instruction, it stores the index of the instruction on the second tape, forming a history. Once the machine enters a state which would have been halting for the original machine, the output of the computation is copied onto the third tape. Then, the machine runs the computation backwards and erases the history tape. The halting configuration contains the input on the first tape, blanks on the second tape, and the output on the third tape. □

We establish the following notation for a 3-tape reversible Turing machine $\mathcal{M}$ given by the construction in this lemma. A **valid initial configuration** of $\mathcal{M}$ is of the form $(\lambda\ \iota\ x, \lambda, \lambda)$, where x in the domain, λ is the empty string, and ι is the initial state of $\mathcal{M}$. From the proof above, observe that a **final (halting)** configuration is of the form $(x, \lambda, \lambda\ q_f\ y)$, with q_f a halting state of $\mathcal{M}$. Also, because of the reversibility assumption, all the chains in $Conf(\mathcal{M})$ are either finite or ω-chains (the order type of the natural numbers). In particular, this means that $Conf(\mathcal{M})$ is well-founded. We call an element of in-degree 0 a **base** (of a chain). The set of valid initial or final configurations is regular. We classify the components (chains) of $Conf(\mathcal{M})$ as follows:

- **Terminating computation chains**: finite chains whose base is a valid initial configuration; that is, one of the form $(\lambda\ \iota\ x, \lambda, \lambda)$, for $x \in \Sigma^\star$.
- **Non-terminating computation chains**: infinite chains whose base is a valid initial configuration.
- **Unproductive chains**: chains whose base is not a valid initial configuration.

Configuration spaces of reversible Turing machines are locally finite graphs (graphs of finite degree) and well-founded. Hence, the following proposition guarantees that their ordinal heights are small. The proof is left to the reader.

Proposition 1. *If $G = (A, E)$ is a locally finite graph then E is well-founded and the ordinal height of E is not above ω, or E has an infinite chain.* □

4.2 Automatic well-founded relations of high rank

Theorem 2. *For each computable ordinal $\alpha < \omega_1^{CK}$, there is an automatic well-founded relation $\mathcal{A}$ whose ordinal height is greater than α*

Proof. The proof of the theorem uses properties of Turing machines and their configuration spaces. We take a computable well-founded relation whose ordinal height is α, and "embed" it into an automatic well-founded relation with similar ordinal height.

By Lemma 2, let $\mathcal{C} = (C, L_\alpha)$ be a computable well-founded relation of ordinal height α. We assume without loss of generality that $C = \Sigma^\star$ for some finite alphabet Σ. Let $\mathcal{M}$ be the Turing machine computing the relation L_α.

On each pair (x, y) from the domain, $\mathcal{M}$ halts and outputs "yes" or "no" . By Lemma 6, we can assume that $\mathcal{M}$ is reversible. Recall that $Conf(\mathcal{M}) = (D, E)$ is an automatic graph. We define the domain of our automatic structure to be $A = \Sigma^\star \cup D$. The binary relation of the automatic structure is:

$$R = E \cup \{(x, (\lambda\, \iota\, (x, y), \lambda, \lambda)) : x, y \in \Sigma^\star\} \cup \{(((x, y), \lambda, \lambda\, q_f \text{ "yes" }), y) : x, y \in \Sigma^\star\}.$$

Intuitively, the structure $(A; R)$ is a stretched out version of (C, L_α) with infinitely many finite pieces extending from elements of C, and with disjoint pieces which are either finite chains or chains of type ω. The structure $(A; R)$ is automatic because its domain is a regular set of words and the relation R is recognisable by a 2-tape automaton. We should verify, however, that R is well-founded. Let $Y \subset A$. If $Y \cap C \neq \emptyset$ then since (C, L_α) is well-founded, there is $x \in Y \cap C$ which is L_α-minimal. The only possible elements u in Y for which $(u, x) \in R$ are those which lie on computation chains connecting some $z \in C$ with x. Since each such computation chain is finite, there is an R-minimal u below x on each chain. Any such u is R-minimal for Y. On the other hand, if $Y \cap C = \emptyset$, then Y consists of disjoint finite chains and chains of type ω. Any such chain has a minimal element, and any of these elements are R-minimal for Y. Therefore, $(A; R)$ is an automatic well-founded structure.

We now consider the ordinal height of $(A; R)$. For each element $x \in C$, an easy induction on $r_C(x)$, shows that $r_{\mathcal{C}}(x) \leq r_{\mathcal{A}}(x) \leq \omega + r_{\mathcal{C}}(x)$. We denote by $\ell(a, b)$ the (finite) length of the computation chain of $\mathcal{M}$ with input (a, b). For any element $a_{x,y}$ in the computation chain which represents the computation of $\mathcal{M}$ determining whether $(x, y) \in R$, we have $r_{\mathcal{A}}(x) \leq r_{\mathcal{A}}(a_{x,y}) \leq r_{\mathcal{A}}(x) + \ell(x, y)$. For any element u in an unproductive chain of the configuration space, $0 \leq r_{\mathcal{A}}(u) < \omega$. Therefore, since $C \subset A$, $r(\mathcal{C}) \leq r(\mathcal{A}) \leq \omega + r(C)$. □

5 Automatic Structures and Scott Rank

The Scott rank of a structure is introduced in the proof of Scott's Isomorphism Theorem [22]. Here we follow the definition of Scott rank from [6].

Definition 4. *For structure $\mathcal{A}$ and tuples $\bar{a}, \bar{b} \in A^n$ (of equal length), define*

- $\bar{a} \equiv^0 \bar{b}$ *if $\bar{a}, \bar{b}$ satisfy the same quantifier-free formulas in the language of $\mathcal{A}$;*
- *For $\alpha > 0$, $\bar{a} \equiv^\alpha \bar{b}$ if for all $\beta < \alpha$, for each $\bar{c}$ (of arbitrary length) there is $\bar{d}$ such that $\bar{a}, \bar{c} \equiv^\beta \bar{b}, \bar{d}$; and for each $\bar{d}$ (of arbitrary length) there is $\bar{c}$ such that $\bar{a}, \bar{c} \equiv^\beta \bar{b}, \bar{d}$.*

Then, the **Scott rank** *of the tuple $\bar{a}$, denoted by $\mathcal{SR}(\bar{a})$, is the least β such that for all $\bar{b} \in A^n$, $\bar{a} \equiv^\beta \bar{b}$ implies that $(\mathcal{A}, \bar{a}) \cong (\mathcal{A}, \bar{b})$. Finally, the Scott rank of $\mathcal{A}$, denoted by $\mathcal{SR}(\mathcal{A})$, is the least α greater than the Scott ranks of all tuples of $\mathcal{A}$.*

Example 1. $\mathcal{SR}(\mathbb{Q}, \leq) = 1$, $\mathcal{SR}(\omega, \leq) = 2$, and $\mathcal{SR}(n \cdot \omega, \leq) = n + 1$.

Configuration spaces of reversible Turing machines are locally finite graphs. By the Proposition below, they all have low Scott Rank.

Proposition 2. *Let $G = (V, E)$ be a locally finite graph, then $SR(G) \leq 3$.*

Proof. The neighbourhood of diameter n of a subset U, denoted $B_n(U)$, is defined as follows: $B_0(U) = U$ and $B_n(U)$ is the set of $v \in V$ which can be reached from U by n or fewer edges. The proof of the proposition relies on two lemmas whose proofs are left to the reader.

Lemma 7. *Let $\bar{a}, \bar{b} \in V$ be such that $\bar{a} \equiv^2 \bar{b}$. Then for all n, there is a bijection of the n-neighbourhoods around $\bar{a}, \bar{b}$ which sends $\bar{a}$ to $\bar{b}$ and which respects E.*

Lemma 8. *Let $G = (V, E)$ be a graph. Suppose $\bar{a}, \bar{b} \in V$ are such that for all n, $(B_n(\bar{a}), E, \bar{a}) \cong (B_n(\bar{b}), E, \bar{b})$. Then there is an isomorphism between the component of G containing $\bar{a}$ and that containing $\bar{b}$ which sends $\bar{a}$ to $\bar{b}$.*

To prove the proposition, we note that for any $\bar{a}, \bar{b}$ in V such that $\bar{a} \equiv^2 \bar{b}$, Lemmas 7 and 8 yield an isomorphism from the component of $\bar{a}$ to the component of $\bar{b}$ that maps $\bar{a}$ to $\bar{b}$. Hence, if $\bar{a} \equiv^2 \bar{b}$, there is an automorphism of G that maps $\bar{a}$ to $\bar{b}$. Therefore, for each $\bar{a} \in V$, $SR(\bar{a}) \leq 2$, so $SR(G) \leq 3$. □

Let $\mathcal{C} = (C; R_1, \ldots, R_m)$ be a computable structure. We construct an automatic structure $\mathcal{A}$ whose Scott rank is (close to) the Scott rank of $\mathcal{C}$. Since the domain of $\mathcal{C}$ is computable, we assume that $C = \Sigma^\star$ for some finite Σ. The construction of $\mathcal{A}$ involves connecting the configuration spaces of Turing machines computing relations $R_1, \ldots, R_m$. Note that Proposition 2 suggests that the high Scott rank of the resulting automatic structure is the main part of the construction because it is not provided by the configuration spaces themselves. We detail the construction for R_i. Let $\mathcal{M}_i$ be a Turing machine for R_i. By a simple modification of the machine we assume that $\mathcal{M}_i$ halts if and only if its output is "yes" . By Lemma 6, we can also assume that $\mathcal{M}_i$ is reversible. We now modify the configuration space $Conf(\mathcal{M}_i)$ so as to respect the isomorphism type of $\mathcal{C}$. This will ensure that the construction (almost) preserves the Scott rank of $\mathcal{C}$. We use the terminology from Subsection 4.1.

Smoothing out unproductive parts. The length and number of unproductive chains is determined by the machine $\mathcal{M}_i$ and hence may differ even for Turing machines computing the same set. In this stage, we standardize the format of this unproductive part of the configuration space. We add ω-many chains of length n (for each n) and ω-many copies of ω. This ensures that the (smoothed) unproductive section of the configuration space of any Turing machine will be isomorphic and preserves automaticity.

Smoothing out lengths of computation chains. We turn our attention to the chains which have valid initial configurations at their base. The length of each finite chain denotes the length of computation required to return a "yes" answer. We will smooth out these chains by adding "fans" to each base. For this, we connect to each base of a computation chain a structure which consists of ω many chains of each finite length. To do so we follow Rubin [21]: consider the structure whose domain is $0^\star 01^\star$ and whose relation is given by xEy if and only if $|x| = |y|$ and y is the least lexicographic successor of x. This structure has a

finite chain of every finite length. As in Lemma 1, we take the ω-fold disjoint union of the structure and identify the bases of all the finite chains. We get a "fan" with infinitely many chains of each finite size whose base can be identified with a valid initial computation state. Also, the fan has an infinite component if and only if R_i does not hold of the input tuple corresponding to the base. The result is an automatic graph, $Smooth(R_i) = (D_i, E_i)$, which extends $Conf(\mathcal{M}_i)$.

Connecting domain symbols to the computations of the relation. We apply the construction above to each R_i in the signature of $\mathcal{C}$. Taking the union of the resulting automatic graphs and adding vertices for the domain, we have the structure $(\Sigma^\star \cup \cup_i D_i, E_1, \ldots, E_n)$ (where we assume that the D_i are disjoint). Assume that each $\mathcal{M}_i$ has a different initial state, and denote it by ι_i. We add n predicates F_i to the signature of the automatic structure connecting the elements of the domain of $\mathcal{C}$ with the computations of the relations R_i:

$$F_i = \{(x_0, \ldots, x_{m_i-1}, (\lambda\ \iota_i\ (x_0, \ldots, x_{m_i-1}), \lambda, \lambda)) \mid x_0, \ldots, x_{m_i-1} \in \Sigma^\star\}.$$

Note that for $\bar{x} \in \Sigma^\star$, $R_i(\bar{x})$ if and only if $F_i(\bar{x}, (\lambda\ \iota_i\ \bar{x}, \lambda, \lambda))$ holds and all E_i chains emanating from $(\lambda\ \iota_i\ \bar{x}, \lambda, \lambda)$ are finite. We have built the automatic structure

$$\mathcal{A} = (\Sigma^\star \cup \cup_i D_i, E_1, \ldots, E_n, F_1, \ldots, F_n).$$

Two technical lemmas are used to show that the Scott rank of $\mathcal{A}$ is close to α:

Lemma 9. *For $\bar{x}, \bar{y}$ in the domain of $\mathcal{C}$ and for ordinal α, if $\bar{x} \equiv^\alpha_{\mathcal{C}} \bar{y}$ then $\bar{x} \equiv^\alpha_{\mathcal{A}} \bar{y}$.*

Lemma 10. *If $\bar{x} \in \Sigma^\star \cup \cup_i D_i$, there is $\bar{y} \in \Sigma^\star$ with $\mathcal{SR}_{\mathcal{A}}(\bar{x}\bar{x}'\bar{u}) \leq 2 + \mathcal{SR}_{\mathcal{C}}(\bar{y})$.*

Putting these together, we conclude that $\mathcal{SR}(\mathcal{C}) \leq \mathcal{SR}(\mathcal{A}) \leq 2 + \mathcal{SR}(\mathcal{C})$. Applying the above construction to the computable structures of Scott rank ω_1^{CK} and $\omega_1^{CK} + 1$ built by Harrison [12] and Knight and Millar [18], we get automatic structures of Scott rank $\omega_1^{CK}, \omega_1^{CK} + 1$. We also apply the construction to [10], where it is proved that there are computable structures with Scott ranks above each computable ordinal. In this case, we get the following theorem.

Theorem 3. *For each infinite computable ordinal α, there is an automatic structure of Scott rank at least α.*

6 Cantor-Bendixson Rank of Automatic Successor Trees

In this section we show that there are automatic successor trees of high Cantor-Bendixson (CB) rank. Recall the definitions of partial order trees and successor trees from Section 1. Note that if $(T, \leq)$ is an automatic partial order tree then the successor tree (T, S), where the relation S is defined by $S(x, y) \iff (x < y)\ \&\ \neg\exists z(x < z < y)$, is automatic.

Definition 5. *The* **derivative** *of a tree T, $d(T)$, is the subtree of T whose domain is $\{x \in T :$ x lies on at least two infinite paths in $T\}$. By induction, $d^0(T) = T$, $d^{\alpha+1}(T) = d(d^\alpha(T))$, and for γ a limit ordinal, $d^\gamma(T) = \cap_{\beta<\gamma} d^\beta(T)$. The* **CB rank** *of the tree, $CB(T)$, is the least α such that $d^\alpha(T) = d^{\alpha+1}(T)$.*

The CB ranks of automatic partial order trees are finite [16]. This is not true of automatic successor trees. The main theorem of this section provides a general technique for building trees of given CB ranks. It uses the fact that for each $\alpha < \omega_1^{CK}$ there is a computable successor tree of CB rank α. This fact can be proven by recursively coding up computable trees of increasing CB rank.

Theorem 4. *For $\alpha < \omega_1^{CK}$ there is an automatic successor tree of CB rank α.*

Proof. Suppose we are given $\alpha < \omega_1^{CK}$. Take a computable tree R_α of CB rank α. We use the same construction as in the case of well-founded relations (see the proof of Theorem 2). The result is a stretched out version of the tree R_α, where between each two elements of the original tree we have a coding of their computation. In addition, extending from each $x \in \Sigma^\star$ we have infinitely many finite computation chains. Those chains which correspond to output "no" are not connected to any other part of the automatic structure. Finally, there is a disjoint part of the structure consisting of chains whose bases are not valid initial configurations. By the reversibility assumption, each unproductive component of the configuration space is isomorphic either to a finite chain or to an ω-chain. Moreover, the set of invalid initial configurations which are the base of such an unproductive chain is regular. We connect all such bases of unproductive chains to the root and get an automatic successor tree, T_α.

We now consider the CB rank of T_α. Note that the first derivative removes all the subtrees whose roots are at distance 1 from the root and are invalid initial computations. This occurs because each of the invalid computation chains has no branching and is not connected to any other element of the tree. Next, if we consider the subtree of T_α rooted at some $x \in \Sigma^\star$, we see that all the paths which correspond to computations whose output is "no" vanish after the first derivative. Moreover, $x \in d(T_\alpha)$ if and only if $x \in d(R_\alpha)$ because the construction did not add any new infinite paths. Therefore, after one derivative, the structure is exactly a stretched out version of $d(R_\alpha)$. Likewise, for all $\beta < \alpha$, $d^\beta(T_\alpha)$ is a stretched out version of $d^\beta(R_\alpha)$. Hence, $CB(T_\alpha) = CB(R_\alpha) = \alpha$. □

Acknowledgement

We thank Moshe Vardi who posed the question about ranks of automatic well-founded relations. We also thank Anil Nerode and Frank Stephan with whom we discussed Scott and Cantor-Bendixson ranks of automatic structures.

References

1. Benedikt, M. and L. Libkin. Tree extension algebras: logics, automata, and query languages. *Proceedings of LICS 02*, 203-212, 2002.
2. Bennett, C.H. Logical Reversibility of Computation. *IBM Journal of Research and Development*, 525-532, 1973.
3. Blaas, A. and Y. Gurevich. Program Termination and Well Partial Orderings. *ACM Transactions on Computational Logic*, 1-25, 2006.

4. Büchi, J.R. Weak second-order arithmetic and finite automata. *Zeitschrift Math. Logik und Grundlagen det Mathematik*, 66-92, 1960.
5. Büchi, J.R. On a decision method in restricted second-order arithmetic. *Proc. International Congress on Logic, Methodology and Philosophy of Science, 1960* (E. Nagel, P. Suppes, A. Tarski, Eds.), 1-12, Stanford University Press, 1962.
6. Calvert, W., S.S. Goncharov, and J.F. Knight. Computable structures of Scott rank ω_1^{CK} in familiar classes. *Advances in Logic (Proceedings of the North Texas Logic Conference)* Contemporary Mathematics **425**, 49-66, American Mathematical Society, 2007.
7. Delhommé, C. Automaticité des ordinaux et des graphes homogènes. *C.R. Acadèmie des sciences Paris, Ser. I* **339**, 5-10, 2004.
8. Eilenberg, S. *Automata, Languages, and Machines (Vol. A)*, Academic Press (New York), 1974.
9. Epstein, D.B.A., et al. *Word Processing in Groups*, A.K. Peters Ltd. (Natick, Massachusetts), 1992.
10. Goncharov, S.S. and J.F. Knight. Computable structure and non-structure theorems. *Algebra and Logic* **41**, 351-373, 2002.
11. Harizanov, V.S. Pure Computable Model Theory. *Handbook of Recursive Mathematics* (Yu. Ershov, S. Goncharov, A. Nerode, J. Remmel, eds.), 3-114, North-Holland (Amsterdam), 1998.
12. Harrison, J. Recursive Pseudo Well-Orderings. *Transactions of the American Mathematical Society* **131: 2**, 526-543, 1968.
13. Hodgson, B.R. On Direct Products of Automaton Decidable Theories. *Theoretical Computer Science* **19**, 331-335, North-Holland, 1982.
14. Khoussainov, B. and A. Nerode. Automatic presentations of structures. *Lecture Notes in Computer Science* **960**, 367-392, 1995.
15. Khoussainov, B. and R.A. Shore. Effective Model Theory: The Number of Models and Their Complexity. *Models and Computability, Invited Papers from LC 1997* (S.B. Cooper and J.K. Truss, eds.) LMSLNS **259**, 193-240, Cambridge University Press (Cambridge, England), 1999.
16. Khoussainov, B., S. Rubin, and F. Stephan. On automatic partial orders. *Proceedings of 18th LICS*, 168-177, 2003.
17. Khoussainov, B., S. Rubin, and F. Stephan. Automatic linear orders and trees. *ACM Transactions on Computational Logic* **6** Number 4, 675-700, 2005.
18. Knight, J.F. and J. Millar. Computable Structures of Rank ω_1^{CK}. Submitted to *Journal of Mathematical Logic*; posted on arXiv 25 Aug 2005.
19. Lohrey, M. Automatic structures of bounded degree. *Proceedings of LPAR 03* LNAI **2850**, 344-358, 2003.
20. Rabin, M.O. Decidability of Second-Order Theories and Automata on Infinite Trees. *Transactions of the American Mathematical Society* **141**, 1-35, 1969.
21. Rubin, S. *Automatic Structures*, PhD Thesis, University of Auckland, 2004.
22. Scott, D. Logic with Denumerably Long Formulas and Finite Strings of Quantifiers. *The Theory of Models* (J. Addison, L. Henkin, A. Tarski, eds.), 329-341, North-Holland, 1965.
23. Vardi, M.Y. and P. Wolper. Automata-Theoretic Techniques for Modal Logics of Programs. *Proceedings of 16th STOC*, 446-456, 1984.
24. Vardi, M.Y. An Automata-Theoretic Approach to Linear Temporal Logic. *Logics for Concurrency: Structure versus Automata* Lecture Notes in Computer Science **1043**, 238-266, Springer-Verlag, 1996.
25. Vardi, M.Y. Model Checking for Database Theoreticians. *Proceedings of ICDL 5*, 2005.