The Journal of Symbolic Logic
Volume 00, Number 0, XXX 0000

SOME NATURAL DECISION PROBLEMS IN AUTOMATIC GRAPHS

DIETRICH KUSKE AND MARKUS LOHREY

Abstract. For automatic and recursive graphs, we investigate the following problems:

(A) existence of a Hamiltonian path and existence of an infinite path in a tree
(B) existence of an Euler path, bounding the number of ends, and bounding the number of infinite branches in a tree
(C) existence of an infinite clique and an infinite version of set cover

The complexity of these problems is determined for automatic graphs and, supplementing results from the literature, for recursive graphs. Our results show that these problems

(A) are equally complex for automatic and for recursive graphs (Σ^1_1-complete),
(B) are moderately less complex for automatic than for recursive graphs (complete for different levels of the arithmetic hierarchy),
(C) are much simpler for automatic than for recursive graphs (decidable and Σ^1_1-complete, resp.).

§1. Introduction. The theory of *recursive structures* has its origins in computability theory. A structure is recursive, if its domain is a recursive set of naturals, and every relation is again recursive. Starting with the work of Manaster and Rosenstein [32] and Bean [2, 3], infinite variants of classical graph problems for finite graphs were studied for recursive graphs. It is not surprising that these problems are mostly undecidable for recursive graphs. This motivates the search for the precise level of undecidability. It turned out that some of the problems reside on low levels of the arithmetic hierarchy (e.g., existence of an Euler path belongs to Σ^0_4 [14]), whereas others are complete for Σ^1_1 (e.g., existence of a Hamiltonian path [18]) – the first level of the analytic hierarchy.

In computer science, in particular in the area of automatic verification, focus has shifted in recent years from arbitrary recursive graphs to subclasses that have more amenable algorithmic properties. An important example for this is the class of *automatic graphs* [7, 8, 21]. A graph is called automatic if its sets of nodes and edges can be accepted by finite automata (with synchronously moving heads). One of the main motivations for investigating automatic graphs is the fact that every automatic graph has a decidable first-order theory [21], this result extends to first-order logic with infinity and modulo quantifiers [8, 24]. Hence, in some sense, automatic structures are "simple". On the other hand,

1991 *Mathematics Subject Classification.* This is not required.
Key words and phrases. This is not required.
The second author acknowledges support from the DFG-project GELO.

0022-4812/00/0000-0000/$00.00

Khoussainov, Nies, and Rubin have shown that the isomorphism problem for automatic graphs is Σ^1_1-complete [22], i.e., this problem has the maximal possible complexity. For locally finite automatic graphs, the isomorphism problem becomes Π^3_0-complete [35]. Khoussainov and Minnes proved that also model theoretically, automatic structures can be rather complex [20].

In this paper, we are interested in the question whether natural graph problems are simpler for automatic graphs than for recursive graphs, or whether the complexity is the same? From our results it will become clear that there is no general answer to this question. We will encounter three different scenarios:

(A) Section 3 exhibits two natural problems that are known to be Σ^1_1-complete for recursive graphs, and that remain so for automatic graphs. These problems are the existence of a Hamiltonian path and existence of an infinite branch in a successor tree. In these cases, we strengthen known Σ^1_1-lower bounds from recursive to automatic graphs.

(B) Section 4 presents problems that are moderately less complex for automatic than for recursive graphs. More specifically, we show that existence of an Euler path is D^0_3-complete (D^0_3 is the class of all set differences of two Σ^0_3-sets) for recursive graphs, but only Π^0_2-complete for automatic graphs. For recursive graphs, upper and lower bounds of Σ^0_4 and Π^0_3 were stated in [14].

A similar situation occurs when we ask for the number of ends, an important concept in combinatorial group theory, see e.g. [9]. The number of ends of a graph is the supremum of the number of *infinite* connected components that remain after removing an arbitrary finite set of edges. We prove that existence of at most k ends is Π^0_3-complete for recursive graphs and only Π^0_2-complete for automatic graphs (the undecidability for $k = 1$ and automatic graphs was shown in [31]).

Finally, also estimating the number of infinite branches in a finitely branching tree follows the same pattern of becoming moderately simpler: for any $k > 0$, existence of at most k infinite branches drops from Π^0_3-completeness (for recursive trees) to Π^0_2-completeness (for automatic trees), and existence of infinitely many infinite branches drops from Π^0_4-completeness to Π^0_3-completeness.

(C) Finally, Section 5 demonstrates how natural problems that are Σ^1_1-complete for recursive graphs become decidable for automatic graphs. This is known to be the case for the existence of an infinite branch in an order tree [26, 23] and, more generally, for the infinite clique problem [19, 34]. We present one more problem with this status, which is an infinitary version from [19] of the classical set cover problem. For this decidability result, we present a fragment FSO of second order logic that is decidable for automatic graphs. The idea of this fragment is that second order quantification $\exists X : \alpha$ is only allowed if α requires X to be infinite and contains membership statements $\overline{y} \in X$ only negatively. Although this is quite restrictive, it suffices to handle the three problems mentioned above.

Most of the results in this paper were announced in the extended abstracts [30, 28].

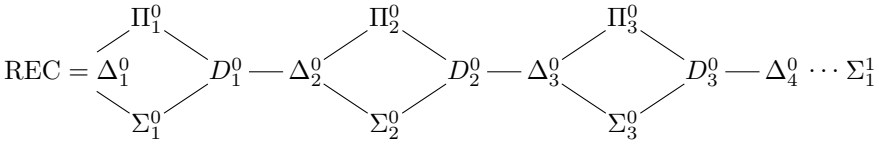

FIGURE 1.

§2. Preliminaries.

2.1. Recursion theory. Details on the arithmetical and analytical hierarchy can be found for instance in [26, 27, 33]. With Σ_n^0 we denote the n^{th} (existential) level of the *arithmetical hierarchy*; it is the class of all subsets $A \subseteq \mathbb{N}$ such that there exists a decidable predicate $P \subseteq \mathbb{N}^{n+1}$ with

$$A = \{a \in \mathbb{N} \mid \exists x_1 \forall x_2 \cdots Q x_n : (a, x_1, \ldots, x_n) \in P\},$$

where $Q = \exists$ ($Q = \forall$) for n odd (even). The set of complements of Σ_n^0-sets is denoted by Π_n^0. Moreover, the class D_n^0 consists of all sets $K \setminus L$ for $K, L \in \Sigma_n^0$, i.e., it is the class of all intersections of a Σ_n^0-set and a Π_n^0-set.

Recall that Σ_1^1 is the first level of the *analytical hierarchy*. More precisely, it is the class of all subsets of $\mathbb{N}$ of the form $\{n \in \mathbb{N} \mid \exists A \subseteq \mathbb{N} : \varphi(A, n)\}$, where $\varphi(A, n)$ is a formula of first-order arithmetic, which may use A as a unary predicate.

The inclusion between the above classes are depicted in Fig. 1, where REC denotes the class of recursive sets and $\Delta_n^0 = \Sigma_n^0 \cap \Pi_n^0$. All inclusions are known to be strict and also the union of all arithmetical classes is properly contained in Σ_1^1.

If not otherwise stated, a Turing machine is always a deterministic Turing machine M that accepts by halting; its language is denoted by $L(M)$. We identify a Turing machine with its Gödel-index with respect to some fixed Gödel numbering of Turing machines. We use the following classes of Turing machines; cf. [27] for the completeness results:

1. TOTAL denotes the class of Turing machines that halt on every input. This set TOTAL is Π_2^0-complete.
2. FIN denotes the class of Turing machines that halt for only finitely many inputs. This set FIN is Σ_2^0-complete.
3. COF denotes the class of Turing machines that halt for almost all inputs (i.e., diverge for only finitely many inputs). This set COF is Σ_3^0-complete.
4. $\overline{\text{COF}}$ denotes the class of Turing machines that diverge for infinitely many inputs. Since $\overline{\text{COF}}$ is the complement of COF, this set $\overline{\text{COF}}$ is Π_3^0-complete.

2.2. Infinite graphs. For details on graph theory see [10]. A *directed graph* is a pair (V, E) where V is the possibly infinite set of nodes and $E \subseteq V \times V$ is the set of edges with $u \neq v$ for all edges $(u, v) \in E$. An *undirected graph* is a pair $G = (V, E)$, where V is the (possibly infinite) set of nodes and $E \subseteq \binom{V}{2}$ is the set of edges. In the following, when just speaking of a graph, we always mean an undirected graph. For a directed graph $G = (V, E)$, we denote by $\overline{G}$ its undirected version $(V, \{\{u, v\} \mid (u, v) \in E\})$.

Let $G = (V, E)$ be a graph. If $\{u, v\} \in E$, then we say that u and v are *neighbors*. The *order* of a vertex $v \in V$ is the number of its neighbors. The *degree* of a graph G is the supremum of the orders of its vertices; if this supremum is finite, we say the graph has *bounded degree*. If it is only required that every node has finite order, then G is called *locally finite*. The graph G is *planar* if it can be embedded in the Euclidean plane without crossing edges and without accumulation points.

A *finite path* in a directed (resp. undirected) graph $G = (V, E)$ is a sequence $[v_1, v_2, \ldots, v_n]$ of nodes such that $(v_i, v_{i+1}) \in E$ (resp. $\{v_i, v_{i+1}\} \in E$) for all $1 \leq i < n$; it is *simple* if the nodes $v_1, \ldots, v_n$ are mutually distinct. The nodes v_1 and v_n are the end points of this path. An *infinite (simple) path* in G is an infinite sequence $[v_1, v_2, \ldots]$ such that every initial segment is a finite (simple) path.

A graph $G = (V, E)$ is connected if for all $u, v \in V$ distinct there exists a finite path with end points u and v. A directed graph G is *connected* if $\overline{G}$ is connected. For a set of edges $H \subseteq E$ of $G = (V, E)$, let $f(H)$ be the number of infinite connected components of $(V, E \setminus H)$. The *number of ends of* G is the maximum of all $f(H)$ for $H \subseteq E$ finite (if this maximum exists) and ∞ otherwise.

An *Euler path* of an *infinite* undirected graph G is an infinite path $[v_1, v_2, \ldots]$ in G that passes every edge of G exactly once, i.e., the mapping $i \mapsto \{v_i, v_{i+1}\}$ is a bijection from $\mathbb{N}$ onto the set of edges E. A graph with an Euler path is called *Eulerian*.[1] Euler's well-known characterization of Eulerian finite graphs [12] was extended by Erdős, Grünwald, and Vazsonyi as follows:

THEOREM 2.1 ([11]). *An infinite countable graph $G = (V, E)$ without isolated nodes is Eulerian if and only if it satisfies the following conditions:*

(1) G is connected.
(2) G has a vertex of odd or infinite order.
(3) G has at most one vertex of odd order.
(4) G has only one end.

The restriction on graphs without isolated vertices can easily be lifted:

COROLLARY 2.2. *An infinite countable graph $G = (V, E)$ is Eulerian if and only if it satisfies the following conditions:*

(E1) G is edge-connected (i.e., for any two edges $\{u, v\}, \{u', v'\} \in E$, there exists a path with end points u and u').
(E2) G has a vertex of odd or infinite order.
(E3) G has at most one vertex of odd order.
(E4) G has only one end.

PROOF. First let P be an Euler path in G and let $V' \subseteq V$ denote the set of non-isolated vertices of G. Since P is an Euler path, the set V' is infinite and P is an Euler path in the graph $G' = (V', E)$ that therefore satisfies (1-4) from Theorem 2.1. Now (E1-4) follow immediately from (1-4).

[1] Note that every Eulerian path has infinitely many edges.

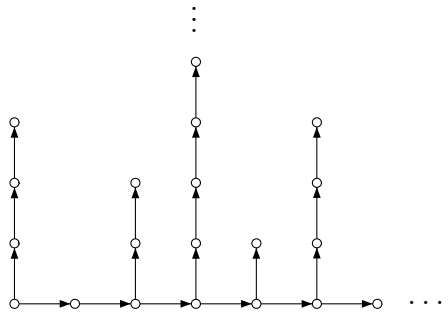

FIGURE 2. A comb with one infinite tooth, the only spine is the horizontal ray.

Conversely suppose G satisfies (E1-4) and let V', as before, be the set of non-isolated vertices. If G has a vertex of infinite order, then V' is infinite. If G is locally finite, it has (by (E2) and (E3)) a unique node of odd degree. But then G has infinitely many edges implying that V' is infinite. Since G satisfies (E1-4), the infinite graph (V', E) satisfies (1-4) from Theorem 2.1 and has therefore an Euler path P which is also an Euler path of G. ⊣

A *successor tree* is a *directed* graph $T = (V, E)$ such that there exists a root node $r \in V$ with the following properties:

- There does not exist $v \in V$ with $(v, r) \in E$.
- For every $v \in V$ there exists a unique path with end points r and v.

T is *n-branching* ($n \in \mathbb{N}$) if $|\{w \in V \mid (v, w) \in E\}| \leq n$ for all $v \in V$; it is *finitely branching* if $\{w \in V \mid (v, w) \in E\}$ is finite for all $v \in V$. An *infinite branch* in T is an infinite path $[v_0, v_1, v_2, \dots]$ in T, where v_0 is the root node of T. If T is a finitely branching successor tree, then the number of infinite branches of T equals the number of ends of the undirected graph $\overline{T}$.[2] A *comb* is a 2-branching successor tree that has an infinite branch containing all the branching points (i.e., all those vertices u with two distinct vertices v, w with $(u, v), (u, w) \in E$); any such infinite branch is called a *spine*, the complement of a (fixed) spine is formed of the *teeth*. Note that a comb may have at most two spines. Fig. 2 shows a comb.

An *order tree* is a partial order $(A, \preceq)$ such that there exists a least element $r \in A$ and the set $\{a \in A \mid a \preceq b\}$ is finite and linearly ordered for every $b \in A$. Alternatively, an order tree can be defined as the reflexive transitive closure of a successor tree. In the following, when we just speak of a tree, we always mean a successor tree.

Recall that a *Hamiltonian path* in a *finite* graph G is a finite path in G that visits every node of G exactly once. A *Hamiltonian path* of an *infinite* graph G is an infinite path in G that visits every node of G exactly once. In other words, it is an infinite path $[v_1, v_2, \dots]$ such that the mapping $i \mapsto v_i$ ($i \in \mathbb{N}$) is a bijection from $\mathbb{N}$ onto the set of nodes. Some authors call a Hamiltonian path

[2]This is false for arbitrary trees. A tree T may have infinitely many ends but not a single infinite branch.

of an infinite graph G a *spanning ray* of G. References on some graph theoretic results on Hamiltonian paths in infinite graphs can be found in [10].

2.3. Recursive graphs and automatic graphs. A *recursive (directed) graph* is a (directed) graph $G = (V, E)$ such that V and E are recursive subsets of $\mathbb{N}$ and $\binom{\mathbb{N}}{2}$ or $\mathbb{N}^2$, respectively. In case G is infinite, one can w.l.o.g. assume that $V = \mathbb{N}$. A recursive graph is *highly recursive* if it is locally finite and for every node v one can compute a list of v's finitely many neighbours. We will also consider a class of graphs in-between recursive and highly recursive graphs: A graph (V, E) is *very recursive* if (i) it is recursive and (ii) one can compute, from a node v, its order (which may be ∞). Note that G is highly recursive if and only if it is very recursive and locally finite (it does not suffice to require G to be recursive and locally finite). A recursive *directed* graph G is *highly recursive* if the graph $\overline{G}$ is highly recursive.

Next we introduce automatic graphs, more details can be found in [21, 8]. Let us fix $n \in \mathbb{N}$ and a finite alphabet Γ. Let $\# \notin \Gamma$ be an additional padding symbol. For words $w_1, w_2, \ldots, w_n \in \Gamma^*$ we define the *convolution* $w_1 \otimes w_2 \otimes \cdots \otimes w_n$, which is a word over the alphabet $(\Gamma \cup \{\#\})^n$, as follows: Let $w_i = a_{i,1}a_{i,2}\cdots a_{i,k_i}$ with $a_{i,j} \in \Gamma$ and $k = \max\{k_1, \ldots, k_n\}$. For $k_i < j \leq k$ define $a_{i,j} = \#$. Then $w_1 \otimes \cdots \otimes w_n = (a_{1,1}, \ldots, a_{n,1}) \cdots (a_{1,k}, \ldots, a_{n,k})$. Thus, for instance $aba \otimes bbabb = (a,b)(b,b)(a,a)(\#,b)(\#,b)$. An n-ary relation $R \subseteq (\Gamma^*)^n$ is called *automatic* if the language $\{w_1 \otimes \cdots \otimes w_n \mid (w_1, \ldots, w_n) \in R\}$ is a regular language.

Now let $\mathcal{A} = (A, (R_i)_{i \in J})$ be a relational structure with finitely many relations, where $R_i \subseteq A^{n_i}$. A tuple (Γ, L, h) is called an *automatic presentation* for $\mathcal{A}$ if

- Γ is a finite alphabet,
- $L \subseteq \Gamma^*$ is a regular language,
- $h : L \to A$ is a surjective function,
- the relation $\{(u, v) \in L \times L \mid h(u) = h(v)\}$ is automatic, and
- the relation $\{(u_1, \ldots, u_{n_i}) \in L^{n_i} \mid (h(u_1), \ldots, h(u_{n_i})) \in R_i\}$ is automatic for every $i \in J$.

This presentation is called *injective*, if h is injective (and hence bijective). We say that $\mathcal{A}$ is *automatic* if there exists an automatic presentation for $\mathcal{A}$. It is known that every automatic structure has an injective automatic presentation [21]. Since directed graphs are relational structures (with one binary relation), this defines the notion of an *automatic directed graph*. A graph (V, E) is *automatic* if the directed graph $(V, \{(u, v) \in V^2 \mid \{u, v\} \in E\})$ is automatic.

In contrast to recursive graphs, automatic graphs have some nice algorithmic properties. In [21] it was shown that every first-order definable relation in an automatic structure is effectively automatic (this result extends to first-order logic with infinity and modulo quantifiers [8, 24]). Hence, the first-order theory of every automatic structure is decidable. If (V, E) is an automatic graph, then for a given node $v \in V$ one can effectively compute a finite automaton that accepts the set of neighbours of v. Thus, an automatic graph is very recursive. In contrast to these positive results, several strong undecidability results (see e.g. [22, 35]) show that algorithmic methods for automatic structures are still quite limited.

A notational remark. A recursive (directed) graph G is determined by a pair of (Gödel indices of) Turing machines that decide the set of nodes and the set of edges, respectively, of G. A highly recursive (directed) graph G is given by (the Gödel index of) a Turing machine M that, on input of $n \in \mathbb{N}$, decides whether n is a node of G and in the positive case computes a tuple of the neighbors of n in G. Finally, an automatic directed graph is given by (a natural number that encodes) two finite automata that accept the set of nodes and edges, respectively.

In the following, we will often make statements like "For graphs from X, property Y belongs to $\mathcal{C}$" or " ... is $\mathcal{C}$-hard" where X is a class of graphs and $\mathcal{C}$ is some class from the arithmetical/analytical hierarchy. Formally, the first means "There is a set $L \in \mathcal{C}$ such that for every input M *that describes a graph* $G \in X$, we have $M \in L$ iff G has property Y". Similarly, the second means "For all $L \in \mathcal{C}$, there exists a computable function f such that, for all inputs w, $f(w)$ *describes a graph* $G \in X$, and $w \in L$ iff G has property Y".

Finally, we will always identify a finitary object (like words, tuples of words, Turing machines etc) with its Gödel number.

§3. Σ^1_1-complete problems. In this section we will concentrate on automatic graphs. For two classical problems, which are known to be Σ^1_1-complete for recursive graphs, we will prove Σ^1_1-completeness also for automatic graphs. In Section 3.1 we show that the existence of a Hamiltonian path in an automatic graph (which can even be assumed to be planar and of bounded degree) is Σ^1_1-complete. Finally, in Section 3.2 we prove Σ^1_1-completeness for the existence of an infinite path in an automatic (successor) tree.

3.1. Hamiltonicity for planar automatic graphs of bounded degree. For recursive graphs, the following is known

THEOREM 3.1. *The existence of a Hamiltonian path is Σ^1_1-complete for the following classes of graphs:*

- *highly recursive graphs of bounded degree (Harel [18])*
- *planar recursive graphs (Hirst and Harel [19])*

The aim of this section is to extend the results from [18, 19] to the class of planar *automatic* graphs of bounded degree. More precisely, we will prove:

THEOREM 3.2. *For planar automatic graphs of bounded degree, the existence of a Hamiltonian path is Σ^1_1-complete.*

Since every automatic graph of bounded degree is highly recursive, this will imply that the problem is still Σ^1_1-complete for the intersection of the two classes in Theorem 3.1.

Note that the upper bound Σ^1_1 in Theorem 3.2 follows immediately from the corresponding result for general recursive graphs (Theorem 3.1). For the lower bound we will use a special variant of the tiling problem [5, 37] that was introduced by Harel.

3.1.1. Tilings. Our main tool for proving Σ^1_1-hardness of the existence of a Hamiltonian path in planar automatic graphs of bounded degree is the *recurring tiling problem* [16, 17]:

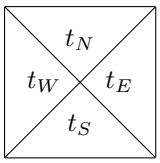

FIGURE 3. A tile

An instance of the recurring tiling problem consists of (i) a finite set of *colors* $C = \{c_0, c_1, \ldots, c_n\}$, (ii) a distinguished color c_0, and (iii) a set $\mathcal{T} \subseteq C^4$ of *tile types*. For a tile type $t \in \mathcal{T}$ we write $t = (t_W, t_N, t_E, t_S)$ ("W" for west, "S" for south, "N" for north, "E" for east), see Fig. 3 for a visualization.

A mapping $f : \mathbb{N}^2 \to \mathcal{T}$ is an *admissible tiling* if, for every $(i, j) \in \mathbb{N}^2$, we have $f(i, j)_N = f(i + 1, j)_S$ and $f(i, j)_E = f(i, j + 1)_W$. A *recurring tiling* is an admissible tiling f such that for infinitely many $j \in \mathbb{N}$, we have $f(0, j)_S = c_0$. Now the recurring tiling problem asks whether a given problem instance has a recurring tiling. Harel has shown the following result:

THEOREM 3.3 ([16]). *The recurring tiling problem is* Σ^1_1*-complete.*

The recurring tiling problem turned out be very useful for proving certain satisfiability problems in logic to be hard for Σ^1_1 [15].

In the rest of Section 3.1, we will reduce the recurring tiling problem to the existence of a Hamiltonian path in a planar automatic graph of bounded degree. This proves Theorem 3.2 by Theorem 3.3.

3.1.2. Building blocks. Let us first introduce several building blocks (gadgets) from which we will assemble our final planar automatic graph of bounded degree. These building blocks are variants of graphs taken from the NP-hardness proof for the Hamiltonian path problem in finite planar graphs [13].

Exclusive or. Consider the finite plane graph X in Fig. 4 (first picture). It has a Hamiltonian path from u_1 to u_2 (and similarly from v_1 to v_2) indicated in the second picture. Now suppose G' is some graph containing the edges u' and v'. Then we build a graph G as follows: in the disjoint union of G' and X, delete the edges u' and v' and connect their endpoints to u_1 and u_2 (to v_1 and v_2, resp., see Fig. 4, third picture). Now suppose H is a Hamiltonian path in G with no endpoint in X. Suppose u_1 is the first vertex from X in H. Then the restriction of H to X has to coincide with the Hamiltonian path from u_1 to u_2. Hence H gives rise to a Hamiltonian path H' in G' that coincides with H on G but passes through the edge u' instead of taking the detour through X. Note that H' cannot contain the edge v'. Conversely, every Hamiltonian path H' of G' that contains the edge u' but not the edge v' induces a Hamiltonian path H of G in a similar way. Joining X to the graph G' in this manner restricts the Hamiltonian paths to those that either contain the edge u' or the edge v', but not both. This also explains the name X: this graph acts as an "exclusive-or". Note that, if G' is planar and the two edges u' and v' belong to the same face,

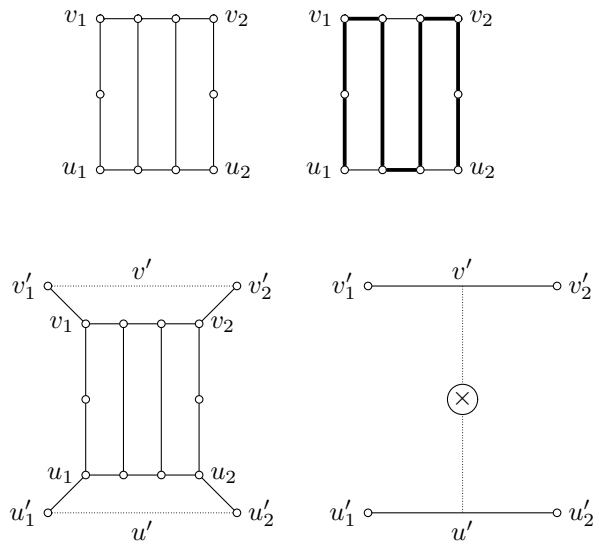

FIGURE 4. The graph X, its use and abbreviation

then also G is planar. Since we will make repeated use of this construction, we abbreviate it as in Fig. 4, fourth picture.

Boolean functions. In the following, we abbreviate the interval $\{1, \dots, n\}$ with $[n]$. Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function. Then [13] constructs a planar graph (V, E) together with distinguished edges $e_1, \dots e_n \in E$ such that $f(b_1, \dots, b_n) = 1$ iff (V, E) has a Hamiltonian cycle H with $\{i \in [n] \mid b_i = 1\} = \{i \in [n] \mid e_i \in H\}$. Since the graph can be constructed in polynomial time from a Boolean formula in 3CNF, NP-completeness of the existence of a Hamiltonian path follows. Here, we modify the construction from [13] slightly in order to place the edges e_i and two vertices u and v in a specified order at the boundary of the outer face of (V, E).

THEOREM 3.4. *There exists a constant c such that for given $k, \ell, n \in \mathbb{N}$ and $F \subseteq 2^{[k+\ell+n]}$, a finite plane graph G_F of degree at most c with the following properties can be constructed:*

(1) At the boundary of the outer face, we find (in this counter-clockwise order) edges $e_1, \dots e_k$, a vertex u, edges $e_{k+1}, \dots, e_{k+\ell}$, a vertex v, and edges $e_{k+\ell+1}, \dots, e_{k+\ell+n}$.

(2) For every $M \subseteq [k + \ell + n]$, $M \in F$ iff there is a Hamiltonian path H from u to v in G_F such that $M = \{i \mid e_i \text{ belongs to } H\}$.

PROOF. First, the lemma is shown for $k = n = 0$. To achieve this, build a propositional formula φ with variables $x_1, \dots, x_\ell$ that describes the set F (i.e., $\varphi(b_1, \dots, b_\ell) = 1$ iff $\{i \mid b_i = 1\} \in F$). To this formula, apply the construction from [13] whose result G is depicted at [13, page 711]. We simplify this graph as follows: for each variable x_i, the graph G contains the left graph from Fig. 5. Replace this occurrence by the right graph from Fig. 5. The resulting graph

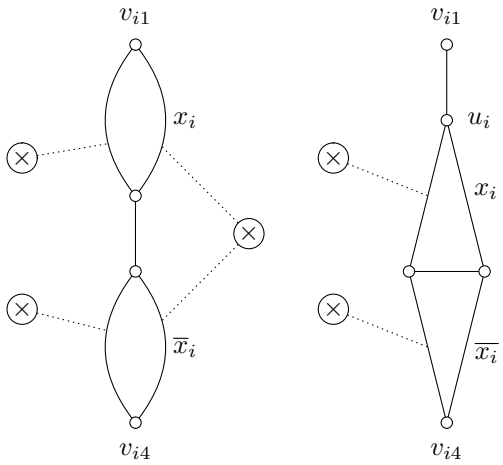

FIGURE 5. Simplification of the graph from [13]

is G_F. Then the proof from [13] yields for all $M \subseteq [\ell]$: $M \in F$ iff G_F has a Hamiltonian cycle C such that $M = \{i \mid C \text{ passes through the edge labeled } x_i\}$.

Let e_i denote the edge labeled x_i, let $v = v_{11}$ and $u = u_1$. From [13, page 711] one observes that G has a face whose boundary contains all the edges e_i and the vertices v_{i1} and we can assume that they appear in counter-clockwise order $e_1, e_2, \ldots, e_\ell$. Considering this face as the outer face, we obtain the picture from Fig. 6 (for $\ell = 6$) where the boundary of the outer face is indicated.

Now consider the general case $k, \ell, n \geq 0$. Then the above construction yields a graph G_0 satisfying (2) for the tuple $(0, k+\ell+n, 0)$ (see Fig. 6 for $k = \ell = n = 2$). The construction of G_F from this graph is indicated in Fig. 7. Note that in the graph G_F, along the boundary of the outer face, we find in counter-clockwise order the edges and nodes

$$e'_1, e'_2, \ldots, e'_k, u', e'_{k+1}, e'_{k+2}, \ldots, e'_{k+\ell}, v', e'_{k+\ell+1}, e'_{k+\ell+2}, \ldots, e'_{k+\ell+n} \ .$$

First, let H be a Hamiltonian path from u' to v' in G_F. Consider some i with $1 \leq i \leq k$ or $k + \ell < i \leq k + \ell + n$. Then the path H contains the edge e'_i iff it contains the edge e_i (more precisely, iff it enters and leaves the corresponding copy of the XOR-graph X from the endpoints of the edge e_i). In addition, the restriction H_0 of H to G_0 is a Hamiltonian path from u to v in G_0. Hence we have $\{i \mid e'_i \text{ belongs to } H\} = \{i \mid e_i \text{ belongs to } H_0\} \in F$. Conversely let $M \in F$. Then there exists a Hamiltonian path H_0 from u to v in G_0 such that $M = \{i \mid e_i \text{ belongs to } H_0\}$. This path can be extended uniquely to a Hamiltonian path H from u' to v' in G_F such that e'_i belongs to H iff e_i belongs to H_0. Hence we get $M = \{i \mid e'_i \text{ belongs to } H\}$. ⊣

Infinity checking. Next consider Fig. 8 – it depicts a graph A that is connected to some context via the edges a, b, ℓ, ℓ', r, and r'. If the complete graph has a Hamiltonian path, then locally, it has to be of one of the four forms depicted in Fig. 9.

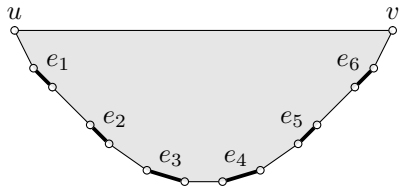

FIGURE 6. Graph G_0 from the proof of Thm. 3.4

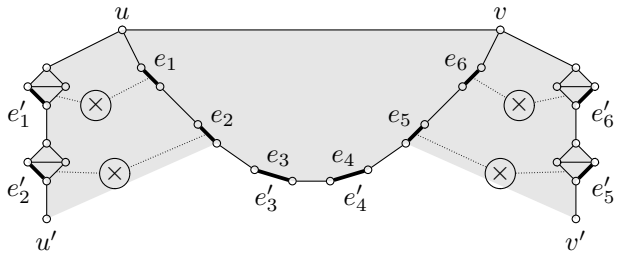

FIGURE 7. Graph G from the proof of Thm. 3.4

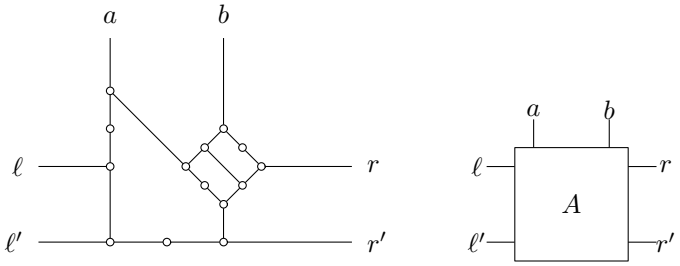

FIGURE 8. The graph A and its abbreviation

Now consider Fig. 10, where we define a planar graph L that consists of infinitely many copies of the graph A arranged in a line. Suppose L is part of a graph G such that only the a-edges and b-edges of the copies of A connect L and its infinite complement in G. Suppose furthermore that G has a Hamiltonian path H. Then H has to enter and leave L infinitely often. Since the possibilities to pass A are restricted as shown in Fig. 9, any such visit has to look as described in Fig. 11, i.e., the path enters from a into some copy of A, moves left to some copy of A (possibly without doing any step), moves all the way back until it leaves the initial A-copy via its b-edge.

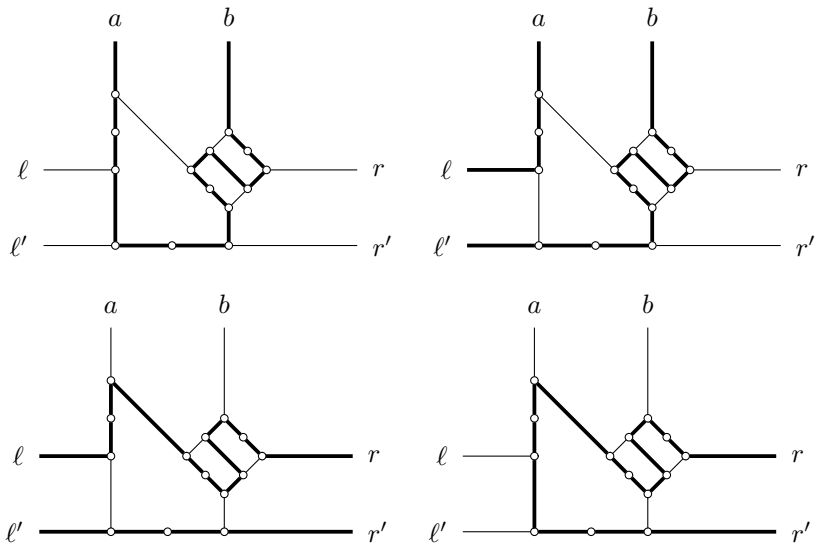

FIGURE 9. Paths through the graph A

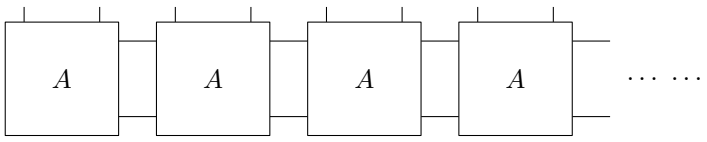

FIGURE 10. The infinite graph L

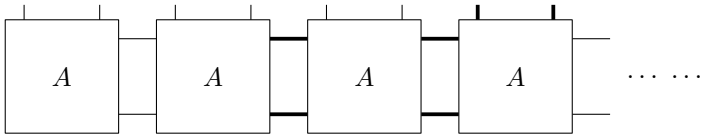

FIGURE 11. A visit of a Hamiltonian path to the graph L

3.1.3. Assembling. From an instance of the recurring tiling problem, we construct in this section a planar graph G of bounded degree that has a Hamiltonian path iff the instance of the recurring tiling problem admits a solution. In the next section, we will argue that G is automatic.

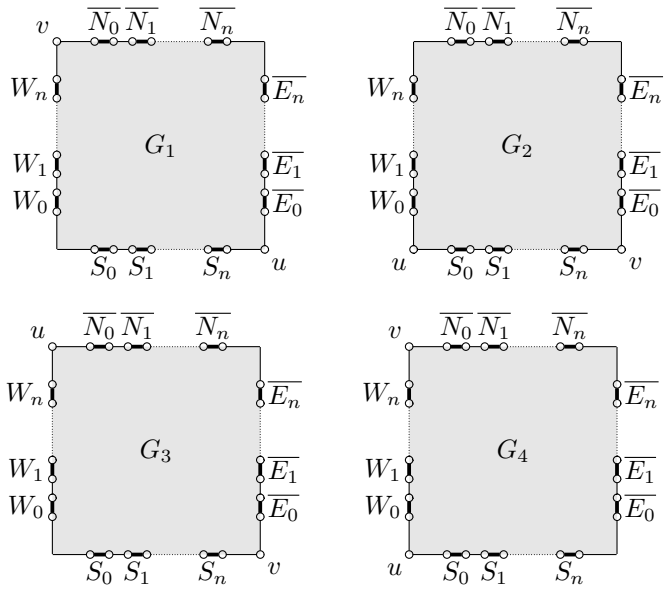

FIGURE 12. The graphs G_x

So, we fix a finite set $C = \{c_0, c_1, \ldots, c_n\}$ of colors, a distinguished color c_0, and a set $\mathcal{T} \subseteq C^4$ of tile types.

Next let

$$\mathcal{V} = \{W_0, W_1, \ldots, W_n, S_0, S_1, \ldots, S_n, \overline{N_0}, \overline{N_1}, \ldots, \overline{N_n}, \overline{E_0}, \overline{E_1}, \ldots, \overline{E_n}\}.$$

We will describe tile types by certain subsets of $\mathcal{V}$ where W_i expresses that the left color is c_i, and $\overline{N_i}$ denotes that the top color is *not* c_i (S_i and $\overline{E_i}$ refer to the bottom and right color and are to be understood similarly). More precisely, the tile $d = (c_i, c_j, c_k, c_\ell)$ is denoted by the set

$$\mathbb{S}_d = \{W_i\} \cup \{\overline{N_m} \mid m \neq j\} \cup \{\overline{E_m} \mid m \neq k\} \cup \{S_\ell\} \ .$$

Now let $F = \{\mathbb{S}_d \mid d \in \mathcal{T}\}$ be the descriptions of all the tile types d in $\mathcal{T}$. Then, by Theorem 3.4, there are finite plane graphs G_1, G_2, G_3, and G_4 with the following properties:

1. at the outer face, we find edges e for $e \in \mathcal{V}$ and nodes u and v in the order indicated in Fig. 12
2. $M \in F$ iff there exists a Hamiltonian path H of G_x from u to v such that $M = \{v \in \mathcal{V} \mid v \text{ belongs to } H\}$ (for all $1 \leq x \leq 4$ and $M \subseteq \mathcal{V}$).

Next we choose mutually disjoint graphs $G(k, \ell)$ (for $k, \ell \in \mathbb{N}$) such that

$$G(k,\ell) \cong \begin{cases} G_1 & \text{if } k+\ell \text{ is even and } k > 0 \\ G_2 & \text{if } k+\ell \text{ is odd and } \ell = 0 \\ G_3 & \text{if } k+\ell \text{ is odd and } \ell > 0 \\ G_4 & \text{if } k+\ell \text{ is even and } k = 0 \ , \end{cases}$$

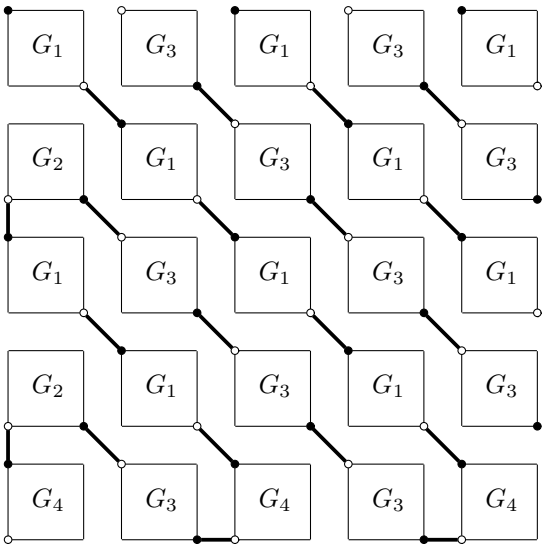

FIGURE 13. First step in global construction - the graph G^1

see Fig. 13 where the square that is reached by going k steps up and ℓ steps to the right represents $G(k, \ell)$.

Then $u(k, \ell)$ and $v(k, \ell)$ refer to the nodes u and v of the graph $G(k, \ell)$; similarly, $e(k, \ell)$ for $e \in \mathcal{V}$ refers to the edge e of the graph $G(k, \ell)$. In the disjoint union of these graphs $G(k, \ell)$, we connect the node $v(k, \ell)$ by a new edge with the following node:

$$\begin{aligned} u(k+1, \ell) &\text{ for } k+\ell \text{ even and } \ell = 0 \\ u(k+1, \ell-1) &\text{ for } k+\ell \text{ even and } \ell > 0 \\ u(k-1, \ell+1) &\text{ for } k+\ell \text{ odd and } k > 0 \\ u(k, \ell+1) &\text{ for } k+\ell \text{ odd and } k = 0 \end{aligned}$$

The result G^1 of this construction is visualized in Fig. 13 where the vertices $u(k, \ell)$ are denoted by empty nodes and $v(k, \ell)$ by filled nodes.

Next, for all $k, \ell \in \mathbb{N}$ and $0 \leq i \leq n$, we replace the edges $\overline{E_i}(k, \ell)$ and $W_i(k, \ell+1)$ by a copy of the exclusive-or graph X. Similarly, the edges $\overline{N_i}(k, \ell)$ and $S_i(k+1, \ell)$ are replaced by a copy of the graph X, see Fig. 14 for a visualization. We denote the resulting graph with G^2.

In a third step, we add to G^2 the graph L from Fig. 10. To connect it to the graph constructed so far, the start node of the edge a of the i^{th} copy of A in L is the left node of the edge $S_0(0, i)$; analogously, the start node of b is the right node of $S_0(0, i)$. The resulting graph is referred to as G^3.

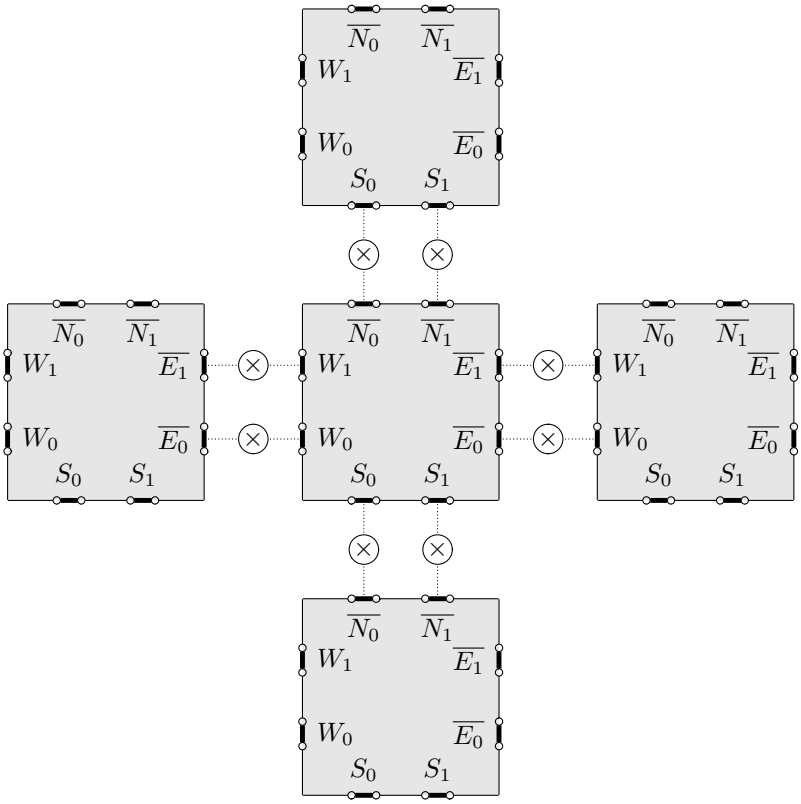

FIGURE 14. Second step in global construction – the graph G^2 (for two colors c_0 and c_1)

Finally, we add to G^3 a new node $\perp$ together with an edge between $\perp$ and $u(0,0)$. Since this is really the final graph, we call it G.

We now claim that the constructed graph G has a Hamiltonian path if and only if the set of tile types $\mathcal{T}$ admits a recurring tiling. First suppose there is a recurring tiling $f : \mathbb{N} \times \mathbb{N} \to \mathcal{T}$. Let $k, \ell \in \mathbb{N}$ and $f(k,\ell) = (c_W, c_N, c_E, c_S)$. Then the graph $G(k,\ell) \in \{G_x \mid 1 \leq i \leq 4\}$ has a Hamiltonian path $H(k,\ell)$ from $u(k,\ell)$ to $v(k,\ell)$ such that for all $1 \leq i \leq n$

1. the edge S_i belongs to $H(k,\ell)$ iff $c_S = c_i$,
2. the edge W_i belongs to $H(k,\ell)$ iff $c_W = c_i$,
3. the edge $\overline{N_i}$ belongs to $H(k,\ell)$ iff $c_N \neq c_i$, and
4. the edge $\overline{E_i}$ belongs to $H(k,\ell)$ iff $c_E \neq c_i$.

Then we find a Hamiltonian path H_1 of the infinite graph G^1 in Fig. 13 by concatenating these Hamiltonian paths suitably:

$$H_1 = H(0,0), H(1,0), H(0,1), H(0,2), H(1,1), H(2,0) \ldots$$

Since the tiling f is admissible, we get

$$\begin{aligned}\overline{E_i}(k,\ell) \notin H_1 &\iff f(k,\ell)_E = c_i \\ &\iff f(k,\ell+1)_W = c_i \\ &\iff W_i(k,\ell+1) \in H_1\end{aligned}$$

and similarly $\overline{N_i}(k,\ell) \notin H_1$ iff $S_i(k+1,\ell) \in H_1$. Hence the Hamiltonian path H_1 can be extended to a Hamiltonian path H_2 of the graph G^2 obtained from G^1 by adding all the copies of the exclusive-or graph X. Observe also that f is recurring, i.e., there are infinitely many $\ell \in \mathbb{N}$ with $f(0,\ell)_S = c_0$. For every such ℓ, the path H_1 passes through the edge $S_0(0,\ell)$. Instead of passing through this edge, we now enter the graph L (Fig. 10) via the edge a of the ℓ^{th} copy of A and leave it via its edge b. We can ensure that after this visit, all nodes of L to the left of the ℓ^{th} copy of A have been visited (cf. Fig. 11). This results in a Hamiltonian path H_3 of the graph G^3 starting in $u(0,0)$. Prepending the node $\perp$ gives a Hamiltonian path H of the final graph G.

Conversely, let H be a Hamiltonian path of the final graph G. Since $\perp$ has degree 1, it has to start in $\perp$ – deleting $\perp$ from H gives a Hamiltonian path H_3 of G^3 that starts in $u(0,0)$. Since G^3 contains infinitely many nodes outside of L, this path has to enter and leave L infinitely often. Any such visit has to enter via the edge a of some copy of A and leave via the edge b of the same copy of A (or vice versa, see Fig. 11). Hence, deleting all the vertices of L from the path H, we obtain a Hamiltonian path H_2 of the graph G^2 that contains infinitely many edges of the form $S_0(0,\ell)$. Recall that G^2 is obtained from G^1 by replacing some pairs of edges by the exclusive-or graph X. Hence, the restriction of H_2 to the nodes of G^1 gives rise to a Hamiltonian path H_1 of G^1 that

(a) contains infinitely many edges of the form $S_0(0,\ell)$,
(b) contains the edge $W_i(k,\ell+1)$ iff it does not contain the edge $\overline{E_i}(k,\ell)$, and
(c) contains the edge $S_i(k+1,\ell)$ iff it does not contain the edge $\overline{N_i}(k,\ell)$

for all $0 \leq i \leq n$ and $k,\ell \in \mathbb{N}$. Since H_1 has to pass through all the graphs $G(k,\ell)$, it has to be of the form

$$H(0,0), H(1,0), H(0,1), H(0,2), H(1,1), H(2,0) \ldots$$

where $H(k,\ell)$ is a Hamiltonian path of the graph $G(k,\ell)$ from $u(k,\ell)$ to $v(k,\ell)$. Now we are ready to define the mapping $f : \mathbb{N}^2 \to C^4$: set

(1) $f(k,\ell)_W = c_i$ iff $H(k,\ell)$ contains the edge $W_i(k,\ell)$,
(2) $f(k,\ell)_N = c_i$ iff $H(k,\ell)$ does not contain the edge $\overline{N_i}(k,\ell)$,
(3) $f(k,\ell)_E = c_i$ iff $H(k,\ell)$ does not contain the edge $\overline{E_i}(k,\ell)$, and
(4) $f(k,\ell)_S = c_i$ iff $H(k,\ell)$ contains the edge $S_i(k,\ell)$.

Since $H(k,\ell)$ is a Hamiltonian path of $G(k,\ell)$ from $u(k,\ell)$ to $v(k,\ell)$, we get $f(k,\ell) \in \mathcal{T}$ from the construction of the graphs G_1, G_2, G_3, G_4. We get

$$\begin{aligned}f(k,\ell)_E = c_i &\overset{(3)}{\iff} \overline{E_i}(k,\ell) \text{ does not belong to } H(k,\ell) \\ &\overset{(b)}{\iff} W_i(k,\ell+1) \text{ belongs to } H(k,\ell+1) \\ &\overset{(1)}{\iff} f(k,\ell+1)_W = c_i\end{aligned}$$

and similarly $f(k,\ell)_N = f(k+1,\ell)_S$ follows from (2), (c), and (4). Thus, f is an admissible tiling. Since H_1 contains infinitely many edges of the form $S_0(0,\ell)$, there are infinitely many $\ell \in \mathbb{N}$ such that $S_0(0,\ell)$ belongs to $H(0,\ell)$, i.e., $f(0,\ell)_S = c_0$.

Thus, we showed that indeed the graph G contains a Hamiltonian path iff the set of tiles $\mathcal{T}$ admits a recurring tiling.

REMARK 3.5. There also exists the variant of two-way Hamiltonian paths in infinite graphs. A two-way Hamiltonian path in $G = (V,E)$ is a two-way infinite sequence $p = [\dots, v_{-1}, v_0, v_1, \dots]$ such that $(v_i, v_{i+1}) \in E$ for all $i \in \mathbb{Z}$ and for every node $v \in V$ there exists exactly one $i \in \mathbb{Z}$ such that $v = v_i$. From the previous construction, it follows that also the question whether a given planar automatic graph of bounded degree has a two-way Hamiltonian path is Σ^1_1-complete. Take the disjoint union of two copies of our main graph G and connect the two $\perp$-nodes with an edge. The resulting graph G' has a two-way Hamiltonian path if and only if G has a (one-way) Hamiltonian path. Moreover, since G is automatic (see next Section), and the class of automatic graphs is closed under disjoint unions and finite distortion, G' is automatic as well.

3.1.4. G is automatic. Clearly, the graph G constructed in the previous section is planar and has bounded degree. So, it remains to prove that it is automatic.

Note that the graph G has a highly regular structure. It results from the infinite grid $\mathbb{N} \times \mathbb{N}$ by replacing each grid point by a finite graph and connecting these finite graphs in a regular pattern. It is not surprising that such a graph is automatic, in particular since the grid is automatic. In this section, we will provide some more formal arguments for the automaticity of G.

Recall that G can be obtained from $\mathbb{N} \times \mathbb{N}$ by replacing every grid point $(k,\ell) \in \mathbb{N} \times \mathbb{N}$ by a finite graph $G'(k,\ell)$. This graph is one of the graphs G_1, G_2, G_3, G_4 together with copies of the XOR-graph X that connect $G(k,\ell)$ with $G(k+1,\ell)$ and $G(k,\ell+1)$. For $k = 0$, we moreover have to add a copy of the graph A from Fig. 8. Whether and how $G'(k,\ell)$ is obtained from G_i only depends on the parity of $k+\ell$ (i.e., the parity of k and ℓ) and whether k and ℓ are zero or non-zero, respectively.

The alphabet of our presentation consists of the letters $\triangleright$, $\triangleleft$, a, b, and the nodes of the graphs G_1, G_2, G_3, and G_4 (extended by the XOR-graphs and the graph A). Then, the node set of G can be represented by the regular language

$$L = \{\triangleright [ab]_k v ([ab]_\ell)^{\mathrm{rev}} \triangleleft \mid k,\ell \geq 0,\ v \text{ is a node of } G'(k,\ell)\}$$

where $[w]_k$ is the prefix of length k of the infinite word w^ω and w^{rev} is the word w reversed. Then k is odd iff $[ab]_k$ ends with a, $k = 0$ iff $\triangleright [ab]_k = \triangleright$ and similarly parity and zeroness of ℓ can be described. Hence L is regular. Now let $V = \triangleright [ab]_k v ([ab]_\ell)^{\mathrm{rev}} \triangleleft$ and $W = \triangleright [ab]_m w ([ab]_n)^{\mathrm{rev}} \triangleleft$ be two such words from L. If $|k-m| > 1$ or $|\ell - n| > 1$, then $\{V,W\}$ is no edge of our graph. Otherwise, it only depends on the letters v and w and their neighboring letters in V and W, resp., whether $\{V,W\} \in E$. Hence, one can define a finite (symmetric) semi-Thue system R such that

- $V \leftrightarrow_R W$ iff $\{V,W\}$ is an edge of G, for all $V, W \in L$, and

- if $V \leftrightarrow_R W \in L$, then $V \in L$ for all words V and W (i.e., L is a regular connected component of the rewrite graph associated with R).

This semi-Thue system can be translated into a synchronous automaton that recognizes the edge set of G.

Hence we reduced the recurring tiling problem to the existence of a Hamiltonian path in a planar automatic graph of bounded degree. Because of Theorem 3.3, this proves Theorem 3.2.

3.2. Infinite paths in automatic trees. The fundamental Σ^1_1-complete problem in recursion theory is the existence of an infinite branch in a recursive tree where it does not make a difference whether the tree is an order or a successor tree [33, Thm. 16.XX]. Surprisingly, existence of an infinite branch is decidable for automatic *order* trees [25]. Here, we show that the problem is Σ^1_1-complete for automatic *successor* trees. The proof idea is to transform a recursive successor tree into an automatic one by adding the computation (i.e., sequence of transitions) that verifies the edge (u, v) as a path between the nodes u and v; a similar idea was used in [20, 25].

THEOREM 3.6. *For automatic successor trees, existence of an infinite branch is* Σ^1_1*-complete.*

PROOF. Membership in Σ^1_1 is easy to see. We prove the lower bound in three steps. We start with the following well known Σ^1_1-complete problem P_1 [17]:

INPUT: A *nondeterministic* Turing machine M with a distinguished state q_r.
QUESTION: Does M have an infinite computation starting with a blank tape and the initial state that visits the state q_r infinitely often?

In a first step we reduce this problem to the following problem P_2:

INPUT: A *nondeterministic* Turing machine M with a distinguished state q_r.
QUESTION: Does there exist a configuration c such that M has an infinite computation starting in c that visits the state q_r infinitely often?

Let M be a nondeterministic Turing machine with a distinguished state q_r. We construct a nondeterministic Turing machine M' by modifying M as follows: The machine M' has an additional tape T, where a sequence of transitions of M is stored. This sequence of M-transitions is simulated by M' step by step (this phase is deterministic). For this, a pointer to the additional tape T is moved one cell to the right after every step. After the last transition from T is simulated, the machine M' continues to simulate M nondeterministically until a configuration with control state q_r is reached (in this phase, the new transitions are added to the content of T). Once the control state q_r is assumed, the pointer to T is set back to the left end of T and the tape, where the current M-configuration is stored, is set back to the initial blank configuration of M. This modification ensures that when starting M' in an arbitrary configuration, it either blocks after finitely many steps or it will finally start simulating nondeterministically a computation of M that starts with the initial blank configuration. Hence, M' allows a computation starting from some configuration that visits the state q_r infinitely many times if and only if M allows a computation starting from the initial blank configuration that visits q_r infinitely often. Hence the problem P_2 is Σ^1_1-hard.

In the second step, we reduce P_2 to the following problem P_3:

INPUT: An automatic structure $T = (V, E, U)$, where (V, E) is a successor tree and $U \subseteq V$ is a unary relation.
QUESTION: Does (V, E) contain an infinite branch that visits infinitely many nodes from U?

Let M be a nondeterministic Turing machine with state set Q, tape alphabet Γ, and set of instructions $\delta \subseteq Q \times \Gamma \times Q \times \Gamma \times \{L, R\}$. We will construct an automatic structure $T = (L, E, U)$, which has the above properties if and only if M has an infinite computation starting from some configuration that visits the state $q_r \in Q$ infinitely often.

As usual, configurations of M are encoded by words from the regular language $C = \Gamma^* Q \Gamma^+$.[3] Let $\mathrm{Pref}(C)$ be the set of all prefixes of words from C. For configurations $c, d \in C$ and an instruction $t \in \delta$ we write $c \vdash_t d$ if c evolves to d by executing instruction t; this is an automatic relation. Let $\Sigma = Q \cup \Gamma$ (we assume that $Q \cap \Gamma = \emptyset$) and let $\Sigma' = \{a' \mid a \in \Sigma\}$ be a disjoint copy of Σ. For a configuration $c \in C$ and $1 \leq i \leq \ell = |c|$ let $c[i]$ be the i^{th} symbol in c and let

$$c^{(i)} = c[1] \cdots c[i-1] c[i]' c[i+1] \cdots c[\ell],$$

i.e., $c^{(i)}$ results from c by replacing the i^{th} symbol in c by its primed copy. Finally, every instruction $t \in \delta$ becomes an additional symbol.

We will define an automatic presentation for a tree, which is basically a "stretched" version of the computation tree of M, where every edge of the computation tree is replaced by a finite branch.[4] Let

$$\begin{aligned} L \;&= (C \cdot \delta)^+ (C \cup \{\varepsilon\}) \\ &\quad \cup\; (C \cdot \delta)^* \cdot \{c^{(i)}\, t\, d \mid c \in C, 1 \leq i \leq |c|, t \in \delta, d \in \mathrm{Pref}(C)\}, \end{aligned}$$

which is regular. On L we define an automatic relation $\to$ as the smallest relation such that the following holds, where $n \geq 1$, $c, c_1, \ldots, c_n \in C$, $t, t_1, \ldots, t_n \in \delta$, $1 \leq i \leq \ell = |c_n|$, and $d \in \mathrm{Pref}(C)$:

$$c_1 t_1 \cdots c_{n-1} t_{n-1} c_n t_n \to c_1 t_1 \cdots c_{n-1} t_{n-1} c_n^{(1)} t_n \tag{1}$$

$$c_1 t_1 \cdots c_{n-1} t_{n-1} c_n^{(i)} t_n d \to c_1 t_1 \cdots c_{n-1} t_{n-1} c_n^{(i+1)} t_n d c_n[i] \quad \text{if } i < \ell \tag{2}$$

$$c_1 t_1 \cdots c_{n-1} t_{n-1} c_n^{(\ell)} t_n d \to c_1 t_1 \cdots c_{n-1} t_{n-1} c_n t_n d c_n[\ell] \tag{3}$$

$$c_1 t_1 \cdots c_{n-1} t_{n-1} c_n t_n d \to c_1 t_1 \cdots c_{n-1} t_{n-1} c_n t_n c\, t \quad \text{if } d \vdash_{t_n} c \tag{4}$$

The relation $\to$ is automatic, basically because every relation $\vdash_t$ is an automatic relation. With (1), (2), and (3) we copy the last configuration c_n, whereas (4) executes the last instruction t_n and guesses a new instruction t. Let $U = \{w \in L \mid q_r \text{ appears in the last configuration of } w\}$. The graph $(L, \to)$ is easily seen to be a forest such that M has a computation, where the state q_r appears infinitely often if and only if $(L, \to)$ has an infinite branch that contains infinitely many nodes from U. Finally, consider the successor tree $(L \cup \{\varepsilon\}, E)$ where

$$E = \{(\varepsilon, u) \mid u \in L, \neg \exists v : v \to u\} \cup \to,$$

[3] Here and in the following, A^+ denotes the set of all non-empty words over the set A.
[4] Similar ideas were used in [22, 20] to encode complex behaviour in automatic graphs.

i.e., ε becomes a new root together with edges to all nodes from L that do not have incoming $\rightarrow$-edges. Then M has a computation, where the state q_r appears infinitely often if and only if $(L \cup \{\varepsilon\}, E)$ has an infinite branch that contains infinitely many nodes from U. The relation E is again automatic since $\{u \in L \mid \neg\exists v : v \rightarrow u\}$ is regular.

In the final step, we reduce P_3 to the following problem P_4, i.e., to the problem whose Σ^1_1-hardness we want to prove:

INPUT: An automatic successor tree (V, E)
QUESTION: Does (V, E) contain an infinite branch?

Let us fix an automatic successor tree (L, E) and a regular subset $U \subseteq L$ of nodes. W.l.o.g. assume that the root node of the tree is ε (this is the case in the tree constructed in the previous paragraph) and that $\varepsilon \in U$. Let

$$L' = L \otimes a^*$$

for some new symbol a. The number n in a word $w \otimes a^n \in L \otimes a^*$ serves as a counter that gives the remaining steps until the set U is visited next. Let

$$\begin{aligned} E' \;=\; & \{(u \otimes a^n, v \otimes a^{n-1}) \mid n \geq 1, (u, v) \in E\} \;\cup \\ & \{(u \otimes \varepsilon, v \otimes a^n) \mid u \in U, (u, v) \in E, n \geq 0\}. \end{aligned}$$

This relation is automatic and (L', E') is a tree with root node $\varepsilon \otimes \varepsilon$. Moreover, we claim that there exists an infinite E-branch that visits infinitely many times the set U if and only if there exists an infinite E'-branch. First assume that $[w_1, w_2, w_3, \dots]$ $(w_1 = \varepsilon)$ is an infinite E-branch such that there exist $i_1 < i_2 < i_3 < \cdots$ with $w_{i_k} \in U$ for all $k \geq 1$. Since $\varepsilon \in U$ we can choose $i_1 = 1$. Let p_k be the finite E'-path $[w_{i_k} \otimes \varepsilon, w_{i_k+1} \otimes a^{i_{k+1}-i_k-1}, \dots, w_{i_{k+1}-1} \otimes a]$. Then $p_1, p_2, p_3, \dots$ is an infinite E'-branch. On the other hand, if $[w_1 \otimes a^{i_1}, w_2 \otimes a^{i_2}, \dots]$ is an infinite E'-branch, then there exist $k_1 < k_2 < \cdots$ such that $i_{k_j} = 0$ for all $j \geq 1$. Hence, $w_{k_j} \in U$ for all $j \geq 1$. Moreover, $[w_1, w_2, \dots]$ is an infinite E-branch. $\dashv$

§4. Problems in the arithmetic hierarchy. In this section, we will consider several graph problems, which turn out to be complete for certain levels of the arithmetic hierarchy, both for recursive and automatic graphs. More precisely, we will consider the following problems:

- Does a given graph have an Euler path?
- Does a given graph have at most k ends (for some number k)?
- Does a given finitely branching tree have at most k infinite branches (for some number k)?
- Does a given finitely branching tree have infinitely many infinite branches?

We will study the complexity of these questions for both recursive and automatic graphs. While these problems are still undecidable for automatic graphs, they will turn out to be easier for automatic graphs than for recursive graphs (w.r.t. the level in the arithmetic hierarchy).

4.1. Upper bounds in the arithmetic hierarchy. Recall the definition of very recursive graphs from Section 2.3. The following proposition gives upper bounds for testing whether a given (very) recursive graph has at most k ends.

These upper bounds (for $k = 1$) will be crucial for our upper bounds concerning Euler paths.

PROPOSITION 4.1. *Let $k > 0$.*

(1) For recursive graphs, the existence of at most k ends belongs to Π^0_3.
(2) For very recursive graphs, the existence of at most k ends belongs to Π^0_2.

PROOF. (1) Consider the following Π^0_3-formula

$$\forall H \subseteq E \text{ finite } \forall x_0, x_1, \ldots, x_k \in V :$$
$$\bigvee_{0 \le i < j \le k} \exists \text{ path from } x_i \text{ to } x_j \text{ in } (V, E \setminus H) \quad \vee$$
$$\exists Z \subseteq V \text{ finite} : \Big(\bigvee_{0 \le i \le k} x_i \in Z \Big) \wedge \forall \{z_1, z_2\} \in E \setminus H (z_1 \in Z \leftrightarrow z_2 \in Z),$$

expressing that of every $k+1$ vertices $x_0, \ldots x_k$, two belong to the same connected component or one belongs to some finite connected component of $(V, E \setminus H)$. Hence, this formula expresses that there are at most k ends.

(2) Now assume (V, E) to be very recursive and let $H \subseteq E$ and $Z \subseteq V$ be finite. If Z contains a vertex of infinite order, it cannot be a union of connected components of $(V, E \setminus H)$. Otherwise, for each $z \in Z$, one can compute a list of its neighbours in (V, E). Then, one can check effectively, whether there exist $z_1 \in Z$ and $\{z_1, z_2\} \in E \setminus H$ with $z_2 \in V \setminus Z$. Thus, the property to have at most k ends is in Π^0_2. ⊣

The characterization of Eulerian infinite graphs from Cor. 2.2 together with Prop. 4.1 gives the following upper bounds:

PROPOSITION 4.2. *The following holds:*

(1) For recursive graphs, existence of an Euler path is in D^0_3.
(2) For locally finite recursive graphs, existence of an Euler path is in Π^0_3.
(3) For very recursive graphs, existence of an Euler path is in Π^0_2.

PROOF. (1) It is an easy exercise to express (E1) from Cor. 2.2 in Π^0_2. The existence of a vertex of odd order is expressible in Σ^0_2 and the existence of a vertex of infinite order in Σ^0_3. Hence (E2) is in Σ^0_3 and (E3) in Π^0_2. By Prop. 4.1, (E4) is expressible in Π^0_3. Hence the existence of an Euler path is a conjunction of Σ^0_3- and Π^0_3-properties and therefore in D^0_3.

(2) This holds since all the properties in the previous paragraph except the existence of a vertex of infinite order are in Π^0_3.

(3) Assume (V, E) to be very recursive. Then the number of neighbors of a node x is computable from x. Hence (E2) is expressible in Σ^0_1 and (E3) in Π^0_1. Again, by Prop. 4.1, (E4) is expressible in Π^0_2. ⊣

4.2. Lower bounds in the arithmetic hierarchy. In this section we will establish lower complexity bounds. We will present two constructions (Lemma 4.3 and 4.10), from which all but one of the lower bounds will be deduced.

4.2.1. Recursive graphs. Our first main construction concerns recursive graphs. Recall that $\overline{G}$ is the undirected version of the directed graph G.

LEMMA 4.3. *From two Turing machines M_1 and M_2, one can compute a connected recursive directed graph $G(M_1, M_2)$ such that*

(a) $M_1 \in \mathrm{COF}$ *if and only if* $\overline{G(M_1, M_2)}$ *has more than one end.*
(b) $M_2 \in \mathrm{COF}$ *if and only if* $\overline{G(M_1, M_2)}$ *has a vertex of infinite order.*
(c) *If* M_2 *does not halt for any input, then* $G(M_1, M_2)$ *is a comb with a recursive spine.*

Proof. An example of the following construction can be found following the proof. Let B denote the set of nonempty words $\#c_0\#c_1 \dots \#c_n$ where each c_i is the halting computation (if it exists) of the machine M_1 with input $m + i$ (for some $m \in \mathbb{N}$). Then the set of vertices V of $G(M_1, M_2)$ is given by $V = \mathbb{N} \cup B$. We also fix a computable bijection $f : \mathbb{N}^3 \to \mathbb{N}$. The set of edges of $G(M_1, M_2)$ is given by:

(1) $(n, n+1) \in E$, $(n, \#c) \in E$ for all $n \in \mathbb{N}$ and c the halting computation of M_1 on input n (if this halting computation exists).
(2) $(w, w\#c) \in E$ for all $w, w\#c \in B$.
(3) $(n, f(k, \ell, m)) \in E$ iff the following hold:
  (3.1) $m = n$,
  (3.2) M_2 halts for each of the inputs $n, n+1, \dots, n+k$ after at most ℓ computation steps, and
  (3.3) there exists $0 \leq j \leq k$ such that M_2 halts for the input $n + j$ after precisely ℓ computation steps.

This graph is recursive since also the bijection $f^{-1} : \mathbb{N} \to \mathbb{N}^3$ is computable and since condition (3) requires finitely many checks.

Note that the vertices of the form $\mathbb{N}$ together with the edges from (1) between them form an infinite path. The node $\#c_0\#c_1 \dots \#c_n \in B$ is connected to this ray via a path (formed by the prefixes of the form $\#c_0\#c_1 \dots \#c_i$ of this word) to the vertex m, the input of the halting computation c_0. Hence $\overline{G(M_1, M_2)}$ is connected.

(a) First suppose that $M_1 \in \mathrm{COF}$, i.e., M_1 halts for almost all inputs. Then there exists $m \in \mathbb{N}$ such that M_1 halts for all inputs $m, m+1, m+2, \dots$, i.e., there are halting computations c_k for $k \geq m$ on input k. But then the set of words $\#c_m\#c_{m+1} \dots \#c_k$ for $k \geq m$ forms an infinite path. Deleting the edge between m and $\#c_m$ therefore leaves two infinite connected components, i.e., $\overline{G(M_1, M_2)}$ has at least two ends (it actually has infinitely many ends).

  Conversely suppose $\overline{G(M_1, M_2)}$ has more than one end. Since the nodes from $\mathbb{N}$ are connected by a ray-like structure, the graph $\overline{G(M_1, M_2)}$ has to have an infinite path formed by nodes from B. But this implies that there are infinitely many consecutive inputs $m, m+1, m+2, \dots$ that allow a halting computation of M_1, i.e., $M_1 \in \mathrm{COF}$.

(b) Suppose $M_2 \in \mathrm{COF}$ halts for all inputs $m \geq n$. Let, for $m \geq n$, ℓ_m be the maximal length of a halting computation with input between n and m. Then $(n, f(m-n, \ell_m, n)) \in E$, i.e., n has infinite order.

  Conversely, suppose $\overline{G(M_1, M_2)}$ contains a vertex of infinite order. By the very construction, every node from B has at most two neighbors in $\overline{G(M_1, M_2)}$. Hence there exists a vertex $n \in \mathbb{N}$ of infinite order. Note that the neighbors of n are $n-1$ (if $n > 0$), $n+1$, the halting computation of M_1 with input n (if it exists), possibly the node n' with $f^{-1}(n) = (k', \ell', n')$,

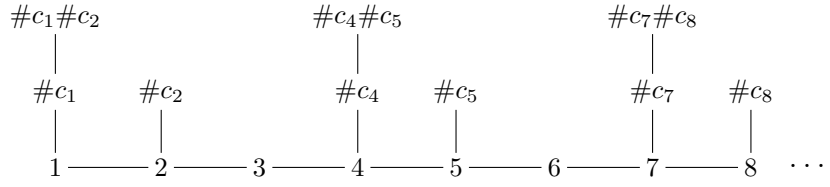

FIGURE 15. A graph $\overline{G(M_1, M_2)}$ with $M_1 \notin \mathrm{COF}$

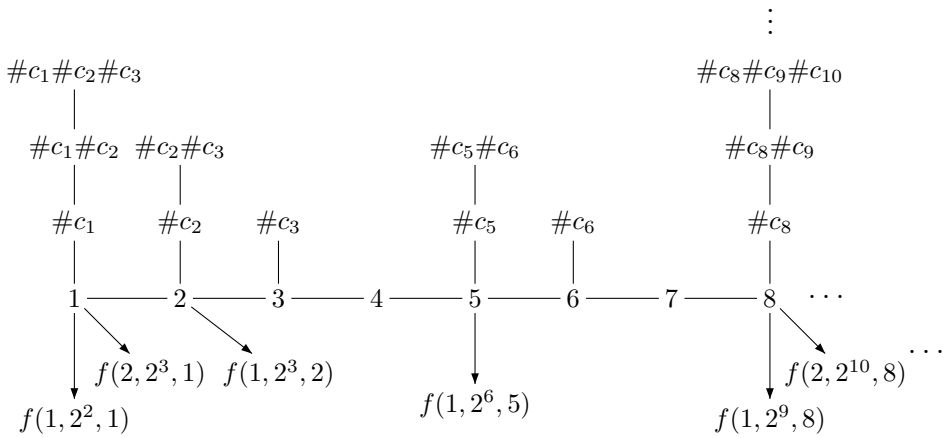

FIGURE 16. A graph $\overline{G(M_1, M_2)}$ with $M_1 \in \mathrm{COF}$

and some nodes of the form $f(k, \ell, n)$ with $k, \ell \in \mathbb{N}$. Since n has infinitely many neighbors, there are therefore mutually distinct pairs $(k_i, \ell_i) \in \mathbb{N}^2$ for $i \in \mathbb{N}$ with $(n, f(k_i, \ell_i, n)) \in E$. By condition (3.2) and (3.3), $k_i = k_j$ implies $\ell_i = \ell_j$. Hence, for every $m \geq n$ there exists $k \geq m - n$ and ℓ such that $(n, f(k, \ell, n)) \in E$ ensuring that M_2 halts for all inputs $n, n+1, n+2, \ldots, n+(m-n), \ldots, n+k$. Thus, M_2 halts for all inputs $m \geq n$, implying $M_2 \in \mathrm{COF}$.

(c) If M_2 never stops, then condition (3) does never hold. Hence $G(M_1, M_2)$ is a comb, the set $\mathbb{N}$ of nodes forms a recursive spine.

⊣

EXAMPLE 4.4. Let us assume that M_2 is a Turing machine that never halts, hence condition (3) does not add any edges. Now let $M_1 \notin \mathrm{COF}$ be a Turing machine that diverges on all inputs of the form $3n$ for $n \geq 1$ and halts on all other inputs. Then (a finite part) of $\overline{G(M_1, M_2)}$ is shown in Fig. 15. All teeth of this comb are finite.

Now assume that $M_1 = M_2 \in \mathrm{COF}$ are Turing machines that do not halt on input 4 and 7 and halt on all other inputs n in 2^n steps. Then (a finite part) of $\overline{G(M_1, M_2)}$ is shown in Fig. 16 where the nodes and edges on top of the spine $\mathbb{N}$

are justified by (1) and (2). The arrows beneath the spine indicate edges that end in another node of the spine.

Since M_1 halts on all inputs $n \geq 8$, every tooth that starts in a node $n \geq 8$ is infinite. Hence, $\overline{G(M_1, M_2)}$ has more than one end. Since also M_2 halts on all inputs $n \geq 8$ in 2^n steps, the node 8 is connected to all nodes of the form $f(k, 2^{k+8}, 8)$, i.e., it has infinite order.

PROPOSITION 4.5. *For recursive graphs, existence of an Euler path is D_3^0-hard.*

PROOF. Let $L \in D_3^0$. Then there exist languages $L_1 \in \Pi_3^0$ and $L_2 \in \Sigma_3^0$ with $L = L_1 \cap L_2$. Since $\overline{\text{COF}}$ and COF are complete for Π_3^0 and Σ_3^0, resp. (see Section 2.1), there are reductions f_i from L_i to $\overline{\text{COF}}$ and COF, resp. Even more, $(f_1(w), f_2(w)) \in \overline{\text{COF}} \times \text{COF}$ iff $w \in L$, i.e., $\overline{\text{COF}} \times \text{COF}$ is hard for D_3^0 and it suffices for our result to reduce this direct product to the set of recursive graphs with an Euler path.

To this end, let M_1 and M_2 be Turing machines and consider $\overline{G(M_1, M_2)}$ from Lemma 4.3. In this graph, replace every edge $e = \{a, b\}$ by four edges $\{a, x_e\}$, $\{x_e, b\}$, $\{a, y_e\}$, and $\{y_e, b\}$. Then the resulting graph G is recursive, connected, and without nodes of odd order. Therefore, G satisfies (E1) and (E3). In addition, it satisfies (E2) iff it has a vertex of infinite degree iff $M_2 \in \text{COF}$ by Lemma 4.3. Finally, G satisfies (E4) iff it has at most one end iff $M_1 \notin \text{COF}$ by Lemma 4.3 iff $M_1 \in \overline{\text{COF}}$. ⊣

PROPOSITION 4.6. *For planar recursive graphs of degree 4, existence of an Euler path is Π_3^0-hard.*

PROOF. First, fix a Turing machine M_2 that never halts. Now let M be a Turing machine. Then the comb $G(M, M_2)$ contains a recursive spine S. Let G be obtained from $\overline{G(M, M_2)}$ by replacing every edge $e = \{a, b\}$ that is not in the spine S by four edges $\{a, x_e\}$, $\{x_e, b\}$, $\{a, y_e\}$, and $\{y_e, b\}$. Then G is recursive (since the spine S is recursive) and connected, i.e., satisfies (E1). Since M_2 never halts, $G(M, M_2)$ is a comb implying that G has degree 4. Note that the root is the only node of $\overline{G(M, M_2)}$ that is adjacent to an odd number of edges from the spine. Hence, in G, the root of $G(M, M_2)$ is the only vertex of odd degree. This shows that G satisfies (E2) and (E3). By Lemma 4.3, it satisfies (E4) if and only if $M \in \overline{\text{COF}}$. Since $\overline{\text{COF}}$ is Π_3^0-complete, the result follows. ⊣

PROPOSITION 4.7. *For recursive combs, existence of only one infinite branch is Π_3^0-hard.*

PROOF. Again, fix a Turing machine M_2 that never halts. Now let M be a Turing machine. Then $G(M, M_2)$ is a comb. Furthermore, the tree $G(M, M_2)$ has only one infinite branch iff $\overline{G(M, M_2)}$ has at most one end iff $M \notin \text{COF}$ (by Lemma 4.3) iff $M \in \overline{\text{COF}}$. Since $\overline{\text{COF}}$ is Π_3^0-hard, the result follows. ⊣

The following lemma prepares the proof that the existence of infinitely many infinite branches is Π_4^0-complete. Recall that in Section 2.1 we agreed that, if not otherwise stated, a Turing machine accepts by halting. In the following lemma, we use *always halting* Turing machines. Such a machine terminates on every input, but it may reject the input by entering a rejecting state. Thus, a set is recursive if and only if it is accepted by an always halting Turing machine.

LEMMA 4.8. *There is an algorithm that, from an always halting Turing machine N, computes a Turing machine N' (that accepts by halting) such that the following are equivalent:*
(1) $\forall k\,\exists \ell\,\forall m \exists n : N$ *accepts* (k,ℓ,m,n)
(2) $\exists^\infty \ell\,\forall m : N'$ *halts on input* (ℓ, m)

PROOF. From $k \in \mathbb{N}$, one can construct an always halting Turing machine $N(k)$ with $L(N(k)) = \{(\ell,m,n) \mid N \text{ accepts } (k,\ell,m,n)\}$. Using the proof that COF is Σ^0_3-complete from [27, Lect. 36], the machine $N(k)$ can be transformed into a Turing machine $A(N(k))$ (which accepts by halting) such that statement (1) is equivalent to $\forall k \in \mathbb{N} : A(N(k)) \in \text{COF}$, i.e., to

$$\forall k\,\exists \ell\,\forall m > \ell : A(N(k)) \text{ halts on input } m\ .$$

Choosing ℓ minimal, we obtain the equivalent statement

$$\forall k\,\exists \ell\,\forall m > \ell : A(N(k)) \text{ halts on } m \wedge (\ell = 0 \vee A(N(k)) \text{ does not halt on } \ell)$$

that, for every $k \in \mathbb{N}$, allows exactly one $\ell = \ell_k$. By taking the infinitely many words $\ell_1 \cdots \ell_k \in \mathbb{N}^+$ $(k \geq 1)$ we obtain the equivalent statement

$$\exists^\infty \overline{\ell} \in \mathbb{N}^+\,\forall m : \bigwedge_{1 \leq i \leq |\overline{\ell}|} \left(\begin{array}{l} m > \overline{\ell}[i] \to A(N(i)) \text{ halts on } m \ \wedge \\ \overline{\ell}[i] = 0 \vee A(N(i)) \text{ does not halt on } \overline{\ell}[i] \end{array} \right)$$

(recall that $\overline{\ell}[i]$ denotes the i^{th} symbol of the word $\overline{\ell} \in \mathbb{N}^+$). Note that, given N and i, the Turing machine $A(N(i))$ can be computed. Hence the matrix of this formula is the conjunction of a Σ^0_1- and a Π^0_1-formula. Starting with $\forall m$, we therefore get a formula from Π^0_2 that can be formulated as

$$\forall m : N'' \text{ halts on input } (\overline{\ell}, m)$$

for some effectively computable Turing machine N''. Since the word $\overline{\ell}$ can be encoded as a single number ℓ, we obtain the Turing machine N' from the theorem as desired. ⊣

PROPOSITION 4.9. *For recursive combs, existence of infinitely many infinite branches is Π^0_4-hard.*

PROOF. Let $L \in \Pi^0_4$ be arbitrary. Then there exists an always halting Turing machine M such that

$$L = \{j \mid \forall k\,\exists \ell\,\forall m \exists n : M \text{ accepts } (j,k,\ell,m,n)\}\ .$$

We will reduce the set L to the set of recursive combs with infinitely many infinite branches. To this end, let $j \in \mathbb{N}$. From j, we can construct an always halting Turing machine N with $L(N) = \{(k,\ell,m,n) \mid M \text{ accepts } (j,k,\ell,m,n)\}$. By Lemma 4.8, we can transform N into a machine N' such that $j \in L$ if and only if

$$\exists^\infty \ell\,\forall m : N' \text{ halts on input } (\ell, m)\ . \tag{5}$$

Define $R \subseteq \mathbb{N}^2$ by $(\ell, m) \in R$ if and only if N' halts on input (ℓ, m). For $(\ell, m) \in R$, let $c_{\ell,m}$ be the halting computation of N' with input (ℓ, m). Then let V denote the computable set

$$\mathbb{N} \cup \{\#c_{\ell,0}\#c_{\ell,1}\#\cdots c_{\ell,m} \mid \ell, m \in \mathbb{N}, (\ell,0), (\ell,1), \ldots, (\ell,m) \in R\}\ .$$

Furthermore, we have the following edges:
(1) $(\ell, \ell+1) \in E$ for all $\ell \in \mathbb{N}$
(2) $(\ell, \#c_{\ell,0}) \in E$ for all $(\ell, 0) \in R$
(3) $(\#c_{\ell,0}\#c_{\ell,1}\#\cdots c_{\ell,m}, \#c_{\ell,0}\#c_{\ell,1}\#\cdots c_{\ell,m+1}) \in E$ if $(\ell,0),\ldots,(\ell,m+1) \in R$
Then (V,E) is a comb with spine the natural numbers. Attached to the natural number ℓ, we have an infinite tooth if and only if, for all $m \in \mathbb{N}$, we have $(\ell, m) \in R$. Hence, (V,E) has infinitely many branches iff (5) holds, i.e., iff $j \in L$. ⊣

4.2.2. Automatic graphs. As in Section 3.2, the construction of the following lemma on automatic graphs uses again configuration graphs of Turing machines.

LEMMA 4.10. *From a Turing machine M, one can compute an automatic comb $T(M)$ with regular spine such that the number of infinite teeth of $T(M)$ equals $|\mathbb{N} \setminus L(M)|$.*

PROOF. Let M_1 be a Turing machine that simulates M and recalls the transitions of its computation. Then, two different configurations of M_1 cannot have the same successor configuration and the configuration graph of M_1 cannot have cycles. Moreover, also backward infinite paths cannot exist. Hence, the configuration graph of M_1 is a disjoint union of finite and infinite paths that start in a node without predecessor. Furthermore, $L(M_1) = L(M)$. The machine M_1 is a *reversible* version of M [4].

There is still a problem with the configuration graph of M_1: There may be infinite paths, where the first configuration is not a legal *initial configuration* of the machine M_1[5] (so called garbage computations). In order to avoid this problem, we next construct a *self-stabilizing* version M_2 of M_1 (see also [31]) as follows. The machine M_2 is obtained from M_1 by adding two counters. Initially, the first counter is set to 0 and the second counter is set to 1. Incrementing the first counter in every step, the machine then simulates M_1 until the first counter equals the second one. At this point, the machine simulates M_1 backwards (which is possible since M_1 is reversible) until the first counter is 0 or it cannot simulate a backward step. If, at this point, the machine is not in an initial configuration of M_1, it stops. Otherwise, it increments the second counter and proceeds as before.

The configuration graph of M_2 is, again, a disjoint union of finite and infinite paths. Moreover, there is a bijection between $\mathbb{N} \setminus L(M_1) = \mathbb{N} \setminus L(M)$ and the set of infinite paths of the configuration graph of M_2. Let

$$S = \{c \mid c \text{ is a configuration of } M_2, \neg\exists c' : c' \vdash_{M_2} c\}$$

be the set of all *source configurations*, where as usual $c' \vdash_{M_2} c$ means that M_2 can transform configuration c' in one step into configuration c. Clearly, every initial configuration of M_2 belongs to S.

Now consider the following graph $T(M) = (V,E)$ whose vertices are the configurations of M_2. For two configurations c, c', we have $(c,c') \in E$ iff $c \vdash_{M_2} c'$ or

[5]The set of initial configurations of M_1 is $q_0\Sigma^+$, where q_0 is the initial state of M_1 and Σ is the input alphabet.

- $c, c' \in S$ and
- c' is the length-lexicographically minimal source configuration length-lexicographically larger than c.

The graph $T(M)$ is thus obtained from the configuration graph of M_2 by placing the source configurations in an ω-chain, i.e., it is a comb. The infinite teeth of this comb are the infinite paths of the configuration graph of M_2. Thus, the number of infinite teeth of $T(M)$ equals $|\mathbb{N} \setminus L(M)|$.

Recall that the relation $\vdash_{M_2}$ as well as the length-lexicographic order on the configurations of M_2 are automatic. Moreover, since S is a regular language, it follows that the edge relation E is automatic. Hence the comb $T(M)$ is automatic [8]. ⊣

By Lemma 4.10, the comb $T(M)$ has only one infinite path if and only if $M \in$ TOTAL. As an immediate consequence of the Π^0_2-hardness of TOTAL (see Section 2.1), we obtain:

PROPOSITION 4.11. *For automatic combs, existence of only one infinite branch is Π^0_2-hard.*

PROPOSITION 4.12. *For planar automatic graphs of degree* 4*, existence of an Euler path is Π^0_2-hard.*

PROOF. Let M be a Turing machine. In the graph $\overline{T(M)}$ (where $T(M)$ is the comb from Lemma 4.10), replace every edge $e = \{a, b\}$ that does not belong to the spine by four edges $\{a, x_e\}$, $\{x_e, b\}$, $\{a, y_e\}$, and $\{y_e, b\}$. Then the resulting graph G is connected, planar, automatic (since the spine of $T(M)$ is regular), and of degree 4. All its nodes except the root of $T(M)$ have even degree and the degree of the root is 1 or 3. Hence, G satisfies (E1), (E2), and (E3). It therefore has an Euler path iff it satisfies (E4) iff $\overline{T(M)}$ has only one end iff M halts for all inputs, i.e., iff $M \in$ TOTAL. Since the set TOTAL is Π^0_2-hard, the result follows. ⊣

4.3. Completeness in the arithmetic hierarchy. We summarize our main results from Section 4.

THEOREM 4.13. *Existence of an Euler path is*

1. *D^0_3-complete for recursive graphs,*
2. *Π^0_3-complete for (planar) locally finite recursive graphs (of degree* 4*),*
3. *Π^0_2-complete for (planar) very recursive graphs (of degree 4), and*
4. *Π^0_2-complete for (planar) automatic graphs (of degree* 4*).*

PROOF. The first statement follows immediately from Prop. 4.2(1) and 4.5, the second from Prop. 4.2(2) and 4.6, and the third from Prop. 4.2(3) and Prop. 4.12 since every automatic graph is very recursive. The last statement follows again from Prop. 4.2(3) and Prop. 4.12. ⊣

Concerning Theorem 4.13(1) and (3), [14] mentions upper (Σ^0_4 and Π^0_2, resp.) and lower bounds (Π^0_3 and both Σ^0_1 and Π^0_1, resp.) and asks for the exact complexities that we provide here. Actually, (3) is stated without proof in [18], where unpublished work of Beigel and Gasarch is cited. In [18], it is also spuriously stated (without proof) that the existence of an Euler path in a recursive graph

is Π^0_3-complete which is (by (1) and (2)) only true for locally finite recursive graphs. To our knowledge, (2) and (4) have not been considered before.

Recall that the existence of an infinite branch in a recursive tree is Σ^1_1-complete and the same holds for automatic trees by Theorem 3.6. On the other hand, by König's lemma, every infinite *finitely branching* tree contains an infinite branch. Since infinity of an automatic structure is decidable [8], it follows that the existence of at least one infinite branch in an automatic finitely branching tree is decidable. The following shows that bounding the number of infinite branches is difficult for both, recursive and automatic trees.

THEOREM 4.14. *Let $k > 0$. Existence of at most k infinite branches is*

1. *Π^0_3-complete for recursive finitely branching trees and*
2. *Π^0_2-complete for automatic and for very recursive finitely branching trees.*

In both cases, hardness holds even for combs.

PROOF. Containment in Π^0_3 and Π^0_2, respectively, follow from Prop. 4.1 since, for a finitely branching tree T, the number of ends of $\overline{T}$ equals the number of infinite branches of T. Hardness for $k = 1$ is shown in Prop. 4.7 and Prop. 4.11, respectively. To reduce the case $k = 1$ to the general case, just add $k - 1$ many infinite branches to a recursive or automatic comb. ⊣

THEOREM 4.15. *Let $k > 0$. Existence of at most k ends is*

1. *Π^0_3-complete for (planar) recursive graphs (of degree 3).*
2. *Π^0_2-complete for automatic and for very recursive planar graphs (of degree 3).*

PROOF. Containment was shown in Prop. 4.1, hardness follows immediately from Theorem 4.14 and the fact that the number of ends of $\overline{T}$ and of infinite branches of T coincide for every finitely branching tree T. ⊣

For the property of having infinitely many infinite branches, we obtain:

THEOREM 4.16. *Existence of infinitely many infinite branches is*

1. *Π^0_4-complete for recursive finitely branching trees.*
2. *Π^0_3-complete for automatic and for very recursive finitely branching trees.*

In both cases, hardness holds even for combs.

PROOF. The upper bounds follow from Theorem 4.14, since T has infinitely many infinite branches if and only if for every $k > 0$, T does not have at most k many infinite branches. The lower bound in (1) was shown in Prop. 4.9.

For the lower bound in (2), note that for the comb $T(M)$ from Lemma 4.10 we have: $T(M)$ has infinitely many infinite branches if and only if $\mathbb{N} \setminus L(M)$ is infinite if and only if $M \in \overline{\mathrm{COF}}$. Since $\overline{\mathrm{COF}}$ is Π^0_3-complete, the result follows. ⊣

§5. Decidable problems for automatic graphs. Every first-order definable relation in an automatic structure has a regular set of representatives [21]. This holds even for the extension $\mathrm{FO}[\infty, \mathrm{mod}]$ of first-order logic by the infinity quantifier and modulo quantifiers. Next, formalizing ideas from [25] (cf. also [34]), we introduce a further extension FSO of $\mathrm{FO}[\infty, \mathrm{mod}]$ with this nice behavior. The logic FSO is a fragment of second order logic, therefore the notation

FSO. At the end of this section, FSO will be used to prove some graph problems decidable for automatic graphs that are Σ^1_1-complete for recursive graphs.

First, recall that the set of FO[∞, mod]-formulas (over a certain signature of relation symbols) is the least set M satisfying:

(i) Every atomic first-order formula over the signature belongs to M.
(ii) M is closed under boolean combinations.
(iii) If $\varphi \in M$, x is a first-order variable, and $0 \leq r < n$, then $\exists x : \varphi$, $\exists^\infty x : \varphi$, and $\exists^{(r,n)} x : \varphi$ belong to M.

Here, $\exists^\infty x : \varphi$ expresses that there are infinitely many x with the property φ, whereas $\exists^{(r,n)} x : \varphi$ expresses that the number of x satisfying φ is finite and congruent r modulo n.

Finally, the set of FSO-formulas (over a certain signature of relation symbols) is the least set M satisfying in addition

(iv) Every formula $(x_1, \ldots, x_n) \in X$ belongs to M (where x_i are elementary variables and X is an n-ary relation variable)
(v) If $\varphi \in M$ and X is an n-ary second-order variable such that $\forall X, Y : X \subseteq Y \wedge \varphi[X/Y] \rightarrow \varphi$ is a tautology[6], then $\exists X$ infinite $: \varphi$ belongs to M.

In the last case, $\exists X$ infinite $: \varphi$ means that there exists an infinite n-ary relation satisfying φ. The fact that $\forall X, Y : X \subseteq Y \wedge \varphi[X/Y] \rightarrow \varphi$ is a tautology implies that if φ is satisfied for some relation Y then it is also satisfied for every smaller (w.r.t. inclusion) relation.

It should be noted that the way we defined FSO-formulas, implies that the set of all FSO-formulas is not decidable. To make the set of FSO-formulas decidable, we could, in the addition formation rule, also require that φ does not contain an occurrence of a formula $(x_1, \ldots, x_n) \in X$ such that X is free in φ and this occurrence lies within an even number of negations.

Our treatment of the logic FSO uses the concept of a word comb that was first used in the proof of [23, Lemma 8.6]: a *word comb* is a set of words $\{s_0 s_1 s_2 \cdots s_{i-1} t_i \mid i \in \mathbb{N}\}$ where $s_i, t_i \in \Gamma^+$ and $|t_i| < |s_i|$ for all $i \in \mathbb{N}$.[7] Then we have:

LEMMA 5.1. *Let Γ be finite and $X \subseteq \Gamma^+$ be infinite. Then there exists a word comb $Y \subseteq X$.*

PROOF. Let $t_0 \in X$ be arbitrary. Now suppose we defined $s_0, \ldots, s_{j-1}$ and $t_0, \ldots, t_j$ such that

(1) $s_0 s_1 \cdots s_{i-1} t_i \in X$ for all $0 \leq i \leq j$
(2) $|s_i| > |t_i|$ for all $0 \leq i < j$
(3) $X \cap s_0 s_1 \cdots s_{j-1} \Gamma^+$ is infinite.

Since there are only finitely many words of length $|t_j|+1$, (3) implies the existence of a word $s_j \in \Gamma^+$ with $|s_j| = |t_j| + 1$ such that $X \cap s_0 s_1 \cdots s_j \Gamma^+$ is infinite ensuring (2) and (3) for $j+1$. Choose $t_{j+1} \in \Gamma^+$ with $s_0 s_1 \cdots s_j t_{j+1} \in X$

[6]Here, $\varphi[X/Y]$ is the formula that results from φ by replacing every free occurrence of X by Y. Morever, that $\forall X, Y : X \subseteq Y \wedge \varphi[X/Y] \rightarrow \varphi$ is a tautology means that this formula is true in every structure and for every interpretation of free variables.

[7]Such a set of words looks like a comb when equipped with the prefix order.

arbitrary. Since this ensures (1), we can proceed by induction. Then the set of words $s_0 s_1 \cdots s_{i-1} t_i$ is a word comb $Y \subseteq X$. $\dashv$

This lemma allows to prove the following result.

THEOREM 5.2. *From a given automatic presentation* (Γ, L, h) *of an automatic structure* $\mathcal{A}$ *and an* FSO*-formula* $\varphi(x_1, \ldots, x_n)$*, we can effectively construct an automaton for the convolution of the relation*

$$\{(u_1, \ldots, u_n) \in L^n \mid \mathcal{A} \models \varphi(h(u_1), \ldots, h(u_n))\}.$$

PROOF. Suppose k is an upper bound for the arity of all the relation variables used in φ. Consider the extension of the structure $\mathcal{A}$ by the set of all ℓ-tuples for $\ell \leq k$ and the relations "x is an ℓ-tuple" and "a is the i^{th} entry in the ℓ-tuple x". Note that an automatic presentation of this extension can be easily computed. Hence, in the following, we can assume that the formula φ uses only second-order quantification over unary relations X.

Since every automatic structure has an injective presentation (see Section 2.3), we can identify the representing words with the elements of the structure, i.e., the language L of the presentation (Γ, L, h) is the underlying set of the structure $\mathcal{A}$ and h is the identity mapping. Then word combs that are contained in L are special subsets of the automatic structure $\mathcal{A}$. Let $\mathcal{A} = (L, (R_i)_{i \in I})$, thus, every R_i is an automatic relation over the alphabet Γ.

Next consider the structure $\mathcal{A}_C = (L \cup C, (R_i)_{i \in I}, C, \mathrm{el})$ where C is the set of all word combs that are contained in L and el is the set of pairs $(w, c) \in L \times C$ with $w \in c$.

We provide an injective ω-automatic presentation[8] $(\Delta, L_\omega, h_\omega)$ for $\mathcal{A}_C$: The alphabet of this presentation is $\Delta = \Gamma \cup \{\#\} \cup (\Gamma \times (\Gamma \cup \{\#\}))$. The underlying language is

$$L_\omega = L\#^\omega \cup L_C,$$

where $L\#^\omega$ represents the elements of $\mathcal{A}$ and $L_C \subseteq (\Gamma \times (\Gamma \cup \{\#\}))^\omega$ represents all word combs contained in L. We define L_C as the set of all ω-words

$$\otimes(s_0, t_0) \otimes (s_1, t_1) \otimes (s_2, t_2) \cdots$$

over $\Gamma \times (\Gamma \cup \{\#\})$ with $s_i, t_i \in \Gamma^+$, $|t_i| < |s_i|$, and $s_0 s_1 \cdots s_{i-1} t_i \in L$ for all $i \in \mathbb{N}$. The mapping $h_\omega : L_\omega \to L \cup C$ is defined by $h_\omega(w\#^\omega) = w$ for $w \in L$ and

$$h_\omega(\otimes(s_0, t_0) \otimes (s_1, t_1) \otimes (s_2, t_2) \cdots) = \{s_0 s_1 \cdots s_{i-1} t_i \mid i \in \mathbb{N}\}$$

for ω-words from L_C (note that this is well-defined since the t_i-blocks are separated by at least one occurrence of # because $|t_i| < |s_i|$). From a deterministic finite automaton accepting the language L with set of states Q, it is not hard to build deterministic Büchi-automata with state sets $Q \cup \{\top\}$ and $Q \times (Q \cup \{\top\})$, respectively, that accept $L\#^\omega$ and L_C, respectively. Hence L_ω as well as $L_C = \{w \in L_\omega \mid h_\omega(w) \in C\}$ are ω-regular. Similarly, one finds a Büchi-automaton for the h_ω-preimage of the automatic relation R_i from a finite automaton for R_i. Finally, note that $h_\omega(w\#^\omega) \in h_\omega(c)$ for $w \in L$ and $c = \otimes(s_0, t_0) \otimes (s_1, t_1) \otimes (s_2, t_2) \cdots \in L_C$ iff there exists $i \in \mathbb{N}$ with $w = s_0 s_1 \cdots s_{i-1} t_i$

[8] ω-automatic presentations and ω-automatic structures [8] are defined analogously to their automatic counterparts. Finite words are replaced by ω-words (the convolution of ω-words is defined in the obvious way). Ordinary finite automata are replaced by Büchi-automata.

which can be checked by a nondeterministic Büchi-automaton with two tracks. This concludes the description of an ω-automatic presentation $(\Delta, L_\omega, h_\omega)$ for the structure $\mathcal{A}_C$.

Now let $\varphi(x_1, \dots, x_n)$ be an FSO-formula in the language of the automatic structure $\mathcal{A}$. Let the FO$[\infty, \mathrm{mod}]$-formula $\varphi_C(x_1, \dots, x_n)$ be obtained from $\varphi(x_1, \dots, x_n)$ by replacing every

- set quantification $\exists X$ infinite : α by $\exists x : (x \in C \wedge \alpha)$, every
- first-order quantification $\exists x : \alpha$ by $\exists x : (x \notin C \wedge \alpha)$ (and similarly for $\exists^\infty x : \alpha$ and $\exists^{(r,n)} x : \alpha$), and every
- atomic subformula $y \in X$ (for X a unary relation variable) by $(y, x) \in \mathrm{el}$.

Then $\varphi_C(x_1, \dots, x_n)$ is an FO$[\infty, \mathrm{mod}]$-formula in the language of the ω-automatic structure $\mathcal{A}_C$.

Now let H be the set of all tuples $(u_1, \dots, u_n) \in (L\#^\omega)^n$ of ω-words over Γ such that

$$\mathcal{A}_C \models \varphi_C(u_1, \dots, u_n) \ .$$

Then, by [29, 1], the relation H is effectively ω-automatic. Since $H \subseteq (L\#^\omega)^n$ is ω-automatic, the set

$$H' = \{(v_1, \dots, v_n) \in L^n \mid (v_1\#^\omega, \dots, v_n\#^\omega) \in H\}$$

is automatic. Note that H' is the set of tuples $(v_1, \dots, v_n) \in L^n$ satisfying $\varphi(x_1, \dots, x_n)$ in $\mathcal{A}$ under the restriction that second-order quantification is restricted to word combs. But this is equivalent to saying $\mathcal{A} \models \varphi(v_1, \dots, v_n)$. For this, note that $\forall X, Y : X \subseteq Y \wedge \alpha[X/Y] \to \alpha$ is a tautology whenever $\exists X$ infinite : α is a subformula of φ. Lemma 5.1 implies that there exists an infinite set $A \subseteq L$ satisfying α if and only if there is a word comb satisfying α. $\dashv$

Since the emptiness of an effectively regular language is decidable, we obtain the following as an immediate consequence.

COROLLARY 5.3. *There exists an algorithm that, on input of an automatic presentation of an automatic structure $\mathcal{A}$ and an* FSO*-sentence φ, determines whether $\mathcal{A} \models \varphi$.*

A variation of the proof of Theorem 5.2 yields the following result.

THEOREM 5.4. *From a given automatic presentation (Γ, L, h) of an automatic structure $\mathcal{A}$ and an* FSO*-formula $\alpha(X)$ with X an n-ary relation variable such that (i) $\forall X, Y : X \subseteq Y \wedge \alpha[X/Y] \to \alpha$ is a tautology and (ii) $\mathcal{A} \models \exists X$ infinite $: \alpha$, one can effectively construct an automatic relation $H \subseteq L^n$ such that $h(H)$ is infinite and $\mathcal{A} \models \alpha(h(H))$.*

PROOF. We use the notations from the proof of Theorem 5.2. In particular, we consider the ω-automatic presentation $(\Delta, L_\omega, h_\omega)$ of the structure $\mathcal{A}_C$. From $\mathcal{A} \models \exists X$ infinite : α, we obtain $\mathcal{A}_C \models \exists x(x \in C \wedge \alpha_C)$ (where α_C is obtained from α in the same way that φ_C resulted from φ in the proof of Theorem 5.2). Now it follows from [29, 1] that $\{u \in K \mid \mathcal{A}_C \models \alpha_C(h_\omega(u))\}$ is effectively ω-regular and nonempty. Hence one can effectively find words $v \in \Delta^*$ and $w \in \Delta^+$

such that $vw^\omega \in K$ and $\mathcal{A}_C \models \alpha_C(h_\omega(vw^\omega))$. Since $vw^\omega \in K$, there exist words $s_i, t_i \in \Gamma^+$ such that

$$vw^\omega = \otimes(s_0, t_0) \otimes (s_1, t_1) \otimes (s_2, t_2) \cdots ,$$

$|t_i| < |s_i|$, and $s_0 s_1 \cdots s_{i-1} t_i \in L$ for all $i \in \mathbb{N}$. From the words v and w, one can construct a finite automaton for the language $H = \{s_0 s_1 \cdots s_{i-1} t_i \mid i \in \mathbb{N}\}$. Then, from $\mathcal{A}_C \models \alpha_C(h_\omega(vw^\omega))$, we get $\mathcal{A} \models \alpha(H)$. ⊣

We use the above Corollary 5.3 and Theorem 5.4 to show that two problems are decidable for automatic structures. In the more general setting of recursive structures, they are Σ^1_1-complete as shown by Hirst and Harel [19].

COROLLARY 5.5 (cf. [34, Theorem 3.20]). *It is decidable whether an automatic graph contains an infinite clique. If an infinite clique exists, a regular set of representatives of an infinite clique can be computed effectively.*

PROOF. Consider the formula $\exists X$ infinite $\forall x, y : (x, y \in X \Rightarrow (x, y) \in E)$. ⊣

The second problem is the infinite version of maximal set cover considered by Hirst and Harel. It asks whether, given a set $X = \{X_i \mid i \in \mathbb{N}\}$ of sets $X_i \subseteq \mathbb{N}$, there exists $A \subseteq \mathbb{N}$ with $\bigcup_{a \in A} X_a = \mathbb{N}$ and $\mathbb{N} \setminus A$ infinite. Note that the collection X can be represented as a set of pairs E with $(i, j) \in E$ iff $j \in X_i$. Then there exists A as required iff the directed graph $(\mathbb{N}, E)$ satisfies $\exists B$ infinite $\forall j \exists i : i \notin B \wedge (i, j) \in E$ (then A is the complement of B). Hence we get:

COROLLARY 5.6. *The infinite version of maximal set cover is decidable if the collection X is given as an automatic set of pairs. In case a set cover as required exists, one can compute a set cover.*

§6. Further graph classes. Let us finally consider two other classes of infinite graphs:

Recursively enumerable graphs. A graph (V, E) is *recursively enumerable* if $V \subseteq \mathbb{N}$ and $E \subseteq \binom{\mathbb{N}}{2}$ are recursively enumerable. The existence of a Hamiltonian path in a recursively enumerable graph is again in Σ^1_1 which, since every recursive graph is recursively enumerable, implies Σ^1_1-completeness. Hence the problem does not become more complicated in this case. In order to show the analogous statement for the statements in Section 4, we need the following construction: Let (V, E) be a recursively enumerable graph. Let M_V (resp. M_E) be a Turing-machine that halts on input n (resp. $n\#m$ with $n < m$) if and only if $n \in V$ (resp. $\{n, m\} \in E$). Now let V' be the union of the set of accepting computations of M_V and the set of accepting computations of M_E. Moreover, let $E' \subseteq \binom{V'}{2}$ be the set of all $\{c_1, c_2\}$ such that for some $n \in \mathbb{N}$: (i) c_1 is an accepting computation on input n and (ii) for some $m \in \mathbb{N}$, either $n < m$ and c_2 is an accepting computation on input $n\#m$ or $m < n$ and c_2 is an accepting computation on input $m\#n$. Then (V', E') is a recursive graph that is obtained from (V, E) by replacing every edge by a path of length 2. This construction shows that Theorem 4.13(1), 4.14(1), 4.15(1), and 4.16(1) hold verbatim for recursively enumerable graphs.

Highly recursive, rational, and tree-automatic graphs. Note that for all graph theoretic properties considered in Sections 3 and 4, very recursive and automatic graphs are complete for the same classes of the analytical and arithmetical hierarchy. Hence the same holds for all classes of graphs in between these two classes. The most prominent examples are highly recursive, rational, and tree-automatic graphs (cf. [36, 6]).

REFERENCES

[1] V. Bárány, Ł. Kaiser, and S. Rubin, *Cardinality and counting quantifiers on omega-automatic structures*, ***STACS 2008***, IFIB Schloss Dagstuhl, 2008, pp. 385–396.

[2] D. R. Bean, *Effective coloration*, this Journal, vol. 41 (1976), no. 2, pp. 469–480.

[3] ———, *Recursive Euler and Hamilton paths*, ***Proceedings of the American Mathematical Society***, vol. 55 (1976), no. 2, pp. 385–394.

[4] C. H. Bennett, *Logical reversibility of computation*, ***IBM Journal of Research and Development***, vol. 17 (1973), pp. 525–532.

[5] R. Berger, *The undecidability of the domino problem*, ***Mem. Amer. Math. Soc. No.***, vol. 66 (1966), p. 72.

[6] A. Blumensath, *Automatic structures*, ***Technical report***, RWTH Aachen, 1999.

[7] A. Blumensath and E. Grädel, *Automatic structures*, ***LICS 2000***, IEEE Computer Society Press, 2000, pp. 51–62.

[8] ———, *Finite presentations of infinite structures: Automata and interpretations*, ***Theory of Computing Systems***, vol. 37 (2004), no. 6, pp. 641–674.

[9] W. Dicks and M. J. Dunwoody, ***Groups acting on graphs***, Cambridge University Press, 1989.

[10] R. Diestel, ***Graph theory, third edition***, Springer, 2006.

[11] P. Erdös, T. Grünwald, and E. Vazsonyi, *Über Euler-Linien unendlicher Graphen*, ***Journal of Mathematics and Physics***, vol. 17 (1938), no. 2, pp. 59–75.

[12] L. Euler, *Solutii problematis ad geometriam situs pertinentis*, ***Commentarii Academiae Scientiarum Petropolitanae***, vol. 8 (1736), pp. 128–140.

[13] M. R. Garey, D. S. Johnson, and R. E. Tarjan, *The planar Hamiltonian circuit problem is NP-complete*, ***SIAM Journal on Computing***, vol. 5 (1976), no. 4, pp. 704–714.

[14] W. Gasarch, *A survey of recursive combinatorics*, ***Handbook of recursive mathematics, volume 2*** (Yu. L. Ershov, S. S. Goncharov, V. W. Marek, A. Nerode, and J. Remmel, editors), Studies in Logic and the Foundations of Mathematics, no. 139, Elsevier, 1998, pp. 1041–1176.

[15] D. Harel, *Recurring dominoes: making the highly undecidable highly understandable*, ***Annals of Discrete Mathematics***, vol. 24 (1985), pp. 51–72.

[16] D. Harel, *A simple undecidable domino problem (or, a lemma on infinite trees, with applications)*, ***Proc. Logic and Computation Conference***, Clayton, 1984.

[17] D. Harel, *Effective transformations on infinite trees, with applications to high undecidability, dominoes, and fairness*, ***Journal of the Association for Computing Machinery***, vol. 33 (1986), no. 1, pp. 224–248.

[18] ———, *Hamiltonian paths in infinite graphs*, ***Israel Journal of Mathematics***, vol. 76 (1991), no. 3, pp. 317–336.

[19] T. Hirst and D. Harel, *Taking it to the limit: on infinite variants of NP-complete problems*, ***Journal of Computer and System Sciences***, vol. 53 (1996), pp. 180–193.

[20] B. Khoussainov and M. Minnes, *Model theoretic complexity of automatic structures*, ***TAMC 2008***, Lecture Notes in Computer Science, no. 4978, Springer, 2008, pp. 514–525.

[21] B. Khoussainov and A. Nerode, *Automatic presentations of structures*, ***LCC: International Workshop on Logic and Computational Complexity***, Lecture Notes in Computer Science, no. 960, 1995, pp. 367–392.

[22] B. Khoussainov, A. Nies, S. Rubin, and F. Stephan, *Automatic structures: richness and limitations*, ***Logical Methods in Computer Science***, vol. 3 (2007), no. 2, pp. 2:2, 18 pp.

(electronic).

[23] B. Khoussainov, S. Rubin, and F. Stephan, *Automatic partial orders*, ***LICS 2003***, IEEE Computer Society Press, 2003, pp. 168–177.

[24] ———, *Definability and regularity in automatic structures*, ***STACS 2004***, Lecture Notes in Computer Science, no. 2996, Springer, 2004, pp. 440–451.

[25] B. Khoussainov, S. Rubin, and F. Stephan, *Automatic linear orders and trees*, ***ACM Transactions on Computational Logic***, vol. 6 (2005), no. 4, pp. 675–700.

[26] S.C. Kleene, *Recursive predicates and quantifiers*, ***Trans. Amer. Math. Soc.***, vol. 53 (1943), pp. 41–73.

[27] D. Kozen, ***Theory of Computation***, Springer, 2006.

[28] D. Kuske and M. Lohrey, *Euler paths and ends in automatic and recursive graphs*, ***AFL 2008***, Hungarian Academy of Sciences, 2008, pp. 245–256.

[29] ———, *First-order and counting theories of ω-automatic structures*, this Journal, vol. 73 (2008), pp. 129–150.

[30] ———, *Hamiltonicity of automatic graphs*, ***IFIP-TCS 2008***, Springer, 2008, pp. 445–459.

[31] O. Ly, *Automatic graphs and D0L-sequences of finite graphs*, ***Journal of Computer and System Sciences***, vol. 67 (2003), no. 3, pp. 497–545.

[32] A. B. Manaster and J. G. Rosenstein, *Effective matchmaking (recursion theoretic aspects of a theorem of Philip Hall)*, ***Proceedings of the London Mathematical Society. Third Series***, vol. 25 (1972), pp. 615–654.

[33] H. Rogers, ***Theory of recursive functions and effective computability***, McGraw-Hill, 1968.

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

[35] S. Rubin, *Automatic structures*, ***Ph.D. thesis***, University of Auckland, 2004.

[36] W. Thomas, *A Short Introduction to Infinite Automata*, ***DLT 2001*** , Lecture Notes in Computer Science, no. 2295, Springer, 2001, pp. 130–144.

[37] H. Wang, *Proving theorems by pattern recognition*, ***Bell Syst. Tech. J.***, vol. 40 (1961), pp. 1–41.

UNIVERSITÄT LEIPZIG
INSTITUT FÜR INFORMATIK
POSTFACH 100920
D-04009 LEIPZIG
GERMANY
E-mail: {kuske,lohrey}@informatik.uni-stuttgart.de