Ramsey Quantifiers over Automatic Structures: Complexity and Applications to Verification

Pascal Bergsträßer
Department of Computer Science, TU Kaiserslautern
Kaiserslautern, Germany

Moses Ganardi
Max Planck Institute for Software Systems (MPI-SWS)
Kaiserslautern, Germany

Anthony W. Lin
Department of Computer Science, TU Kaiserslautern
Max Planck Institute for Software Systems (MPI-SWS)
Kaiserslautern, Germany

Georg Zetzsche
Max Planck Institute for Software Systems (MPI-SWS)
Kaiserslautern, Germany

ABSTRACT

Automatic structures are infinite structures that are finitely represented by synchronized finite-state automata. This paper concerns specifically automatic structures over finite words and trees (ranked/unranked). We investigate the "directed version" of Ramsey quantifiers, which express the existence of an infinite directed clique. This subsumes the standard "undirected version" of Ramsey quantifiers. Interesting connections between Ramsey quantifiers and two problems in verification are firstly observed: (1) reachability with Büchi and generalized Büchi conditions in regular model checking can be seen as Ramsey quantification over transitive automatic graphs (i.e., whose edge relations are transitive), (2) checking monadic decomposability (a.k.a. recognizability) of automatic relations can be viewed as Ramsey quantification over co-transitive automatic graphs (i.e., the complements of whose edge relations are transitive). We provide a comprehensive complexity landscape of Ramsey quantifiers in these three cases (general, transitive, co-transitive), all between NL and EXP. In turn, this yields a wealth of new results with precise complexity, e.g., verification of subtree/flat prefix rewriting, as well as monadic decomposability over tree-automatic relations. We also obtain substantially simpler proofs, e.g., for NL complexity for monadic decomposability over word-automatic relations (given by DFAs).

CCS CONCEPTS

• **Theory of computation** → **Logic and verification**; **Regular languages**; **Tree languages**.

KEYWORDS

Ramsey quantifier, automatic structures, recurrent reachability, monadic decomposability

ACM Reference Format:
Pascal Bergsträßer, Moses Ganardi, Anthony W. Lin, and Georg Zetzsche. 2022. Ramsey Quantifiers over Automatic Structures: Complexity and Applications to Verification. In *37th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS) (LICS '22), August 2–5, 2022, Haifa, Israel.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3531130.3533346

LICS '22, August 2–5, 2022, Haifa, Israel
© 2022 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9351-5/22/08.
https://doi.org/10.1145/3531130.3533346

1 INTRODUCTION

Automatic structures are infinite structures whose domains are *regular sets* (represented by finite automata over finite words/trees) and whose relations are *regular relations* (represented by synchronized finite word/tree automata) [5, 6]. They have been intensively studied in the logic and automata community, and have been also applied to infinite-state verification, especially the area of regular model checking. In this paper, we are interested in algorithmic aspects of the *Ramsey quantifiers* [29], which state the existence of infinite cliques. [Ramsey quantifiers are also known in model theory as Magidor-Malitz quantifiers, e.g., see [14].] Similar clique-like objects were also considered in the proofs of two seemingly different problems over automatic structures; namely, the problems of verification of liveness properties in regular model checking [32, 33], as well as monadic decomposability of regular relations (a.k.a. finite recognizability) [3, 7]. In this paper, we investigate a more general notion of quantifiers that generalize the classic Ramsey quantifiers and these aforementioned clique-like objects studied in the context of liveness and monadic decomposability. Through our new notion of quantifiers, we establish a comprehensive picture of the computational complexity landscape of Ramsey quantifiers over automatic structures, as well the aforementioned verification problems. We first discuss the state-of-the-art of these seemingly disconnected problems.

Ramsey Quantifiers over Automatic Structures. Blumensath and Grädel were the first to systematically study automatic structures [5, 6]. A fundamental fact is that, given a first-order formula $\varphi(\boldsymbol{x})$ and a word/tree automatic structure $\mathfrak{A}$ (with domain A), one can effectively compute a synchronized word/tree automaton representing the set $[\![\varphi]\!]_{\mathfrak{A}} = \{\boldsymbol{a} \in A^{|\boldsymbol{x}|} \mid \mathfrak{A} \models \varphi(\boldsymbol{a})\}$ of solutions of φ. In other words, regular relations are *effectively closed under all first-order operations.* Consequently, first-order (FO) model checking over automatic structures is decidable.

In the seminal paper [5] on automatic structures, it was already observed that FO can be enriched with the quantifier "there exist infinitely many" — i.e., $\exists^\infty x\colon \varphi(x, \boldsymbol{y})$, which is true iff there exist infinitely many x such that $\varphi(x, \boldsymbol{y})$ — while preserving the above effective closure property and decidability of model checking. In fact, assuming nondeterministic automata as finite representations of $[\![\varphi]\!]_{\mathfrak{A}}$, one can compute $[\![\exists^\infty x\colon \varphi]\!]_{\mathfrak{A}}$ in polynomial-time. A few years later, in the case of word automatic structures, Rubin [29] studied Ramsey quantifiers, which generalize $\exists^\infty$ by enforcing that these infinitely many elements form an infinite undirected clique, and showed that Ramsey quantifiers preserve regularity as well, meaning FO extended with Ramsey quantifiers is still decidable. Upon closer inspection, Rubin's construction runs in doubly exponential time. *Is this optimal? Does the same extend to tree-automatic structures?*

Liveness in Regular Model Checking. Regular model checking (RMC) is a generic verification framework that exploits regular languages and relations (e.g., over finite/ω-words or trees) as symbolic representations of infinite systems [1, 2, 23]. Various flavors of automata and transducers for representing the transition relations are employed, e.g., word/tree automatic relations (or subsets thereof), ω-automatic relations (or subsets thereof), and rational relations. Since safety and reachability are undecidable in RMC (e.g., over automatic graphs), one focus of RMC has been to develop acceleration/widening techniques, which are semi-algorithms for computing reachability sets (i.e., $post^*(S)$) and reachability relations (i.e., transitive closure R^* of the edge relation in the graph), that may terminate on many interesting cases. Some of these semi-algorithms have general completeness and termination guarantee, e.g., bounded local-depth acceleration for automatic relations are guaranteed to compute reachability relations for pushdown systems (PDS) and ground-tree rewrite systems (GTRS) [1, 22], while flattable acceleration for Presburger-definable relations is guaranteed to compute reachability relations for reversal-bounded counter systems and 2-dimensional vector addition systems with states [19].

Reachability sets/relations can be directly used to solve safety. The challenge of verifying liveness is the necessity to deal with genuinely infinite paths (with no repeated configurations). To and Libkin [32, 33] showed that one can decide liveness (in the form of recurrent reachability) over word- and tree-automatic graphs, when the transitive closure R^* of the edge relation is additionally supplied (e.g., by the aforementioned acceleration methods or otherwise). The algorithm runs in time polynomial in the size of the problem, with R^* supplied as part of inputs. Their technique uses a kind of "staircase argument" combined with Ramsey's Theorem to construct a Büchi automaton that represents some witnessing infinite runs.

Monadic Decomposability and Recognizability. A classic task in the theory of finite-state transductions is the problem to decide whether a given regular relation $R \subseteq (\Sigma^*)^k$ is *recognizable*, i.e., if it can be expressed as a finite union of cartesian products of regular languages (in symbols: $R = \bigcup_{i=1}^{n} L_{i,1}\times\cdots\times L_{i,k}$ for some $n \in \mathbb{N}$ and regular sets $L_{i,j} \subseteq \Sigma^*$). In the formal verification terminology [36], such a relation is said to be *monadically decomposable*, i.e., that it can be expressed as a Boolean combination of monadic predicates. The first important result was by Stearns [31] and Valiant [34]: Their algorithms for checking regularity of deterministic pushdown automata imply that given deterministic rational relation—i.e., a relation $R \subseteq \Sigma^* \times \Sigma^*$ recognized by *deterministic asynchronous automata*, which is strictly more general than binary automatic relations—can be checked to be recognizable in doubly exponential time. This decidability was extended to general k-ary deterministic rational relations by Carton et al. [7], which yields decidability as well for the subclass of automatic relations. As noted by Löding and Spinrath [26], the complexity of the algorithm for automatic relations in [7] runs in doubly exponential time. Using their new polynomial-time algorithm for checking regularity for deterministic visibly pushdown automata, Löding and Spinrath showed that this could be improved to single exponential time for binary automatic relations. The complexity for automatic relations was fully settled by Barceló et al. [3] by showing that this problem is NL-complete (resp. PSPACE-complete) when R is presented as a deterministic (resp. nondeterministic) automaton. The proof technique in [3] is an extremely intricate refinement and analysis of the staircase argument used by To and Libkin [32] for recurrent reachability for automatic relations.

Contributions. Our account of the state-of-the-art of the aforementioned three research directions seems to suggest that there might be some connections between them. To what extent are they connected? Is there a more fundamental notion that unifies them? These questions are hitherto open, but as we shall see in this paper the answer is a resounding yes. We pinpoint that the *directed Ramsey quantifiers*—which ask for the existence of infinite directed cliques (instead of infinite undirected cliques as in [29])—is a fundamental concept that underlies the above three problems, and lets us study them under the same umbrella, while inferring the optimal complexity and even new results. On the one hand, the directed Ramsey quantifiers subsume the standard Ramsey quantifiers. On the other hand, recurrent reachability over automatic graphs [32] can be seen as a Ramsey quantifier over a transitive binary relation, whereas monadic decomposability over automatic relations [3, 7, 26] can be construed as a Ramsey quantifier over co-transitive binary relations. Our results are summarized in Table 1.

Firstly, from the proof by Barceló et al. [3], it is possible to infer that the Ramsey quantifier can be evaluated on regular relations in NL, which substantially improves the doubly exponential-time algorithm of Rubin [29]. Unfortunately, their argument relies on an intricate Ramsey argument on the transition monoid of the automaton. Our contribution is a substantially simpler argument that avoids the use of the transition monoid altogether, which we show to generalize to the case of tree-regular relations (which is not the case with the proof of [3]). More precisely, our approach divides the proof for regular relations into two steps: (i) First, we argue that one can assume infinite cliques witnessed by accepting runs that form a *comb of combs*. (ii) Then, we argue that the runs can be "merged" together so that it can be witnessed by a single run of a polynomial-size Büchi automaton. This way, we obtain the same complexity as [3].

For tree-regular relations we can easily extend step (i). The comb of combs structure of the accepting runs can be witnessed by an alternating Büchi tree automaton, which yields the complexity of EXP for the Ramsey quantifier on tree-regular relations. However, step (ii) is provably impossible over tree-regular relations, since as we show, the infinite clique problem is EXP-hard. For the special cases of transitive and co-transitive relations we need further separate arguments that enable us to evaluate the Ramsey quantifier in P. The case for transitive relations can be inferred from the proof in [32], but not so for the co-transitive case.

Finally, we apply our results to decidability and complexity of recurrent reachability with generalized Büchi conditions, and automatic structures over unranked trees. We show, for example, decidability (in fact in polynomial-time) of recurrent reachability of subtree/flat prefix rewriting, answering an open question by Löding and Spelten [25] and decidability (in fact, EXP-completeness) of recurrent reachability with generalized Büchi conditions of ground tree rewrite systems, answering an open question by Löding [24].

Organization. We provide a more detailed summary of our main results in Section 2. We fix notation and basic terminologies in Section 3. We then start with the word case in Section 4 and proceed to the tree case in Section 5. Applications and generalizations to unranked trees are given in, respectively, Section 6 and Section 7.

The full version of this paper is available at [4].

2 DETAILED SUMMARY OF MAIN RESULTS

To improve readability, we provide a detailed summary of our main results in this section before we take a deeper dive into the proofs. Unless otherwise specified, the completeness results mentioned in this section (and Table 1) hold for NFAs and DFAs in the word case and NTAs, D↑TAs, and D↓TAs in the tree case. We define the directed Ramsey quantifier:

Definition 2.1. Let $\mathfrak{A}$ be a structure with domain A. The *Ramsey quantifier* $\exists^{\text{ram}}$ over an $\mathfrak{A}$-formula φ with $k+2$ free variables is defined for all $\boldsymbol{c} \in A^k$ by $\mathfrak{A} \models \exists^{\text{ram}} x, y\colon \varphi(x, y, \boldsymbol{c})$ if and only if there is an infinite sequence $(a_i)_{i\geq 1}$ of pairwise distinct elements $a_i \in A$ so that $\mathfrak{A} \models \varphi(a_i, a_j, \boldsymbol{c})$ for all $1 \leq i < j$.

We deviate from the definition of the Ramsey quantifier found in the literature, see [12], requiring $\mathfrak{A} \models \varphi(a_i, a_j, \boldsymbol{c})$ *for all* $i \neq j$, in the definition above. Over (tree-)regular relations the two quantifier definitions can be simulated by each other, see [4]. Furthermore, there are also higher-dimensional versions $\exists^{d\text{-ram}}$ of the Ramsey quantifier, which will not be considered in this paper.

Evaluating Ramsey quantifiers. If R is a binary (tree-) regular relation, then evaluating $\exists^{\text{ram}} x, y\colon R(x, y)$ is the problem of checking whether R contains an *infinite (directed) clique*, i.e., an infinite sequence $(a_i)_{i\geq 1}$ of distinct elements of A such that $(a_i, a_j) \in R$ for all $1 \leq i < j$. It follows from [3] that the infinite clique problem over word-regular relations is NL-complete. We provide a much simpler proof by considering a slightly more general setting. Instead of the infinite clique problem we consider the evaluation of the Ramsey quantifier on a $(k+2)$-ary (tree-)regular relation $R \subseteq A^{k+2}$, i.e., compute an automaton for $[\![\exists^{\text{ram}} x, y\colon R(x, y, z)]\!] = \{\boldsymbol{c} \in A^k \mid \exists^{\text{ram}} x, y\colon R(x, y, \boldsymbol{c})\}$.

Theorem 2.2. *Given a regular relation $R \subseteq (\Sigma^*)^{k+2}$ by an NFA $\mathcal{A}$*[1]*, one can construct an NFA for the relation $[\![\exists^{\text{ram}} x, y\colon R(x, y, z)]\!]$ in logspace. In particular, the infinite clique problem over regular relations is in* NL.

We show that the complexity of the infinite clique problem increases from NL to EXP when considered over tree-regular relations given by NTAs or D↓TAs. Let $\mathcal{T}_\Sigma$ denote the set of ranked trees over alphabet Σ.

[1]In this and the following theorems, the parameter k is part of the input.

Theorem 2.3. *The infinite clique problem over tree-regular relations $R \subseteq \mathcal{T}_\Sigma \times \mathcal{T}_\Sigma$ is* EXP*-complete if R is given as NTA or D↓TA, and* P*-complete if R is given as D↑TA.*

For the exponential lower bound, we present a reduction from intersection nonemptiness for NTAs and D↓TAs. This is surprising, because an analogue reduction in the word case does not exist: This would yield a PSPACE lower bound for the infinite clique problem over words, but the latter belongs to NL.

For the exponential upper bound of Theorem 2.3, we prove the tree analogue of Theorem 2.2. It even holds when the relation R is given as an alternating tree automaton (ATA), which allows us to apply it to recurrent reachability with *generalized Büchi condition.*

Theorem 2.4. *Given an ATA (D↑TA) $\mathcal{A}$ for a tree-regular relation $R \subseteq (\mathcal{T}_\Sigma)^{k+2}$, one can construct in exponential (polynomial) time an NTA for the relation $[\![\exists^{\text{ram}} x, y\colon R(x, y, z)]\!]$.*

If we make further assumptions on the relation R, we obtain a better complexity for NTAs. We say that a $(k+2)$-ary relation R over A is *transitive* if the binary relation $\{(a, b) \mid (a, b, \boldsymbol{c}) \in R\}$ is transitive for all $\boldsymbol{c} \in A^k$.

Theorem 2.5. *Given an NTA $\mathcal{A}$ for a transitive tree-regular $R \subseteq (\mathcal{T}_\Sigma)^{k+2}$, one can construct in polynomial time an NTA for the relation $[\![\exists^{\text{ram}} x, y\colon R(x, y, z)]\!]$. In particular, the infinite clique problem over transitive tree-regular relations is in* P.

A binary relation $R \subseteq A \times A$ is *co-transitive* if its complement $(A \times A) \setminus R$ is a transitive relation.

Theorem 2.6. *The infinite clique problem over co-transitive tree-regular relations $R \subseteq \mathcal{T}_\Sigma \times \mathcal{T}_\Sigma$ given as NTA is* P*-complete.*

In Section 7 we show by a reduction that the Ramsey quantifier can be evaluated over unranked tree-regular relations with the same complexity as in the ranked case.

Recurrent reachability. Since reachability in automatic graphs is in general undecidable [6], we will instead use *transitive paths*, i.e., infinite sequences $(a_i)_{i\geq 1}$ with $(a_i, a_j) \in R$ for all $1 \leq i < j$. Given sets $L_1, \ldots, L_k \subseteq A$ we write $\mathit{Rec}(L_1, \ldots, L_k)[R]$ for the set of all initial vertices a_1 of transitive paths $(a_i)_{i\geq 1}$ that visit each set L_j infinitely often. *Recurrent reachability with generalized Büchi condition* is the problem of testing $a_1 \in \mathit{Rec}(L_1, \ldots, L_k)[R]$ for a given (tree-)regular relation $R \subseteq A \times A$, (tree-)regular languages $L_1, \ldots, L_k \subseteq A$, and an initial element $a_1 \in A$. If $k = 1$ this problem is simply called *recurrent reachability.*

Since the infinite clique problem and recurrent reachability are logspace equivalent (Proposition 6.1), we obtain:

Corollary 2.7. *Recurrent reachability is* NL*-complete over regular relations. It is* EXP*-complete over tree-regular relations given by NTAs or D↓TAs, and* P*-complete if the tree-regular relations are transitive or given by D↑TAs.*

We also apply Theorems 2.2 and 2.4 to obtain tight upper bounds for recurrent reachability with generalized Büchi condition. The lower bounds result from a reduction from intersection nonemptiness.

Theorem 2.8. *Recurrent reachability with generalized Büchi condition is* PSPACE*-complete over regular relations, and* EXP*-complete over tree-regular relations.*

	regular relations	tree-regular relations
Automaton construction for Ramsey quantifier	logspace	exponential time poly-time for transitive relations
Recurrent reachability & infinite clique	NL-complete*	EXP-complete for NTA, D↓TA P-complete for transitive* or co-transitive relations or D↑TA
Recurrent reachability with generalized Büchi condition	PSPACE-complete	EXP-complete
Monadic decomposability	NL-complete for DFA* PSPACE-complete for NFA*	P-complete for D↑TA, D↓TA EXP-complete for NTA

Table 1: Complexity results. Those marked with * were known, but we provide simpler proofs. The other results are new.

Monadic Decomposability. Recall that a relation $R \subseteq A^k$ is *monadically decomposable* if it is of the form $\bigcup_{i=1}^{n} A_{i,1} \times \cdots \times A_{i,k}$ for some $n \in \mathbb{N}$ and (tree-)regular languages $A_{i,j}$. The traditional approach to deciding monadic decomposability [3, 7, 10, 20, 26] is to associate with R certain equivalence relations $\sim_j$ for $1 \leq j \leq k$ such that R is monadically decomposable if and only if each $\sim_j$ has finite index. An equivalence relation has infinite index if and only if there exist infinitely many elements that are pairwise in different equivalence classes which is witnessed by an infinite clique in the complement relation. Therefore, monadic decomposability amounts to checking that $\sim_j$'s complement $\nsim_j$ does not have an infinite clique for any j. If R is given by a DFA (resp. NFA), then one can construct an NFA for each $\nsim_j$ in logspace (resp. in PSPACE) and thus Theorem 2.2 yields a tight upper bound:

Corollary 2.9. *Given a regular relation $R \subseteq (\Sigma^*)^k$ by a DFA (resp. NFA), it is* NL*-complete (resp.* PSPACE*-complete) to decide whether R is monadically decomposable.*

While this approach yields optimal complexity for words, this is, unexpectedly, not the case for trees. For a tree-regular relation given as D↓TA or D↑TA (resp. NTA), one can also construct an NTA for each $\nsim_j$ in logspace (resp. PSPACE). Then, applying Theorem 2.4 would yield an EXP (resp. 2EXP) algorithm. However, perhaps surprisingly, monadic decomposability for trees has much lower complexity:

Corollary 2.10. *Given a tree-regular relation $R \subseteq \mathcal{T}_\Sigma^k$ by a D↓TA or D↑TA (resp. NTA), it is* P*-complete (resp.* EXP*-complete) to decide whether R is monadically decomposable.*

To get the P (resp. EXP) algorithm, we exploit the co-transitivity of each $\nsim_j$ and apply Theorem 2.6 instead of Theorem 2.4. This shows the importance of the co-transitivity notion: In the word case, monadic decomposability requires only the generic clique detection, but the tree case is more nuanced—we need one algorithm for the general case and a specialized algorithm for co-transitive relations.

3 PRELIMINARIES

We assume familiarity with the basic models of (non)deterministic and alternating finite automata on words and trees as well as with standard complexity classes (e.g., NL, P, PSPACE, EXP). We refer the reader to the textbooks [8, 16] for more details. We often abbreviate a finite or infinite sequence of elements $a_1, a_2, \ldots$ by a boldface letter $\boldsymbol{a}$.

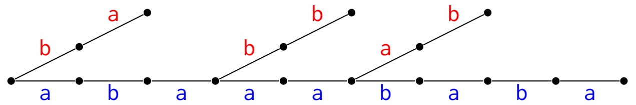
Trees. A *tree domain* is a nonempty set $D \subseteq \mathbb{N}^*$ such that (i) D is prefix closed, i.e., $uv \in D$ implies $u \in D$, (ii) for all $v \in \mathbb{N}^*$ and $j \leq i$ if $vi \in D$, then $vj \in D$, and (iii) each node $v \in D$ has only finitely many *children* $vi \in D$ where $i \in \mathbb{N}$. An *unranked tree* over an alphabet Σ is a function $t\colon \mathrm{dom}(t) \to \Sigma$ where $\mathrm{dom}(t)$ is a finite tree domain. A *ranked alphabet* is a finite alphabet Σ where every symbol $a \in \Sigma$ has a rank $\mathrm{rk}(a) \in \mathbb{N}$. A *ranked tree* is an unranked tree t such that every node $v \in \mathrm{dom}(t)$ has $\mathrm{rk}(t(v))$ many children. We denote the set of all ranked and unranked trees over Σ by $\mathcal{T}_\Sigma$ and $\mathcal{U}_\Sigma$, respectively.

Let $x \notin \Sigma$ be a *variable*. The set $\mathcal{C}_\Sigma$ of all *contexts* over Σ contains all unranked trees over $\Sigma \cup \{x\}$ such that every node $u \in \mathrm{dom}(t)$ with $t(u) = x$ is a leaf, called *hole*. We partition $\mathrm{dom}(t) = \mathrm{nodes}(t) \cup \mathrm{holes}(t)$ into nodes and holes. The size of a context t is $|\mathrm{nodes}(t)|$. For contexts $s, t_1, \ldots, t_n$ with $|\mathrm{holes}(s)| = n$ we denote by $s[t_1, \ldots, t_n]$ the context obtained by replacing the i-th hole in lexicographic order by t_i. For two contexts $s_1, s_2 \in \mathcal{C}_\Sigma$, we call s_1 a *prefix* of s_2, denoted by $s_1 \leq_\mathrm{p} s_2$, if $s_1[t_1, \ldots, t_n] = s_2$ for some contexts $t_1, \ldots, t_n$. If $n > 0$ and each context t_i has size at least one, then s_1 is a *proper prefix* of s_2, denoted by $s_1 <_\mathrm{p} s_2$.

We define an *infinite unranked tree* and an *infinite ranked tree* as in the finite case but with infinite domains. We denote the set of all finite and infinite unranked trees over the alphabet Σ by $\mathcal{U}_\Sigma^\infty$ and the set of all finite and infinite ranked trees over Σ by $\mathcal{T}_\Sigma^\infty$.

Regular and tree-regular languages. A *nondeterministic finite automaton* (NFA) over the alphabet Σ is a tuple $\mathcal{A} = (Q, \Sigma, \Delta, q_0, F)$ where Q is a finite set of states, $\Delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times Q$ is a transition relation, $q_0 \in Q$ is an initial state, and $F \subseteq Q$ is a set of final states. We denote by $L(\mathcal{A}) \subseteq \Sigma^*$ the *regular language* recognized by $\mathcal{A}$. In our algorithms, the alphabet Σ is *not* part of the representation of an automaton. Instead, we will always work with the subalphabet of all symbols occurring in the transitions. This will be important later when the implicitly given alphabet is significantly smaller.

A *nondeterministic (top-down) tree automaton* (NTA) over the ranked alphabet Σ is a tuple $\mathcal{A} = (Q, \Sigma, \Delta, q_0)$ where Q is a finite set of states, $q_0 \in Q$ is an initial state, and $\Delta \subseteq \bigcup_{a \in \Sigma} Q \times \{a\} \times Q^{\mathrm{rk}(a)}$ is a transition relation. A *run* of $\mathcal{A}$ on a tree $t \in \mathcal{T}_\Sigma$ is a tree $\rho \in \mathcal{T}_Q$ with $\mathrm{dom}(\rho) = \mathrm{dom}(t)$ such that $\rho(\varepsilon) = q_0$ and



$$\begin{bmatrix}b\\a\end{bmatrix}\begin{bmatrix}a\\b\end{bmatrix}\begin{bmatrix}\perp\\a\end{bmatrix}\#\begin{bmatrix}b\\a\end{bmatrix}\begin{bmatrix}b\\a\end{bmatrix}\#\begin{bmatrix}a\\b\end{bmatrix}\begin{bmatrix}b\\a\end{bmatrix}\begin{bmatrix}\perp\\b\end{bmatrix}\begin{bmatrix}\perp\\a\end{bmatrix}\#\dots$$

Figure 1: An example word comb ba, ababb, abaaaab, . . . and its encoding as an infinite word.

$(\rho(u), t(u), \rho(u1), \dots, \rho(ur)) \in \Delta$ for all nodes $u \in \mathrm{dom}(\rho)$ with $\mathrm{rk}(t(u)) = r$. As before, $L(\mathcal{A})$ is the set of trees recognized by $\mathcal{A}$, i.e., the set of all trees t such that there exists a run of $\mathcal{A}$ on t. A set of trees is called *tree-regular* if there is an NTA that recognizes it. We will also use the notions of *deterministic finite automata* (DFA), *deterministic bottom-up* (D↑TA), and *deterministic top-down tree automata* (D↓TA). Moreover, *alternating automata* will be formally introduced in later sections.

Regular and tree-regular relations. Let Σ be a finite alphabet and let $\Sigma_\perp = \Sigma \cup \{\perp\}$ where $\perp \notin \Sigma$ is a fresh symbol. For words $w_1, \dots, w_k \in \Sigma^*$ with $w_i = a_{i,1} \dots a_{i,n_i}$ and $n := \max\{n_i \mid 1 \le i \le k\}$ we define their convolution

$$w_1 \otimes \dots \otimes w_k := \begin{bmatrix} w_1 \\ \vdots \\ w_k \end{bmatrix} := \begin{pmatrix} a'_{1,1} \\ \vdots \\ a'_{k,1} \end{pmatrix} \dots \begin{pmatrix} a'_{1,n} \\ \vdots \\ a'_{k,n} \end{pmatrix} \in (\Sigma_\perp^k)^*$$

where $a'_{i,j} = a_{i,j}$ if $j \le n_i$ and $a'_{i,j} = \perp$ otherwise. A relation $R \subseteq (\Sigma^*)^k$ is *recognized* by an NFA $\mathcal{A}$ if $L(A) = \{w_1 \otimes \dots \otimes w_k \mid (w_1, \dots, w_k) \in R\}$. In that case we call R *regular.*

We extend the definitions to tree-regular relations. For an alphabet Σ we set again $\Sigma_\perp = \Sigma \cup \{\perp\}$ where $\perp \notin \Sigma$ is a fresh symbol. Let ε be the *empty tree* with $\mathrm{dom}(\varepsilon) := \emptyset$. Given k trees $t_1, \dots, t_k \in \mathcal{U}_\Sigma \cup \{\varepsilon\}$ we define their convolution $t = t_1 \otimes \dots \otimes t_k \in \mathcal{U}_{\Sigma_\perp^k} \cup \{\varepsilon\}$ with $\mathrm{dom}(t) = \bigcup_{i=1}^k \mathrm{dom}(t_i)$ and $t(v) = (t'_1(v), \dots, t'_k(v))$ where $t'_i(v) = t_i(v)$ if $v \in \mathrm{dom}(t_i)$ and $t'_i(v) = \perp$ otherwise. Observe that the degree of a node v in $t_1 \otimes \dots \otimes t_k$ is the maximum degree of v in a tree t_i such that $v \in \mathrm{dom}(t_i)$. Similar to the word case, we also write the convolution of trees as a column vector. If all t_i are ranked trees, then also t is a ranked tree with $\mathrm{rk}(a_1, \dots, a_k) := \max\{\mathrm{rk}(a_i) \mid 1 \le i \le k\}$ for all $(a_1, \dots, a_k) \in \Sigma_\perp^k$ where $\mathrm{rk}(\perp) := 0$. A relation $R \subseteq \mathcal{T}_\Sigma^k$ is *recognized* by an NTA $\mathcal{A}$ if the tree language $\{t_1 \otimes \dots \otimes t_k \mid (t_1, \dots, t_k) \in R\}$ is recognized by $\mathcal{A}$. In that case we call R *tree-regular.*

Regular and tree-regular relations are effectively closed under first-order operations (Boolean operations and projections). A relational structure $\mathfrak{A}$ is *automatic (tree-automatic)* if its universe and all its relations are regular (tree-regular).

4 WORD-AUTOMATIC STRUCTURES

We first consider Ramsey quantifiers over word-regular relations. We show that if $R \subseteq (\Sigma^*)^{k+2}$ is a regular relation, then an automaton for $\{\boldsymbol{c} \mid \exists^{\mathrm{ram}} x, y \colon R(x, y, \boldsymbol{c})\}$ can be constructed in logarithmic space (Theorem 2.2).

Word combs. The first step is to observe that, when looking for infinite cliques in R, one can restrict to combs: An infinite sequence $\boldsymbol{v}$ of words is called a *comb* if there exist infinite sequences $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ of words with $v_i = \beta_1 \dots \beta_{i-1}\alpha_i$ and $1 \le |\alpha_i| \le |\beta_i|$ for all $i \ge 1$. The pair $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ is called a *generator* of $\boldsymbol{v}$. We remark that the choice of the generator is not unique. Any infinite subsequence of a comb is again a comb. In fact, the following lemma is well-known, see [18, Lemma 5.1].

Lemma 4.1. *Any sequence $\boldsymbol{w}$ of pairwise distinct words w_i over a finite alphabet Σ contains a comb as a subsequence.*

In contrast to arbitrary infinite sequences of words, combs can be encoded naturally by infinite words. If $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ is a generator we call the infinite word

$$\mathrm{enc}(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \begin{bmatrix}\alpha_1\\\beta_1\end{bmatrix} \# \begin{bmatrix}\alpha_2\\\beta_2\end{bmatrix} \# \dots \in ((\Sigma_\perp \times \Sigma) \cup \{\#\})^\omega$$

the *encoding of* $(\boldsymbol{\alpha}, \boldsymbol{\beta})$, or also an encoding of $\boldsymbol{v}$.

Comb of combs. The next goal would be to construct a Büchi automaton which reads an encoding of a comb $\boldsymbol{v}$ and verifies that $v_i \otimes v_j$ has an accepting run $\rho(v_i, v_j)$ for all $1 \le i < j$. In general, this is challenging since it is not clear how a finite automaton can keep track of infinitely many runs (let alone, an automaton of polynomial size). Instead we will show that every infinite clique contains an infinite subclique whose accepting runs can be arranged in a dag of constant width, and can therefore be recognized by a polynomial-sized Büchi automaton.

Consider a comb $\boldsymbol{v}$ with generator $(\boldsymbol{\alpha}, \boldsymbol{\beta})$. First observe that the convolutions $v_i \otimes v_j$ can be written as

$$\begin{bmatrix}v_i\\v_j\end{bmatrix} = \begin{bmatrix}\beta_1\\\beta_1\end{bmatrix} \dots \begin{bmatrix}\beta_{i-1}\\\beta_{i-1}\end{bmatrix}\begin{bmatrix}\alpha_i\\\beta_i\end{bmatrix}\begin{bmatrix}\varepsilon\\\beta_{i+1}\end{bmatrix} \dots \begin{bmatrix}\varepsilon\\\beta_{j-1}\end{bmatrix}\begin{bmatrix}\varepsilon\\\alpha_j\end{bmatrix} \tag{1}$$

and can hence be arranged in a trie displayed in Figure 2, that we call *comb of combs.* The next insight is that we can ensure that the accepting runs $\rho(v_i, v_j)$ on the convolutions $v_i \otimes v_j$ match this comb of combs structure, after replacing $\boldsymbol{v}$ by an infinite subsequence. Roughly speaking, the runs look like as if the automaton for R would be *deterministic.* For example, all runs $\rho(v_1, v_j)$ for $j > 1$ share a common prefix which is a run on $\alpha_1 \otimes \beta_1$, and all runs $\rho(v_i, v_j)$ for $1 < i < j$ share a common prefix which is a run on $\beta_1 \otimes \beta_1$.

A *run* of an NFA $\mathcal{A} = (Q, \Sigma, \Delta, q_{\mathrm{in}}, F)$ on a nonempty word $a_1 \dots a_n \in \Sigma^*$ (runs on the empty word are not needed here) is a sequence $(q_0, a_1, q_1)(q_1, a_2, q_2) \dots (q_{n-1}, a_n, q_n)$ of triples in $Q \times \Sigma \times Q$ such that there exists a path from q_{i-1} to q_i in $\mathcal{A}$ labeled with a_i for all $1 \le i \le n$. A run is *accepting* if $q_0 = q_{\mathrm{in}}$ and $q_n \in F$.

We define a *decomposition* of a word $w \in \Sigma^*$ as $w = u_1 \dots u_n$ where $u_i \in \Sigma^*$. The decomposition in Equation (1) is called the $(\boldsymbol{\alpha}, \boldsymbol{\beta})$-decomposition of $v_i \otimes v_j$. We say that a decomposition of a run $\rho = \rho_1 \dots \rho_n$ of an NFA is *compatible* with a decomposition $w = u_1 \dots u_n$ of a word if ρ_i is a run on u_i for all $i \in [1, n]$.

We say that a generator $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ of a comb $\boldsymbol{v}$ is *coarser* than a generator $(\boldsymbol{\gamma}, \boldsymbol{\delta})$ of a comb $\boldsymbol{w}$ if there exist indices $k_1 < k_2 < \dots$ such that $v_i = w_{k_i}$ and $\beta_1 \dots \beta_i = \delta_1 \dots \delta_{k_i}$ for all $i \ge 1$. In this case we also say that $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ is the coarsening of $(\boldsymbol{\gamma}, \boldsymbol{\delta})$ defined by the subsequence $\boldsymbol{v}$ of $\boldsymbol{w}$.

Lemma 4.2. *Let $\boldsymbol{w}$ be a comb generated by $(\boldsymbol{\gamma}, \boldsymbol{\delta})$ that forms an infinite clique in a regular relation $R \subseteq \Sigma^* \times \Sigma^*$ given as an NFA $\mathcal{A}$. There exist a coarsening $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ of $(\boldsymbol{\gamma}, \boldsymbol{\delta})$ that generates a comb $\boldsymbol{v}$, accepting runs $\rho(v_i, v_j)$ of $\mathcal{A}$ on $v_i \otimes v_j$, and runs $\kappa_i, \lambda_i, \mu_{i,j}, \nu_{i,j}$ such that*

$$\rho(v_i, v_j) = \kappa_1 \dots \kappa_{i-1} \lambda_i \mu_{i,i+1} \dots \mu_{i,j-1} \nu_{i,j}$$

is a decomposition compatible with the $(\boldsymbol{\alpha}, \boldsymbol{\beta})$-decomposition of $v_i \otimes v_j$ for all $i < j$.

Proof. Let $\boldsymbol{w}$ be a comb generated by $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ that forms an infinite clique in R and $\rho(w_i, w_j)$ be an accepting run of $\mathcal{A}$ on $w_i \otimes w_j$ for all $1 \le i < j$. We establish the run structure as illustrated in Figure 3 column-wise.

Assume we already defined $(\kappa_i)_{i<n}$, $(\lambda_i)_{i<n}$, $(\mu_{i,j})_{i<j<n}$, and $(\nu_{i,j})_{i<j<n}$ for some $n \ge 1$ such that

$$\rho(w_i, w_j) = \kappa_1 \dots \kappa_{i-1} \lambda_i \mu_{i,i+1} \dots \mu_{i,n-1} \tau(w_i, w_j)$$
$$\rho(w_{i'}, w_{j'}) = \kappa_1 \dots \kappa_{n-1} \sigma(w_{i'}, w_{j'})$$

for runs $\tau(w_i, w_j)$, $\sigma(w_{i'}, w_{j'})$ for all $1 \le i \le n-1 < j$ and $n-1 < i' < j'$. For all $1 \le i < n$ define $\nu_{i,n} := \tau(w_i, w_n)$.

We now define $\mu_{i,n}$ successively for each $1 \le i < n$. In step i we apply the pigeonhole principle to get an infinite subsequence $\boldsymbol{v}$ of $\boldsymbol{w}$ starting with $w_1, \dots, w_n$ such that all runs $\tau(v_i, v_j)$ for $j > n$ have a common prefix $\mu_{i,n}$ which is a run on $\varepsilon \otimes \beta_n$. At the end of step i we replace $\boldsymbol{w}$ by $\boldsymbol{v}$ and we replace $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ by the coarsening defined by $\boldsymbol{v}$.

Next we define λ_n. By the pigeonhole principle there exists an infinite subsequence $\boldsymbol{v}$ of $\boldsymbol{w}$ starting with $w_1, \dots, w_n$ such that all runs $\sigma(v_n, v_j)$ for $j > n$ have a common prefix λ_n which is a run on $\alpha_n \otimes \beta_n$. Again we replace $\boldsymbol{w}$ by $\boldsymbol{v}$ and $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ by the coarsening defined by $\boldsymbol{v}$.

Finally, by Ramsey's theorem there is an infinite subsequence $\boldsymbol{v}$ of $\boldsymbol{w}$ starting with $w_1, \dots, w_n$ such that all runs $\sigma(v_i, v_j)$ for $n < i < j$ have a common prefix κ_n which is a run on $\beta_n \otimes \beta_n$. We replace $\boldsymbol{w}$ by $\boldsymbol{v}$ and $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ by the coarsening defined by $\boldsymbol{v}$.

In the limit we obtain the desired decomposition of the runs $\rho(v_i, v_j)$ and the generator $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ of a comb $\boldsymbol{v}$ that is coarser than the initial generator of $\boldsymbol{w}$. □

It is not hard to see that such a comb of combs structure can be simulated by an alternating Büchi automaton, which would only yield a PSPACE-solution for the infinite clique problem. The following key lemma states that the runs $\mu_{i,j}$, $\nu_{i,j}$ from Lemma 4.2 can be chosen independently from i, which reduces the width of the run dag of the alternating automaton to a constant.

Lemma 4.3. *If $\boldsymbol{w}$ is an infinite clique in a regular relation $R \subseteq \Sigma^* \times \Sigma^*$ given as an NFA $\mathcal{A}$, then there exist a generator $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ for a subsequence $\boldsymbol{v}$ of $\boldsymbol{w}$, accepting runs $\rho(v_i, v_j)$ of $\mathcal{A}$ on $v_i \otimes v_j$, and runs $\kappa_i, \lambda_i, \mu_j, \nu_j$ such that*

$$\rho(v_i, v_j) = \kappa_1 \dots \kappa_{i-1} \lambda_i \mu_{i+1} \dots \mu_{j-1} \nu_j$$

is a decomposition compatible with the $(\boldsymbol{\alpha}, \boldsymbol{\beta})$-decomposition of $v_i \otimes v_j$ for all $i < j$.

Proof. Suppose that R has an infinite clique. Then there exist an infinite clique $\boldsymbol{w}$ in R generated by $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ and runs $\kappa_i, \lambda_i, \mu_{i,j}, \nu_{i,j}$ for $i < j$ as in Lemma 4.2.

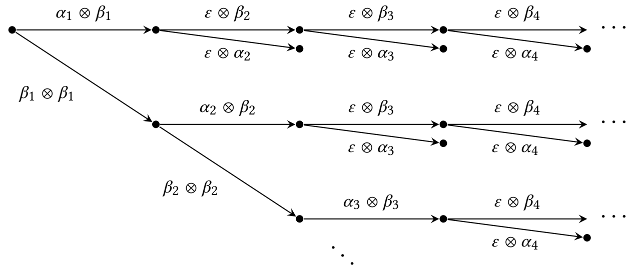

$\alpha_1 \otimes \beta_1$
$\varepsilon \otimes \beta_2$
$\varepsilon \otimes \beta_3$
$\varepsilon \otimes \beta_4$
$\varepsilon \otimes \alpha_2$
$\varepsilon \otimes \alpha_3$
$\varepsilon \otimes \alpha_4$
$\beta_1 \otimes \beta_1$
$\alpha_2 \otimes \beta_2$
$\varepsilon \otimes \beta_3$
$\varepsilon \otimes \beta_4$
$\varepsilon \otimes \alpha_3$
$\varepsilon \otimes \alpha_4$
$\beta_2 \otimes \beta_2$
$\alpha_3 \otimes \beta_3$
$\varepsilon \otimes \beta_4$
$\varepsilon \otimes \alpha_4$

Figure 2: If $\boldsymbol{v}$ is a comb of the form $v_i = \beta_1 \dots \beta_{i-1} \alpha_i$ then the convolutions $v_i \otimes v_j$ for all $i < j$ form a *comb of combs*.

It remains to ensure that $\mu_{i,j} = \mu_{i',j}$ and $\nu_{i,j} = \nu_{i',j}$ for all $i < i' < j$. To do so, consider the initial state of the run $\mu_{i,j+1}$. By Ramsey's theorem there exist indices $k_1 < k_2 < \dots$ such that all runs μ_{k_i,k_j+1} have the same initial state. We define $v_i = w_{k_i}$ for all $i \ge 1$, and

$$\tilde{\kappa}_i = \kappa_{k_{i-1}+1} \dots \kappa_{k_i} \qquad \tilde{\lambda}_i = \kappa_{k_{i-1}+1} \dots \kappa_{k_i-1} \lambda_{k_i}$$
$$\tilde{\mu}_{i+1} = \mu_{k_1,k_i+1} \dots \mu_{k_1,k_{i+1}} \qquad \tilde{\nu}_{i+1} = \mu_{k_1,k_i+1} \dots \mu_{k_1,k_{i+1}-1} \nu_{k_1,k_{i+1}}$$

for all $i \ge 1$ where $k_0 := 0$. Observe that the composition $\tilde{\lambda}_i \tilde{\mu}_{i+1}$ forms a valid run since μ_{k_i,k_i+1} and μ_{k_1,k_i+1} have the same initial state. Then $\tilde{\kappa}_1 \dots \tilde{\kappa}_{i-1} \tilde{\lambda}_i \tilde{\mu}_{i+1} \dots \tilde{\mu}_{j-1} \tilde{\nu}_j$ is an accepting run on $v_i \otimes v_j$. Furthermore, this run decomposition is compatible with the $(\boldsymbol{\delta}, \boldsymbol{\gamma})$-decomposition of $v_i \otimes v_j$ where the generator $(\boldsymbol{\delta}, \boldsymbol{\gamma})$ is defined as

$$\delta_i = \beta_{k_{i-1}+1} \dots \beta_{k_i}$$
$$\gamma_i = \beta_{k_{i-1}+1} \dots \beta_{k_i-1} \alpha_{k_i}$$

for all $i \ge 1$ where $k_0 := 0$. This concludes the proof. □

A *nondeterministic Büchi automaton* (NBA) has the same format as an NFA $\mathcal{B} = (Q, \Sigma, \Delta, q_0, F)$. An infinite word $w \in \Sigma^\omega$ is *accepted* by $\mathcal{B}$ if there exists an accepting run $(q_0, a_1, q_1)(q_1, a_2, q_2) \dots \in \Delta^\omega$ with $w = a_1 a_2 \dots$ and $q_i \in F$ for infinitely many $i \ge 0$.

Proposition 4.4. *Given an NFA $\mathcal{A}$ for a relation $R \subseteq (\Sigma^*)^2$, one can construct in logarithmic space a Büchi automaton $\mathcal{B}$ over the alphabet $(\Sigma_\perp \times \Sigma) \cup \{\#\}$ such that:*

- *If $\boldsymbol{w}$ is an infinite clique in R then $\mathcal{B}$ accepts an encoding of a comb $\boldsymbol{v}$ which is a subsequence of $\boldsymbol{w}$.*
- *If $\mathcal{B}$ accepts an encoding of a comb $\boldsymbol{w}$ then $\boldsymbol{w}$ is an infinite clique in R.*

Proof. Given an NFA $\mathcal{A} = (Q, \Sigma_\perp^2, q_{\text{in}}, \Delta, F)$, we add to $\mathcal{A}$ a fresh state $\perp$ and transitions $\perp \xrightarrow{(a,b)} q$ for all $(a, b) \in \Sigma_\perp^2$ and $q \in Q_\perp$ where $Q_\perp := Q \cup \{\perp\}$.

The Büchi automaton $\mathcal{B} = (Q_\perp^4, \Sigma_\perp^2, q_{\text{in}}^{\mathcal{B}}, \Delta^{\mathcal{B}}, Q_\perp^4)$ simulates the runs $\kappa_j, \lambda_j, \mu_j, \nu_j$ from Lemma 4.3 in the four components. Its initial state is $q_{\text{in}}^{\mathcal{B}} = (q_{\text{in}}, q_{\text{in}}, \perp, \perp)$ and it contains the following transtions:

- $(p, s, q, t) \xrightarrow{(a,b)}_{\mathcal{B}} (p', s', q', t')$ for all transitions $p \xrightarrow{(b,b)}_{\mathcal{A}} p'$, $s \xrightarrow{(a,b)}_{\mathcal{A}} s'$, $q \xrightarrow{(\perp,b)}_{\mathcal{A}} q'$, $t \xrightarrow{(\perp,a)}_{\mathcal{A}} t'$,
- $(p, q, q, t) \xrightarrow{\#}_{\mathcal{B}} (p, p, q, q)$ for all $p, q \in Q$, $t \in F$,

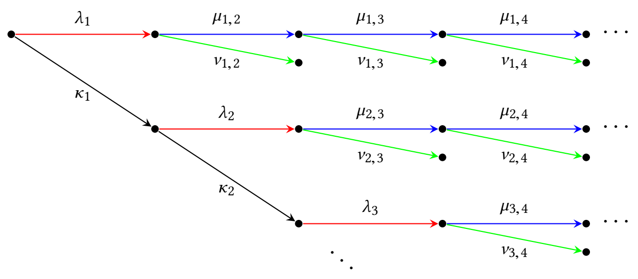


Figure 3: One can always find an infinite comb clique $\boldsymbol{v}$ with accepting runs which can be decomposed in the form $\rho(v_i, v_j) = \kappa_1 \dots \kappa_{i-1}\lambda_i\mu_{i,i+1} \dots \mu_{i,j-1}\nu_{i,j}$.

- $(p_1, p_2, p_3, p_4) \xrightarrow{\varepsilon}_{\mathcal{B}} (q_1, q_2, q_3, q_4)$ if $p_i = q_i$ or $p_i \xrightarrow{\varepsilon}_{\mathcal{A}} q_i$ for all $i \in [1, 4]$.

The desired Büchi automaton is a product automaton of $\mathcal{B}$ and a Büchi automaton which verifies that the input word is a valid comb encoding $\mathrm{enc}(\boldsymbol{\alpha}, \boldsymbol{\beta})$. □

Proof of Theorem 2.2. First observe that we can construct in log-space an NFA C over Σ such that (i) for every infinite clique $\boldsymbol{w}$ of R some element w_i is accepted by C, and (ii) if w is accepted by C then w belongs to an infinite clique of R. To be more precise, C accepts $\alpha_1 \in \Sigma^*$ if and only if some encoding $\mathrm{enc}(\boldsymbol{\alpha}, \boldsymbol{\beta})$ is accepted by the Büchi automaton $\mathcal{B}$ from Proposition 4.4. This can be done in log-space as follows: First we construct a Büchi automaton $\hat{C}$ over $\Sigma \cup \{\#\}$ which accepts all words of the form $\alpha_1\#^\omega$ such that an encoding $\mathrm{enc}(\boldsymbol{\alpha}, \boldsymbol{\beta})$ is accepted by $\mathcal{B}$. Then $\hat{C}$ is turned into an NFA C (which does not read the suffix $\#^\omega$) by replacing #-transitions by ε-transitions. Furthermore C tracks the number of final states visited so far and accepts if and only if this number exceeds the number of states in $\hat{C}$.

Given an NFA $\mathcal{A}$ for $R \subseteq (\Sigma^*)^{k+2}$. We first construct an NFA $\mathcal{A}'$ over $\Sigma_\perp^{2k+2}$ which accepts the regular binary relation

$$R' = \{(u \otimes c_1 \otimes \dots \otimes c_k, v \otimes c_1 \otimes \dots \otimes c_k) \mid (u, v, \boldsymbol{c}) \in R\}.$$

Note that $\mathcal{A}'$ can be constructed in logspace since it is obtained by taking each transition of $\mathcal{A}$ and duplicating the $\boldsymbol{c}$-coordinates and moving the v-coordinate. Let C' be the NFA described above which accepts at least one word from each infinite R'-clique and only accepts elements of infinite R'-cliques. Projecting away the first component yields the desired NFA for $\{\boldsymbol{c} \in (\Sigma^*)^k \mid \exists^{\mathrm{ram}} x, y \colon R(x, y, \boldsymbol{c})\}$. □

5 TREE-AUTOMATIC STRUCTURES

5.1 EXP-hardness

In this section we prove the exponential lower bound from Theorem 2.3 for the infinite clique problem over trees. This lower bound is surprising since over words, the infinite clique problem can be reduced to the emptiness of (word) Büchi automata, which is NL-complete. This is not the case in the tree case since emptiness of Büchi tree automata is P-complete.

We start with an intutive explanation of the lower bound. To prove the upper bound in the word case, we used the fact that

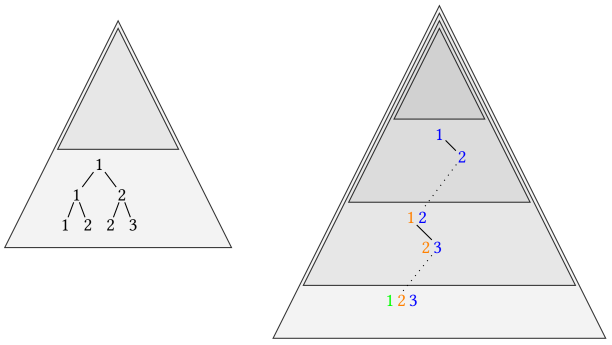


Figure 4: Left: Illustration of the tree automaton $\mathcal{T}$, tracking the number of right directions modulo n. Right: A path on which the runs of $\mathcal{T}$ are disjoint.

we can assume cliques $\boldsymbol{v}$ whose runs $\rho(v_i, v_j)$ can be merged into a *single* global run (Lemma 4.3). Over tree regular relations this is not the case anymore. Consider a deterministic top-down tree automaton $\mathcal{T}$, which behaves as follows on the convolution $t \otimes t'$ of two binary trees t, t' with $\mathrm{dom}(t) \subsetneq \mathrm{dom}(t')$: Starting from every node on the fringe of t, the automaton tracks the number of times it moves to a right child, modulo some number n; see Figure 4 for a depiction. Now consider an increasing sequence of binary trees $t_1, t_2, \dots$, and the unique runs $\rho_{i,j}$ of $\mathcal{T}$ on $t_i \otimes t_j$. Figure 4 illustrates that we can always find a path on which the runs $\rho_{1,n}, \rho_{2,n}, \dots, \rho_{n-1,n}$ are disjoint. This behavior indicates that it is difficult to witness the existence of infinite cliques by a polynomially-sized Büchi tree automaton.

We extend this idea to a reduction from the intersection non-emptiness problem for tree automata, which is known to be EXP-complete [9]: Given an NTA $\mathcal{A} = (Q, \Sigma, \Delta, q_0)$ and states $q_1, \dots, q_n \in Q$, decide whether $\bigcap_{i=1}^n L(\mathcal{A}_{q_i})$ is non-empty. Here, $\mathcal{A}_{q_i}$ denotes the NTA $\mathcal{A}$ with initial state q_i.

We construct a relation R on decorated trees, which are obtained from a binary tree by attaching to every inner node u a ranked tree $\delta(u)$ over Σ. Let $\Gamma := \Sigma \cup \{a, c\}$ be a ranked alphabet with $\mathrm{rk}(a) = 3$, $\mathrm{rk}(c) = 0$. A *decorated tree* is a tree $t \in \mathcal{T}_\Gamma$ such that $t(\varepsilon) \in \{a, c\}$ and for all $u \in \mathrm{dom}(t)$ we have

- $t(u1) = t(u2) \in \{a, c\}$ and $t(u3) \in \Sigma$ if $t(u) = a$ and
- $t(ui) \in \Sigma$ for all $i \in [1, \mathrm{rk}(t(u))]$ if $t(u) \in \Sigma$.

We denote by $a(t) := \{u \in \mathrm{dom}(t) \mid t(u) = a\}$ the nodes of t labeled with a and by $ac(t) := \{u \in \mathrm{dom}(t) \mid t(u) \in \{a, c\}\}$ the nodes labeled with a or c. The *decoration* of t is a function $\delta_t \colon a(t) \to \mathcal{T}_\Sigma$ such that $\delta_t(u) = t_{\downarrow u3}$ where $t_{\downarrow v}$ denotes the subtree of t rooted in $v \in \mathrm{dom}(t)$.

Let $\mathcal{A}' = (Q', \Gamma, \Delta', p_1)$ be the NTA where $Q' := Q \cup \{p_1, \dots, p_n\}$ and Δ' contains all transitions from Δ and the transitions

$$p_i \xrightarrow{a} (p_i, p_{i+1}, q_i) \text{ for all } i \in [1, n] \text{ where } p_{n+1} := p_1,$$

$$p_i \xrightarrow{c} () \text{ for all } i \in [1, n].$$

We define the tree-regular relation $R \subseteq \mathcal{T}_\Gamma \times \mathcal{T}_\Gamma$ such that $(s, t) \in R$ if and only if s and t are decorated trees with $ac(s) \subseteq a(t)$ and $\mathcal{A}'$ accepts $t_{\downarrow u}$ for all $u \in \min(a(t) \setminus a(s))$. Here, the minimum is defined with respect to prefix ordering. It is easy to construct an NTA that recognizes R in logspace.

It remains to show that $\bigcap_{i=1}^{n} L(\mathcal{A}_{q_i}) \neq \emptyset$ if and only if R contains an infinite clique. For the "only if" direction let $t \in \mathcal{T}_\Sigma$ be a tree that is accepted by $\mathcal{A}_{q_i}$ for all $i \in [1,n]$. For all $i \geq 0$ we define the decorated tree $t_i \in \mathcal{T}_\Gamma$ such that $ac(t_i) = \bigcup_{j=0}^{i}\{1,2\}^j$ and $\delta_{t_i}(u) = t$ for all $u \in a(t_i)$. It is easy to verify that $(t_i, t_j) \in R$ for all $i < j$.

Conversely, consider a sequence of decorated trees $t_i \in \mathcal{T}_\Gamma$ for $i \geq 0$ with $(t_i, t_j) \in R$ for all $i < j$. We define nodes $v_1, \dots, v_n$ with

- $v_i \in \min(a(t_i) \setminus a(t_{i-1}))$ for all $i \in [1,n]$ and
- $v_{i+1} = v_i 21^{k_i}$ for all $i \in [1, n-1]$ and some $k_i \geq 0$.

We can choose $v_1 \in \min(a(t_1) \setminus a(t_0))$ arbitrary which defines $v_2, \dots, v_n$ uniquely. Since $(t_{i-1}, t_n) \in R$ and $v_i \in \min(a(t_n) \setminus a(t_{i-1}))$, the subtree $t_{n\downarrow v_i}$ is accepted by $\mathcal{A}'$ for all $i \in [1,n]$. By definition of $\mathcal{A}'$ there exist accepting runs on $t_{n\downarrow v_n}$ starting from $p_1, \dots, p_n$. Therefore, the tree $\delta_{t_n}(v_n) \in \mathcal{T}_\Sigma$ is accepted by $\mathcal{A}$ starting from all states $q_1, \dots, q_n$.

We note that EXP-hardness already holds if R is given by a D↓TA since intersection nonemptiness is EXP-hard already for D↓TAs [30], and if the automaton $\mathcal{A}$ is a D↓TA, then the constructed relation R from the proof can also be recognized by a D↓TA. Moreover, the reduction can be adapted to recurrent reachability by setting the target set to $\mathcal{T}_\Sigma$, which proves the exponential lower bound in Corollary 2.7.

5.2 Tree combs

To prove the upper bounds for tree-regular relations, we extend the notion of combs to the tree case. Here a tree t_i is decomposed vertically in the form $t_i = \beta_1 \dots \beta_{i-1}\alpha_i$ where β_1 is a context, $\beta_2, \dots, \beta_{i-1}$ are forests of contexts, and α_i is a forest of trees, see Figure 5 for an abstract illustration.

A *context forest* of width n is a finite sequence $\tau = (c_i)_{1 \leq i \leq n}$ of contexts $c_i \in C_\Sigma$. Context forests of width 1 are regarded as contexts. We say that τ is *nontrivial* if $n \geq 1$ and $|c_i| \geq 1$ for all $1 \leq i \leq n$. If the c_i are trees in $\mathcal{U}_\Sigma$, we call τ just a *forest*. We define the concatenation of a context $\tau_1 \in C_\Sigma$, where $|\mathrm{holes}(\tau_1)| = n$, with a context forest $\tau_2 = (c_1, \dots, c_n) \in \mathcal{F}_\Sigma$ of width n by $\tau_1\tau_2 := \tau_1[c_1, \dots, c_n]$. We write $\tau_1\tau_2 \dots \tau_n$ for a context τ_1 and context forests $\tau_2, \dots, \tau_n$ assuming left-associativity. Here we implicitly assume that the width of τ_i matches $|\mathrm{holes}(\tau_1 \dots \tau_{i-1})|$. If t is a tree and s is a context, we write $t \triangleleft s$ if $\mathrm{dom}(t) \cap \mathrm{holes}(s) = \emptyset$. For a forest $\alpha = (t_i)_{i \leq n}$ and a context forest $\beta = (s_i)_{i \leq n}$ we also write $\alpha \triangleleft \beta$ if $t_i \triangleleft s_i$ for all $1 \leq i \leq n$.

An infinite sequence $\boldsymbol{t} = (t_i)_{i \geq 1}$ of ranked trees is called a *comb* if there is a sequence of forests $\boldsymbol{\alpha} = (\alpha_i)_{i \geq 1}$ and a sequence of nontrivial context forests $\boldsymbol{\beta} = (\beta_i)_{i \geq 1}$ such that for all $i \geq 1$ we have $t_i = \beta_1 \dots \beta_{i-1}\alpha_i$ and $\alpha_i \triangleleft \beta_i$. The pair $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ is called *generator* of the comb. Since the trees t_i are ranked, also the trees in α_i and the contexts in β_i are ranked.

The property $\alpha_i \triangleleft \beta_i$ should be compared to the property $|\alpha_i| \leq |\beta_i|$ in word combs. It ensures that every forest α_i does not touch any context forest α_j, β_j for $j \neq i$.

Lemma 5.1 (Combs lemma over trees). *Any sequence $\boldsymbol{t}$ of pairwise distinct ranked trees $t_i \in \mathcal{T}_\Sigma$ over a finite ranked alphabet Σ contains a comb as a subsequence.*

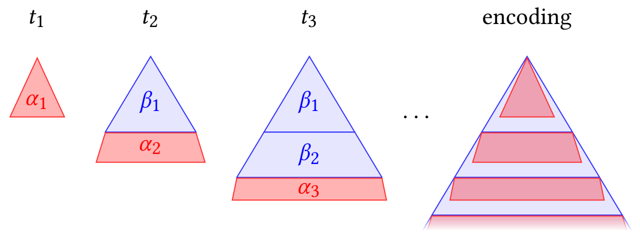


Figure 5: An example tree comb and its encoding as an infinite tree. In this example the generator satisfies $\mathrm{dom}(\alpha_i) \subseteq \mathrm{dom}(\beta_i)$ whereas general generators only satisfy $\alpha_i \triangleleft \beta_i$.

Proof. It suffices to show that for any infinite set $T \subseteq \mathcal{T}_\Sigma$ there exists a comb over T. Consider the following finitely branching infinite tree whose nodes are contexts from C_Σ. The root is the context x. The children of a context s are the contexts of the form $s[t_1, \dots, t_n]$ where each t_i is a context of size one. Observe that all trees in $\mathcal{T}_\Sigma$ occur as nodes in the infinite tree. The set of all ancestors of trees in T form an infinite subtree, which contains an infinite path $s_0 <_p s_1 <_p s_2 <_p \dots$ by Kőnig's Lemma. For all $i \geq 1$ there exists a tree $t_i \in T$ which contains s_{i-1} as a prefix.

Since the minimal level of a hole in s_i is strictly increasing, for every $i \geq 1$ there exists a $j \geq i$ with $t_i \triangleleft s_j$. Hence one can inductively construct indices $1 = k_1 < k_2 < \dots$ such that $t_{k_i+1} \triangleleft s_{k_{i+1}}$ for all $i \geq 1$. Then $(t_{k_i+1})_{i \geq 1}$ is a comb where the generator $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ is defined such that $s_{k_{i+1}} = \beta_1 \dots \beta_i$ and $t_{k_i+1} = \beta_1 \dots \beta_{i-1}\alpha_i$ for $i \geq 1$. The comb property $\alpha_i \triangleleft \beta_i$ follows from $t_{k_i+1} \triangleleft s_{k_{i+1}}$. □

To define the encoding $\mathrm{enc}(\boldsymbol{\alpha}, \boldsymbol{\beta})$ of a comb generator we need a few more definitions. For a tree t and context s with $t \triangleleft s$ we define the convolution $t \otimes s$ as before but every $(\bot, x)$ is replaced by x. That is, $t \otimes s$ is again a context. We extend the convolution in a natural way to forests and context forests of the same width. If $\tau = (c_1, \dots, c_n)$ is a context forest let $\bar{\tau}$ be obtained from τ by attaching a new #-labeled root to each of the n contexts c_i. We can now define the *encoding* of a comb with generator $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ as the infinite tree

$$\mathrm{enc}(\boldsymbol{\alpha}, \boldsymbol{\beta}) := (\alpha_1 \otimes \beta_1)(\overline{\alpha_2 \otimes \beta_2})(\overline{\alpha_3 \otimes \beta_3}) \dots$$

over the ranked alphabet $\Omega := \Sigma_\bot^2 \cup \{\#\}$. See Figure 5 for an illustration of the encoding. Here, the forests α_i are colored red and the context forests β_i are colored blue. It is not hard to see that the set Enc_Σ of all comb encodings is a regular language of infinite trees.

5.3 Arbitrary relations

An *alternating tree automaton* (ATA) over the ranked alphabet Σ is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0)$ where Q is a set of states, $q_0 \in Q$ is an initial state, and $\delta\colon Q \times \Sigma \to \mathcal{B}^+(Q \times \mathbb{N})$ is a transition function with $\delta(q, a) \in \mathcal{B}^+(Q \times \{1, \dots, \mathrm{rk}(a)\})$ for all $q \in Q$, $a \in \Sigma$. Here, $\mathcal{B}^+(Q \times \mathbb{N})$ denotes the set of positive propositional formulas over the set of variables $Q \times \mathbb{N}$. For a set S of variables and formula φ we denote by $S \models \varphi$ that if the variables in S are set to true and the variables not in S are set to false, then φ is satisfied.

We will use a nonstandard definition of runs of ATAs. Firstly, we consider runs on both trees and contexts. Secondly, each node in the run also carries the labels of its children, with the purpose of predetermining the states in the context holes. A *run* of $\mathcal{A}$ on a nontrivial ranked context $t \in C_\Sigma$ is a context ρ over the alphabet $\mathbb{N}^* \times Q \times 2^{Q\times\mathbb{N}}$ such that $\rho(\varepsilon) = (\varepsilon, q_0, S)$ for some $S \subseteq Q \times \mathbb{N}$ and for each node $u \in \text{nodes}(\rho)$ with $\rho(u) = (w, q, S)$ and $S = \{(q_1, c_1), \ldots, (q_r, c_r)\}$ we have that $S \models \delta(q, t(w))$ and u has $r \geq 0$ children such that $\rho(ui) = (wc_i, q_i, S_i)$ for some $S_i \subseteq Q \times \mathbb{N}$ if $wc_i \in \text{nodes}(t)$, and $\rho(ui) = x$, otherwise. Note that an NTA can be seen as a special ATA where for all $q \in Q$ and $a \in \Sigma$, the transition formula $\delta(q, a)$ is a disjunction of conjunctions $\bigwedge_{i=1}^{\text{rk}(a)}(q_i, i)$.

We define a *decomposition* of a context t as $t = \tau_1 \ldots \tau_n$ where the τ_i are context forests. For a generator $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ of a comb $\boldsymbol{s}$ we define the $(\boldsymbol{\alpha}, \boldsymbol{\beta})$-decomposition of $s_i \otimes s_j$ as in the word case. We say that a decomposition of a run $\rho = \rho_1 \ldots \rho_n$ of an ATA is *compatible* with a decomposition $t = \tau_1 \ldots \tau_n$ of a context if $\rho_1 \ldots \rho_i$ is a run on $\tau_1 \ldots \tau_i$ for all $i \in [1, n]$. Note that the above definition of a run ensures that $\rho_1 \ldots \rho_i$ already determines the first two components of the root labels of ρ_{i+1} for all $i < n$.

We say that a generator $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ of a comb $\boldsymbol{s}$ is *coarser* than a generator $(\boldsymbol{\gamma}, \boldsymbol{\delta})$ of a comb $\boldsymbol{t}$ if there exist indices $k_1 < k_2 < \ldots$ such that $s_i = t_{k_i}$ and $\beta_1 \ldots \beta_i = \delta_1 \ldots \delta_{k_i}$ for all $i \geq 1$. In this case we also say that $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ is the coarsening of $(\boldsymbol{\gamma}, \boldsymbol{\delta})$ defined by the subsequence $\boldsymbol{s}$ of $\boldsymbol{t}$.

Lemma 5.2. *Let $\boldsymbol{t}$ be a comb generated by $(\boldsymbol{\gamma}, \boldsymbol{\delta})$ that forms an infinite clique in a tree-regular relation $R \subseteq \mathcal{T}_\Sigma \times \mathcal{T}_\Sigma$ given as an ATA $\mathcal{A}$. There exist a coarsening $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ of $(\boldsymbol{\gamma}, \boldsymbol{\delta})$ that generates a comb $\boldsymbol{s}$, runs $\rho(s_i, s_j)$ of $\mathcal{A}$ on $s_i \otimes s_j$, and context forests $\kappa_i, \lambda_i, \mu_{i,j}, \nu_{i,j}$ such that*

$$\rho(s_i, s_j) = \kappa_1 \ldots \kappa_{i-1} \lambda_i \mu_{i,i+1} \ldots \mu_{i,j-1} \nu_{i,j}$$

is a decomposition compatible with the $(\boldsymbol{\alpha}, \boldsymbol{\beta})$-decomposition of $s_i \otimes s_j$ for all $i < j$.

Proof. The proof is similar to the proof of Lemma 4.2 in the word case. We emphasize that the pigeonhole principle and Ramsey's theorem can be applied as in the word case since the unique prefixes of the runs of the ATA that are runs on a given context have bounded size. Further note that since a run on a forest is not defined, instead of considering only the suffixes $\tau(t_i, t_j)$ and $\sigma(t_i, t_j)$ in the inductive step, we have to consider the whole run $\rho(t_i, t_j)$ and extend the common prefix that is already fixed. ◻

An *alternating Büchi tree automaton* (ABTA) over the ranked alphabet Σ is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ where Q, Σ, δ, and q_0 are as in the definition of an ATA and $F \subseteq Q$ is a set of final states. A *run* of $\mathcal{A}$ on a finite or infinite tree $t \in \mathcal{T}_\Sigma^\infty$ is a tree $\rho \in \mathcal{U}_{\mathbb{N}^*\times Q}^\infty$ such that $\rho(\varepsilon) = (\varepsilon, q_0)$ and for all $u \in \text{dom}(\rho)$ with r children and $\rho(u) = (w, q)$ there is a satisfying assignment $S = \{(q_1, c_1), \ldots, (q_r, c_r)\} \models \delta(q, t(w))$ of pairwise distinct (q_i, c_i) such that $\rho(ui) = (wc_i, q_i)$ for all $i \in [1, r]$. A run ρ is *accepting* if every infinite path of ρ contains infinitely many nodes with labels in $\mathbb{N}^* \times F$.

Proposition 5.3. *Given an ATA $\mathcal{A}$ for a tree-regular relation $R \subseteq (\mathcal{T}_\Sigma)^2$, one can construct in polynomial time an ABTA $\mathcal{B}$ over the ranked alphabet $\Omega = \Sigma_\perp^2 \cup \{\#\}$ such that we have:*

- *If $\boldsymbol{t}$ is an infinite clique in R, then $\mathcal{B}$ accepts an encoding of a comb $\boldsymbol{s}$ which is a subsequence of $\boldsymbol{t}$.*
- *If $\mathcal{B}$ accepts $t \in \mathcal{T}_\Omega^\infty$, then t is an encoding of a comb $\boldsymbol{t}$ that is an infinite clique in R.*

Proof. Let $\mathcal{A} = (Q, \Sigma_\perp^2, \delta, q_0)$ be the ATA that recognizes R. We construct an ABTA $\mathcal{B}$ over Ω which accepts precisely all comb encodings $\text{enc}(\boldsymbol{\alpha}, \boldsymbol{\beta})$ with the properties from Lemma 5.2. The state set of $\mathcal{B}$ is $Q \times \{1, 2, 3, 4\}$, representing four different modes. In the first mode it simulates κ_j on $\beta_j \otimes \beta_j$, in the second mode it simulates λ_j on $\alpha_j \otimes \beta_j$, in the third mode it simulates $\mu_{i,j}$ on $\varepsilon \otimes \beta_j$, and in the fourth mode it simulates $\nu_{i,j}$ on $\varepsilon \otimes \alpha_j$. Figure 3 illustrates the simulation.

For all $q \in Q$ and $\binom{a}{b} \in \Sigma_\perp^2$ we set

$$\delta_{\mathcal{B}}((q,1), \tbinom{a}{b}) := \delta_1(q, \binom{b}{b}) \qquad \delta_{\mathcal{B}}((q,2), \tbinom{a}{b}) := \delta_2(q, \tbinom{a}{b})$$
$$\delta_{\mathcal{B}}((q,3), \tbinom{a}{b}) := \delta_3(q, \tbinom{\perp}{b}) \qquad \delta_{\mathcal{B}}((q,4), \tbinom{a}{b}) := \delta_4(q, \tbinom{\perp}{a})$$

where $\delta_i(q, \sigma)$ is the formula $\delta(q, \sigma)$ where each variable (p, c) is replaced by $((p, i), c)$. At the holes of $\alpha_j \otimes \beta_j$ labeled with #, i.e., the points where $\alpha_{j+1} \otimes \beta_{j+1}$ starts, the simulations in modes 1, 2, and 3 split up. For all $q \in Q$ we define

$$\delta_{\mathcal{B}}((q,1), \#) := ((q,1),1) \wedge ((q,2),1)$$
$$\delta_{\mathcal{B}}((q,2), \#) := \delta_{\mathcal{B}}((q,3), \#) := ((q,3),1) \wedge ((q,4),1)$$

Finally, we add a new initial state $q_0^{\mathcal{B}}$ which spawns simulations of $\mathcal{A}$ in mode 1 and 2, i.e., for all $q \in Q$ and $\binom{a}{b} \in \Sigma_\perp^2$ we define

$$\delta_{\mathcal{B}}(q_0^{\mathcal{B}}, \tbinom{a}{b}) := \delta_1(q_0, \binom{b}{b}) \wedge \delta_2(q_0, \tbinom{a}{b}).$$

Finally, we intersect $L(\mathcal{B})$ with the tree-regular language Enc_Σ, which concludes the proof. ◻

If R is given by a D↑TA we can even compute in polynomial time a *nondeterministic* Büchi tree automaton (NBTA) for the representation of infinite cliques. The proof idea is that the runs $\mu_{i,j}$ and $\nu_{i,j}$ in Lemma 5.2 only depend on j.

Proposition 5.4. *Given a D↑TA $\mathcal{A}$ for a tree-regular relation $R \subseteq (\mathcal{T}_\Sigma)^2$, one can construct in polynomial time an NBTA $\mathcal{B}$ over the ranked alphabet $\Omega = \Sigma_\perp^2 \cup \{\#\}$ such that we have:*

- *If $\boldsymbol{t}$ is an infinite clique in R, then $\mathcal{B}$ accepts an encoding of a comb $\boldsymbol{s}$ which is a subsequence of $\boldsymbol{t}$.*
- *If $\mathcal{B}$ accepts $t \in \mathcal{T}_\Omega^\infty$, then t is an encoding of a comb $\boldsymbol{t}$ that is an infinite clique in R.*

We are ready to prove Theorem 2.4. If $\mathcal{A}$ is an ATA, we use the ABTA $\mathcal{B}$ from Proposition 5.3 and transform it into an NBTA in exponential time [27, Theorem 1.2]. If $\mathcal{A}$ is a D↑TA $\mathcal{A}$ we use the NBTA $\mathcal{B}$ from Proposition 5.4. We then construct an NTA C over Σ which accepts $t_1 \in \mathcal{T}_\Sigma$ if and only if the encoding of some comb $(t_i)_{i\geq 1}$ is accepted by $\mathcal{B}$. The rest of the proof is analogous to the proof of Theorem 2.2, see [4] for details.

5.4 Transitive relations

In this section we show that if we assume that R is transitive, then the Ramsey quantifier can be evaluated in polynomial time (Theorem 2.5).

Proposition 5.5. *Given an NTA $\mathcal{A}$ for a transitive tree-regular relation $R \subseteq (\mathcal{T}_\Sigma)^2$, one can construct in polynomial time an NBTA $\mathcal{B}$ over the ranked alphabet Ω such that:*

- *If $\boldsymbol{t}$ is an infinite clique in R, then $\mathcal{B}$ accepts an encoding of a comb $\boldsymbol{s}$ which is a subsequence of $\boldsymbol{t}$.*
- *If $\mathcal{B}$ accepts $t \in \mathcal{T}_\Omega^\infty$, then t is an encoding of a comb $\boldsymbol{t}$ that is an infinite clique in R.*

For the proof we view $\mathcal{A}$ as an ATA and construct the ABTA $\mathcal{B}$ as in the proof of Proposition 5.3 which accepts precisely all comb encodings $\mathrm{enc}(\boldsymbol{\alpha}, \boldsymbol{\beta})$ with the properties from Lemma 5.2. Since R is transitive, ensuring that $\boldsymbol{v}$ is a clique merely requires to check that $(v_i, v_{i+1}) \in R$ for each i. Therefore, the set of runs we need to detect on an encoding $\mathrm{enc}(\boldsymbol{\alpha}, \boldsymbol{\beta})$ are as in Figure 3, but without all the runs $\mu_{i,j}$. In the resulting comb of combs, all rows have finite length. In terms of the constructed ABTA, this means we can omit all states in mode 3. Then any run contains for each node v of the input tree at most three run nodes referring to v. Thus, we can apply a standard powerset construction to convert C into an equivalent NBTA of polynomial size by restricting to subsets of states of size at most three. We refer to [4] for more details. Using the polynomially-sized NBTA $\mathcal{B}$ we can prove Theorem 2.5 analogously to Theorem 2.4.

5.5 Co-transitive relations

Recall that a binary relation $R \subseteq A \times A$ is *co-transitive* if its complement $\bar{R} = (A \times A) \setminus R$ is transitive. Next we show Theorem 2.6.

A context β is called *monadic* if it has exactly one hole. We will show that, if a co-transitive relation has an infinite clique, then there exists one which is a comb generated by a *monadic generator* $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ in which all forests β_i are monadic contexts. This also implies that all α_i are trees.

Lemma 5.6. *If a co-transitive tree-regular relation $R \subseteq \mathcal{T}_\Sigma \times \mathcal{T}_\Sigma$ has an infinite clique over a tree-regular language $L \subseteq \mathcal{T}_\Sigma$, then there exists an infinite clique $\boldsymbol{t}$ of R over L and a nontrivial monadic context β with $t_1 \lessdot \beta$ and $\beta \leq_{\mathrm{p}} t_i$ for all $i \geq 2$.*

Proof. Let $\mathcal{A} = (Q, \Sigma_\perp^2, \Delta, F)$ and $\mathcal{B} = (P, \Sigma, \Lambda, G)$ be D↑TAs for R and L, respectively. Suppose that $\boldsymbol{t}$ is an infinite clique in R over L. For $j \geq 2$ let c_j be the unique context with $\mathrm{nodes}(c_j) = \mathrm{nodes}(t_1) \cap \mathrm{nodes}(t_j)$ and $c_j \leq_{\mathrm{p}} t_j$. Notice that c_j is nontrivial since t_1 and t_j contain the root. Furthermore we have $t_1 \lessdot c_j$ since any hole $u \in \mathrm{holes}(c_j)$ is contained in $\mathrm{nodes}(t_j) \setminus \mathrm{nodes}(c_j)$. Since there are only finitely many such choices for c_j, by reducing $\boldsymbol{t}$ to a subsequence which starts with t_1 we can assume that $c_j = c$ for all $j \geq 2$ for some context c. Suppose that $v^1, \dots, v^n$ are the holes of c in lexicographical order, and $t_j = c[t_j^1, \dots, t_j^n]$ for some trees t_j^k. Again, by reducing $\boldsymbol{t}$ to a subsequence starting with t_1, we can further assume that $(t_j^k)_{j \geq 2}$ is an infinite sequence of pairwise distinct trees for each $k \in [1, n]$. Indeed, if $(t_j^k)_{j\geq 2}$ contains only finitely many distinct trees for some k, then some tree t must occur infinitely often in the sequence $(t_j^k)_{j \geq 2}$, say $t = t_{\ell_2}^k = t_{\ell_3}^k = \cdots$ for some $1 = \ell_1 < \ell_2 < \cdots$. We then extend c by plugging t into the hole v^k and we replace $\boldsymbol{t}$ by $(t_{\ell_j})_{j \geq 1}$. Clearly, duplicates in a sequence $(t_j^k)_{j \geq 2}$ that contains infinitely many distinct elements can also be removed by restricting to a subsequence.

For all $i < j$ we have

$$t_i \otimes t_j = \begin{cases} (t_1 \otimes c)[\varepsilon \otimes t_j^1, \dots, \varepsilon \otimes t_j^n], & 1 = i < j, \\ (c \otimes c)[t_i^1 \otimes t_j^1, \dots, t_i^n \otimes t_j^n], & 1 < i < j, \end{cases}$$

where $t_1 \otimes c$ and $c \otimes c$ are naturally viewed as contexts with n holes. For $j \geq 2$ consider the accepting run ρ_j of $\mathcal{A}$ on $t_1 \otimes t_j$ and the accepting run π_j of $\mathcal{B}$ on t_j, and color each index j by the tuple $(\rho_j(v^1), \dots, \rho_j(v^n), \pi_j(v^1), \dots, \pi_j(v^n))$. By the pigeonhole principle we can pick numbers $1 = \ell_1 < \ell_2 < \dots$ such that $\{\ell_2, \ell_3, \dots\}$ is monochromatic. We then replace $\boldsymbol{t}$ by $(t_{\ell_i})_{i \geq 1}$. Hence, the accepting runs of $\mathcal{A}$ on $t_1 \otimes t_j$ $(j \geq 2)$ visit the same states $r^1, \dots, r^n$ in the nodes $v^1, \dots, v^n$. Similarly, the accepting runs of $\mathcal{B}$ on the trees t_j visit the same states $p^1, \dots, p^n$ in the nodes $v^1, \dots, v^n$. Therefore

$$(t_1, c[t_{j_1}^1, \dots, t_{j_n}^n]) \in R \qquad \text{for any } j_1, \dots, j_n \geq 2 \text{ and}$$
$$c[t_{j_1}^1, \dots, t_{j_n}^n] \in L \qquad \text{for any } j_1, \dots, j_n \geq 2.$$

For $1 < i < j$ consider an accepting run of $\mathcal{A}$ on $t_i \otimes t_j$ and let $q_{i,j}^k$ be the state reached in node v^k. By Ramsey's theorem we can assume that there exist states $q^1, \dots, q^n \in Q$ such that $q_{i,j}^k = q^k$ for all $1 < i < j$ (again, after replacing $\boldsymbol{t}$ by a subsequence starting with t_1). Observe that $\mathcal{A}$ accepts the context $c \otimes c$ if it starts in nodes $v^1, \dots, v^k$ with the states $q^1, \dots, q^k$, respectively.

For every $0 \leq k \leq n$ define the tree

$$s_k = c[t_3^1, \dots, t_3^k, t_2^{k+1}, \dots, t_2^n].$$

We have $(s_0, s_n) = (t_2, t_3) \in R$. There must be an index $1 \leq k \leq n$ with $(s_{k-1}, s_k) \in R$ since otherwise by transitivity of $\bar{R}$ we would have $(s_0, s_n) \notin R$. Define the context

$$\beta = c[t_3^1, \dots, t_3^{k-1}, x, t_2^{k+1}, \dots, t_2^n].$$

Then we have that $(\beta[t_2^k], \beta[t_3^k]) \in R$. This is witnessed by an accepting run on their convolution, which reaches state q^k at node v^k. This implies that $(\beta[t_i^k], \beta[t_j^k]) \in R$ for all $i < j$, since the run of $\mathcal{A}$ on $t_i^k \otimes t_j^k$ also reaches q^k. Hence, the context β together with the trees t_1 and $\beta[t_i^k]$ for $i \geq 2$ satisfy the claim. Moreover, $t_1 \lessdot c$ and $c \leq_{\mathrm{p}} \beta$ implies $t_1 \lessdot \beta$. □

Repeated applications of Lemma 5.6 yields the desired infinite clique:

Lemma 5.7. *If a co-transitive tree-regular relation R has an infinite clique then there exists an infinite clique $\boldsymbol{t}$ of R generated by a monadic generator $(\boldsymbol{\alpha}, \boldsymbol{\beta})$.*

Proof. Let $n \in \mathbb{N}$ and suppose we have inductively constructed trees $\alpha_1, \dots, \alpha_n$, and nontrivial monadic contexts $\beta_1, \dots, \beta_n$, with $\alpha_i \lessdot \beta_i$ for all $1 \leq i \leq n$, such that there exist trees $(t_i')_{i > n}$ such that $(t_i)_{i \geq 1}$ is an infinite clique in R where $t_i = \beta_1 \beta_2 \dots \beta_{i-1} \alpha_i$ if $i \leq n$, and $t_i = \beta_1 \beta_2 \dots \beta_n t_i'$ if $i > n$. Let $\beta := \beta_1 \beta_2 \dots \beta_n$. Then $(t_i')_{i > n}$ is an infinite clique in the relation $R' = \{(s', t') \mid (\beta[s'], \beta[t']) \in R\}$. It is easy to see that R' is again tree-regular and also co-transitive since transitivity is preserved from $\bar{R}$ to $\bar{R}'$ via $\beta[t] \mapsto t$. Furthermore all trees t_i' for $i > n$ belong to the tree-regular language $L = \bigcap_{i=1}^n \{t \mid (t_i, \beta[t]) \in R\}$. We can apply Lemma 5.6 and obtain a tree α_{n+1}, a nontrivial monadic context β_{n+1} with $\alpha_{n+1} \lessdot \beta_{n+1}$, and trees $(t_i'')_{i > n+1}$ such that α_{n+1} together with $\beta_{n+1}[t_i'']$ for $i > n+1$ form an infinite clique in R'. Furthermore all trees $\beta_{n+1}[t_i'']$ for $i > n+1$ belong to L. Hence $t_1, \dots, t_n$ together with $\beta[\alpha_{n+1}]$ and $\beta[\beta_{n+1}[t_i'']]$ for $i > n+1$ form an infinite clique in R. By induction we then obtain the desired sequences $\boldsymbol{\alpha}, \boldsymbol{\beta}$. □

We can now prove Theorem 2.6. Given an NTA $\mathcal{A}$ for a co-transitive relation R. Using Lemmas 5.2 and 5.7, we can prove a statement similar to Lemma 4.3 for tree combs which are generated by a monadic generator. In particular, all context forests $\kappa_i, \lambda_i, \mu_j$ have exactly one hole, and hence $\kappa_i, \lambda_i, \mu_j, \nu_j$ are in fact contexts. Now we can construct in polynomial time a Büchi tree automaton $\mathcal{B}$ which accepts all comb encodings $\mathrm{enc}(\boldsymbol{\alpha}, \boldsymbol{\beta})$ of a monadic generator $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ for which runs of the form $\kappa_j, \lambda_j, \mu_j, \nu_j$ as above exist. To this end, $\mathcal{B}$ consists of four components in which the runs $\kappa_j, \lambda_j, \mu_j, \nu_j$ are simulated. The detailed construction can be found in [4].

The lower bound of Theorem 2.6 follows by a logspace reduction from monadic decomposability for D↑TAs which is P-complete (see Section 6.2).

6 APPLICATIONS

6.1 Recurrent reachability with generalized Büchi condition

The proof of the following Proposition can be found in [4].

Proposition 6.1. *The infinite clique problem and recurrent reachability are logspace equivalent over (tree-)regular relations. Moreover, the logspace reduction from recurrent reachability to the infinite clique problem preserves transitivity of relations and determinism of automata.*

Using Proposition 6.1 we obtain tight complexity bounds for recurrent reachability over (transitive) (tree-)regular relations. We can even compute an automaton for the set $Rec(L)[R]$ of initial elements given automata for R and L.

Corollary 6.2. *If R is a binary (tree-)regular relation and L is a (tree-)regular language given by NFAs (NTAs), then one can construct an NFA (NTA) for $Rec(L)[R]$ in logspace (exponential time). The construction works in polynomial time if R and L are given by D↑TAs or if R is transitive.*

Proof. We can define $Rec(L)[R]$ by the formula

$$\begin{aligned}\varphi(x) = \exists^{\mathrm{ram}} y, z\colon\, & R(x,y) \wedge L(y) \wedge R(y,z)\\ \vee\, \exists y\colon\, & R(x,y) \wedge L(y) \wedge R(y,y).\end{aligned}$$

Here, the first disjunct (beginning with $\exists^{\mathrm{ram}}$) captures infinite paths visiting infinitely many configurations, whereas the second (beginning with $\exists$) captures infinite paths with only finitely many (distinct) configurations.

If R and L are given by NFAs, we can construct in logspace an NFA for $Rec(L)[R]$ using the closure properties of regular relations and Theorem 2.2. Over trees, we use Theorems 2.4 and 2.5 to construct an NTA for $Rec(L)[R]$ in exponential or polynomial time depending on whether R is transitive and how R, L are given. □

For recurrent reachability with generalized Büchi condition we show that over words the complexity increases from NL to PSPACE, while over trees it stays in EXP (Theorem 2.8).

For both the word and the tree case we reduce the generalized version to the classical version. We first observe that $a_0 \in Rec(L_1, \dots, L_k)[R]$ if and only if there is a sequence $\boldsymbol{a}$ such that $(a_i, a_j) \in R$ for all $0 \le i < j$ and $a_i \in L_{((i-1) \bmod k)+1}$ for all $i \ge 1$. We define a (tree-)regular relation $R' \subseteq A^k \times A^k$ that checks if a tuple $(a_1, \dots, a_{2k})$ forms a clique of size $2k+1$ in R starting with a_0 such that $a_i \in L_i$ for all $i \in [1,k]$. In the word case the NFA $\mathcal{A}'$ for R' can be constructed in PSPACE using a product construction. In the tree case we can avoid the exponential blow-up for the product automaton by using ATAs. To make this work, we have to reduce the size of the alphabet for the ATA $\mathcal{A}'$. This can be achieved by encoding a tuple $(\sigma_1, \dots, \sigma_k) \in \Sigma_\perp^k$ of symbols by a path $\sigma_1(\sigma_2(\dots\sigma_k(\#_m)\dots))$ where $\#_m$ is used as delimiter symbol of rank $m := \max\{\mathrm{rk}(\sigma_i) \mid 1 \le i \le k\}$. Then the ATA $\mathcal{A}'$ can be constructed in polynomial time. Now it holds that $a_0 \in Rec(L_1, \dots, L_k)[R]$ if and only if $\exists^{\mathrm{ram}} x, y\colon R'(x,y) \vee \exists x\colon R'(x,x)$ is valid. By Theorem 2.2 (resp. Theorem 2.4) validity of the first disjunct of φ can be checked in nondeterministic logspace (resp. exponential time) given $\mathcal{A}'$. It is easy to see that validity of the second disjunct of φ can also be checked in nondeterministic logspace (resp. exponential time) given $\mathcal{A}'$. This yields a PSPACE-algorithm in the word case and an EXP-algorithm in the tree case. Details are in [4].

For the lower bounds we reduce from the intersection nonemptiness problem of (tree-)regular languages $L_1, \dots, L_k \subseteq A$, which is known to be PSPACE-complete over words [15] and EXP-complete over trees [8, Theorem 11]. We define the (tree-)regular relation $R \subseteq A \times A$ such that $(a,b) \in R$ if and only if $a = c$ or $a = b$ where $c \in A$ is some fixed element. Then $L_1 \cap \cdots \cap L_k \neq \emptyset$ if and only if $c \in Rec(L_1, \dots, L_k)[R]$.

For $k = 1$, the previous construction yields a reduction from nonemptiness for D↑TAs, which is P-complete, to recurrent reachability over transitive tree-regular relations given by D↑TAs, proving the P-hardness in Theorem 2.3 and Corollary 2.7.

The lower bound in the word case of Corollary 2.7 follows by a logspace reduction from monadic decomposability for DFAs which is NL-complete (see Section 6.2).

In [24] Löding shows that the reachability relation $\to^*$ for regular ground tree rewrite systems (RGTRS) is tree-regular and an NTA for $\to^*$ can be constructed in polynomial time. Hence, by Theorem 2.8 recurrent reachability with generalized Büchi condition is EXP-complete for RGTRSs where hardness for GTRSs can be shown by a similar reduction as above from intersection nonemptiness.

Corollary 6.3. *Given an RGTRS and NTAs for tree-regular languages $L_1, \dots, L_k$, one can construct in exponential time an NTA recognizing $Rec(L_1, \dots, L_k)[\to^*]$.*

6.2 Monadic Decomposability

In the following we reduce monadic decomposability to the infinite clique problem over co-transitive relations, proving Corollary 2.9 and Corollary 2.10. A k-ary relation R over words or trees is monadically decomposable if and only if for all $1 \le j \le k$ the equivalence relations $\sim_j$ on $(\Sigma^*)^j$ (or $\mathcal{T}_\Sigma^j$) have finite index, where two tuples $\boldsymbol{u} = (u_1, \dots, u_j)$, $\boldsymbol{v} = (v_1, \dots, v_j)$ are $\sim_j$-equivalent if and only if

$$\forall \boldsymbol{w} = (w_{j+1}, \dots, w_k)\colon [(\boldsymbol{u}, \boldsymbol{w}) \in R \iff (\boldsymbol{v}, \boldsymbol{w}) \in R],$$

see for example [7, Proof of Proposition 3.9]. If the given automaton $\mathcal{A}$ for R is a DFA, D↑TA, or D↓TA, then one can compute automata $\mathcal{A}_{\nsim_j}$ for the complements $\nsim_j$ of $\sim_j$ in logspace, using the fact that $\boldsymbol{u} \nsim_j \boldsymbol{v}$ is equivalent to

$$\exists \boldsymbol{w}\colon \big((\boldsymbol{u}, \boldsymbol{w}) \in R \wedge (\boldsymbol{v}, \boldsymbol{w}) \notin R\big) \vee \big((\boldsymbol{u}, \boldsymbol{w}) \notin R \wedge (\boldsymbol{v}, \boldsymbol{w}) \in R\big).$$

If $\mathcal{A}$ is an NFA or NTA, then this is possible in polynomial space, by determinizing $\mathcal{A}$ and using closure properties of regular relations. Then, apply Theorem 2.2 (Theorem 2.6) to $\mathcal{A}_{\not\sim_j}$ to check in NL (resp. P) for an infinite clique in $\not\sim_j$.

We now prove the lower bounds by a reduction from the universality problem for DFAs, NFAs, D↑TAs, and NTAs, and the emptiness problem for D↓TAs.

Lemma 6.4. *Given a binary regular relation $R \subseteq \Sigma^* \times \Sigma^*$ by an NFA (resp. DFA), it is* PSPACE*-hard (resp.* NL*-hard) to decide whether R is monadically decomposable. Given a binary tree-regular relation $R \subseteq \mathcal{T}_\Sigma \times \mathcal{T}_\Sigma$ by an NTA (resp. D↑TA), it is* EXP*-hard (resp.* P*-hard) to decide whether R is monadically decomposable.*

Proof. We give a logspace reduction from the universality problem which is known to be PSPACE-complete for NFAs, NL-complete for DFAs, P-complete for D↑TAs, and EXP-complete for NTAs. To ease notation, we only consider the word case and remark that the tree case is analogous. Recall that the universality problem asks whether for a given regular language $L \subseteq \Sigma^*$ it holds that $L = \Sigma^*$. Let $L \subseteq \Sigma^*$ be a regular language given by an NFA (resp. DFA) $\mathcal{A}$. We define the regular relation

$$R_L := \{(u \otimes v, w) \mid u \in L \text{ or } v = w \in \Sigma^*\}.$$

It is easy to construct an NFA (resp. DFA) that recognizes R_L in logarithmic space from $\mathcal{A}$. Note that for DFAs the disjunction can be realized with a product construction. It remains to show that R_L is monadically decomposable if and only if $L = \Sigma^*$.

If $L = \Sigma^*$, it holds that $R_L = \{(u \otimes v, w) \mid u, v, w \in \Sigma^*\}$ which is clearly monadically decomposable.

For the converse assume that there exists $u_0 \in \Sigma^* \setminus L$. Then the intersection of R_L with the monadically decomposable relation $\{(u_0 \otimes v, w) \mid v, w \in \Sigma^*\}$ is the relation $\{(u_0 \otimes v, w) \mid v = w \in \Sigma^*\}$, which is not monadically decomposable. Since monadically decomposable relations are closed under intersection, it follows that R_L is not monadically decomposable. □

Lemma 6.5. *Given a binary tree-regular relation $R \subseteq \mathcal{T}_\Sigma \times \mathcal{T}_\Sigma$ by a D↓TA, it is* P*-hard to decide whether R is monadically decomposable.*

Proof. We give a logspace reduction from the emptiness problem for D↓TAs which is known to be P-complete [35]. Let $\mathcal{A} = (Q, \Sigma, \Delta, q_0)$ be a D↓TA. We construct a D↓TA $\mathcal{A}' = (Q, \Gamma_\perp^2, \Delta', q_0)$ recognizing a binary tree-regular relation R' over $\Gamma := \Sigma \cup \{\#\}$ where $\# \notin \Sigma$ is a symbol of rank 1 as follows. We define the transition relation Δ' such that

- $q \xrightarrow{(a,a)}_{\mathcal{A}'} (q_1, \dots, q_r)$ for all $q \xrightarrow{a}_{\mathcal{A}} (q_1, \dots, q_r)$,
- $q_0 \xrightarrow{(\#,\#)}_{\mathcal{A}'} q_0$

Clearly, $\mathcal{A}'$ can be constructed in logspace from $\mathcal{A}$.

It is easy to see that $R' \subseteq \{(t,t) \mid t \in \mathcal{T}_\Gamma\}$. Moreover, it holds that R' is finite if and only if $L(\mathcal{A}) = \emptyset$. Indeed, if there exists $t \in L(\mathcal{A})$, then $(t_n, t_n) \in R'$ for all $n \geq 0$ where t_n is the resulting tree when padding a chain of #-symbols of length n to the root of t. Since every finite relation is monadically decomposable and every infinite subrelation of $\{(t,t) \mid t \in \mathcal{T}_\Gamma\}$ is clearly not monadically decomposable, it holds that R' is monadically decomposable if and only if $L(\mathcal{A}) = \emptyset$. □

7 UNRANKED TREE-AUTOMATIC STRUCTURES

In this section we consider the unranked tree analogue of Theorems 2.4 and 2.5. Furthermore, we consider an application of the results to recurrent reachability in subtree and flat prefix rewriting systems. *Unranked tree-regular* languages and relations are recognized by nondeterministic unranked tree automata (NUTAs), see e.g. [8].

Theorem 7.1. *Given an unranked tree-regular $R \subseteq (\mathcal{U}_\Sigma)^{k+2}$ by an NUTA $\mathcal{A}$, an NUTA for the relation $[\![\exists^{\text{ram}} x, y \colon R(x, y, z)]\!]$ can be constructed in polynomial time if R is transitive and in exponential time otherwise. Hence, the infinite clique problem over (transitive) unranked tree-regular relations is in* EXP *(*P*).*

The proof can be found in [4]. It uses the *first-child next-sibling encoding*, a standard regularity-preserving transformation from unranked trees to binary trees (see e.g., [11, 21, 28]). As over ranked trees, Theorem 7.1 implies:

Corollary 7.2. *For a binary unranked tree-regular relation R and an unranked tree-regular language L given by NUTAs, one can construct an NUTA recognizing $Rec(L)[R]$ in polynomial time if R is transitive and in exponential time otherwise.*

In [25] Löding and Spelten introduce tree rewriting systems over unranked trees called subtree and flat prefix rewriting systems (SFPRS). In [25] it is shown that the reachability relation $\to^*$ for (regular) SFPRSs is an unranked tree-regular relation. Moreover, it can be observed that the NUTA for $\to^*$ can be constructed in polynomial time. Since $\to^*$ is transitive, we can apply Corollary 7.2 to obtain that recurrent reachability for (regular) SFPRSs is decidable in polynomial time:

Corollary 7.3. *For a (regular) SFPRS and an unranked tree-regular language L given as NUTA, one can construct an NUTA recognizing $Rec(L)[\to^*]$ in polynomial time.*

Let FO(SFPRS) be the first-order theory over unranked trees with the reachability relation $\to^*$ and the one-step reachability relation $\to$ for (regular) SFPRSs. In [25] it is shown that the structure of FO(SFPRS) is unranked tree-automatic which means that FO(SFPRS) is decidable.

Let FO(SFPRS + Rec) be the theory FO(SFPRS) enriched by the recurrent reachability operator: For a formula φ in FO(SFPRS + Rec) with one free variable we define the recurrent reachability operator $Rec(\varphi)$ as formula with one free variable such that $Rec(\varphi)(t)$ is true if and only if $t \in Rec(L)[\to^*]$ for any $t \in \mathcal{U}_\Sigma$ where L is the unranked tree-regular language defined by φ.

Corollary 7.4. *The theory* FO(SFPRS + Rec) *is decidable.*

For future work, we propose to investigate if Corollary 7.2 could be applied to other classes of tree rewriting systems over unranked trees, e.g., the class Trs_0 of tree rewrite systems with positive guards [13], which was applied for the analysis of HTML5 applications.

8 CONCLUSION AND FUTURE WORKS

We have identified directed Ramsey quantifiers as a fundamental notion that underlies the standard notion of Ramsey quantifiers, recurrent reachability, and monadic decomposability. We have also shown that the notion of *comb of combs* can be used to obtain substantially simpler proofs in case of word-automatic relations, and can be generalized to tree-automatic relations, allowing us to derive new results for Ramsey quantifiers, recurrent reachability and monadic decomposability (with applications to generalized Büchi conditions and unranked tree-automatic relations). There are many natural research directions. In particular, we pinpoint that Ramsey quantifiers over ω-automatic relations, as well as recurrent reachability over transitive ω-automatic relations, is still a major open problem [17], although monadic decomposability is known to be decidable [26]. One possible approach is to consider the subclass of ω-automatic relations that are definable over the theory of mixed integer-real linear arithmetic $\langle \mathbb{R}; \mathbb{Z}, 1, 0, <, + \rangle$, for which the problem of Ramsey quantifiers and recurrent reachability, to be the best our knowledge, is still an open problem.

ACKNOWLEDGMENTS

Pascal Bergsträßer and Anthony Lin are supported by the ERC Starting Grant 759969 (AV-SMP).

REFERENCES

[1] Parosh Aziz Abdulla, Bengt Jonsson, Pritha Mahata, and Julien d’Orso. 2002. Regular Tree Model Checking. In *Computer Aided Verification, 14th International Conference, CAV 2002,Copenhagen, Denmark, July 27-31, 2002, Proceedings (Lecture Notes in Computer Science, Vol. 2404)*, Ed Brinksma and Kim Guldstrand Larsen (Eds.). Springer, 555–568. https://doi.org/10.1007/3-540-45657-0_47

[2] Parosh Aziz Abdulla, Bengt Jonsson, Marcus Nilsson, and Mayank Saksena. 2004. A Survey of Regular Model Checking. In *CONCUR 2004 - Concurrency Theory, 15th International Conference, London, UK, August 31 - September 3, 2004, Proceedings (Lecture Notes in Computer Science, Vol. 3170)*, Philippa Gardner and Nobuko Yoshida (Eds.). Springer, 35–48. https://doi.org/10.1007/978-3-540-28644-8_3

[3] Pablo Barceló, Chih-Duo Hong, Xuan Bach Le, Anthony W. Lin, and Reino Niskanen. 2019. Monadic Decomposability of Regular Relations. In *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece (LIPIcs, Vol. 132)*, Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi (Eds.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 103:1–103:14. https://doi.org/10.4230/LIPIcs.ICALP.2019.103

[4] Pascal Bergsträßer, Moses Ganardi, Anthony W. Lin, and Georg Zetzsche. 2022. Ramsey Quantifiers over Automatic Structures: Complexity and Applications to Verification. *CoRR* abs/2205.09015 (2022). https://doi.org/10.48550/arXiv.2205.09015 arXiv:2205.09015

[5] Achim Blumensath and Erich Grädel. 2000. Automatic Structures. In *15th Annual IEEE Symposium on Logic in Computer Science, Santa Barbara, California, USA, June 26-29, 2000.* IEEE Computer Society, 51–62. https://doi.org/10.1109/LICS.2000.855755

[6] Achim Blumensath and Erich Grädel. 2004. Finite Presentations of Infinite Structures: Automata and Interpretations. *Theory Comput. Syst.* 37, 6 (2004), 641–674. https://doi.org/10.1007/s00224-004-1133-y

[7] Olivier Carton, Christian Choffrut, and Serge Grigorieff. 2006. Decision problems among the main subfamilies of rational relations. *RAIRO Theor. Informatics Appl.* 40, 2 (2006), 255–275. https://doi.org/10.1051/ita:2006005

[8] Hubert Comon, Max Dauchet, Rémi Gilleron, Florent Jacquemard, Denis Lugiez, Christof Löding, Sophie Tison, and Marc Tommasi. 1997. Tree Automata Techniques and Applications. (1997).

[9] Thom W. Frühwirth, Ehud Shapiro, Moshe Y. Vardi, and Eyal Yardeni. 1991. Logic Programs as Types for Logic Programs. In *Proceedings of the Sixth Annual Symposium on Logic in Computer Science (LICS ’91), Amsterdam, The Netherlands, July 15-18, 1991.* IEEE Computer Society, 300–309. https://doi.org/10.1109/LICS.1991.151654

[10] Seymour Ginsburg and Edwin H Spanier. 1966. Bounded regular sets. *Proc. Amer. Math. Soc.* 17, 5 (1966), 1043–1049. https://doi.org/10.1090/S0002-9939-1966-0201310-3

[11] Georg Gottlob, Christoph Koch, Reinhard Pichler, and Luc Segoufin. 2005. The complexity of XPath query evaluation and XML typing. *J. ACM* 52, 2 (2005), 284–335. https://doi.org/10.1145/1059513.1059520

[12] Erich Grädel. 2020. Automatic Structures: Twenty Years Later. In *LICS ’20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science, Saarbrücken, Germany, July 8-11, 2020*, Holger Hermanns, Lijun Zhang, Naoki Kobayashi, and Dale Miller (Eds.). ACM, 21–34. https://doi.org/10.1145/3373718.3394734

[13] Matthew Hague, Anthony Widjaja Lin, and C.-H. Luke Ong. 2015. Detecting redundant CSS rules in HTML5 applications: a tree rewriting approach. In *Proceedings of the 2015 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications, OOPSLA 2015, part of SPLASH 2015, Pittsburgh, PA, USA, October 25-30, 2015.* 1–19. https://doi.org/10.1145/2814270.2814288

[14] Yair Hayut. 2017. Magidor-Malitz reflection. *Arch. Math. Log.* 56, 3-4 (2017), 253–272. https://doi.org/10.1007/s00153-017-0522-2

[15] Dexter Kozen. 1977. Lower Bounds for Natural Proof Systems. In *18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA, 31 October - 1 November 1977.* IEEE Computer Society, 254–266. https://doi.org/10.1109/SFCS.1977.16

[16] Dexter Kozen. 1997. *Automata and computability.* Springer.

[17] Dietrich Kuske. 2010. Is Ramsey’s Theorem omega-automatic?. In *27th International Symposium on Theoretical Aspects of Computer Science, STACS 2010, March 4-6, 2010, Nancy, France (LIPIcs, Vol. 5)*, Jean-Yves Marion and Thomas Schwentick (Eds.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 537–548. https://doi.org/10.4230/LIPIcs.STACS.2010.2483

[18] Dietrich Kuske and Markus Lohrey. 2010. Some natural decision problems in automatic graphs. *J. Symb. Log.* 75, 2 (2010), 678–710. https://doi.org/10.2178/jsl/1268917499

[19] Jérôme Leroux and Grégoire Sutre. 2006. Flat counter automata almost everywhere!. In *Software Verification: Infinite-State Model Checking and Static Program Analysis, 19.02. - 24.02.2006 (Dagstuhl Seminar Proceedings, Vol. 06081)*, Parosh Aziz Abdulla, Ahmed Bouajjani, and Markus Müller-Olm (Eds.). Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany. http://drops.dagstuhl.de/opus/volltexte/2006/729

[20] Leonid Libkin. 2003. Variable independence for first-order definable constraints. *ACM Trans. Comput. Log.* 4, 4 (2003), 431–451. https://doi.org/10.1145/937555.937557

[21] Leonid Libkin. 2005. Logics for Unranked Trees: An Overview. In *Automata, Languages and Programming, 32nd International Colloquium, ICALP 2005, Lisbon, Portugal, July 11-15, 2005, Proceedings (Lecture Notes in Computer Science, Vol. 3580)*, Luís Caires, Giuseppe F. Italiano, Luís Monteiro, Catuscia Palamidessi, and Moti Yung (Eds.). Springer, 35–50. https://doi.org/10.1007/11523468_4

[22] Anthony Widjaja Lin. 2012. Accelerating tree-automatic relations. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2012, December 15-17, 2012, Hyderabad, India.* 313–324. https://doi.org/10.4230/LIPIcs.FSTTCS.2012.313

[23] Anthony W. Lin and Philipp Rümmer. 2021. Regular Model Checking Revisited. In *Model Checking, Synthesis, and Learning - Essays Dedicated to Bengt Jonsson on The Occasion of His 60th Birthday.* 97–114. https://doi.org/10.1007/978-3-030-91384-7_6

[24] Christof Löding. 2006. Reachability Problems on Regular Ground Tree Rewriting Graphs. *Theory Comput. Syst.* 39, 2 (2006), 347–383. https://doi.org/10.1007/s00224-004-1170-6

[25] Christof Löding and Alex Spelten. 2007. Transition Graphs of Rewriting Systems over Unranked Trees. In *Mathematical Foundations of Computer Science 2007, 32nd International Symposium, MFCS 2007, Ceský Krumlov, Czech Republic, August 26-31, 2007, Proceedings (Lecture Notes in Computer Science, Vol. 4708)*, Ludek Kucera and Antonín Kucera (Eds.). Springer, 67–77. https://doi.org/10.1007/978-3-540-74456-6_8

[26] Christof Löding and Christopher Spinrath. 2019. Decision Problems for Subclasses of Rational Relations over Finite and Infinite Words. *Discret. Math. Theor. Comput. Sci.* 21, 3 (2019). http://dmtcs.episciences.org/5141

[27] David E. Muller and Paul E. Schupp. 1995. Simulating Alternating Tree Automata by Nondeterministic Automata: New Results and New Proofs of the Theorems of Rabin, McNaughton and Safra. *Theor. Comput. Sci.* 141, 1&2 (1995), 69–107. https://doi.org/10.1016/0304-3975(94)00214-4

[28] Frank Neven. 2002. Automata, Logic, and XML. In *Computer Science Logic, 16th International Workshop, CSL 2002, 11th Annual Conference of the EACSL, Edinburgh, Scotland, UK, September 22-25, 2002, Proceedings (Lecture Notes in Computer Science, Vol. 2471)*, Julian C. Bradfield (Ed.). Springer, 2–26. https://doi.org/10.1007/3-540-45793-3_2

[29] Sasha Rubin. 2008. Automata Presenting Structures: A Survey of the Finite String Case. *Bull. Symb. Log.* 14, 2 (2008), 169–209. https://doi.org/10.2178/bsl/1208442827

[30] Helmut Seidl. 1994. Haskell Overloading is DEXPTIME-Complete. *Inf. Process. Lett.* 52, 2 (1994), 57–60. https://doi.org/10.1016/0020-0190(94)00130-8

[31] Richard Edwin Stearns. 1967. A Regularity Test for Pushdown Machines. *Inf. Control.* 11, 3 (1967), 323–340. https://doi.org/10.1016/S0019-9958(67)90591-8

[32] Anthony Widjaja To and Leonid Libkin. 2008. Recurrent Reachability Analysis in Regular Model Checking. In *Logic for Programming, Artificial Intelligence, and Reasoning, 15th International Conference, LPAR 2008, Doha, Qatar, November 22-27, 2008. Proceedings.* 198–213. https://doi.org/10.1007/978-3-540-89439-1_15

[33] Anthony Widjaja To and Leonid Libkin. 2010. Algorithmic Metatheorems for Decidable LTL Model Checking over Infinite Systems. In *Foundations of Software*

Science and Computational Structures, 13th International Conference, FOSSACS 2010, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2010, Paphos, Cyprus, March 20-28, 2010. Proceedings. 221–236. https://doi.org/10.1007/978-3-642-12032-9_16

[34] Leslie G. Valiant. 1975. Regularity and Related Problems for Deterministic Pushdown Automata. *J. ACM* 22, 1 (1975), 1–10. https://doi.org/10.1145/321864.321865

[35] Margus Veanes. 1997. *On computational complexity of basic decision problems of finite tree automata.* Technical Report. UPMAIL Technical Report 133, Uppsala University, Computing Science Department.

[36] Margus Veanes, Nikolaj Bjørner, Lev Nachmanson, and Sergey Bereg. 2017. Monadic Decomposition. *J. ACM* 64, 2 (2017), 14:1–14:28. https://doi.org/10.1145/3040488