Universal Quantification Makes Automatic Structures Hard to Decide

Christoph Haase
Department of Computer Science, University of Oxford, UK

Radosław Piórkowski
Department of Computer Science, University of Oxford, UK

Abstract

Automatic structures are structures whose universe and relations can be represented as regular languages. It follows from the standard closure properties of regular languages that the first-order theory of an automatic structure is decidable. While existential quantifiers can be eliminated in linear time by application of a homomorphism, universal quantifiers are commonly eliminated via the identity $\forall x \,.\, \Phi \equiv \neg(\exists x \,.\, \neg\Phi)$. If Φ is represented in the standard way as an NFA, a priori this approach results in a doubly exponential blow-up. However, the recent literature has shown that there are classes of automatic structures for which universal quantifiers can be eliminated by different means without this blow-up by treating them as first-class citizens and not resorting to double complementation. While existing lower bounds for some classes of automatic structures show that a singly exponential blow-up is unavoidable when eliminating a universal quantifier, it is not known whether there may be better approaches that avoid the naïve doubly exponential blow-up, perhaps at least in restricted settings.

In this paper, we answer this question negatively and show that there is a family of NFA representing automatic relations for which the minimal NFA recognising the language after eliminating a single universal quantifier is doubly exponential, and deciding whether this language is empty is ExpSpace-complete.

2012 ACM Subject Classification Theory of computation → Logic

Keywords and phrases automatic structures, universal projection, state complexity, tiling problems

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2023.13

Funding This work is part of a project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 852769, ARiAT).

Acknowledgements We would like to thank the anonymous reviewers for their comments and suggestions, which helped us to improve the presentation of this paper. We are grateful to Alessio Mansutti for spotting a bug in Section 4.

1 Introduction

Quantifier elimination is a standard technique to decide logical theories. A logical theory $\mathcal{T}$ admits quantifier elimination whenever for every quantifier free conjunction of literals $\Phi(x, y_1, \ldots, y_n)$ of $\mathcal{T}$ there is a quantifier free formula $\Psi(y_1, \ldots, y_n)$ such that $\mathcal{T} \models \exists x. \Phi \leftrightarrow \Psi$. Universal quantifiers can then be eliminated simply by applying the duality $\forall x \,.\, \Phi \equiv \neg(\exists x \,.\, \neg\Phi)$. If the formula Ψ above is effectively computable then $\mathcal{T}$ is decidable. For quantifier elimination procedures, the computationally most expensive step is the elimination of an existential quantifier, since negating a formula can be performed on a syntactic level.

Automatic structures [11, 12, 2] are a family of first-order structures whose corresponding first-order theory can be decided using automata-theoretic methods, as an alternative approach to syntactic quantifier elimination. In their simplest variant, automatic structures are relational first-order structures whose universe is isomorphic to a regular language

$L \subseteq \Sigma^*$ over some alphabet Σ, and whose n-ary relations are interpreted as regular languages over $(\Sigma^n)^*$. It follows that the set of all satisfying assignments of a quantifier-free formula $\Phi(x_1, \ldots, x_{m+1})$ can be obtained as the language $\mathcal{L}(\mathcal{A}) \subseteq (\Sigma^{m+1})^*$ of some finite-state automaton $\mathcal{A}$. In this setting, eliminating existential quantifiers is easy. In order to obtain a finite-state automaton whose language encodes the satisfying assignments to $\exists x_{m+1} . \Phi$, it suffices to apply the homomorphism induced by the mapping $h \colon (\Sigma^{m+1}) \to (\Sigma^m)$ such that $h(u_1, \ldots, u_{m+1}) \coloneqq (u_1, \ldots, u_m)$ to $\mathcal{L}(\mathcal{A})$. This can be performed in linear time, even when $\mathcal{A}$ is non-deterministic. However, if $\mathcal{A}$ is non-deterministic then computing a finite-state automaton whose language encodes the complement of Φ is computationally difficult and may lead to an automaton with $2^{\Omega(|\mathcal{A}|)}$ many states. In particular, due to double complementation, eliminating a universal quantifier may *a priori* lead to an automaton with $2^{2^{\Omega(|\mathcal{A})|}}$ many states. Notable examples of automatic structures are Presburger arithmetic [17], the first-order theory of the structure $\langle \mathbb{N}, 0, 1, +, = \rangle$, and its extension Büchi arithmetic [6, 4, 5]. Tool suites such as Lash [1], Tapas [15] and Walnut [16] are based on the automata-theoretic approach and have successfully been used to decide challenging instances of Presburger arithmetic and Büchi arithmetic from various application domains. Those tools eliminate universal quantifiers via double complementation.

Yet another approach to deciding Presburger arithmetic is based on manipulating semi-linear sets [10, 8], which are generalisations of ultimately periodic sets to arbitrary tuples of integers in $\mathbb{N}^d$. They are similar to automata-based methods in terms of the computational difficulty of existential projection and complementation: the former is easy whereas the latter is difficult.

For certain classes of automatic structures, it is possible to avoid eliminating universal quantifiers via existential projection and negation. For example, it was shown in [7] that deciding sentences of quantified integer programming $\exists \bar{x}_1 \, \forall \bar{x}_2 \ldots \exists \bar{x}_n \, . \, A \cdot \bar{x} \geq \bar{b}$ is complete for the n-th level of the polynomial hierarchy. The upper bound was obtained by manipulating so-called hybrid linear sets, which characterise the sets of integer solutions of systems of linear equations $A \cdot \bar{x} \geq \bar{b}$. A key technique introduced in [7] is called *universal projection* and enables directly eliminating universal quantifiers instead of resorting to double complementation and existential projection. Given $S \subseteq \mathbb{N}^{d+k}$, the universal projection of S onto the first d coordinates is defined as

$$\pi_d^{\forall}(S) \coloneqq \left\{ \bar{u} \in \mathbb{N}^d \;\middle|\; (\bar{u}, \bar{v}) \in S \text{ for all } \bar{v} \in \mathbb{N}^k \right\}.$$

It is shown in [7] that if S is a hybrid linear set then $\pi_d^{\forall}(S)$ is a hybrid linear set that can be obtained as a finite intersection of existential projections of certain hybrid linear sets. Moreover, the growth of the constants in the description of the hybrid linear set is only polynomial. Neither syntactic quantifier elimination nor automata-based methods are powerful enough to derive those tight upper bounds for quantified integer programming.

Another example is a recent paper of Boigelot et al. [3] showing that, in an automata-theoretic approach for a fragment of Presburger arithmetic with uninterpreted predicates, a universal projection step can directly be carried out on the automata level without complementation and only results in a singly exponential blowup.

Those positive algorithmic and structural results are specific to Presburger arithmetic and leave open the option that it may be possible to establish analogous results for general automatic structures. The starting point of this paper is the question of whether, given a non-deterministic finite automaton $\mathcal{A}$ whose language $\mathcal{L}(\mathcal{A}) \subseteq (\Sigma^{d+k})^*$ encodes the set of solutions of some quantifier-free formula Φ, there is a more efficient way to eliminate a (block of) universally quantified variable(s) than to first complement $\mathcal{A}$, next to perform an existential projection step, and finally to complement the resulting automaton again, especially in the light of the results of [7, 8]. Such a method would have direct consequences for tools such as WALNUT which perform the aforementioned sequence of operations in order to eliminate universal quantifiers. In particular, WALNUT is not restricted to automata resulting from formulas of linear arithmetic and allows users to directly specify a finite-state automaton when desired.

For better or worse, however, as the main result of this paper, we show that deciding whether the universal projection $\pi_d^{\forall}(\mathcal{L}(\mathcal{A}))$ of some language regular language $\mathcal{L}(\mathcal{A}) \subseteq \left(\Sigma^{d+k}\right)^*$ is empty is complete for EXPSPACE. In particular, the lower bound already holds for $d = k = 1$, meaning that, in general, even for fixed-variable fragments of automatic structures, there is no algorithmically more efficient way to eliminate a single universal quantifier than the naïve one. The challenging part is to show the EXPSPACE lower bound, which requires an involved reduction from a tiling problem. This reduction also enables us to show that there is a family $\left(\mathcal{A}_n\right)_{n\in\mathbb{N}}$ of non-deterministic finite automata such that $|\mathcal{A}_n| = O\left(n^3\right)$ and the smallest non-deterministic finite automaton recognising the universal projection of $\mathcal{L}(\mathcal{A}_n)$ has $\Omega\left(2^{2^n}\right)$ many states.

2 Preliminaries

2.1 Regular languages and their compositions

For a word $w = a_1a_2\cdots a_n \in \Sigma^*$, we write $w[i]$ to denote its i-th letter a_i, and $w[i,j]$ to denote the infix $a_ia_{i+1}\cdots a_j$ $(i \leq j)$. We write $|w|$ for the length of w. A *proper suffix* of w is any infix $w[i,n]$ for some $1 < i \leq n$.

Regular expressions. A *regular expression* over the alphabet Σ is a term featuring Kleene star, concatenation and union operations, as well as $\emptyset$ and all symbols from Σ as constants:

$$\mathcal{E}, \mathcal{E}' ::\equiv \mathcal{E}^* \mid \mathcal{E}\cdot\mathcal{E}' \mid \mathcal{E} + \mathcal{E}' \mid \emptyset \mid a \text{ for every } a \in \Sigma$$

For notational convenience, we also use sets of symbols $A \subseteq \Sigma$ as constants, and a k-fold concatenation $\mathcal{E}^k$ for every $k \in \mathbb{N}$; we also drop the concatenation dot most of the time. The language $\mathcal{L}(\mathcal{E}) \subseteq \Sigma^*$ is defined by structural induction, by interpreting constants as $\mathcal{L}(\emptyset) := \emptyset$ and $\mathcal{L}(a) := \{a\}$, and using the standard semantics of the three operations. The class of languages definable by regular expressions is called *regular languages*. The size $|\mathcal{E}|$ of a regular expression $\mathcal{E}$ is defined recursively as 1 plus the sizes of its subexpressions. For $\rho\colon \Sigma \to \Gamma$ and a regular expression $\mathcal{E}$, $\rho(\mathcal{E})$ is a regular expression over Γ obtained through substituting every constant $a \in \Sigma$ appearing in $\mathcal{E}$ by $\rho(a)$.

Finite-state automata. Regular languages can also be represented by *non-deterministic finite-state automata* (NFA). Such an automaton is a tuple $\mathcal{A} = (Q, \Sigma, \delta, Q_{\mathrm{I}}, Q_{\mathrm{F}})$, where Q is a finite non-empty set of *states*, Σ is a finite *alphabet*, $\delta \subseteq Q \times \Sigma \times Q$ is the *transition relation*, $Q_{\mathrm{I}} \subseteq Q$ is the set of *initial states*, and $Q_{\mathrm{F}} \subseteq Q$ is the set of *final states*. A triple $(p, a, q) \in Q \times \Sigma \times Q$ is called a *transition* and denoted as $p \xrightarrow{a} q$. A *run* of $\mathcal{A}$ from a state q_0 to a state q_n $(n \in \mathbb{N})$ on a word $w = a_1a_2\cdots a_n \in \Sigma^*$ is a finite sequence of transitions $\left(q_{i-1} \xrightarrow{a_i} q_i\right)_{1\leq i\leq n}$ such that $q_{i-1} \xrightarrow{a_i} q_i \in \delta$ for every i. A word $w \in \Sigma^*$ is *accepted* by $\mathcal{A}$ if there exists a run of $\mathcal{A}$ from some $q_{\mathrm{I}} \in Q_{\mathrm{I}}$ to $q_{\mathrm{F}} \in Q_{\mathrm{F}}$ over w. The *language* of $\mathcal{A}$ is defined as $\mathcal{L}(\mathcal{A}) := \{w \in \Sigma^* \mid w \text{ is accepted by } \mathcal{A}\}$. We define the size of $\mathcal{A}$ as $|\mathcal{A}| := |Q| + |Q|^2 \cdot |\Sigma|$. This definition only depends on Q and Σ and ensures that $|\mathcal{A}| \geq |Q| + |\delta| \cdot |\Sigma|$. Subsequently, we will implicitly apply the well-known fact that the number of states of an NFA accepting the complement of $\mathcal{L}(\mathcal{A})$ is bounded by $2^{|Q|}$.

Below we state, without proofs, a few folklore properties of NFA:

▶ **Fact 1** (NFA closed under language union). *For any* NFA $\mathcal{A}, \mathcal{B}$ *over* Γ*, there exists an* NFA $\mathcal{A} \oplus \mathcal{B}$ *of size* $O(|\mathcal{A}| + |\mathcal{B}|)$ *such that* $\mathcal{L}(\mathcal{A} \oplus \mathcal{B}) = \mathcal{L}(\mathcal{A}) \cup \mathcal{L}(\mathcal{B})$*.*

▶ **Fact 2** (NFA closed under inverse language homomorphisms). *For any* NFA $\mathcal{A}$ *and a homomorphic mapping* $\rho\colon \Sigma^* \to \Gamma^*$*, there exists an* NFA $\rho^{-1}(\mathcal{A})$ *of size* $O(|\mathcal{A}|)$ *such that* $\mathcal{L}\big(\rho^{-1}(\mathcal{A})\big) = \rho^{-1}(\mathcal{L}(\mathcal{A}))$*.*

▶ **Fact 3** (NFA closed under concatenation of languages). *For any* NFA $\mathcal{A}, \mathcal{B}$ *there exists an* NFA $\mathcal{A} \odot \mathcal{B}$ *of size* $O(|\mathcal{A}| + |\mathcal{B}|)$ *s.t.* $\mathcal{L}(\mathcal{A} \odot \mathcal{B}) = \mathcal{L}(\mathcal{A}) \cdot \mathcal{L}(\mathcal{B}) := \{u \cdot v \mid u \in \mathcal{L}(\mathcal{A}) \text{ and } v \in \mathcal{L}(\mathcal{B})\}$*.*

▶ **Fact 4** (translating regular expressions into NFA). *For any regular expression* $\mathcal{E}$*, there exists an* NFA $\mathcal{A}(\mathcal{E})$ *such that* $|\mathcal{A}(\mathcal{E})| = O(|\mathcal{E}|)$ *and* $\mathcal{L}(\mathcal{A}(\mathcal{E})) = \mathcal{L}(\mathcal{E})$ *(see [19]).*

Filters. A *filter* is an auxiliary term introduced to simplify the proofs in Section 3, allowing for a modular design of regular languages. Fix a finite alphabet Σ and let $\Phi := \{\top, \bot\}$. Define homomorphisms $\psi_{\text{in}}, \psi_{\text{out}}\colon (\Sigma \times \Phi)^* \to \Sigma^*$ by their actions on a single letter

$$\psi_{\text{in}}(a, b) := a \qquad\qquad \psi_{\text{out}}(a, \top) := a \qquad \psi_{\text{out}}(a, \bot) := \varepsilon\,.$$

(output every symbol from Σ) (output only symbols paired with $\top$)

A filter over an alphabet Σ is any language $F \subseteq (\Sigma \times \Phi)^*$. It induces a binary *input-output relation* $\mathcal{R}(F) \subseteq \Sigma^* \times \Sigma^*$ between input words u and their subsequences v:

$$(u, v) \in \mathcal{R}(F) \quad \overset{\text{def}}{\iff} \quad u = \psi_{\text{in}}(w) \text{ and } v = \psi_{\text{out}}(w) \text{ for some } w \in F\,.$$

We define $F(u) := \{v \mid (u, v) \in \mathcal{R}(F)\}$ to be the set of all possible outputs of F on u.

Filtering regular expressions. A *filtering regular expression* $\mathcal{F}$ over alphabet Σ is any regular expression over $\Sigma \times \Phi$. We write $\mathcal{F}(w) := \mathcal{L}(\mathcal{F})(w)$. To simplify the notation, we only write the Σ component of the constants, and underline parts of the expression. A symbol a appearing in an underlined fragment represents a pair $(a, \top)$, and in a fragment which is not underlined a pair $(a, \bot)$. Intuitively, underlined portions correspond to parts of the words being output. We apply the same notational convention to words $w \in (\Sigma \times \Phi)^*$. Additionally, for $\rho\colon \Sigma \to \Gamma$, we abuse the notation and extend it to the naturally defined homomorphism of type $\Sigma \times \Phi \to \Gamma \times \Phi$, which just preserves the coordinate belonging to Φ.

▶ **Example 5.** Fix $A = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \ldots, \mathtt{z}\}$. Consider a filtering regular expression $\mathcal{F}$ and a word w, both over $A \cup \{\llcorner\!\lrcorner\}$:

$$\mathcal{F} := (\underline{A}\, A^* \llcorner\!\lrcorner)^* \underline{A}\, A^* \qquad\qquad w := \mathtt{nondeterministic{\llcorner\!\lrcorner}finite{\llcorner\!\lrcorner}automaton}\,.$$

We have:

$$\begin{aligned}
&\mathcal{F}(w) = \{\mathtt{nfa}\}\,,\\
&\quad \mathcal{F} = \Big((A \times \{\top\}) \cdot (A \times \{\bot\})^* \cdot (\llcorner\!\lrcorner, \bot)\Big)^* \cdot (A \times \{\top\}) \cdot (A \times \{\bot\})^*\,,\\
&\mathcal{L}(\mathcal{F}) \ni \underline{\mathtt{n}}\mathtt{ondeterministic{\llcorner\!\lrcorner}}\underline{\mathtt{f}}\mathtt{inite{\llcorner\!\lrcorner}}\underline{\mathtt{a}}\mathtt{utomaton}\,.
\end{aligned}$$

▶ **Fact 6.** *For every filtering regular expression* $\mathcal{F}$ *and* w*,* $\mathcal{F}(w) = \mathcal{L}(\mathcal{A}(\mathcal{F}))(w)$*.*

2.2 Automatic relations

Let Σ be a finite alphabet such that $\# \notin \Sigma$. We denote by $\Sigma_\# := \Sigma \cup \{\#\}$. Let $w_1, \ldots, w_k \in \Sigma^*$ such that $w_i = a_{i,1}a_{i,2}\cdots a_{i,\ell_i}$, and $\ell := \max\{\ell_1, \ldots, \ell_k\}$. For all $1 \le i \le k$ and $\ell_i < j \le \ell$, set $a_{i,j} := \#$. The *convolution* $w_1 \otimes w_2 \otimes \cdots \otimes w_k$ of $w_1, \ldots, w_k$ is defined as

$$w_1 \otimes w_2 \otimes \cdots \otimes w_k := \begin{bmatrix} a_{1,1} \\ \vdots \\ a_{k,1} \end{bmatrix} \begin{bmatrix} a_{1,2} \\ \vdots \\ a_{k,2} \end{bmatrix} \cdots \begin{bmatrix} a_{1,\ell} \\ \vdots \\ a_{k,\ell} \end{bmatrix} \subseteq \left(\Sigma_\#^k\right)^* .$$

For $R \subseteq (\Sigma^*)^k$ and $L \subseteq \left(\Sigma_\#^k\right)^*$ define

$$\begin{aligned} &\mathit{Rel2Lang}(R) := \{w_1 \otimes w_2 \otimes \cdots \otimes w_k \mid (w_1, w_2, \ldots, w_k) \in R\}, \\ &\mathit{Lang2Rel}(L) := \{(w_1, w_2, \ldots, w_k) \mid w_1 \otimes w_2 \otimes \cdots \otimes w_k \in L\}. \end{aligned}$$

In this paper, we say that a relation $R \subseteq (\Sigma^*)^k$ is *automatic* whenever $\mathit{Rel2Lang}(R)$ is regular. In the sequel, we assume that $\mathit{Rel2Lang}(R)$ is given by some NFA $\mathcal{A}_R = (Q, \Sigma_\#^k, \delta, Q_{\mathrm{I}}, Q_{\mathrm{F}})$.

Clearly, not every NFA $\mathcal{A} = (Q, \Sigma_\#^k, \delta, Q_{\mathrm{I}}, Q_{\mathrm{F}})$ is associated with an automatic relation $R \subseteq \Sigma^k$ since there are *a priori* no restrictions on the occurrences of the padding symbol "#". The language $L_{\text{✗}} \subseteq (\Sigma_\#^k)^*$ of all incorrect words that cannot be obtained as a convolution of words $w_1, \ldots, w_k \in \Sigma^*$ can be characterized by the following regular expression:

$$\left(\Sigma_\#^k\right)^* \cdot \left(\{\#\}^k + \sum_{1 \le i \le k} \left(\left(\Sigma_\#^{i-1} \times \{\#\} \times \Sigma_\#^{k-i}\right) \cdot \left(\Sigma_\#^{i-1} \times \Sigma \times \Sigma_\#^{k-i}\right)\right)\right) \cdot \left(\Sigma_\#^k\right)^* .$$

This regular expression "guesses" that either a letter consisting solely of k # symbols occurs, or in some row of a word in $\left(\Sigma_\#^k\right)^*$ a "#" symbol is followed by a symbol in Σ. The language of this regular expression can be implemented by an NFA with $k+2$ many states. Hence, the complement $L_{\text{✓}} := \overline{L_{\text{✗}}}$ of $L_{\text{✗}}$, characterizing all "good" words, can be recognized by an NFA with 2^{k+2} many states. For the sake of readability, we do not parameterize $L_{\text{✗}}$ explicitly with k; the relevant k will always be clear from the context.

The *(existential) projection* of $R \subseteq (\Sigma^*)^{d+k}$ onto the first d components is defined as

$$\pi_d^{\exists}(R) := \left\{\bar{u} \in (\Sigma^*)^d \;\middle|\; (\bar{u}, \bar{w}) \in R \text{ for some } \bar{w} \in (\Sigma^*)^k\right\}.$$

The dual of existential projection is *universal projection*:

$$\pi_d^{\forall}(R) := \left\{\bar{u} \in (\Sigma^*)^d \;\middle|\; (\bar{u}, \bar{w}) \in R \text{ for all } \bar{w} \in (\Sigma^*)^k\right\}.$$

It is clear that $\pi_d^{\forall}(R) = \overline{\pi_d^{\exists}(\overline{R})}$. We overload the projection notation for languages

$$\pi_d^{\exists}(L) := \mathit{Rel2Lang}\big(\pi_d^{\exists}(\mathit{Lang2Rel}(L))\big) \qquad \pi_d^{\forall}(L) := \mathit{Rel2Lang}\big(\pi_d^{\forall}(\mathit{Lang2Rel}(L))\big).$$

In this article, given $\mathcal{A}_R$ such that $\mathit{Rel2Lang}(R) = \mathcal{L}(\mathcal{A}_R) \subseteq \left(\Sigma_\#^{d+k}\right)^*$, we are concerned with the computational complexity of deciding whether $\pi_d^{\forall}(R) = \emptyset$, measured in terms of $|\mathcal{A}_R|$. In Sections 3 and 5 we will prove the following.

▶ **Theorem 7.** *Deciding whether $\pi_d^{\forall}(R) \neq \emptyset$ for an automatic relation $R \subseteq (\Sigma^*)^{d+k}$ with an associated* NFA *$\mathcal{A}_R$ is* EXPSPACE*-complete. The lower bound already holds for $d = k = 1$.*

3 Emptiness after universal projection is ExpSpace-hard

3.1 Tiling problems

Let $\mathcal{T} \subseteq_{\text{fin}} \mathbb{N}^4$ be a set of *tiles* with colours coded as tuples of numbers in top–right–bottom–left order. We define natural projections *top*, *right*, *bottom*, *left*: $\mathbb{N}^4 \to \mathbb{N}$ to access individual colours of a tile, and let $\mathit{colours}(\mathcal{T}) := \mathit{top}(\mathcal{T}) \cup \mathit{right}(\mathcal{T}) \cup \mathit{bottom}(\mathcal{T}) \cup \mathit{left}(\mathcal{T})$.

▶ **Example 8** (a tile)**.** A tile $t = (2, 4, 3, 3)$ is drawn as

with various auxiliary background shades corresponding to colour values.

A $\mathcal{T}$-*tiling of size* $(h, w) \in \mathbb{N}^2_+$ is any $h \times w$ matrix $T = [t_{i,j}]_{i,j} \in \mathcal{T}^{h\times w}$. It is *valid*, whenever colours of the neighboring tiles match:

$$\mathit{bottom}(t_{i,j}) = \mathit{top}(t_{i+1,j}) \qquad \text{for every } 1 \le i \le h-1 \text{ and } 1 \le j \le w, \tag{1}$$

$$\mathit{right}(t_{i,j}) = \mathit{left}(t_{i,j+1}) \qquad \text{for every } 1 \le i \le h \quad \text{and } 1 \le j \le w-1. \tag{2}$$

See Figure 1 on page 14 for an example of a valid tiling. A $\mathcal{T}$-*tiling of width* $w \in \mathbb{N}_+$ is any tiling in $\mathcal{T}^{h\times w}$ for some $h \in \mathbb{N}_+$. We define

$$\mathcal{T}^{\star\times w} := \bigcup_{h\in\mathbb{N}_+} \mathcal{T}^{h\times w}.$$

Additionally, for two distinguished tiles $t_\nearrow, t_\swarrow \in \mathcal{T}$, let $(\mathcal{T}, t_\nearrow, t_\swarrow)$-tiling be any $\mathcal{T}$-tiling with $t_\nearrow$ placed in its top-right corner, and $t_\swarrow$ in its bottom-left corner.

▶ **Problem 9.** *CorridorTiling*

Input: A 4-tuple $(\mathcal{T}, t_\nearrow, t_\swarrow, n)$, where
- $\mathcal{T} \subseteq_{\text{fin}} \mathbb{N}^4$ is a finite set of tiles,
- $t_\nearrow, t_\swarrow \in \mathcal{T}$,
- $n \in \mathbb{N}$ given in unary.

Question: Does there exist a valid $(\mathcal{T}, t_\nearrow, t_\swarrow)$-tiling of width 2^n?

By $\mathbb{T} \subset \mathcal{P}_{\text{fin}}(\mathbb{N}^4) \times \mathbb{N}^4 \times \mathbb{N}^4 \times \mathbb{N}_+$ we denote the set of all valid instances of the above problem.

▶ **Fact 10.** *CorridorTiling (Problem 9) is ExpSpace-hard.*

It is part of the folklore of the theory of computation that tiling problems can simulate the computation of Turing machines, the width of the requested tiling corresponding to the length of tape the machine is allowed to use. ExpSpace-completeness of a variant similar to the one above is sketched in [18].

3.2 The reduction

We prove Theorem 7 by a reduction from *CorridorTiling*. We will show that the ExpSpace-hardness occurs in the simplest case of universal projection – projecting a binary relation to get a unary one. Intuitively, for each instance $\mathcal{I} = (\mathcal{T}, t_\nearrow, t_\swarrow, n)$ of *CorridorTiling*, we want to construct an automaton $\mathcal{A}_\mathcal{I}$ such that $\pi^\forall_1(\mathcal{L}(\mathcal{A}_\mathcal{I}))$ is not empty if, and only if, $\mathcal{I}$ is a YES-instance. Formally, we provide a family of LogSpace-constructible NFA $(\mathcal{A}_\mathcal{I})_{\mathcal{I}\in\mathbb{T}}$, each of size $O(n^3)$, over the alphabet $(\Sigma_\mathcal{I} \cup \{\#\})^2$ for some $\Sigma_\mathcal{I}$ and representing relation $\mathit{Lang2Rel}(\mathcal{L}(\mathcal{A}_\mathcal{I})) \subseteq (\Sigma^*_\mathcal{I})^2$ such that

$$\pi^\forall_1(\mathcal{L}(\mathcal{A}_\mathcal{I})) \neq \emptyset \iff \text{there exists a valid } (\mathcal{T}, t_\nearrow, t_\swarrow)\text{-tiling of width } 2^n. \tag{3}$$

For the rest of this section, we fix an instance $\mathcal{I} = (\mathcal{T}, t_{\nearrow}, t_{\swarrow}, n) \in \mathbb{T}$. Due to technical reasons, we assume that $n \geq 6$. Note that every instance $(\mathcal{T}, t_{\nearrow}, t_{\swarrow}, n)$ with $n < 6$ can be easily transformed into $(\mathcal{T}', t'_{\nearrow}, t'_{\swarrow}, 6)$, while preserving the (non)existence of a valid tiling.

In Section 3.3, we define $\Sigma_{\mathcal{I}}$, specify a language $L_{\mathcal{I}} \in \Sigma_{\mathcal{I}}^*$, and prove that:

▶ **Lemma 11.** *$L_{\mathcal{I}} \neq \emptyset \iff$ there exists a valid $(\mathcal{T}, t_{\nearrow}, t_{\swarrow})$-tiling of width 2^n.*

In turn in Section 3.4, we construct in LogSpace an NFA $\mathcal{A}_{\mathcal{I}}$ such that

▶ **Lemma 12.** $\pi_1^{\forall}(\mathcal{L}(\mathcal{A}_{\mathcal{I}})) = L_{\mathcal{I}}$.

This completes the proof of Theorem 7, the correctness of the reduction stemming directly from Lemmas 11 and 12.

3.3 Word encoding of tilings

Here, we provide $\Sigma_{\mathcal{I}}$ and an encoding $enc_{\mathcal{I}} \colon \mathcal{T}^{\star \times 2^n} \to \Sigma_{\mathcal{I}}^*$. Then we define $L_{\mathcal{I}}$ as an intersection of six conditions, and prove Lemma 11 by showing that it coincides with the language of encodings of valid tilings.

Let $N_n := \mathbb{N} \cap [0, n]$. Additionally, let $N_n^{\circledast k} := \{i \in N_n \mid i \circledast k\}$ for $\circledast \in \{<, =, >\}$ and $k \in \mathbb{N}$ (to be used in the next section). The alphabet $\Sigma_{\mathcal{I}}$ consists of three groups of symbols – tiles from $\mathcal{T}$, numbers from N_n, and auxiliary symbols:

$$\Sigma_{\mathcal{I}} := \mathcal{T} \cup N_n \cup \{\mathtt{A}, [\![,]\!], \langle, \rangle\}\,.$$

Above, the symbol $\mathtt{A}$ is a mnemonic – it marks places in Section 3.4 where we enforce "for-all"-type properties. In what follows, we print some symbols in colours (e.g., $\mathtt{3010}\,t\,\mathtt{20103}$) to assist in understanding the construction – such designations are auxiliary and are not reflected in the alphabet. The encoding of runs makes use of the word $\textsc{Comb}_n \in N_n^*$

$$\textsc{Comb}_n := n\, \textsc{Comb}'_{n-1}\, n\,,$$

where the words $\left(\textsc{Comb}'_i\right)_{0 \leq i \leq n}$ are defined recursively as

$$\begin{aligned} \textsc{Comb}'_0 &:= \mathtt{0} \\ \textsc{Comb}'_i &:= \textsc{Comb}'_{i-1}\, i\, \textsc{Comb}'_{i-1} && \text{for } 0 < i \leq n. \end{aligned}$$

Observe that $\textsc{Comb}_n$ has length exactly $2^n + 1$; that property is important in the upcoming construction.

▶ **Example 13.** $\textsc{Comb}_4$ is $\mathtt{40102010301020104}$ and has length 17.

We define the *encoding* function $enc_{\mathcal{I}} \colon \mathcal{T}^{\star \times 2^n} \to \Sigma_{\mathcal{I}}^*$ in three steps. Let $T = [t_{i,j}]_{i,j} \in \mathcal{T}^{h \times 2^n}$ for some $h \in \mathbb{N}$. The tile $t_{i,j}$ in T is represented as

$$encCell_{\mathcal{I}}(T, i, j) := \langle\, \textsc{Comb}_n[1, j]\, t_{i,j}\, \textsc{Comb}_n[j+1, 2^n+1]\, \mathtt{A} \,\rangle\,,$$

a single row is encoded as

$$encRow_{\mathcal{I}}(T, i) := [\![\prod_{1 \leq j \leq 2^n} encCell_{\mathcal{I}}(T, i, j)\,]\!]\,,$$

and finally, the encoding of the entire tiling is defined as

$$enc_{\mathcal{I}}(T) := \mathtt{A} \prod_{1 \leq i \leq h} encRow_{\mathcal{I}}(T, i)\,.$$

▶ **Example 14.** The tiling $T = [t_{i,j}]_{i,j}$ of size $(2, 2^4)$ is encoded as

$$\mathtt{A}\,\llbracket\langle 4\, t_{1,1}\, 0102010301020104\,\mathtt{A}\rangle \cdots \langle 40102\, t_{1,5}\, 01030\cdots 04\,\mathtt{A}\rangle \cdots \langle 4010201030102010\, t_{1,16}\, 4\,\mathtt{A}\rangle\rrbracket\,\cdot$$
$$\cdot\,\llbracket\langle 4\, t_{2,1}\, 0102010301020104\,\mathtt{A}\rangle \cdots \langle 40102\, t_{2,5}\, 01030\cdots 04\,\mathtt{A}\rangle \cdots \langle 4010201030102010\, t_{2,16}\, 4\,\mathtt{A}\rangle\rrbracket.$$

The word above is written in two lines to make the correspondence to tiling more apparent.

Languages of encodings

Define the language of encodings of valid tilings of width 2^n with $t_\nearrow, t_\swarrow$ in the correct corners

$$\textsc{ValidEnc}_{\mathcal{T}} := \left\{ enc_{\mathcal{T}}(T) \;\middle|\; T \text{ is a valid } (\mathcal{T}, t_\nearrow, t_\swarrow)\text{-tiling of width } 2^n \right\}.$$

In order to express the notion of an encoding of a valid tiling in a more tangible way, below we define languages $\textsc{Cond}^1_{\mathcal{T}}, \ldots, \textsc{Cond}^6_{\mathcal{T}}$, which – as we prove in Lemma 15 – jointly characterise encodings. The first three of them are easily definable with automata of size $O(n)$, the next two guarantee an appropriate width of the encoding, while the last one enforces in a nontrivial way that the vertical colour match.

▶ **Condition 1.** Language $\textsc{Cond}^1_{\mathcal{T}}$ is given by the regular expression

$$\mathcal{E}^1_{\mathcal{T}} := \Big(\llbracket \langle\, n\, \mathcal{T}\, N_n^*\, \mathtt{A} \rangle \Big(\langle\, N_n^*\, \mathcal{T}\, N_n^*\, \mathtt{A}\, \rangle \Big)^* \langle\, N_n^*\, \mathcal{T}\, n\, \mathtt{A} \rangle\rrbracket \Big)^* .$$

Intuitively, encodings consist of rows bounded by $\llbracket$ and $\rrbracket$; each row comprised of cells delimited by $\langle$ and $\rangle$; the first cell begins with the number n followed by a tile, while last one ends with a tile, n and $\mathtt{A}$. As $|\mathcal{E}^1_{\mathcal{T}}| = O(n)$, by Fact 4 the language $\textsc{Cond}^1_{\mathcal{T}}$ is recognised by an NFA $\mathcal{B}^1_{\mathcal{T}} := \mathcal{A}\big(\mathcal{E}^1_{\mathcal{T}}\big)$ of size $O(n)$.

▶ **Condition 2.** The language $\textsc{Cond}^2_{\mathcal{T}}$ is defined by the regular expression

$$\mathcal{E}^2_{\mathcal{T}} := \llbracket \Big(\langle N_n^*\, \mathcal{T}\, N_n^*\, \mathtt{A} \rangle \Big)^* \langle N_n^*\, t_\nearrow\, N_n^*\, \mathtt{A} \rangle \rrbracket\; \Sigma^*_{\mathcal{T}}\; \llbracket \langle N_n^*\, t_\swarrow\, N_n^*\, \mathtt{A} \rangle \Big(\langle N_n^*\, \mathcal{T}\, N_n^*\, \mathtt{A} \rangle \Big)^* \rrbracket .$$

Trivially, this requires the first row of a purported tiling to end with $t_\nearrow$, and the last row to begin with $t_\swarrow$. As in Condition 1, $\textsc{Cond}^2_{\mathcal{T}}$ is recognised by an NFA $\mathcal{B}^2_{\mathcal{T}} := \mathcal{A}\big(\mathcal{E}^2_{\mathcal{T}}\big)$ of size $O(n)$.

▶ **Condition 3.** Let $Q = colours(\mathcal{T})$ and $\mathcal{B}^3_{\mathcal{T}} = (Q, \Sigma_{\mathcal{T}}, \delta, Q, Q)$, where δ has transitions

$$\begin{aligned}
& i \xrightarrow{t} j && \text{for every } i, j \in Q \text{ and } t \in \mathcal{T} \text{ s.t. } \mathit{left}(t) = i \text{ and } \mathit{right}(t) = j,\\
& i \xrightarrow{a} i && \text{for every } i \in Q \text{ and } a \in \Sigma_{\mathcal{T}} \setminus (\mathcal{T} \cup \{\rrbracket\}),\\
& i \xrightarrow{\rrbracket} j && \text{for every } i, j \in Q.
\end{aligned}$$

We set $\textsc{Cond}^3_{\mathcal{T}} := \mathcal{L}\big(\mathcal{B}^3_{\mathcal{T}}\big)$; it contains encodings where tile colours match horizontally.

▶ **Condition 4** (each cell contains a $\textsc{Comb}_n$)**.** The definition of $\textsc{Cond}^4_{\mathcal{T}}$ uses a filtering regular expression $\mathcal{F}^4_{\mathcal{T}}$:

$$\begin{aligned}
\mathcal{F}^4_{\mathcal{T}} &:= \langle\, \underline{N_n^*}\, \mathcal{T}\, \underline{N_n^*\, \mathtt{A}}\, \rangle\, \Sigma^*_{\mathcal{T}}\\
\textsc{Cond}^4_{\mathcal{T}} &:= \left\{ w \in \Sigma^*_{\mathcal{T}} \;\middle|\; \textsc{Comb}_n\, \mathtt{A} \in \mathcal{F}^4_{\mathcal{T}}(v) \text{ for every proper suffix } v \text{ of } w \text{ s.t. } v[1] = \langle \right\}
\end{aligned}$$

▶ **Condition 5** (prefix of a cell and first symbols of following cells' suffixes form a $\textsc{Comb}_n$)**.**

$$\begin{aligned}
\mathcal{F}^5_{\mathcal{T}} &:= \langle\, \underline{N_n^*}\, \mathcal{T}\, \underline{N_n}\, N_n^*\, \mathtt{A}\, \rangle \Big(\langle\, N_n^*\, \mathcal{T}\, \underline{N_n}\, N_n^*\, \mathtt{A}\, \rangle \Big)^* \langle\, N_n^*\, \mathcal{T}\, \underline{N_n}\, N_n^*\, \underline{\mathtt{A}}\, \rangle \rrbracket\, \Sigma^*_{\mathcal{T}}\\
\textsc{Cond}^5_{\mathcal{T}} &:= \left\{ w \in \Sigma^*_{\mathcal{T}} \;\middle|\; \textsc{Comb}_n\, \mathtt{A} \in \mathcal{F}^5_{\mathcal{T}}(v) \text{ for every proper suffix } v \text{ of } w \text{ s.t. } v[1] = \langle \right\}
\end{aligned}$$

▶ **Condition 6** (tile colours match vertically)**.** Let $\blacktriangledown_t := \{t' \in \mathfrak{T} \mid \mathit{top}(t') = \mathit{bottom}(t)\}$ be the set of tiles with the top colour matching to the bottom of a tile t. Define

$$\mathcal{F}^6_{\mathfrak{T}} := \sum_{t\in\mathfrak{T}} \Big(\qquad\qquad \langle\, \underline{N_n^*}\ t\ N_n^*\, \mathtt{A}\,\rangle \left(\langle\, N_n^*\, \mathfrak{T}\, N_n^*\, \mathtt{A}\,\rangle\right)^* \rrbracket \cdot$$

$$\cdot \llbracket \left(\langle\, N_n^*\, \mathfrak{T}\, N_n^*\, \mathtt{A}\,\rangle\right)^* \langle\, N_n^*\, \blacktriangledown_t\, \underline{N_n^*\, \mathtt{A}}\,\rangle \left(\langle\, N_n^*\, \mathfrak{T}\, N_n^*\, \mathtt{A}\,\rangle\right)^* \rrbracket\, \Sigma_{\mathfrak{T}}^* \Big)\,.$$

The expression above was typeset in two lines only to highlight the correspondence between cells in two consecutive rows. Define the language $\textsc{Cond}^6_{\mathfrak{T}}$ as

$$\textsc{Cond}^6_{\mathfrak{T}} := \big\{ w \in \Sigma_{\mathfrak{T}}^* \;\big|\; \textsc{Comb}_n\, \mathtt{A} \in \mathcal{F}^6_{\mathfrak{T}}(v) \text{ for every proper suffix } v \text{ of } w \text{ such that}$$
$$v[1] = \langle \text{ and } v[j] = \llbracket \text{ for some } j \qquad\qquad \big\}$$

Intuitively, requiring $\llbracket$ to appear in v filters out suffixes of the last row.

Define $L_{\mathfrak{T}} := \mathtt{A} \bigcap_{1\le i\le 6} \textsc{Cond}^i_{\mathfrak{T}}$. To prove Lemma 11, it suffices to show the following:

▶ **Lemma 15.** $L_{\mathfrak{T}} = \textsc{ValidEnc}_{\mathfrak{T}}$

Proof. The inclusion $L_{\mathfrak{T}} \supseteq \textsc{ValidEnc}_{\mathfrak{T}}$ is trivial.

Inclusion $L_{\mathfrak{T}} \subseteq \textsc{ValidEnc}_{\mathfrak{T}}$. Take any $u \in L_{\mathfrak{T}}$. Due to Condition 1, it has the form $\mathtt{A} \prod_{1\le i\le h} \llbracket\, v_i \,\rrbracket$, where each $v_i \in (\Sigma_{\mathfrak{T}} \setminus \{\llbracket, \rrbracket\})^*$. We will show that $\llbracket\, v_i \,\rrbracket \in \mathit{Range}(\mathit{encRow}_{\mathfrak{T}}(\cdot))$ for all i. Fix an arbitrary i. Again due to Condition 1, v_i has the form

$$\prod_{1\le j\le w_i} (\langle\, p_{i,j}\, t_{i,j}\, s_{i,j}\, \mathtt{A}\,\rangle)\,,$$

where $w_i \in \mathbb{N}$, $p_{i,j}, s_{i,j} \in N_n^*$, $p_{i,1} = s_{i,w_i} = n$, and $t_{i,j} \in \mathfrak{T}$. Due to Condition 5, we have that all $s_{i,j}$ are nonempty and

$$p_{i,1}\, s_{i,1}[1]\, s_{i,2}[1]\, s_{i,3}[1] \cdots s_{i,w_i}[1] = \textsc{Comb}_n\,. \tag{4}$$

This implies that $w_i = 2^n$. By Condition 5 and Equation (4) we get that $p_{i,j} = \textsc{Comb}_n[1, j]$, and now Condition 4 implies that $s_{i,j} = \textsc{Comb}_n[j+1, 2^n+1]$, so $\llbracket\, v_i \,\rrbracket$ is a valid encoding of a row of length 2^n. Hence u encodes a tiling $T := [t_{i,j}]_{i,j} \in \mathfrak{T}^{h\times 2^n}$. Property (2) in the definition of a valid tiling is now trivially implied by Condition 3, and we only need to show (1). Fix arbitrary pair of tiles $t_{i,j}, t_{i+1,j}$ which are vertical neighbours. Observe that $p_{i,j} s_{i+1,x} = \textsc{Comb}_n \iff x = j$. Therefore, by Condition 6, $\mathit{bottom}(t_{i,j}) = \mathit{top}(t_{i+1,j})$, thus T is a valid $\mathfrak{T}$-tiling, and – by Condition 2 – a valid $(\mathfrak{T}, t_{\nearrow}, t_{\swarrow})$-tiling. ◀

3.4 Construction of the automaton $\mathcal{A}_{\mathfrak{T}}$

Let $\Sigma_{\mathfrak{T},\#} := \Sigma_{\mathfrak{T}} \cup \{\#\}$. Here, we define the NFA $\mathcal{A}_{\mathfrak{T}}$ over $\Sigma^2_{\mathfrak{T},\#}$ and prove Lemma 12, which states that $\pi_1^{\forall}(\mathcal{L}(\mathcal{A}_{\mathfrak{T}})) = L_{\mathfrak{T}}$. The construction we present in this section, however, does not require the full generality of the setting of automatic structures:

- $\mathit{Lang2Rel}(\mathcal{L}(\mathcal{A}_{\mathfrak{T}}))$ only holds for words of the same length, i.e., $\mathcal{A}_{\mathfrak{T}}$ rejects words with #;
- we only use a subset of the alphabet: $\Sigma_{\mathfrak{T}} \times N_n \subseteq \Sigma^2_{\mathfrak{T},\#}$.

For this reason, we begin with a simplifying Lemma 16, which allows us to focus only on words satisfying above properties. Let $\rho_{\mathfrak{T}} \colon (\Sigma_{\mathfrak{T}} \times N_n)^* \to \Sigma_{\mathfrak{T}}^*$ be a homomorphism given by $\rho_{\mathfrak{T}}(a, \cdot) := a$. Additionally, let

$$\rho^{\forall}_{\mathfrak{T}}(L) := \big\{ w \in \Sigma_{\mathfrak{T}}^* \;\big|\; \rho_{\mathfrak{T}}^{-1}(w) \subseteq L \big\}\,.$$

▶ **Lemma 16** (simplification). *For any* NFA $\mathcal{A}'_{\mathcal{J}}$ *over* $\Sigma_{\mathcal{J}} \times N_n$*, there exists an* NFA $\mathcal{A}_{\mathcal{J}}$ *over* $\Sigma^2_{\mathcal{J},\#}$ *such that* $\pi^{\forall}_1(\mathcal{L}(\mathcal{A}_{\mathcal{J}})) = \rho^{\forall}_{\mathcal{J}}(\mathcal{L}(\mathcal{A}'_{\mathcal{J}}))$.

Proof. Take any $\mathcal{A}'_{\mathcal{J}}$ over $\Sigma_{\mathcal{J}} \times N_n$. Let

$$\begin{aligned}
\mathcal{E}_1 &\coloneqq \big(\Sigma^2_{\mathcal{J}}\big)^*(\Sigma_{\mathcal{J}} \times \{\#\})^+ + \big(\Sigma^2_{\mathcal{J}}\big)^*(\{\#\} \times \Sigma_{\mathcal{J}})^+ && (u \otimes v \text{ such that } |u| \neq |v|)\\
\mathcal{E}_2 &\coloneqq \big(\Sigma^2_{\mathcal{J}}\big)^*(\Sigma_{\mathcal{J}} \times (\Sigma_{\mathcal{J}} \setminus N_n))\big(\Sigma^2_{\mathcal{J}}\big)^* && (\text{words with letter from } \Sigma^2_{\mathcal{J}} \setminus \Sigma_{\mathcal{J}} \times \mathbb{N}_n)\\
\mathcal{A}_{\mathcal{J}} &\coloneqq \mathcal{A}'_{\mathcal{J}} \oplus \mathcal{A}(\mathcal{E}_1) \oplus \mathcal{A}(\mathcal{E}_2)\,.
\end{aligned}$$

By definition, a word w belongs to $\pi^{\forall}_1(\mathcal{L}(\mathcal{A}_{\mathcal{J}}))$ whenever for all v the word $w \otimes v$ belongs to $\mathcal{L}(\mathcal{A}_{\mathcal{J}})$. By construction, $\mathcal{L}(\mathcal{A}_{\mathcal{J}})$ contains all $w \otimes v$ where $|v| \neq |w|$ ($\mathcal{E}_1$) or where v is using a symbol from $\Sigma_{\mathcal{J}} \setminus N_n$ ($\mathcal{E}_2$). Hence, the only words which can be missing from $\mathcal{L}(\mathcal{A}_{\mathcal{J}})$ come from $\mathcal{L}(\mathcal{A}'_{\mathcal{J}})$. This implies that $\pi^{\forall}_1(\mathcal{L}(\mathcal{A}_{\mathcal{J}})) = \rho^{\forall}_{\mathcal{J}}(\mathcal{L}(\mathcal{A}'_{\mathcal{J}}))$. ◀

Therefore, we only have to provide $\mathcal{A}'_{\mathcal{J}}$ such that $\rho^{\forall}_{\mathcal{J}}(\mathcal{L}(\mathcal{A}'_{\mathcal{J}})) = L_{\mathcal{J}}$. The construction is modular, based on six NFA corresponding to Conditions 1–6:

▶ **Lemma 17** (modular design). *For any six* NFA $(\mathcal{C}^i_{\mathcal{J}})_{1\le i\le 6}$ *over* $\Sigma_{\mathcal{J}} \times N_n$*, there exists an* NFA $\mathcal{A}'_{\mathcal{J}}$ *of size* $O\big(\sum_{1\le i\le 6}|\mathcal{C}^i_{\mathcal{J}}|\big)$ *over* $\Sigma_{\mathcal{J}} \times N_n$ *such that*

$$\rho^{\forall}_{\mathcal{J}}(\mathcal{L}(\mathcal{A}'_{\mathcal{J}})) = \mathtt{A} \bigcap_{1\le i\le 6} \rho^{\forall}_{\mathcal{J}}\big(\mathcal{L}\big(\mathcal{C}^i_{\mathcal{J}}\big)\big)\,.$$

Proof. Define

$$\begin{aligned}
\mathcal{H} &\coloneqq (\{\mathtt{A}\} \times N_n \setminus \{1, 2, \dots, 6\})\,(\Sigma_{\mathcal{J}} \times N_n)^*\\
\mathcal{A}'_{\mathcal{J}} &\coloneqq \mathcal{A}((\mathtt{A}, 1)) \odot \mathcal{C}^1_{\mathcal{J}} \quad\oplus\quad \mathcal{A}((\mathtt{A}, 2)) \odot \mathcal{C}^2_{\mathcal{J}} \quad\oplus\quad \cdots \quad\oplus\quad \mathcal{A}((\mathtt{A}, 6)) \odot \mathcal{C}^6_{\mathcal{J}} \quad\oplus\quad \mathcal{A}(\mathcal{H})\,.
\end{aligned}$$

Observe that

$$\begin{aligned}
\mathtt{A}w \in \rho^{\forall}_{\mathcal{J}}(\mathcal{L}(\mathcal{A}'_{\mathcal{J}})) \iff \rho^{-1}_{\mathcal{J}}(\mathtt{A}w) \subseteq \mathcal{L}(\mathcal{A}'_{\mathcal{J}}) \iff (\{\mathtt{A}\} \times N_n)\,\rho^{-1}_{\mathcal{J}}(w) \subseteq \mathcal{L}(\mathcal{A}'_{\mathcal{J}}) \iff\\
\iff \forall i \in N_n\,.\,(\mathtt{A}, i)\,\rho^{-1}_{\mathcal{J}}(w) \subseteq \mathcal{L}(\mathcal{A}'_{\mathcal{J}})\,,
\end{aligned}$$

but trivially

$$\begin{aligned}
\mathcal{L}\Big(\mathcal{A}((\mathtt{A}, j)) \odot \mathcal{C}^j_{\mathcal{J}}\Big) \cap (\mathtt{A}, i)\,\rho^{-1}_{\mathcal{J}}(w) &= \emptyset && \text{for any } i \neq j\\
\mathcal{L}(\mathcal{A}(\mathcal{H})) \cap (\mathtt{A}, i)\,\rho^{-1}_{\mathcal{J}}(w) &= \emptyset && \text{for any } i.
\end{aligned}$$

Therefore, $\mathtt{A}w \in \rho^{\forall}_{\mathcal{J}}(\mathcal{L}(\mathcal{A}'_{\mathcal{J}}))$ if, and only if, $\rho^{-1}_{\mathcal{J}}(w) \subseteq \rho^{\forall}_{\mathcal{J}}\big(\mathcal{L}\big(\mathcal{C}^i_{\mathcal{J}}\big)\big)$ for all i, as required. ◀

By definition of $L_{\mathcal{J}}$, it only remains to construct automata $\mathcal{C}^i_{\mathcal{J}}$ such that $\rho^{\forall}_{\mathcal{J}}\big(\mathcal{L}\big(\mathcal{C}^i_{\mathcal{J}}\big)\big) = \textsc{Cond}^i_{\mathcal{J}}$ for $1 \le i \le 6$. The construction is easy for Conditions 1–3:

$$\mathcal{C}^i_{\mathcal{J}} \coloneqq \rho^{-1}_{\mathcal{J}}\big(\mathcal{B}^i_{\mathcal{J}}\big) \qquad\qquad \text{for } i \in \{1, 2, 3\}$$

as $\rho^{\forall}_{\mathcal{J}}\big(\mathcal{L}\big(\rho^{-1}_{\mathcal{J}}(\mathcal{A})\big)\big) = \mathcal{L}(\mathcal{A})$ for any NFA $\mathcal{A}$. Observe that the remaining Conditions 4–6 all speak about "every proper suffix" satisfying some simple regular property. We handle that in a general way. For $L \subseteq (\Sigma_{\mathcal{J}} \times N_n)^*$, define

$$L_{\forall\text{suf}}(L) \coloneqq \big\{w \bigm| v \in \rho^{\forall}_{\mathcal{J}}(L) \text{ for all proper suffixes } v \text{ of } w\big\}$$

▶ **Lemma 18** (recognising "for all proper suffixes"). *For any* NFA $\mathcal{A}$ *over* $\Sigma_{\mathcal{J}} \times N_n$*, there exists an* NFA $\textsc{AllSuf}(\mathcal{A})$ *of size* $O(|\mathcal{A}|)$ *such that*

$$\rho^{\forall}_{\mathcal{J}}(\mathcal{L}(\textsc{AllSuf}(\mathcal{A}))) = L_{\forall\text{suf}}(\mathcal{L}(\mathcal{A}))\,.$$

Proof. Fix any NFA $\mathcal{A} = (Q, \Sigma_{\mathcal{J}} \times N_n, \delta, Q_{\mathrm{I}}, Q_{\mathrm{F}})$. We define $\textsc{AllSuf}(\mathcal{A})$ which guesses the suffix to verify

$$\textsc{AllSuf}(\mathcal{A}) \coloneqq (Q \cup \{s\}, \Sigma_{\mathcal{J}} \times N_n, \delta \cup \delta', \{s\}, Q_{\mathrm{F}} \cup \{s\})$$

for some fresh state $s \notin Q$, and δ' containing transitions $s \xrightarrow{(a,0)} s$ for $a \in \Sigma_{\mathcal{J}}$ and $s \xrightarrow{(a,n)} q$ for $a \in \Sigma_{\mathcal{J}}$, $n \in N_n^{>0}$, $q \in Q_{\mathrm{I}}$. Additionally, let τ be a homomorphism such that $\tau(a) \coloneqq (a, 0)$.

Inclusion "$\subseteq$". Take any $w \in \rho_{\mathcal{J}}^{\forall}(\mathcal{L}(\textsc{AllSuf}(\mathcal{A})))$. Let v be any proper suffix of w. Take any $v' \in \rho_{\mathcal{J}}^{-1}(v)$. We need to show that $v' \in \mathcal{L}(\mathcal{A})$. The word w can be written as uav, for $|u| \geq 0$ and $|a| = 1$. Consider a word $w' = \tau(u)(a,1)v'$. By definition of $\rho_{\mathcal{J}}^{\forall}$, $w' \in \mathcal{L}(\textsc{AllSuf}(\mathcal{A}))$. Let r be an accepting run of $\textsc{AllSuf}(\mathcal{A})$ over w'. By construction, the run stays in state s while reading $\tau(u)$ and goes to some $q \in Q_{\mathrm{I}}$ upon reading $(a, 1)$. Therefore, the remaining suffix of r is an accepting run of $\mathcal{A}$ over v'.

Inclusion "$\supseteq$". Fix $w \in L_{\forall\mathrm{suf}}(\mathcal{L}(\mathcal{A}))$. Take any $w' \in \rho_{\mathcal{J}}^{-1}(w)$. We will show that $w' \in \mathcal{L}(\textsc{AllSuf}(\mathcal{A}))$. Let $u'(a,k)v' \coloneqq w'$ be such that u' is the maximal prefix arising as $\tau(u)$ for some u (possibly empty). Note that $k \neq 0$. By assumption, $v' \in \mathcal{L}(\mathcal{A})$, so there exists an accepting run r_2 of $\mathcal{A}$ over v' starting in some $q \in Q_{\mathrm{I}}$. By construction, there exists a run r_1 from s to q over $u'(a,k)$ in $\textsc{AllSuf}(\mathcal{A})$. Hence the run $r_1 r_2$ accepts w'. ◀

To handle conditions "beginning with $\langle$" and "containing $\llbracket$" appearing as antecedents of implications, we proceed in the vein of the equivalence $a \to b \equiv \neg a \vee b$. Let

$$\mathcal{G}_{\neg\langle} \coloneqq (\Sigma_{\mathcal{J}} \setminus \{\langle\})\, \Sigma_{\mathcal{J}}^{*} \qquad\qquad \mathcal{G}_{\neg\llbracket} \coloneqq (\Sigma_{\mathcal{J}} \setminus \{\llbracket\})^{*}\,.$$

▶ **Lemma 19.** *For $i \in \{4,5,6\}$, given* NFA *$\hat{\mathcal{C}}_{\mathcal{J}}^{i}$ satisfying $\rho^{\forall}(\mathcal{L}(\hat{\mathcal{C}}_{\mathcal{J}}^{i})) = \{w \mid \textsc{Comb}_n\,\mathtt{A} \in \mathcal{F}_{\mathcal{J}}^{i}(w)\}$, one can construct $\mathcal{C}_{\mathcal{J}}^{i}$ of size $O(|\hat{\mathcal{C}}_{\mathcal{J}}^{i}|)$ such that $\rho_{\mathcal{J}}^{\forall}(\mathcal{L}(\mathcal{C}_{\mathcal{J}}^{i})) = \textsc{Cond}_{\mathcal{J}}^{i}$.*

Proof. Fix $\hat{\mathcal{C}}_{\mathcal{J}}^{4}, \hat{\mathcal{C}}_{\mathcal{J}}^{5}, \hat{\mathcal{C}}_{\mathcal{J}}^{6}$ as in the statement of the lemma. We define $\mathcal{C}_{\mathcal{J}}^{i}$ as

$$\begin{aligned}
\mathcal{C}_{\mathcal{J}}^{4} &\coloneqq \textsc{AllSuf}\big(\hat{\mathcal{C}}_{\mathcal{J}}^{4} \oplus \rho_{\mathcal{J}}^{-1}\big(\mathcal{A}(\mathcal{G}_{\neg\langle})\big)\big)\\
\mathcal{C}_{\mathcal{J}}^{5} &\coloneqq \textsc{AllSuf}\big(\hat{\mathcal{C}}_{\mathcal{J}}^{5} \oplus \rho_{\mathcal{J}}^{-1}\big(\mathcal{A}(\mathcal{G}_{\neg\langle})\big)\big)\\
\mathcal{C}_{\mathcal{J}}^{6} &\coloneqq \textsc{AllSuf}\big(\hat{\mathcal{C}}_{\mathcal{J}}^{6} \oplus \rho_{\mathcal{J}}^{-1}\big(\mathcal{A}(\mathcal{G}_{\neg\langle} + \mathcal{G}_{\neg\llbracket})\big)\big)\,.
\end{aligned}$$

The above cases are similar; w.l.o.g. let us focus on $\mathcal{C}^4$. Observe that

$$\rho^{\forall}\big(\mathcal{L}\big(\hat{\mathcal{C}}_{\mathcal{J}}^{4} \oplus \rho_{\mathcal{J}}^{-1}\big(\mathcal{A}(\mathcal{G}_{\neg\langle})\big)\big)\big) = \rho^{\forall}\big(\mathcal{L}\big(\hat{\mathcal{C}}_{\mathcal{J}}^{4}\big)\big) \cup \mathcal{L}\big(\mathcal{G}_{\neg\langle}\big) = \big\{w \mid \textsc{Comb}_n\,\mathtt{A} \in \mathcal{F}_{\mathcal{J}}^{i}(w)\big\} \cup \mathcal{L}\big(\mathcal{G}_{\neg\langle}\big)\,,$$

which directly corresponds to Condition 4, as required. ◀

The essential element needed to define NFA $\hat{\mathcal{C}}_{\mathcal{J}}^{i}$ as in Lemma 19 is an NFA for the language $\{\textsc{Comb}_n\mathtt{A}\}$. First, we define $\textsc{Comb}_n$ as the intersection of languages of $n+1$ regular expressions, then show how that can be concisely represented by an automaton $\mathcal{C}_n$ of size $O(n^2)$ such that $\rho_{\mathcal{J}}^{\forall}(\mathcal{L}(\mathcal{C}_n)) = \{\textsc{Comb}_n\mathtt{A}\}$.

▶ **Definition 20.** *We define $n+1$ regular expressions $\mathcal{E}_i$ over $\Sigma_{\mathcal{J}}$*

$$\begin{aligned}
\mathcal{E}_0 &\coloneqq N_n^{>1}\,\big(\mathtt{0}\,N_n^{>1}\big)^{*}\\
\mathcal{E}_i &\coloneqq N_n^{>i}\,\Big(\big(N_n^{<i}\big)^{*}\,i\,\big(N_n^{<i}\big)^{*}\,N_n^{>i}\Big)^{*} && \textit{for } 0 < i < n\\
\mathcal{E}_n &\coloneqq n\,\big(N_n^{<n}\big)^{*}\,n
\end{aligned}$$

▶ **Lemma 21.** $\{\textsc{Comb}_n\} = \bigcap_{0 \leq i \leq n} \mathcal{L}(\mathcal{E}_i)$.

Proof. It is easy to prove the inclusion "$\subseteq$" by unravelling the definition of $\textsc{Comb}_n$.

Inclusion "$\supseteq$". Take any $w \in \bigcap_{1 \le i \le n} \mathcal{L}(\mathcal{E}_i)$. We will show that $w = \textsc{Comb}_n$.

▷ Claim 22. For $0 \le k \le n-1$, we have $\bigcap_{1 \le i \le k} \mathcal{L}(\mathcal{E}_i) = \mathcal{L}\big(N_n^{>k}\left(\textsc{Comb}'_k\, N_n^{>k}\right)\big)^*$

We prove the claim by induction. The base case is trivial. Fix a word

$$w \in \mathcal{L}\big(N_n^{>k}\left(\textsc{Comb}'_k\, N_n^{>k}\right)\big)^* \cap \mathcal{L}(\mathcal{E}_{k+1}) .$$

It has the form $w = a_1\, \textsc{Comb}'_k\, a_2\, \textsc{Comb}'_k \cdots \textsc{Comb}'_k\, a_m$ for some $m \ge 2$ and $a_1, a_2, \dots, a_m \in N_n^{>k}$. But since $w \in \mathcal{L}(\mathcal{E}_{k+1})$, every other symbol a_i is equal $k+1$ and m is odd. Thus

$$w = a_1\, \underbrace{\textsc{Comb}'_k\,(k+1)\,\textsc{Comb}'_k}_{\textsc{Comb}'_{k+1}} \cdots \underbrace{\textsc{Comb}'_k\,(k+1)\,\textsc{Comb}'_k}_{\textsc{Comb}'_{k+1}}\, a_m$$

We conclude by noticing that $\mathcal{L}(\mathcal{E}_n) \cap \mathcal{L}\Big(N_n^{>(n-1)}\left(\textsc{Comb}'_{n-1}\, N_n^{>(n-1)}\right)\Big)^* = \{\textsc{Comb}_n\}$. ◀

Let us define

$$\mathcal{C}_n \coloneqq \rho_{\mathcal{J}}^{-1}(\mathcal{A}(\mathcal{E}_0)) \odot \mathcal{A}((\mathtt{A}, \mathtt{0})) \oplus \rho_{\mathcal{J}}^{-1}(\mathcal{A}(\mathcal{E}_1)) \odot \mathcal{A}((\mathtt{A}, \mathtt{1})) \oplus \cdots \oplus \rho_{\mathcal{J}}^{-1}(\mathcal{A}(\mathcal{E}_n)) \odot \mathcal{A}((\mathtt{A}, n)) .$$

▶ **Lemma 23.** $\rho_{\mathcal{J}}^{\forall}(\mathcal{L}(\mathcal{C}_n)) = \left(\bigcap_{0 \le i \le n} \mathcal{L}(\mathcal{E}_i)\right) \mathtt{A}$.

Proof. Inclusion "$\subseteq$". Take any $w = u\mathtt{A} \in \rho_{\mathcal{J}}^{\forall}(\mathcal{L}(\mathcal{C}_n))$, and $i \in N_n$. We prove that $u \in \mathcal{L}(\mathcal{E}_i)$. By definition, $\rho_{\mathcal{J}}^{-1}(u\mathtt{A}) \subseteq \mathcal{L}(\mathcal{C}_n)$. Fix a homomorphism $\tau(a) = (a, 0)$. Note that $\tau(u)(\mathtt{A}, i) \in \mathcal{L}(\mathcal{C}_n)$. This can be accepted only by the $\rho_{\mathcal{J}}^{-1}(\mathcal{A}(\mathcal{E}_i)) \odot \mathcal{A}((\mathtt{A}, i))$ component, thus $u \in \mathcal{L}(\mathcal{A}(\mathcal{E}_i)) = \mathcal{L}(\mathcal{E}_i)$, as required.

Inclusion "$\supseteq$". Take any $w = u\mathtt{A} \in \left(\bigcap_{0 \le i \le n} \mathcal{L}(\mathcal{E}_i)\right) \mathtt{A}$. Take any $u'(\mathtt{A}, i) \in \rho_{\mathcal{J}}^{-1}(u\mathtt{A})$. Since $u \in \mathcal{L}(\mathcal{E}_i)$, $u' \in \rho_{\mathcal{J}}^{-1}(\mathcal{L}(\mathcal{E}_i))$, and $u'(\mathtt{A}, i) \in \mathcal{L}\big(\rho_{\mathcal{J}}^{-1}(\mathcal{A}(\mathcal{E}_i)) \odot \mathcal{A}((\mathtt{A}, i))\big)$, as required. ◀

▶ **Definition 24** (NFA $\hat{\mathcal{C}}_{\mathcal{J}}^i$). *Fix $i \in \{4,5,6\}$, NFA $\mathcal{C}_n = (Q^{(1)}, \Sigma \times N_n, \delta^{(1)}, Q_I^{(1)}, Q_F^{(1)})$ (of size $O(n^2)$) and $\mathcal{A}(\mathcal{F}_{\mathcal{J}}^i) = (Q^{(2)}, \Sigma \times \Phi, \delta^{(2)}, Q_I^{(2)}, Q_F^{(2)})$ (of size $O(n)$).*

Define $\hat{\mathcal{C}}_{\mathcal{J}}^i \coloneqq (Q, \Sigma \times N_n, \delta, Q_I, Q_F)$ of size $O(n^3)$, where

$$Q \coloneqq Q^{(1)} \times Q^{(2)}, \qquad Q_I \coloneqq Q_I^{(1)} \times Q_I^{(2)}, \qquad Q_F \coloneqq Q_F^{(1)} \times Q_F^{(2)},$$

and the transition relation is

$$\delta \coloneqq \left\{ (p,q) \xrightarrow{(a,\alpha)} (r,s) \;\middle|\; q \xrightarrow{(a,\top)} s \in \delta^{(2)} \wedge p \xrightarrow{(a,\alpha)} r \in \delta^{(1)} \right\} \cup$$
$$\left\{ (p,q) \xrightarrow{(a,\alpha)} (p,s) \;\middle|\; q \xrightarrow{(a,\bot)} s \in \delta^{(2)} \wedge p \in Q^{(1)} \right\} .$$

Intuitively, $\hat{\mathcal{C}}_{\mathcal{J}}^i$ runs $\mathcal{C}_n$ over the fragments of the input which were underlined by $\mathcal{F}_{\mathcal{J}}^i$.

▶ **Fact 25.** *$w \in \mathcal{L}(\hat{\mathcal{C}}_{\mathcal{J}}^i)$ if, and only if, $\exists v \in \mathcal{L}\big(\rho_{\mathcal{J}}^{-1}(\mathcal{F}_{\mathcal{J}}^i)\big) \,.\, \psi_{\text{in}}(v) = w \wedge \psi_{\text{out}}(v) \in \mathcal{L}(\mathcal{C}_n)$.*

To finish the construction, we need to prove that

▶ **Lemma 26.** *For $i \in \{4,5,6\}$*

$$\rho^{\forall}\big(\mathcal{L}(\hat{\mathcal{C}}_{\mathcal{J}}^i)\big) = \left\{ w \;\middle|\; \textsc{Comb}_n\, \mathtt{A} \in \mathcal{F}_{\mathcal{J}}^i(w) \right\} .$$

As the proofs for $i \in \{4,5,6\}$ are analogous, we focus on the hardest one, and then only comment how it can be adapted for $i \in \{4,5\}$.

Proof ($i = 6$).

A. Inclusion "$\subseteq$". Take any $w \in \rho^{\forall}\big(\mathcal{L}\big(\hat{\mathcal{C}}^6_{\mathcal{J}}\big)\big)$. Define

$$U \coloneqq \big\{u \in \mathcal{L}\big(\mathcal{F}^6_{\mathcal{J}}\big) \mid \psi_{\text{in}}(u) = w\big\}$$

Note that if $U = \emptyset$, then $\mathcal{F}^6_{\mathcal{J}}(w) = \emptyset$, so by Fact 25 $\mathcal{L}\big(\hat{\mathcal{C}}^6_{\mathcal{J}}\big) = \emptyset$, and $\rho^{\forall}\big(\mathcal{L}\big(\hat{\mathcal{C}}^6_{\mathcal{J}}\big)\big) = \emptyset$, a contradiction. Therefore, $U \neq \emptyset$, and $w \in \mathcal{L}\big(\psi_{\text{in}}\big(\mathcal{F}^6_{\mathcal{J}}\big)\big)$, so it has the form

$$\langle\, p\, t\, s\, \mathtt{A} \,\rangle\, \beta \,\rrbracket\, \llbracket\, \langle\, p_1\, t_1\, s_1\, \mathtt{A} \,\rangle \langle\, p_2\, t_2\, s_2\, \mathtt{A} \,\rangle \cdots \langle\, p_k\, t_k\, s_k\, \mathtt{A} \,\rangle\, \rrbracket\, \gamma$$

for some $k \in \mathbb{N}$, $p, p_i, s, s_i \in N_n^*$, $t, t_i \in \mathcal{T}$, $\gamma \in (\Sigma_{\mathcal{J}} \setminus \{\llbracket, \rrbracket\})^*$ and $\gamma \in \Sigma_{\mathcal{J}}^*$. Furthermore, $|U| = k$ and it contains the following underlined words $u_1, \ldots, u_k \in (\Sigma_{\mathcal{J}} \times \Phi)^*$:

$$u_1 = \langle\, \underline{p}\, t\, s\, \mathtt{A} \,\rangle\, \beta \,\rrbracket\, \llbracket\, \langle\, p_1\, t_1\, \underline{s_1\, \mathtt{A}} \,\rangle \langle\, p_2\, t_2\, s_2\, \mathtt{A} \,\rangle \cdots \langle\, p_k\, t_k\, s_k\, \mathtt{A} \,\rangle\, \rrbracket\, \gamma$$
$$u_2 = \langle\, \underline{p}\, t\, s\, \mathtt{A} \,\rangle\, \beta \,\rrbracket\, \llbracket\, \langle\, p_1\, t_1\, s_1\, \mathtt{A} \,\rangle \langle\, p_2\, t_2\, \underline{s_2\, \mathtt{A}} \,\rangle \cdots \langle\, p_k\, t_k\, s_k\, \mathtt{A} \,\rangle\, \rrbracket\, \gamma$$
$$\vdots$$
$$u_k = \langle\, \underline{p}\, t\, s\, \mathtt{A} \,\rangle\, \beta \,\rrbracket\, \llbracket\, \langle\, p_1\, t_1\, s_1\, \mathtt{A} \,\rangle \langle\, p_2\, t_2\, s_2\, \mathtt{A} \,\rangle \cdots \langle\, p_k\, t_k\, \underline{s_k\, \mathtt{A}} \,\rangle\, \rrbracket\, \gamma$$

Consider two cases, depending on the validity of the following assertion

$$\exists u \in U \,.\, \psi_{\text{out}}\big(\rho_{\mathcal{J}}^{-1}(u)\big) \subseteq \mathcal{L}(\mathcal{C}_n)$$

Case A.1: such u exists. Take any such $u \in U$. Observe that $\psi_{\text{out}}(u) \in \rho^{\forall}_{\mathcal{J}}(\mathcal{L}(\mathcal{C}_n)) = \{\textsc{Comb}_n\mathtt{A}\}$. Hence, $\psi_{\text{in}}(u) = w$, $\psi_{\text{out}}(u) = \textsc{Comb}_n\mathtt{A}$, and $u \in \mathcal{L}\big(\mathcal{F}^6_{\mathcal{J}}\big)$. Therefore, $\textsc{Comb}_n\mathtt{A} \in \mathcal{F}^6_{\mathcal{J}}(w)$, as required.

Case A.2: such u does not exist. Therefore, for every $u \in U$, there is some $v_u \in \rho_{\mathcal{J}}^{-1}(u)$ such that $\psi_{\text{out}}(v_u) \notin \mathcal{L}(\mathcal{C}_n)$. Fix any family $(v_u)_{u \in U}$ of such words. Let α_u be the position of the last underlined symbol in u. Fix a word $w' \in \rho_{\mathcal{J}}^{-1}(w)$ such that

$$w'[i] = \begin{cases} \psi_{\text{out}}(v_u[i]) & \text{if } i = \alpha_u \text{ for some } u \\ (w[i], 0) & \text{otherwise} \end{cases}.$$

Observe that w' is properly defined, as positions α_u are pairwise different (corresponding to the last letters of $s_1, s_2, \ldots, s_k$). Since $\rho_{\mathcal{J}}(w') = w$, from assumption $w \in \rho^{\forall}\big(\mathcal{L}\big(\hat{\mathcal{C}}^6_{\mathcal{J}}\big)\big)$ we have that $w' \in \mathcal{L}\big(\hat{\mathcal{C}}^6_{\mathcal{J}}\big)$. By Fact 25, we obtain $v \in \mathcal{L}\big(\rho_{\mathcal{J}}^{-1}\big(\mathcal{F}^6_{\mathcal{J}}\big)\big)$ such that

$$\psi_{\text{in}}(v) = w' \wedge \psi_{\text{out}}(v) \in \mathcal{L}(\mathcal{C}_n)$$

However, $\rho_{\mathcal{J}}(\psi_{\text{out}}(v)) = \rho_{\mathcal{J}}(\psi_{\text{out}}(v_u))$ for some $u \in U$ and last symbols of $\psi_{\text{out}}(v)$ and $\psi_{\text{out}}(v_u)$ are identical. Since by construction $\mathcal{C}_n$ ignores the component N_n of its alphabet $\Sigma_I \times N_n$ for all letters but the last one, we get that

$$\psi_{\text{out}}(v) \in \mathcal{L}(\mathcal{C}_n) \iff \psi_{\text{out}}(v_u) \in \mathcal{L}(\mathcal{C}_n)\,.$$

We conclude that $\psi_{\text{out}}(v) \notin \mathcal{L}(\mathcal{C}_n)$, a contradiction.

B. Inclusion "$\supseteq$". Take any w such that $\textsc{Comb}_n\mathtt{A} \in \mathcal{F}^6_{\mathcal{J}}(w)$. Using definition of $\mathcal{F}^6_{\mathcal{J}}(w)$, fix $v \in \mathcal{L}\big(\mathcal{F}^6_{\mathcal{J}}\big)$ such that $\psi_{\text{in}}(v) = w$ and $\psi_{\text{out}}(v) = \textsc{Comb}_n\mathtt{A}$. We have to show $\rho_{\mathcal{J}}^{-1}(w) \subseteq \mathcal{L}\big(\hat{\mathcal{C}}^6_{\mathcal{J}}\big)$. Take any $w' \in \rho_{\mathcal{J}}^{-1}(w)$. Let $u \in (\Sigma_{\mathcal{J}} \times \mathbb{N}_n \times \Phi)^*$ be the unique word such that $\psi_{\text{in}}(u) = w'$ and $\rho_{\mathcal{J}}(u) = v$. Observe that $\psi_{\text{out}}(w') \in \rho_{\mathcal{J}}^{-1}(\psi_{\text{out}}(w')) \subseteq \mathcal{L}(\mathcal{C}_n)$, thus $w' \in \mathcal{L}\big(\hat{\mathcal{C}}^6_{\mathcal{J}}\big)$, as required. ◀

Proof ($i \in \{4, 5\}$). The proof is analogous to the case $i = 6$. As the cases are distinguished by the filter $\mathcal{F}^i_{\mathcal{J}}$ being used, the only differences are related to the shape of words matched by $\psi_{\text{in}}(\mathcal{F}^i_{\mathcal{J}})$. In particular, the set U for $i \in \{4, 5\}$ is now a singleton containing u_i:

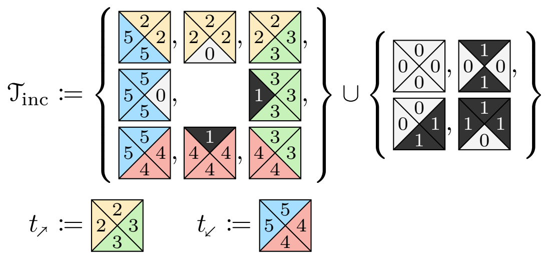
$$u_4 = \langle\, \underline{p}\, t\, \underline{s\, \mathtt{A}}\, \rangle\, \gamma \qquad (i = 4)$$

$$u_5 = \langle\, \underline{p_1}\, t\, \underline{s'_1}\, s_1\, \mathtt{A}\, \rangle \langle\, p_2\, t\, \underline{s'_2}\, s_2\, \mathtt{A}\, \rangle \cdots \langle\, p_{k-1}\, t\, \underline{s'_{k-1}}\, s_{k-1}\, \mathtt{A}\, \rangle \langle\, p_k\, t\, \underline{s'_k\, \mathtt{A}}\, \rangle \rrbracket\, \gamma \qquad (i = 5)$$

The rest of the proof only requires substituting $\mathcal{F}^6_{\mathcal{J}}$ with $\mathcal{F}^4_{\mathcal{J}}$ or $\mathcal{F}^5_{\mathcal{J}}$. ◀

4 NFA of doubly exponential size after universal projection

From the lower bounds established in Section 3.4, it is now easy to construct a family $\left(\mathcal{A}_{(\mathcal{T}_{\text{inc}}, t_{\nearrow}, t_{\swarrow}, n)}\right)_{n \in \mathbb{N}}$ of NFA, each of size $O(n^3)$, such that the smallest NFA after a universal projection step has doubly-exponentially many states. Indeed, let



Intuitively, the colours $0, 1$ vertically represent the counter bits, and horizontally encode the carryover bit. The only valid $(\mathcal{T}_{\text{inc}}, t_{\nearrow}, t_{\swarrow})$-tiling of width n simulates incrementing an $(n-2)$-bit binary counter from 0 to $2^{(n-2)} - 1$; see Figure 1 for an example with $n = 5$. Thus, after a universal projection step, the resulting NFA accepts a single word of length doubly exponential in n.

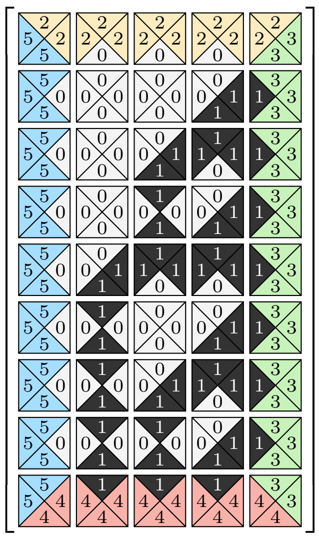


Figure 1 The unique valid $(\mathcal{T}_{\text{inc}}, t_{\nearrow}, t_{\swarrow})$-tiling of width 5.

▶ **Proposition 27.** *The* NFA *for* $\pi^{\forall}_1\left(\mathcal{L}\left(\mathcal{A}_{(\mathcal{T}_{\text{inc}}, t_{\nearrow}, t_{\swarrow}, n)}\right)\right)$ *has size* $\Omega\left(2^{2^n}\right)$.

5 Emptiness after universal projection is in ExpSpace

We now consider algorithmic upper bounds for deciding whether the language of an automatic relation $R \subseteq (\Sigma^*)^{d+k}$ after a universal projection step is non-empty, measured in terms of the size of the associated NFA $\mathcal{A}_R$, which yields the upper bound of Theorem 7.

Define a homomorphism $h\colon (\Sigma_{\#}^{d+k})^* \to (\Sigma_{\#}^{d})^*$ by

$$h(a_1, \dots, a_d, a_{d+1}, \dots, a_{d+k}) \coloneqq (a_1, \dots, a_d).$$

Given an NFA $\mathcal{B}$ over $\Sigma_{\#}^{d+k}$ such that $S \subseteq (\Sigma^*)^{d+k}$ is automatic via $\mathcal{B}$, it is clear that we can compute in linear time an NFA $\mathcal{B}'$ with the same number of states as $\mathcal{B}$ such that $L(\mathcal{B}') = h(\mathcal{L}(\mathcal{B}))$. The homomorphism h acts almost like existential projection, but in general, we do not have that $\pi_d^{\exists}(S)$ is automatic via $\mathcal{B}'$. For instance, suppose that

$$w = \begin{bmatrix} a \\ a \end{bmatrix} \begin{bmatrix} b \\ a \end{bmatrix} \begin{bmatrix} \# \\ c \end{bmatrix} \begin{bmatrix} \# \\ a \end{bmatrix} \in \mathcal{L}(\mathcal{B})\,.$$

Then $h(w) = aa\#\# \notin L_{\checkmark}$ because of the trailing # symbols. To remove them, we define

$$\textit{Strip}(L) \coloneqq \left\{ w \,\middle|\, \text{there exists } v \in (\{\#\}^d)^* \text{ such that } wv \in L \right\}.$$

It is then the case that $\pi_d^{\exists}(S)$ is automatic via $\textit{Strip}(\mathcal{L}(\mathcal{B}')) \cap L_{\checkmark}$. Note that an NFA for $\textit{Strip}(L)$ can be computed in linear time from an NFA for L without changing the set of states by making all states accepting that can reach a final state via a sequence of "$\{\#\}^d$" symbols.

Recall that $\pi_d^{\forall}(R) = \overline{\pi_d^{\exists}(\overline{R})}$, consequently an automatic presentation of $\pi_d^{\forall}(R)$ is given by

$$\overline{\left(\textit{Strip}\left(h\left(\overline{\mathcal{L}(\mathcal{A}_R)}\right)\right) \cap L_{\checkmark}\right)} \cap L_{\checkmark}.$$

Assuming Q is the set of states of $\mathcal{A}_R$, and recalling that $L_{\checkmark} \subseteq (\Sigma_{\#}^d)^*$ is given by an NFA with 2^{d+2} many states, it can easily be checked that the number of states of an NFA whose language gives the universal projection of R is bounded by $2^{\left(2^{|Q|+d+2}\right)+d+2}$.

With those characterisations and estimations at hand, the ExpSpace upper bound stated in Theorem 7 can now easily be established.

▶ **Proposition 28.** *Deciding whether $\pi_d^{\forall}(R) \neq \emptyset$ is in* ExpSpace*, measured in terms of the size of its associated* NFA *$\mathcal{A}_R$.*

Proof. For an ExpSpace algorithm, we first construct an NFA $\mathcal{B} = (Q, \Sigma_{\#}^d, \delta, q_0, F)$ whose language is $\left(\textit{Strip}\left(p_d(\overline{\mathcal{L}(\mathcal{A}_R)})\right) \cap L_{\checkmark}\right)$. We have $|Q| \le 2^{|Q_R|+d+2}$, where Q_R is the set of states of $\mathcal{A}_R$, and hence $\mathcal{B}$ can be constructed in exponential space. It remains to show that non-emptiness of $\overline{\mathcal{L}(\mathcal{B})} \cap L_{\checkmark}$ can be decided in exponential space.

Clearly, we cannot explicitly construct an NFA for this language. Let $\mathcal{A}_{\checkmark} = (S, \Sigma_{\#}^d, \delta_{\checkmark}, s_0, F_{\checkmark})$ be the the NFA for $L_{\checkmark}$, we can however non-deterministically guess a word in $\overline{\mathcal{L}(\mathcal{B})} \cap \mathcal{L}(\mathcal{A}_{\checkmark})$ letter by letter as follows. We keep track of a configuration of the form $(Q', s) \in 2^Q \times S$, which initially is $(\{q_0\}, s_0)$. Then we repeatedly non-deterministically guess some $a \in \Sigma_{\#}^d$ and update (Q', s) to $(\delta(Q', a), \delta_{\checkmark}(s, a))$ until we reach a configuration (Q', s) such that $Q' \cap F = \emptyset$ and $s \in F_{\checkmark}$. Clearly, the word obtained by this sequence of letters is in $\overline{\mathcal{L}(\mathcal{B})}$ and $\mathcal{L}(L_{\checkmark})$. The overall membership in ExpSpace is then a consequence of Savitch's theorem and the observation that the length of the shortest word in $\overline{\mathcal{L}(\mathcal{B})} \cap L_{\checkmark}$ is bounded by $2^{\left(2^{|Q|+d+2}\right)+d+2}$. ◀

6 Conclusion

In this paper, we studied the computational complexity of eliminating universal quantifiers in automatic structures. We showed that, in general, this is a computationally challenging problem whose associated decision problem is ExpSpace-complete. Our result further reinforces the intuition already stemming from [13] that, in general, the alternation of quantifiers requires "complex" automata.

It would be interesting to understand whether it is possible to identify natural sufficient conditions on regular languages for which a universal projection step does not result in a doubly-exponential blow-up and only leads to, e.g., polynomial or singly exponential growth. Results of this kind have been obtained in model-theoretic terms for structures of bounded degree [14, 9], but we are not aware of a systematic study of questions of this kind on the level of regular languages.

References

1 The LASH toolset. `https://people.montefiore.uliege.be/boigelot/research/lash/index.html`.

2 Achim Blumensath and Erich Grädel. Automatic structures. In *Logic in Computer Science, LICS*, pages 51–62. IEEE Computer Society, 2000. `doi:10.1109/LICS.2000.855755`.

3 Bernard Boigelot, Pascal Fontaine, and Baptiste Vergain. First-order quantification over automata. In *Conference on Implementation and Application of Automata, CIAA*, Lect. Notes Comp. Sci. Springer, 2023. To appear. `doi:10.48550/arXiv.2306.04210`.

4 Véronique Bruyère. Entiers et automates finis. *Mémoire de fin d'études*, 1985.

5 Véronique Bruyère, Georges Hansel, Christian Michaux, and Roger Villemaire. Logic and p-recognizable sets of integers. *Bull. Belg. Math. Soc. Simon Stevin*, 1(2):191–238, 1994. `doi:10.36045/bbms/1103408547`.

6 J. Richard Büchi. Weak second-order arithmetic and finite automata. *Math. Logic Quart.*, 6(1-6):66–92, 1960. `doi:10.1002/malq.19600060105`.

7 Dmitry Chistikov and Christoph Haase. On the complexity of quantified integer programming. In *International Colloquium on Automata, Languages, and Programming, ICALP*, volume 80 of *LIPIcs*, pages 94:1–94:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. `doi:10.4230/LIPIcs.ICALP.2017.94`.

8 Dmitry Chistikov, Christoph Haase, and Alessio Mansutti. Geometric decision procedures and the VC dimension of linear arithmetic theories. In *Logic in Computer Science, LICS*, pages 59:1–59:13. ACM, 2022. `doi:10.1145/3531130.3533372`.

9 Antoine Durand-Gasselin and Peter Habermehl. Ehrenfeucht-fraïssé goes elementarily automatic for structures of bounded degree. In *Symposium on Theoretical Aspects of Computer Science, STACS*, volume 14 of *LIPIcs*, pages 242–253. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2012. `doi:10.4230/LIPIcs.STACS.2012.242`.

10 Seymour Ginsburg and Edwin H. Spanier. Semigroups, Presburger formulas, and languages. *Pac. J. Math.*, 16(2):285–296, 1966.

11 Bernard R. Hodgson. On direct products of automaton decidable theories. *Theor. Comput. Sci.*, 19(3):331–335, 1982. `doi:10.1016/0304-3975(82)90042-1`.

12 Bakhadyr Khoussainov and Anil Nerode. Automatic presentations of structures. In *Logical and Computational Complexity, LCC*, volume 960 of *Lect. Notes Comp. Sci.*, pages 367–392. Springer, 1995. `doi:10.1007/3-540-60178-3_93`.

13 Dietrich Kuske. Theories of automatic structures and their complexity. In *Conference on Algebraic Informatics, CAI*, volume 5725 of *Lect. Notes Comp. Sci.*, pages 81–98. Springer, 2009. `doi:10.1007/978-3-642-03564-7_5`.

14 Dietrich Kuske and Markus Lohrey. Automatic structures of bounded degree revisited. *J. Symb. Log.*, 76(4):1352–1380, 2011. `doi:10.2178/jsl/1318338854`.

15 Jérôme Leroux and Gérald Point. TaPAS: The Talence Presburger Arithmetic Suite. In *Tools and Algorithms for the Construction and Analysis of Systems, TACAS*, volume 5505 of *Lect. Notes Comp. Sci.*, pages 182–185. Springer, 2009. `doi:10.1007/978-3-642-00768-2_18`.

16 Hamoon Mousavi. Automatic theorem proving in Walnut. *CoRR*, 2016. `arXiv:1603.06017`.

17 Mojżesz Presburger. Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt. In *Comptes Rendus du I congres de Mathematiciens des Pays Slaves*, pages 92–101. 1929.

18 François Schwarzentruber. The complexity of tiling problems, 2019. `arXiv:1907.00102`.

19 Ken Thompson. Programming techniques: Regular expression search algorithm. *Commun. ACM*, 11(6):419–422, 1968. `doi:10.1145/363347.363387`.