

Weakly Acyclic Automata and Exponential Time Hypotheses: Universality and Related Problems

Jiří Balun, Tomáš Masopust, Petr Osička

Faculty of Science, Palacky University Olomouc, 17. listopadu 12, Olomouc, 779 00, Czechia

Abstract

A nondeterministic finite automaton (NFA) is weakly acyclic if the only cycles in its transition graph are self-loops. Universality for NFAs is PSPACE-complete, which rules out polynomial-time algorithms unless PSPACE = P. This raises the question of whether faster, in particular subexponential-time, algorithms are possible. We show that even for highly restricted subclasses of weakly acyclic NFAs, universality admits no algorithm running in time $2^{o(n)}$ under the Exponential Time Hypothesis (ETH), already over a binary alphabet, and no algorithm running in time $O^*(2^{n/(2+\epsilon)})$ for any $\epsilon > 0$ under the Strong Exponential Time Hypothesis (SETH), over an alphabet growing with the input. Both bounds match the known lower bounds for general NFAs. We further establish analogous lower bounds for related problems, including language inclusion and equivalence, length-restricted universality, and the shortest non-accepted string problem. Our results show that the exponential complexity of the problems is inherent and does not arise from the presence of nontrivial cycles.

Keywords: Weakly acyclic automata, universality, exponential time hypothesis, conditional lower bound, complexity.
2000 MSC: 68Q45, 68Q17, 68Q25

1. Introduction

Language inclusion, equivalence, and universality are fundamental problems in automata theory, with applications across computer science. For nondeterministic finite automata (NFAs), these problems are PSPACE-complete [1]. Assuming that PSPACE $\neq$ P, a natural question is whether these problems admit more efficient, in particular subexponential-time, algorithms. While various techniques improve the practical runtime of algorithms [2, 3, 4, 5, 6, 7], their core relies on NFA determinization, which leads to exponential worst-case complexity. Indeed, the determinization problem is EXPTIME-hard [8], and deciding whether the subset construction yields an exponential blow-up in the number of states is itself PSPACE-hard [9]. The absence of subexponential-time algorithms for these problems has been formalized via conditional lower bounds. Fernau and Krebs [10] showed that, under the Exponential Time Hypothesis (ETH), no algorithm can decide universality for n -state NFAs in time $2^{o(n)}$. This result was later strengthened under the Strong Exponential Time Hypothesis (SETH), ruling out algorithms with running time $O^*(2^{n/(2+\epsilon)})$ for any $\epsilon > 0$ [11]. Oliveira and Wehar [12] analyzed the DFA intersection non-emptiness problem under NC-SETH, implying a stronger result for the DFA union universality problem.

In this paper, we revisit these questions for weakly acyclic NFAs and their variants. An automaton is *weakly acyclic* (or *partially ordered*, poNFA) if its transition relation induces a partial order on states, that is, if the only cycles in its transition diagram are self-loops. A poNFA is *self-loop deterministic* (rpoNFA, from restricted poNFA) if it has no nondeterministic choice between self-loop and non-self-loop transitions. Self-loop deterministic weakly acyclic automata arise naturally in several settings. They play a key role in weakly acyclic diagrams (WADs), which generalize OBDDs to represent infinite languages [13]; in XML schema languages, where their languages can be defined by

Email addresses: jiri.balun01@upol.cz (Jiří Balun), tomas.masopust@upol.cz (Tomáš Masopust), petr.osicka@upol.cz (Petr Osička)

Table 1: Deciding universality for the classes of automata considered in this paper; n denotes the number of states of the input automaton. An entry in the column ETH means that no algorithm running in time $2^{o(n)}$ exists unless ETH fails; an entry in the column SETH means that no algorithm running in time $O^*(2^{n/(2+\epsilon)})$, for any $\epsilon > 0$, exists unless SETH fails. A dash indicates that no such bound is known.

Class	Alphabet	Universality	ETH	SETH
NFA	fixed	PSPACE-C [1]	[10]	–
NFA	growing	PSPACE-C [1]	[10]	[11]
poNFA	fixed	PSPACE-C [14]	[10]	–
poNFA	growing	PSPACE-C [14]	[10]	Thm 19
rpoNFA	fixed	coNP-c [14]	Thm 5	–
rpoNFA	growing	PSPACE-C [14]	Thm 5	Thm 19
ptNFA	fixed	coNP-c [29]	Thm 5	–
ptNFA	growing	PSPACE-C [29]	Thm 5	–

deterministic (one-unambiguous) regular expressions [14]; in regular model checking, where they are used to model and verify properties of distributed systems [15]; in language separation problems [16, 17]; and in discrete-event systems, as simple deadlock-free models for verification [18].

Brzozowski and Fich [19] studied partially ordered DFAs (poDFAs) and showed that they characterize $\mathcal{R}$ -trivial languages. Extending the notion to NFAs, Schwentick, Thérien, and Vollmer [20] proved that poNFAs characterize level 3/2 of the Straubing-Thérien hierarchy [21, 22], an infinite hierarchy of star-free languages with connections to the dot-depth hierarchy [23, 24, 25], to first-order logic [26], and to complexity theory [27]. The languages of poNFAs are also known as *alphabetical pattern constraints* (APC), i.e., regular languages effectively closed under permutation rewriting [15]. Consequently, poNFAs are strictly more expressive than poDFAs, and the natural nondeterministic counterpart of poDFAs is given by self-loop deterministic poNFAs (rpoNFAs) [14].

Level 1 of the Straubing-Thérien hierarchy is known as *piecewise testable languages*, introduced by Simon [28]. These languages are characterized by $\mathcal{J}$ -trivial syntactic monoids and by confluent poDFAs. The corresponding nondeterministic model is given by complete and confluent rpoNFAs, also known as ptNFAs [29].

Despite their restricted structure, deciding universality for poNFAs and their variants discussed above remains PSPACE-complete. For poNFAs, the result holds even over a binary alphabet. However, for rpoNFAs and ptNFAs over a fixed alphabet, the problem drops to coNP-complete [14, 29].

Since universality is a special case of language inclusion and equivalence (with Σ^*), these complexity bounds carry over to the inclusion and equivalence problems as well [29].

1.1. Our contributions

The main result of this paper is that the exponential complexity of universality persists under strong structural restrictions: weakly acyclic NFAs do not admit subexponential-time algorithms under standard complexity assumptions.

Technically, our contributions consist of two reductions with a linear overhead in the number of states, together with the consequences we derive from them. The reductions are purely combinatorial, the lower bounds they imply are conditional on ETH or SETH. Table 1 summarizes how the resulting picture relates to what was known before.

Besides universality itself, two refinements recur throughout the paper, and we introduce them here.

First, we pay particular attention to automata in which *all states are accepting*. For general NFAs, universality remains PSPACE-complete under this restriction [30]; the known reduction, however, introduces nontrivial cycles and therefore does not apply in the weakly acyclic setting. All lower bounds established below hold already for automata with all states accepting, which makes them as strong as possible in this respect.

Second, following Gawrychowski et al. [31], we consider two length-restricted variants of universality. Given an automaton $\mathcal{A}$ over an alphabet Σ and an integer ℓ encoded in binary, *specified-length universality* asks whether $\Sigma^\ell \subseteq L(\mathcal{A})$, and *bounded-length universality* asks whether $\Sigma^{\leq \ell} \subseteq L(\mathcal{A})$. The latter is closely related to the problem of determining the length of the shortest non-accepted string: an integer ℓ is the length of the shortest non-accepted string of a language L if and only if ℓ is the maximum number such that $\Sigma^{\leq \ell-1} \subseteq L$. A dual problem, asking whether

a given automaton accepts some string of a given length, can be solved in time $O(n^\omega (\log n)^{1+\epsilon} \log \ell)$, where ω is the exponent of matrix multiplication [32].

Main results.

- From 3-COLORING on a graph with n vertices and m edges, we construct in time $O(n+m)$ an rpoNFA with $4n-2$ states, all accepting, and a ptNFA with $5n-3$ states, both over a three-letter alphabet, which are universal if and only if the graph admits no proper 3-coloring (Proposition 4). A binary encoding of the alphabet preserves the linear number of states. The ptNFA moreover accepts every string whose length differs from n , which is what makes the length-restricted variants accessible (Corollary 9).
- From a k -CNF formula with n variables and m clauses, we construct in polynomial time an rpoNFA with $2n+1$ states, all accepting, over an alphabet of size $(n+1)m$, which is universal if and only if the formula is satisfiable (Proposition 18). If the formula is unsatisfiable, then the shortest string not accepted by the automaton has length 2^n .

The second reduction relies on a construction of independent interest: an rpoNFA with $2n+1$ states whose configurations, driven by a Zimin string, enumerate all 2^n binary counter values in increasing order (Section 4.1).

Consequences for exponential hardness.

- Unless ETH fails, universality of n -state ptNFAs, and of n -state rpoNFAs with all states accepting, cannot be decided in time $2^{o(n)}$, even over a binary alphabet (Theorem 5). The same lower bound therefore applies to poNFAs and to NFAs.
- Unless SETH fails, universality of n -state rpoNFAs with all states accepting cannot be decided in time $O^*(2^{n/(2+\epsilon)})$ for any $\epsilon > 0$ (Theorem 19). Here the alphabet grows with the input; see Remark 20.
- Both bounds carry over to language inclusion and equivalence (Corollaries 6 and 21), to specified-length and bounded-length universality (Corollaries 9 and 22), and, under SETH, to determining the length of the shortest non-accepted string (Corollary 23).

Complexity results.

- Requiring all states to be accepting does not change the complexity of universality for weakly acyclic automata: for poNFAs and for rpoNFAs, the complexity of the restricted problem coincides with the complexity of the unrestricted one (Theorem 24).
- Specified-length and bounded-length universality are PSPACE-complete for weakly acyclic NFAs over a binary alphabet and for rpoNFAs and ptNFAs over a growing alphabet (Lemma 7). For fixed alphabets, the complexity of both problems drops to coNP-complete for rpoNFAs and ptNFAs (Lemma 8).

Consequences for descriptonal complexity.

- There are n -state rpoNFAs with all states accepting whose shortest non-accepted string has length $2^{(n-1)/2}$.

Relation to previous work. While the ETH-based lower bound of Fernau and Krebs [10] already applies to weakly acyclic NFAs, other existing bounds do not. In particular, the results of Oliveira and Wehar [12], based on Kozen's PSPACE-hardness proof of DFA-union universality [33], rely on nontrivial cycles that are essential for the construction; if all cycles are restricted to self-loops, the problem becomes easier and is known to be coNP-complete [29]. This paper builds on the results of [11], substantially revising and extending them. In particular, all results concerning universality also apply to the notions of opacity studied therein.

Considering the descriptonal complexity result, the best previous bound is by Masopust and Krötzsch [29], who constructed, for all $k, n \geq 1$, a ptNFA over an n -letter alphabet with $n(2k+1)+1$ states whose unique non-accepted string has length $\binom{k+n}{k} - 1$; for a given number N of states, this quantity is maximized for $k = n$, and is then $2^{\Theta(\sqrt{N})}$. Since ptNFAs are complete and confluent rpoNFAs, the same bound holds for rpoNFAs, where it was also obtained

directly [14]. For weakly acyclic NFAs over a five-letter alphabet, Ellul et al. [34] obtained $2^{n/25}$, and $2^{n/75}$ after a binary encoding that destroys weak acyclicity, the latter also holds with all states accepting [30]. Our construction improves the exponent from $\Theta(\sqrt{n})$ to $\Theta(n)$, at the price of an alphabet growing with the input and of a weaker combinatorial property: the automata of [29] reject a single string, whereas ours rejects infinitely many.

2. Preliminaries

We assume familiarity with automata theory [35]. For a set S , its cardinality is denoted by $|S|$, and its power set by 2^S . If $S = \{x\}$ is a singleton, we often write x instead of $\{x\}$. An alphabet Σ is a finite nonempty set of letters. A string over Σ is a finite sequence of letters from Σ . The set of all strings over Σ is denoted by Σ^* ; the empty string is denoted by ε . A language over Σ is a subset of Σ^* . For a string $u \in \Sigma^*$, its length is denoted by $|u|$. We denote by Σ^ℓ the set of strings over Σ of length ℓ , and by $\Sigma^{\leq \ell}$ and $\Sigma^{\geq \ell}$ the sets of strings of length at most ℓ and at least ℓ , respectively.

2.1. Automata

A *nondeterministic finite automaton* (NFA) is a structure $\mathcal{A} = (Q, \Sigma, \delta, I, F)$, where Q is a finite set of states, Σ is an input alphabet, $I \subseteq Q$ is a set of initial states, $F \subseteq Q$ is a set of accepting states, and $\delta: Q \times \Sigma \rightarrow 2^Q$ is a transition function, extended to $2^Q \times \Sigma^*$ in the standard way. For $Y, Y' \subseteq Q$, $p, q \in Q$, and $w \in \Sigma^*$, we write $Y \xrightarrow{w} Y'$ for $\delta(Y, w) = Y'$ and $p \xrightarrow{w} q$ for $q \in \delta(p, w)$. A *run* on w from q_0 is a sequence $q_0 \xrightarrow{w_1} q_1 \xrightarrow{w_2} q_2 \xrightarrow{w_3} \dots \xrightarrow{w_k} q_k$, where $q_i \in Q$, $w_i \in \Sigma^* - \{\varepsilon\}$ for all i , and $w = w_1 w_2 \dots w_k$. The run is *accepting* if $q_0 \in I$ and $q_k \in F$.

The automaton $\mathcal{A}$ is *complete* if, for every letter $a \in \Sigma$ and every state $q \in Q$, the set $\delta(q, a)$ is nonempty. The language accepted by $\mathcal{A}$ is $L(\mathcal{A}) = \{w \in \Sigma^* \mid \delta(I, w) \cap F \neq \emptyset\}$. The NFA $\mathcal{A}$ is *deterministic* (DFA) if $|I| = 1$ and $|\delta(q, a)| = 1$ for every state $q \in Q$ and every letter $a \in \Sigma$. We define DFAs to be complete.

The reachability relation $\leq$ on Q is defined by $p \leq q$ if there exists a string $w \in \Sigma^*$ such that $q \in \delta(p, w)$. An NFA is *partially ordered* (poNFA) if $\leq$ is a partial order. A poNFA with the property that $q \in \delta(q, a)$ implies $\delta(q, a) = \{q\}$ for every state $q \in Q$ and every letter $a \in \Sigma$ is *self-loop deterministic* or *restricted partially ordered* (rpoNFA). The same definitions apply to DFAs, resulting in partially ordered DFAs (poDFAs). An automaton is *confluent* if, for every state $q \in Q$ and every pair of (not necessarily distinct) letters $a, b \in \Sigma$, whenever $s \in \delta(q, a)$ and $t \in \delta(q, b)$, there exists a string $w \in \{a, b\}^*$ such that $\delta(s, w) \cap \delta(t, w) \neq \emptyset$. An rpoNFA is a *ptNFA* if it is complete and confluent. Analogously, we define confluent poDFAs, which are complete by definition. Note that every ptNFA is an rpoNFA and every rpoNFA is a poNFA.

The *depth* of a poNFA $\mathcal{A}$ is the number of transitions on a longest simple path in the underlying graph of $\mathcal{A}$ starting in an initial state. Recall that a path is simple if all states on the path are pairwise distinct. The *configuration* of an NFA $\mathcal{A}$ is a state of the DFA obtained from $\mathcal{A}$ by the standard subset construction [35].

A string $r_n r_{n-1} \dots r_1 \in \{0, 1\}^n$ is the binary representation of a natural number ℓ if $\ell = (r_n r_{n-1} \dots r_1)_2 = \sum_{i=1}^n r_i \cdot 2^{i-1}$.

2.2. CNF-SAT and exponential time hypotheses

A (*Boolean*) *formula* consists of variables, logical connectives conjunction ($\wedge$), disjunction ($\vee$), and negation ($\neg$), and parentheses. A *literal* is a variable or its negation. A *clause* is a disjunction of literals. A formula is in *conjunctive normal form* (CNF) if it is a conjunction of clauses. If each clause has at most k literals, then the formula is in k -CNF. A formula is *satisfiable* if there exists an assignment of truth values to the variables that evaluates the formula to true. Given $k \geq 3$, the k -CNF *Boolean satisfiability problem* (k -SAT) asks whether a given k -CNF formula is satisfiable. If the formula has n variables, then enumerating all 2^n assignments yields an $O(2^n n^k)$ -time algorithm for k -SAT, where the polynomial factor accounts for checking all clauses. We use O^* to suppress polynomial factors, that is, $O^*(g(n)) = O(g(n) \cdot \text{poly}(n))$ for $g(n) \in \omega(\text{poly}(n))$.

Exponential time hypotheses are motivated by the observation that no algorithm is known that solves SAT in the worst case significantly faster than exhaustive search [36]. The exponential time hypothesis states that 3-SAT cannot be solved in subexponential time $2^{o(n)}$, where n is the number of variables in the input formula [36].

Hypothesis 1 (Exponential Time Hypothesis (ETH)). *There exists $\epsilon > 0$ such that 3-SAT cannot be solved in time $O(2^{\epsilon n})$, where n is the number of variables in the formula.*

This formulation still allows algorithms with running time $O(c^n)$ for $c < 2$. The currently fastest 3-SAT algorithm runs in time $O^*(1.30704^n)$ [37, 38]. For larger k , the best known algorithms for k -SAT become slower; for example, the fastest known 4-SAT algorithm runs in time $O^*(1.46899^n)$ [38]. This motivates the strong exponential time hypothesis, which asserts that for every $c < 2$ there exists k such that k -SAT cannot be solved in time $O(c^n)$ [36].

Hypothesis 2 (Strong Exponential Time Hypothesis (SETH)). *For every $\epsilon > 0$, there exists $k \geq 3$ such that k -SAT cannot be solved in time $O(2^{(1-\epsilon)n})$, where n is the number of variables in the formula.*

Both hypotheses imply $P \neq NP$, and SETH implies ETH.

3. Hardness under the Exponential Time Hypothesis

We recall the classical 3-COLORING problem. A 3-coloring of an undirected graph $G = (V, E)$ is a mapping $\mu: V \rightarrow \{a, b, c\}$. The coloring is *proper* if $\mu(u) \neq \mu(v)$ for every edge $\{u, v\} \in E$. The problem asks whether a given graph admits a proper 3-coloring.

Theorem 3 ([10, Theorem 1], [39, Theorem 14.6]). *Unless ETH fails, 3-COLORING on graphs with n vertices and m edges cannot be solved in time $2^{o(n+m)}$.*

Fernau and Krebs [10] showed how to reduce 3-COLORING on a graph with n vertices and m edges to the universality problem for NFAs with $\Theta(n+m)$ states over a binary alphabet. Combined with Theorem 3, this implies that universality for N -state NFAs over a binary alphabet admits no algorithm running in time $2^{o(N)}$, unless ETH fails.

We show that this lower bound can be extended to restricted classes of weakly acyclic automata. The technical core is the following reduction, which is obtained by a modification of the construction of Fernau and Krebs. The reduction records the size of the resulting automata, which is what the consequences depend on.

Proposition 4. *There is an algorithm that, given a graph G with $n \geq 2$ vertices and $m \geq 1$ edges, constructs in time $O(n+m)$ the following two automata over a three-letter alphabet Σ .*

- (a) *An rpoNFA $\mathcal{A}$ with all states accepting, with $4n - 2$ states and $12n + 3m - 15$ transitions, such that a string $w \in \Sigma^*$ is not accepted by $\mathcal{A}$ if and only if $|w| \geq n$ and the prefix of w of length n encodes a proper 3-coloring of G .*
- (b) *A ptNFA $\mathcal{A}'$ with $5n - 3$ states and $15n + 3m - 9$ transitions that accepts every string of length different from n , and such that a string $w \in \Sigma^n$ is not accepted by $\mathcal{A}'$ if and only if w encodes a proper 3-coloring of G .*

In particular, each of $\mathcal{A}$ and $\mathcal{A}'$ is universal if and only if G has no proper 3-coloring. Moreover, automata with the same properties can be constructed over a binary alphabet at the cost of increasing the number of states by a constant factor.

Proof. Given a graph $G = (V, E)$ with vertices $v_1, v_2, \dots, v_n$ and $m \geq 1$ edges. We construct an automaton $\mathcal{A}$ with $\Theta(n)$ states and $\Theta(n+m)$ transitions such that $\mathcal{A}$ is universal if and only if G is not 3-colorable.

- (a) Let $\Sigma = \{a, b, c\}$, where letters represent colors. A string $w = w_1 \dots w_n \in \Sigma^n$ encodes a coloring of G by setting $\mu(v_i) = w_i$. The automaton $\mathcal{A} = (Q, \Sigma, \delta, I, F)$ is defined as follows. The set of states is

$$Q = \{f\} \cup \{q_1, \dots, q_n\} \cup \{x_i \mid x \in \{a, b, c\}, 1 \leq i \leq n-1\}$$

and the transition relation consists of

- $q_i \xrightarrow{x} q_{i+1}$ for $i = 1, \dots, n-1$ and all $x \in \Sigma$,
- $x_i \xrightarrow{y} x_{i-1}$ for $i = 2, \dots, n-1$ and all $x, y \in \Sigma$,
- $x_1 \xrightarrow{x} f$ for all $x \in \Sigma$,
- $f \xrightarrow{x} f$ for all $x \in \Sigma$, and
- for each edge $\{v_i, v_j\} \in E$ with $i < j$, $q_i \xrightarrow{x} x_{j-i}$ for all $x \in \Sigma$.

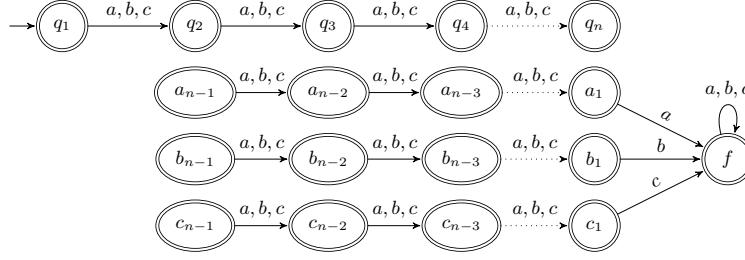


Figure 1: Main part of the rpoNFA from the proof of Proposition 4(a), without transitions corresponding to the edges of G ; all states are accepting.

We set $I = \{q_1\}$ and $F = Q$, see Figure 1.

Intuitively, the automaton simulates a coloring of the vertices and uses the auxiliary states x_i to verify consistency of colors along edges. Let $w = w_1 \cdots w_k \in \Sigma^*$ with $w_i \in \Sigma$. If $k < n$, then the run

$$q_1 \xrightarrow{w_1} q_2 \xrightarrow{w_2} q_3 \xrightarrow{w_3} \cdots \xrightarrow{w_k} q_{k+1} \quad (1)$$

is accepting. Hence, we may assume that $k \geq n$. In this case, no run on w starting from q_1 can remain entirely within the states $\{q_1, \dots, q_n\}$. Instead, every run has a prefix of the form

$$q_1 \xrightarrow{w_1 \cdots w_{i-1}} q_i \xrightarrow{w_i} x_{j-i} \xrightarrow{w_{i+1} \cdots w_{j-1}} x_1, \quad (2)$$

where $\{v_i, v_j\} \in E$ with $i < j$, and $x = w_i$. By construction, we have $w_i = w_j$ if and only if the transition $x_1 \xrightarrow{w_j} f$ is enabled, which is in turn equivalent to the existence of an accepting run on w (continuing as $f \xrightarrow{w_{j+1} \cdots w_k} f$). It follows that a string $w_1 \cdots w_n$ is rejected if and only if it encodes a proper 3-coloring of G . Consequently, G admits a proper 3-coloring if and only if $\mathcal{A}$ is not universal.

The automaton has $4n - 2$ states and $12n + 3m - 15$ transitions. It is weakly acyclic and self-loop deterministic; in particular, its only cycle is the self-loop on f . Since all states are accepting, $\mathcal{A}$ is an rpoNFA with the required properties.

(b) We now modify the construction to obtain a ptNFA. Note that part (a) cannot be directly adapted while keeping all states accepting, since ptNFAs are complete by definition and any complete automaton with all states accepting is trivially universal. To ensure completeness and confluence, we extend the automaton by adding new states $s_2, \dots, s_n$ and the following transitions (see Figure 2):

- $q_n \xrightarrow{x} s_n$ for all $x \in \Sigma$,
- $x_1 \xrightarrow{y} s_2$ for all $x, y \in \Sigma$ such that $x \neq y$,
- $s_n \xrightarrow{x} f$ for all $x \in \Sigma$, and
- $s_i \xrightarrow{x} s_{i+1}$ for $i = 2, \dots, n-1$ and all $x \in \Sigma$.

The state f acts as a sink, which ensures confluence. The automaton is complete by construction. The set of accepting states is $F = \{q_1, \dots, q_n\} \cup \{f\}$.

Let $w = w_1 \cdots w_k \in \Sigma^*$ with $w_i \in \Sigma$. If $k < n$, then the run (1) is accepting. If $k > n$, then the run extends as $q_1 \xrightarrow{w_1 \cdots w_{n-1}} q_n \xrightarrow{w_n} s_n \xrightarrow{w_{n+1}} f \xrightarrow{w_{n+2} \cdots w_k} f$, and is therefore accepting. It remains to consider the case $k = n$. In this case, the extension $q_n \xrightarrow{w_n} s_n$ leads to a non-accepting state. All other runs have a prefix of the form (2). If $w_i \neq w_j$, then the run continues as $x_1 \xrightarrow{w_j} s_2 \xrightarrow{w_{j+1} \cdots w_n} s_{n-j+2}$, and is not accepting. If, on the other hand, $w_i = w_j$, then the run continues as $x_1 \xrightarrow{w_j} f \xrightarrow{w_{j+1} \cdots w_n} f$, and is accepting. Hence, w is rejected if and only if it encodes a proper 3-coloring of G . Consequently, G admits a proper 3-coloring if and only if $\mathcal{A}$ is not universal. The automaton has $5n - 3$ states and $15n + 3m - 9$ transitions, and is a complete and confluent rpoNFA, i.e., a ptNFA.

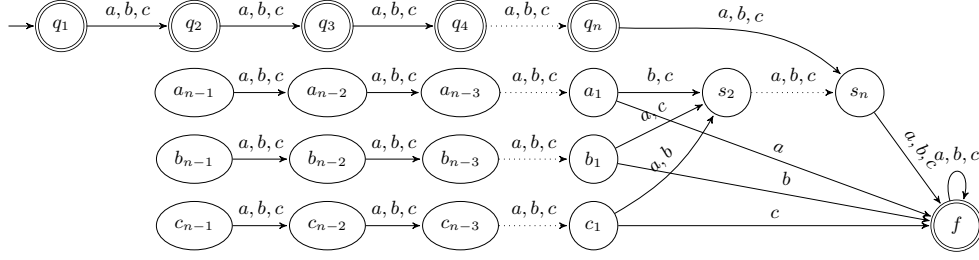


Figure 2: Main part of the ptNFA from the proof of Proposition 4(b), without transitions corresponding to the edges of G .

Finally, we encode the construction over a binary alphabet. We introduce a fresh letter a' duplicating the transitions of a , and use the encoding $a = 00$, $a' = 01$, $b = 10$, and $c = 11$. To avoid introducing nontrivial cycles, the self-loops on f are replaced by self-loops under 0 and 1. The encoding increases the number of states only by a constant factor (at most two new states for every existing state). The acceptance condition is adjusted accordingly. In case (a), all states are accepting. In case (b), a run corresponding to a run that visits only states from $\{q_1, \dots, q_n, s_n, f\}$ in the original automaton is accepting if and only if its length is different from $2n$. $\square$

The announced lower bounds now follow by combining Proposition 4 with Theorem 3.

Theorem 5. *Unless ETH fails, there is no algorithm that decides the following problems for n -state automata in time $2^{o(n)}$.*

- (a) *Universality of rpoNFAs with all states accepting.*
- (b) *Universality of ptNFAs.*

The result holds even if the automata are over a binary alphabet.

Proof. Let G be a graph with n vertices and m edges, and let $\mathcal{A}$ be the automaton provided by Proposition 4; it has $N = \Theta(n)$ states and is universal if and only if G has no proper 3-coloring. Assume that universality of N -state automata of the respective class can be decided in time $2^{o(N)}$. Constructing $\mathcal{A}$ takes time $O(n+m)$, and $2^{o(N)} = 2^{o(n)} \subseteq 2^{o(n+m)}$. Hence 3-COLORING would be solvable in time $2^{o(n+m)}$, contradicting Theorem 3. The claim over a binary alphabet follows from the last part of Proposition 4, since the number of states grows only by a constant factor. $\square$

3.1. Inclusion, equivalence, and length-restricted universality

Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be NFAs over the same alphabet Σ . The *language inclusion problem* is to decide whether $L(\mathcal{A}_1) \subseteq L(\mathcal{A}_2)$. The *language equivalence problem* is to decide whether $L(\mathcal{A}_1) = L(\mathcal{A}_2)$. Both problems are closely related to universality. In particular, universality reduces to inclusion (by testing $\Sigma^* \subseteq L(\mathcal{A})$) and equivalence (by testing $L(\mathcal{A}) = \Sigma^*$). Consequently, the lower bounds of Theorem 5 immediately carry over to language inclusion and equivalence.

Corollary 6. *Unless ETH fails, language inclusion and equivalence for ptNFAs (and for rpoNFAs with all states accepting) over a binary alphabet cannot be decided in time $2^{o(n)}$, where n denotes the maximum number of states of the input automata.* $\square$

We next consider two variants of universality introduced by Gawrychowski et al. [31]. Given an automaton $\mathcal{A}$ over an alphabet Σ and an integer ℓ encoded in binary, the *specified-length universality* problem asks whether $\Sigma^\ell \subseteq L(\mathcal{A})$, while the *bounded-length universality* problem asks whether $\Sigma^{\leq \ell} \subseteq L(\mathcal{A})$. Both problems are known to be PSPACE-complete for NFAs. The hardness can be adapted to weakly acyclic automata, yielding the following.

Lemma 7. *Specified-length and bounded-length universality are PSPACE-complete for poNFAs, rpoNFAs, and ptNFAs. The hardness holds already for poNFAs over a binary alphabet.*

Proof sketch. The PSPACE-hardness for NFAs shown by Gawrychowski et al. [31] builds on the classical construction of regular expressions that accept all invalid or rejecting computations of a polynomial-space bounded deterministic Turing machine [1, Lemma 10.2]. The construction is arranged so that the underlying computation either halts or loops forever. The resulting regular expressions can be translated into poNFAs, which yields PSPACE-hardness for specified-length universality for poNFAs. Moreover, the hardness already holds over a binary alphabet by the construction of Krötzsch et al. [14], who further provide a construction of rpoNFAs from the regular expressions. Since the assumptions on the Turing machine used in [31] do not affect the argument, the result extends to rpoNFAs over a growing alphabet.

Finally, in [29], a ptNFA $\mathcal{A}$ is constructed from a given polynomial-space bounded deterministic Turing machine and its input such that $\mathcal{A}$ accepts all strings whose length differs from $\binom{2^n}{n} - 1$, for some n specified in the proof. Consequently, the only length at which $\mathcal{A}$ may reject a string is $\ell = \binom{2^n}{n} - 1$. Hence, $\mathcal{A}$ is universal if and only if $\Sigma^\ell \subseteq L(\mathcal{A})$. The same reasoning applies to bounded-length universality, since the existence of a rejecting string of length ℓ implies the failure of universality over $\Sigma^{\leq \ell}$. $\square$

For fixed alphabets, the complexity decreases.

Lemma 8. *For rpoNFAs and ptNFAs over a fixed alphabet, specified-length and bounded-length universality are coNP-complete.*

Proof. For hardness, we use Proposition 4(b), which provides a polynomial-time reduction from the complement of 3-COLORING. The parameter ℓ is set to the number of vertices. By the reduction, a graph admits a proper 3-coloring if and only if there exists a string of length ℓ that is not accepted by the constructed automaton.

For membership in coNP, let $\mathcal{A}$ be an rpoNFA over a fixed alphabet Σ of size c , and let k denote its depth. We may assume that $\mathcal{A}$ is complete by adding a sink state if necessary. By [14, Theorem 8], the language $L(\mathcal{A})$ is a union of $\sim_k^{\mathcal{R}}$ -classes, where $\sim_k^{\mathcal{R}}$ is a congruence of finite index. Each equivalence class contains a unique shortest representative, whose length is bounded by $\lambda = \binom{k+c}{k} - 1 = O(k^c)$ [19, 40]. Since c is fixed and k is at most the number of states of $\mathcal{A}$, the bound λ is polynomial in the number of states. Therefore, if there exists a string not in $L(\mathcal{A})$, then there exists such a string of length at most λ . A nondeterministic algorithm can guess such a string and verify in polynomial time that it is not accepted by $\mathcal{A}$, which shows membership in coNP.

The argument directly extends to ptNFAs, which form a subclass of rpoNFAs. $\square$

Finally, we consider the case where the length parameter is bounded by a linear function of the number of states.

Corollary 9. *Let $c \geq 5$ be a constant. For n -state ptNFAs (rpoNFAs, NFAs) over a fixed alphabet, deciding whether $\Sigma^{n/c} \subseteq L(\mathcal{A})$ is coNP-hard and cannot be solved in time $2^{o(n)}$ unless ETH fails. The same holds for $\Sigma^{\leq n/c}$ and $\Sigma^{\geq n/c}$.*

Proof. Membership in coNP follows since a violating string has length n/c , which is polynomial in the number of states and can be guessed and verified in polynomial time.

For hardness and the ETH-based lower bound, we use Proposition 4(b). It yields an automaton $\mathcal{A}$ with $5N - 3$ states such that a graph G has a proper 3-coloring if and only if there exists a string of length N not accepted by $\mathcal{A}$. Moreover, all strings of lengths different from N are accepted.

To enforce $N = \lfloor n/c \rfloor$, we add k unreachable states to $\mathcal{A}$, which does not affect its language. Specifically, we add states $p_1, \dots, p_k$ together with transitions $p_i \xrightarrow{x} f$ for all $x \in \Sigma$. The total number of states becomes $n = 5N - 3 + k$. Choosing k as the smallest integer such that $\lfloor \frac{n}{c} \rfloor = N$ yields $k \geq cN - 5N + 3 = O(N)$. Thus, the padding increases the size only linearly and preserves the correctness of the reduction.

The same argument applies to the bounded-length variants. $\square$

With a small modification, the above corollary holds also for automata over a binary alphabet. Since the binary version of the automaton $\mathcal{A}'$ of Proposition 4(b) has a higher number of states, the constant c has to be at least 10 in that case.

4. Hardness under Strong Exponential Time Hypothesis

We now strengthen the previous results under the Strong Exponential Time Hypothesis (SETH), showing that universality for rpoNFAs cannot be decided significantly faster than by exhaustive search. The proof proceeds in two steps. First, we construct an rpoNFA $\mathcal{A}_n$ whose configurations simulate a binary counter from 0 to $2^n - 1$. The counter value is encoded in the current configuration, that is, in the subset of active states of $\mathcal{A}_n$. Moreover, the automaton accepts all strings except those having the specific prefix $Z_n a_{n+1}$, i.e., it rejects exactly the language $Z_n a_{n+1} \Sigma_{n+1}^*$, where Z_n is a Zimin string.

In the second step, we use this automaton to reduce a k -SAT formula φ with n variables to the universality problem for an rpoNFA $\mathcal{A}_\varphi$. The counter values are interpreted as truth assignments, and additional transitions are used to verify whether clauses are satisfied.

Since the argument is somewhat involved, we first mention four facts that drive it; each of them is stated formally later in this section, and we indicate where.

1. Reading the Zimin string Z_n from the initial configuration, the automaton $\mathcal{A}_n$ passes through the 2^n configurations that encode the counter values $0, 1, \dots, 2^n - 1$, in this order (Lemma 12). Reading any letter other than the expected one immediately activates the sink state q_+ , and reading the terminating letter a_{n+1} in the last configuration empties the configuration; consequently, $\mathcal{A}_n$ rejects a string if and only if the string starts with $Z_n a_{n+1}$ (Lemmas 11 and 13).
2. Since a configuration encoding a counter value contains exactly one of the states x_i^0, x_i^1 , for every i , it can be read as a valuation of the variables of φ . The traversal in Fact 1 therefore enumerates all 2^n valuations.
3. In $\mathcal{A}_\varphi$, every letter is a pair consisting of a letter of Σ_{n+1} and a clause c of φ . The first component drives the counter as in $\mathcal{A}_n$; the second component is a guess whose only effect is that the sink q_+ becomes active if and only if c is *satisfied* by the valuation encoded by the current configuration (Lemma 14). Staying on the rejecting path of Fact 1 thus amounts to exhibiting, at every step, a clause falsified by the current valuation.
4. Hence $\mathcal{A}_\varphi$ is non-universal if and only if the whole rejecting path can be traversed, that is, if and only if every one of the 2^n valuations falsifies some clause, i.e., if and only if φ is unsatisfiable (Lemmas 15, 16, and 17).

4.1. An rpoNFA with a state counter

We consider the alphabet $\Sigma_{n+1} = \{a_1, a_2, \dots, a_{n+1}\}$. The *Zimin string* [41] Z_n over $\{a_1, \dots, a_n\}$ is defined recursively by $Z_1 = a_1$ and $Z_i = Z_{i-1} a_i Z_{i-1}$ for $1 < i \leq n$. The letter a_{n+1} occurs in no Zimin string and plays a special role in what follows: it terminates the counting. Accordingly, the string $Z_n a_{n+1}$, which is of length 2^n , is the shortest string rejected by the automaton constructed below. It is well known that $|Z_n| = 2^n - 1$ [42] and that it satisfies the following property.

Lemma 10 ([43]). *The ℓ th letter of Z_n is a_j , where $j - 1$ equals the number of trailing zeros in the binary representation of ℓ .* $\square$

We now construct an rpoNFA $\mathcal{A}_n = (Q_n, \Sigma_{n+1}, \delta, I, Q_n)$. Let

$$Q_n = \{x_i^0 \mid 1 \leq i \leq n\} \cup \{x_i^1 \mid 1 \leq i \leq n\} \cup \{q_+\} \quad \text{and} \quad I = \{x_i^0 \mid 1 \leq i \leq n\}.$$

The transition relation is defined as follows (see Figure 3):

1. $q_+ \xrightarrow{\sigma} q_+$ for every $\sigma \in \Sigma_{n+1}$.
2. For every state x_i^0 :
 - $x_i^0 \xrightarrow{a_j} x_i^0$ for $1 \leq j < i$,
 - $x_i^0 \xrightarrow{a_i} x_i^1$,
 - $x_i^0 \xrightarrow{a_i} x_j^0$ for $1 \leq j < i$,
 - $x_i^0 \xrightarrow{a_j} q_+$ for $i + 1 \leq j \leq n + 1$.

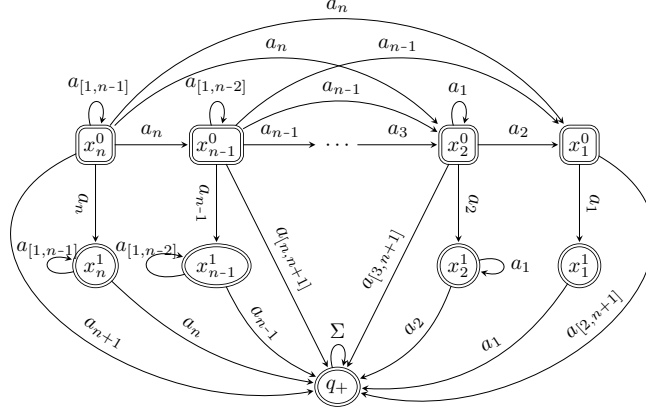


Figure 3: The rpoNFA $\mathcal{A}_n$; initial states are squared and all states are accepting. For positive integers $i \leq j$, $[i, j] = \{i, i+1, \dots, j\}$ and $a_{[i,j]} = \{a_r \mid r \in [i, j]\}$.

3. For every state x_i^1 :

- $x_i^1 \xrightarrow{a_j} x_i^1$ for $1 \leq j < i$,
- $x_i^1 \xrightarrow{a_i} q_+$.

The automaton is an rpoNFA, since no state has a nondeterministic choice between a self-loop and a non-self-loop transition.

Before analyzing $\mathcal{A}_n$, we fix the terminology used throughout this section. Let $0 \leq \ell \leq 2^n - 1$ be an integer with binary representation $\ell = (r_n r_{n-1} \dots r_1)_2$. We say that a configuration $Y \subseteq Q_n$ encodes ℓ if

$$Y = \{x_1^{r_1}, x_2^{r_2}, \dots, x_n^{r_n}\},$$

that is, if for every i the state of Y with lower index i carries, as its upper index, the i th bit of ℓ . Equivalently, Y encodes some integer if and only if $q_+ \notin Y$ and Y contains exactly one of the states x_i^0, x_i^1 for every $i = 1, \dots, n$; the encoded integer is then uniquely determined. Every configuration Y that encodes an integer also determines a valuation e_Y of the variables $x_1, \dots, x_n$, namely $e_Y(x_i) = r_i$; this is used in Section 4.2 below. For example, let $n = 3$. The initial configuration $I = \{x_1^0, x_2^0, x_3^0\}$ encodes $(000)_2 = 0$, the configuration $\{x_1^1, x_2^0, x_3^1\}$ encodes $(101)_2 = 5$, and neither $\{x_1^0, x_1^1, x_3^0\}$ nor $\{x_1^0, x_2^0, x_3^0, q_+\}$ encodes an integer. Reading $Z_3 a_4 = a_1 a_2 a_1 a_3 a_1 a_2 a_1 a_4$ from I , the automaton $\mathcal{A}_3$ passes through the configurations encoding $0, 1, 2, \dots, 7$ in this order, and the final letter a_4 then leads to the empty configuration; see Lemmas 12 and 13.

Lemma 11. Let $Y = \{x_1^{r_1}, \dots, x_n^{r_n}\}$ be a configuration encoding $\ell = (r_n r_{n-1} \dots r_1)_2$, let $a \in \Sigma_{n+1}$, and let $Y \xrightarrow{a} Y'$. Let further a_t denote the $(\ell + 1)$ st letter of $Z_n a_{n+1}$; by Lemma 10, the index t is uniquely determined by ℓ , namely $t - 1$ is the number of trailing zeros in the binary representation of $\ell + 1$. (In particular, $t = n + 1$ if and only if $\ell = 2^n - 1$.)

- (a) If $\ell < 2^n - 1$ and $a = a_t$, then Y' encodes $\ell + 1$.
- (b) If $\ell = 2^n - 1$ and $a = a_t$, then $Y' = \emptyset$.
- (c) If $a \neq a_t$, then $q_+ \in Y'$.

Proof. (a) By Lemma 10, r_t is the rightmost zero of $r_n r_{n-1} \dots r_1$, i.e., $r_n \dots r_1 = r_n \dots r_{t+1} 0 1 \dots 1$. The transition under a_t flips this bit and resets all lower bits, yielding the configuration encoding $\ell + 1$.

(b) In this case, $\ell = 2^n - 1 = (11 \dots 1)_2$ and $a_t = a_{n+1}$. No transitions are defined from states x_i^1 under a_{n+1} , so the configuration becomes empty.

(c) If $a = a_s \neq a_t$, then either $s > t$ or $s < t$. In both cases, a transition to q_+ is enabled, so $q_+ \in Y'$. $\square$

Lemma 12. After reading a prefix of Z_n of length $\ell \leq 2^n - 1$, the configuration of $\mathcal{A}_n$ is $\{x_n^{r_n}, x_{n-1}^{r_{n-1}}, \dots, x_1^{r_1}\}$, where $(r_n r_{n-1} \dots r_1)_2 = \ell$.

Proof. We proceed by induction on ℓ . For $\ell = 0$, the initial configuration is $\{x_n^0, x_{n-1}^0, \dots, x_1^0\}$, which encodes 0. Assume that the claim holds for some $\ell < 2^n - 1$, i.e., after reading a prefix of Z_n of length ℓ , the configuration is $\{x_n^{r_n}, \dots, x_1^{r_1}\}$ with $(r_n \dots r_1)_2 = \ell$. Let a_t be the $(\ell + 1)$ st letter of Z_n (as characterized in Lemma 10). By Lemma 11(a), reading a_t transforms the configuration into $\{x_n^{s_n}, \dots, x_1^{s_1}\}$, where $(s_n \dots s_1)_2 = \ell + 1$. This completes the induction. $\square$

Lemma 13. The automaton $\mathcal{A}_n$ accepts exactly the strings that do not have $Z_n a_{n+1}$ as a prefix.

Proof. Let $w \in \Sigma_{n+1}^*$. If $Z_n a_{n+1}$ is a prefix of w , then after reading Z_n the automaton reaches the configuration encoding $2^n - 1$. The next letter is a_{n+1} , and by Lemma 11(b) the configuration becomes empty. Hence w is rejected.

Otherwise, let u be the longest prefix of w that is also a prefix of Z_n . If $u = w$, then $\mathcal{A}_n$ accepts w . If $u \neq w$, then the next letter differs from the expected one in $Z_n a_{n+1}$. By Lemma 11(c), the state q_+ is reached. Since q_+ is accepting and has self-loops under all letters, the automaton accepts w . $\square$

4.2. Main results

Let φ be a k -CNF formula over variables $\{x_1, \dots, x_n\}$ with a set of clauses C . We construct an rpoNFA $\mathcal{A}_\varphi$ that is universal if and only if φ is satisfiable. The construction is based on a modification of the automaton $\mathcal{A}_n$ from the previous subsection.

Idea. The automaton $\mathcal{A}_n$ accepts all strings except those prefixed with $Z_n a_{n+1}$. While reading $Z_n a_{n+1} = w_1 w_2 \dots w_{2^n}$, it traverses configurations

$$Y_0 \xrightarrow{w_1} Y_1 \xrightarrow{w_2} Y_2 \xrightarrow{w_3} \dots \xrightarrow{w_{2^n}} Y_{2^n}, \quad (3)$$

where $Y_{2^n} = \emptyset$, and for $i < 2^n$ we have $Y_i = \{x_1^{r_1}, \dots, x_n^{r_n}\}$ with $i = (r_n r_{n-1} \dots r_1)_2$. That is, Y_i encodes i in the sense of Section 4.1, and the valuation it determines is $e_{Y_i}(x_j) = r_j$.

We extend the alphabet to $\Sigma_\varphi = \Sigma_{n+1} \times C$. Each transition now carries both a letter and a clause. The construction ensures that for every transition

$$Y_i \xrightarrow{(w_{i+1}, c)} Y_{i+1}$$

along (3), the configuration Y_{i+1} contains q_+ if and only if the clause c is satisfied by e_{Y_i} .

If φ is satisfiable, then for some valuation e_{Y_i} all clauses are satisfied, and hence every choice of $c \in C$ adds q_+ to the configuration. Thus all strings are accepted and $\mathcal{A}_\varphi$ is universal.

Conversely, if φ is unsatisfiable, then for every valuation there exists a clause that is not satisfied. By consistently choosing such clauses, we obtain a string that avoids adding q_+ , and hence $\mathcal{A}_\varphi$ is not universal.

Construction.

The automaton $\mathcal{A}_\varphi = (Q, \Sigma_\varphi, \delta, I, Q)$ is defined as follows. The set of states and the set of initial states are the same as for $\mathcal{A}_n$, that is,

$$Q = \{x_i^0 \mid 1 \leq i \leq n\} \cup \{x_i^1 \mid 1 \leq i \leq n\} \cup \{q_+\} \quad \text{and} \quad I = \{x_i^0 \mid 1 \leq i \leq n\}.$$

Let $\mathcal{L}$ denote the set of literals of φ , and, for $\ell \in \mathcal{L}$, define $\text{cl}(\ell) = \{c \in C \mid \ell \in c\}$. The transition function is defined as follows (see Figure 4 and the example below the proof of Theorem 19 for an illustration):

1. $q_+ \xrightarrow{\sigma} q_+$ for all $\sigma \in \Sigma_\varphi$.
2. For every x_i^0 and $c \in C - \text{cl}(\neg x_i)$:

- $x_i^0 \xrightarrow{(a_j, c)} x_i^0$ for $1 \leq j < i$,
- $x_i^0 \xrightarrow{(a_i, c)} x_i^1$,
- $x_i^0 \xrightarrow{(a_i, c)} x_j^0$ for $1 \leq j < i$,

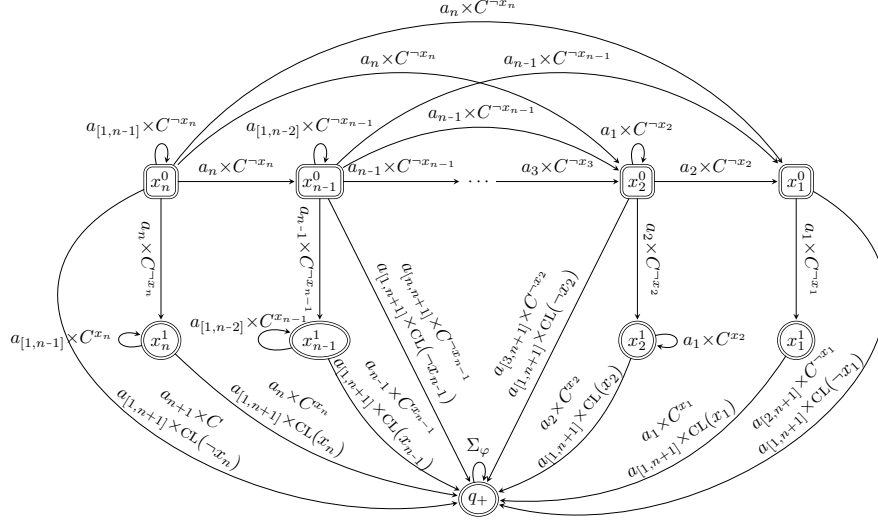


Figure 4: The NFA $\mathcal{A}_\varphi$ of Proposition 18, where the initial states are squared, all states are accepting, and for positive integers $i \leq j$, $[i, j] = \{i, i+1, \dots, j\}$ and $a_{[i,j]} = \{a_r \mid r \in [i, j]\}$; here, for a literal ℓ , C^ℓ stands for $C - \text{CL}(\ell)$.

- $x_i^0 \xrightarrow{(a_j, c)} q_+$ for $i+1 \leq j \leq n+1$.
- 3. For every x_i^0 and $c \in \text{CL}(\neg x_i)$:
 - $x_i^0 \xrightarrow{(a_j, c)} q_+$ for all $1 \leq j \leq n+1$.
- 4. For every x_i^1 and $c \in C - \text{CL}(x_i)$:
 - $x_i^1 \xrightarrow{(a_j, c)} x_i^1$ for $1 \leq j < i$,
 - $x_i^1 \xrightarrow{(a_i, c)} q_+$.
- 5. For every x_i^1 and $c \in \text{CL}(x_i)$:
 - $x_i^1 \xrightarrow{(a_j, c)} q_+$ for all $1 \leq j \leq n+1$.

The automaton is weakly acyclic and self-loop deterministic.

Define $\text{LIT}(x^r)$ by

$$\text{LIT}(x^r) = \begin{cases} \neg x & \text{if } r = 0, \\ x & \text{if } r = 1. \end{cases}$$

The construction can be viewed as modifying $\mathcal{A}_n$ as follows. For every $x \in \{x_i^0 \mid 1 \leq i \leq n\} \cup \{x_i^1 \mid 1 \leq i \leq n\}$ and every $x' \in Q$:

1. Replace each transition $x \xrightarrow{a} x'$ with transitions $x \xrightarrow{(a, c)} x'$ for all c such that $\text{LIT}(x) \notin c$.
2. Add transitions $x \xrightarrow{(a, c)} q_+$ whenever $\text{LIT}(x) \in c$ and $a \in \Sigma_{n+1}$.
3. Keep q_+ as a universal sink, that is, it self-loops under all letters in Σ_φ .

Without loss of generality, we assume that for every literal there exists a clause not containing it; if not, φ is trivially satisfiable and we replace $\mathcal{A}_\varphi$ by a trivial universal rpoNFA.

Note that $\mathcal{A}_\varphi$ is not a ptNFA, since it is neither complete nor confluent.

405 **Lemma 14.** Let $Y = \{x_1^{r_1}, \dots, x_n^{r_n}\}$ be a configuration encoding an integer, and let e_Y be the valuation it determines.

406 Let $(a, c) \in \Sigma_\varphi$, and consider a transition $Y \xrightarrow{(a,c)} Y'$ in $\mathcal{A}_\varphi$ and a transition $Y \xrightarrow{a} Y''$ in $\mathcal{A}_n$.

407 (a) If c is not satisfied by e_Y , then $Y' = Y''$.

408 (b) If c is satisfied by e_Y , then Y' contains q_+ .

409 *Proof.* (a) Since c is not satisfied by e_Y , we have $\text{LIT}(x_i^{r_i}) \notin c$ for all $i = 1, \dots, n$. Hence, a transition $x_i^{r_i} \xrightarrow{(a,c)} p$ exists
410 in $\mathcal{A}_\varphi$ if and only if the transition $x_i^{r_i} \xrightarrow{a} p$ exists in $\mathcal{A}_n$. Therefore $Y' = Y''$.

411 (b) Since c is satisfied by e_Y , there exists i such that $\text{LIT}(x_i^{r_i}) \in c$. By construction, $\mathcal{A}_\varphi$ contains a transition
412 $x_i^{r_i} \xrightarrow{(a,c)} q_+$, and thus $q_+ \in Y'$. □

413 Let $\pi : \Sigma_\varphi^* \rightarrow \Sigma_{n+1}^*$ be the homomorphism given by $\pi((a, c)) = a$, and define

$$\mathbb{W}_\varphi = \{w \in \Sigma_\varphi^* \mid \pi(w) = Z_n a_{n+1}\}.$$

414 **Lemma 15.** $\mathcal{A}_\varphi$ accepts every string that does not have a prefix in $\mathbb{W}_\varphi$.

415 *Proof.* Let $w \in \Sigma_\varphi^*$ have no prefix in $\mathbb{W}_\varphi$. By Lemma 14, either $\mathcal{A}_\varphi$ follows exactly the same sequence of configurations
416 as $\mathcal{A}_n$ on $\pi(w)$, in which case the claim follows from Lemma 13, or $\mathcal{A}_\varphi$ reaches a configuration containing q_+ and
417 hence accepts w . □

418 **Lemma 16.** If φ is satisfiable, then $\mathcal{A}_\varphi$ accepts every string having a prefix in $\mathbb{W}_\varphi$.

419 *Proof.* Let $w \in \Sigma_\varphi^*$ have a prefix in $\mathbb{W}_\varphi$. Let $\ell = (r_n r_{n-1} \dots r_1)_2$ be the smallest number such that the valuation e
420 defined by $e(x_i) = r_i$ satisfies φ . Let u be the prefix of w of length ℓ . Note that u is a proper prefix of w , since $|w| \geq 2^n$
421 and $\ell \leq 2^n - 1$. By Lemma 14, after reading u the automaton $\mathcal{A}_\varphi$ either already contains q_+ (and we are done), or it is
422 in configuration $Y = \{x_1^{r_1}, \dots, x_n^{r_n}\}$. Since $\pi(u)$ is a prefix of Z_n , this follows from Lemma 12. Let the $(\ell + 1)$ st letter
423 of w be (a, c) . Since φ is satisfied by $e = e_Y$, the clause c is satisfied by e_Y . By Lemma 14(b), reading (a, c) adds q_+
424 to the configuration, and hence w is accepted. □

425 **Lemma 17.** If φ is not satisfiable, then there exists a string $w_\varphi \in \mathbb{W}_\varphi$ that is not accepted by $\mathcal{A}_\varphi$.

426 *Proof.* We construct a sequence $\varepsilon = w_0, w_1, \dots, w_{2^n} = w_\varphi$ of prefixes of w_φ such that for every $\ell = 0, \dots, 2^n$, the
427 configuration of $\mathcal{A}_\varphi$ after reading w_ℓ coincides with the configuration of $\mathcal{A}_n$ after reading $\pi(w_\ell)$. The claim then
428 follows from Lemma 13.

429 The construction proceeds by induction on ℓ . The initial configurations coincide by definition. Assume that the
430 claim holds for some $\ell < 2^n$, and let the common configuration be Y . By Lemma 12, we have $Y = \{x_1^{r_1}, \dots, x_n^{r_n}\}$ and
431 $\ell = (r_n r_{n-1} \dots r_1)_2$. Since φ is not satisfiable, there exists a clause c that is not satisfied by e_Y . Let a be the $(\ell + 1)$ st
432 letter of $Z_n a_{n+1}$, and define $w_{\ell+1} = w_\ell(a, c)$. By Lemma 14(a), the configurations after reading $w_{\ell+1}$ in $\mathcal{A}_\varphi$ and
433 $\pi(w_{\ell+1})$ in $\mathcal{A}_n$ coincide. □

434 Combining the above lemmas, we obtain the reduction announced in Section 1.1. As in Section 3, we state the
435 reduction on its own, and derive the conditional lower bounds from it afterwards.

436 **Proposition 18.** Let φ be a k -CNF formula with $n \geq 1$ variables and m clauses. The rpoNFA $\mathcal{A}_\varphi$ can be constructed
437 in time polynomial in the size of φ ; it has all states accepting, $N = 2n + 1$ states, and an alphabet of size $(n + 1)m$, and
438 it is universal if and only if φ is satisfiable. Moreover, if φ is not satisfiable, then every string not accepted by $\mathcal{A}_\varphi$ has
439 length at least 2^n , and there is a string of length $2^n = 2^{\lfloor N/2 \rfloor}$ that is not accepted.

440 *Proof.* The automaton $\mathcal{A}_\varphi$ has the state set Q of size $2n + 1$, all states accepting, and the alphabet $\Sigma_\varphi = \Sigma_{n+1} \times C$ of
441 size $(n + 1)m$; the transitions are determined by membership of literals in clauses and can be listed in polynomial time.
442 If φ is satisfiable, then $\mathcal{A}_\varphi$ accepts every string with a prefix in $\mathbb{W}_\varphi$ by Lemma 16, and every string without such a
443 prefix by Lemma 15; hence $\mathcal{A}_\varphi$ is universal. If φ is not satisfiable, then Lemma 17 provides a string $w_\varphi \in \mathbb{W}_\varphi$ that
444 is not accepted, so $\mathcal{A}_\varphi$ is not universal. Every element of $\mathbb{W}_\varphi$ has length $|Z_n a_{n+1}| = 2^n$, and by Lemma 15 every
445 non-accepted string has a prefix in $\mathbb{W}_\varphi$ and hence length at least 2^n . Finally, $2^n = 2^{\lfloor N/2 \rfloor}$ because $N = 2n + 1$. □

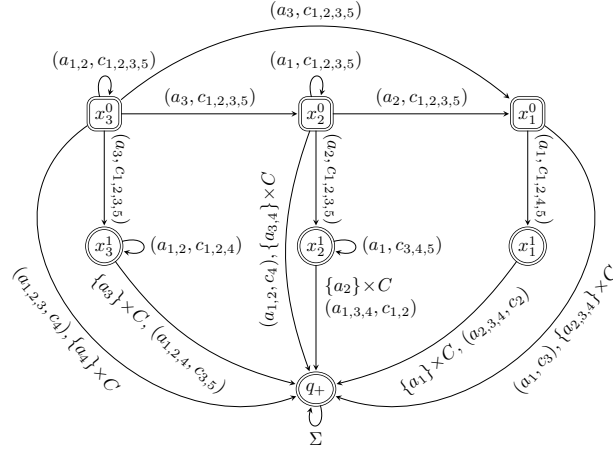


Figure 5: The NFA $\mathcal{A}_\varphi$ illustrating Proposition 18; the initial states are squared and all states are accepting. The notation such as $(a_{1,2}, c_{1,2,3,5})$ is an abbreviation of the set of transitions $\{a_1, a_2\} \times \{c_1, c_2, c_3, c_5\}$.

Theorem 19. *Unless SETH fails, there is no algorithm deciding universality of an N -state rpoNFA with all states accepting that runs in time $O^*(2^{N/(2+\epsilon)})$ for any $\epsilon > 0$.*

Proof. Let φ be a k -CNF formula with n variables. By Proposition 18, the automaton $\mathcal{A}_\varphi$ has $N = 2n + 1$ states, can be constructed in polynomial time, and is universal if and only if φ is satisfiable. Hence, an algorithm deciding universality in time $O^*(2^{N/(2+\epsilon)})$ would yield an algorithm for k -SAT running in time $O(\text{poly}(n)) + O^*(2^{(2n+1)/(2+\epsilon)}) = O^*(2^{(1-\delta)n})$, where $\delta = \epsilon/(2+\epsilon) > 0$. This contradicts SETH. $\square$

Remark 20. In contrast with Proposition 4, the alphabet Σ_φ of Proposition 18 is of size $(n+1)m$ and hence grows with the input. The binary encoding used at the end of the proof of Proposition 4 does not help here: encoding an alphabet of size s over $\{0, 1\}$ replaces each state by a binary tree of depth $\lceil \log s \rceil$ and therefore multiplies the number of states by $\Theta(\log(nm))$. The resulting automaton has $\Theta(n \log(nm))$ states, and the reduction then only rules out algorithms running in time $2^{o(N/\log N)}$, which is subsumed by Theorem 5. Whether the bound of Theorem 19 can be achieved over a fixed alphabet is left open; see also Section 6.

For illustration, we provide a simple example based on a 2-CNF formula $\varphi = (x_2 \vee x_2) \wedge (x_1 \vee x_2) \wedge (\neg x_1 \vee x_3) \wedge (\neg x_2 \vee \neg x_3) \wedge (x_3)$ with three variables x_1, x_2, x_3 and five clauses $c_1 = \{x_2, x_2\}, c_2 = \{x_1, x_2\}, c_3 = \{\neg x_1, x_3\}, c_4 = \{\neg x_2, \neg x_3\}$, and $c_5 = \{x_3\}$. The automaton $\mathcal{A}_\varphi = (Q, \Sigma, \delta, \{x_1^0, x_2^0, x_3^0\}, Q)$ is depicted in Figure 5, where $Q = \{q_+\} \cup \{x_1^0, x_1^1, x_2^0, x_2^1, x_3^0, x_3^1\}$ and $\Sigma = \{a_1, a_2, a_3, a_4\} \times \{c_1, c_2, c_3, c_4, c_5\}$.

Since φ is not satisfiable, the automaton $\mathcal{A}_\varphi$ is not universal. Indeed, the string $(a_1, c_1)(a_2, c_1)(a_1, c_5)(a_3, c_3)(a_1, c_1)(a_2, c_1)(a_1, c_4)(a_4, c_4)$ leads $\mathcal{A}_\varphi$ to the empty configuration; see Figure 6 (left), which depicts the reachable configurations of $\mathcal{A}_\varphi$.

On the other hand, consider the formula $\varphi' = c_1 \wedge c_2 \wedge c_3 \wedge c_4$. Then φ' is satisfiable, and hence the automaton $\mathcal{A}_{\varphi'}$, obtained from $\mathcal{A}_\varphi$ by removing all transitions labeled by letters containing c_5 , is universal. This can be observed from the reachable configurations of $\mathcal{A}_{\varphi'}$ shown in Figure 6 (right).

4.3. Consequences for inclusion, equivalence, and the shortest non-accepted string problems

Since the single-state automaton for the language Σ^* is weakly acyclic, we obtain the following consequence for language inclusion and equivalence.

Corollary 21. *Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be rpoNFAs with n_1 and n_2 states, respectively, and let $n = \max(n_1, n_2)$. Unless SETH fails, there is no algorithm deciding whether $L(\mathcal{A}_1) \subseteq L(\mathcal{A}_2)$ in time $O^*(2^{n/(2+\epsilon)})$, and no algorithm deciding whether $L(\mathcal{A}_1) = L(\mathcal{A}_2)$ in time $O^*(2^{n/(2+\epsilon)})$, for any $\epsilon > 0$.* $\square$

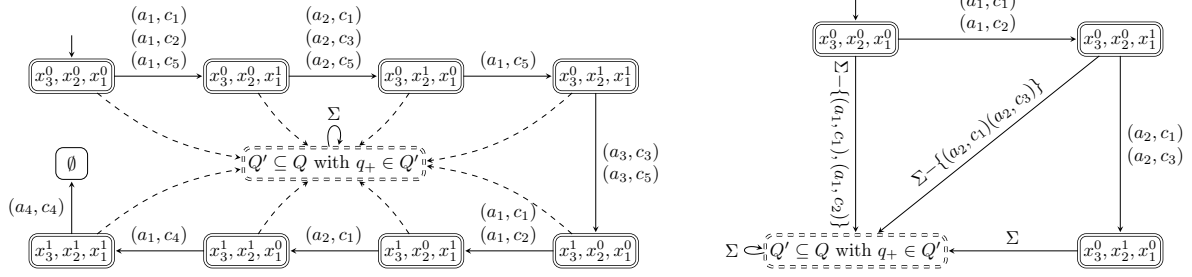


Figure 6: Reachable configurations of $\mathcal{A}_\varphi$ (left) and $\mathcal{A}_{\varphi'}$ (right). All undefined transitions lead to the dashed middle state. Since all states of $\mathcal{A}_\varphi$ are accepting, every configuration except $\emptyset$ is accepting.

The automaton of Proposition 18 rejects, if at all, only strings of length at least 2^n . This yields the SETH counterpart of Corollary 9 for the length-restricted variants of universality.

Corollary 22. *Unless SETH fails, there is no algorithm that, given an N -state rpoNFA $\mathcal{A}$ with all states accepting and an integer ℓ encoded in binary, decides whether $\Sigma^\ell \subseteq L(\mathcal{A})$ in time $O^*(2^{N/(2+\epsilon)})$, for any $\epsilon > 0$. The same holds for $\Sigma^{\leq \ell} \subseteq L(\mathcal{A})$.*

Proof. Let φ be a k -CNF formula with n variables, let $\mathcal{A}_\varphi$ be the automaton of Proposition 18 with $N = 2n + 1$ states, and set $\ell = 2^n$, which is representable in $n + 1$ bits. By Proposition 18, every string not accepted by $\mathcal{A}_\varphi$ has length at least 2^n , and if $\mathcal{A}_\varphi$ is not universal, then some string of length 2^n is not accepted. Hence $\Sigma^\ell \subseteq L(\mathcal{A}_\varphi)$ if and only if $\Sigma^{\leq \ell} \subseteq L(\mathcal{A}_\varphi)$ if and only if $\mathcal{A}_\varphi$ is universal, and the claim follows from Theorem 19. $\square$

Ellul et al. [34] considered the problem of determining, given an NFA $\mathcal{A}$, the minimum length of a string not accepted by $\mathcal{A}$. They showed that there exists an n -state NFA for which the shortest non-accepted string has length 2^{cn} for some constant $0 < c \leq 1$. In particular, they constructed a weakly acyclic NFA over a five-letter alphabet whose shortest non-accepted string has length at least $2^{n/25}$. By binary encoding, they obtained an NFA (no longer weakly acyclic) with a shortest non-accepted string of length at least $2^{n/75}$. This result also holds for NFAs with all states accepting [30]. Krötzsch et al. [14] showed that even binary weakly acyclic NFAs can have exponentially long shortest non-accepted strings, and that there are rpoNFAs over an n -letter alphabet with $n(n + 2)$ states whose shortest non-accepted string has length $\binom{2n}{n} - 1$. As discussed in Section 1.1, the strongest bound of this kind is due to Masopust and Krötzsch [29], who obtained it already for ptNFAs; in terms of the number N of states, it is $2^{\Theta(\sqrt{N})}$.

Our construction from the previous section yields shortest non-accepted strings of length $2^{(n-1)/2}$ for rpoNFAs with all states accepting, over a growing alphabet, thus improving the exponent from $\Theta(\sqrt{n})$ to $\Theta(n)$. As a consequence, we obtain the following corollary.

Corollary 23. *Unless SETH fails, there is no algorithm that, given an N -state rpoNFA, decides whether the shortest non-accepted string has length $2^{\lfloor N/2 \rfloor}$ in time $O^*(2^{N/(2+\epsilon)})$, for any $\epsilon > 0$. The result holds even if all states of the automaton are accepting.*

Proof. We use the automaton $\mathcal{A}_\varphi$ of Proposition 18, which has $N = 2n + 1$ states, where n is the number of variables of φ . By Proposition 18, if φ is not satisfiable, then the shortest non-accepted string of $\mathcal{A}_\varphi$ has length $2^n = 2^{\lfloor N/2 \rfloor}$, and otherwise $\mathcal{A}_\varphi$ is universal and no such string exists. A decision procedure for the former property running in time $O^*(2^{N/(2+\epsilon)})$ would therefore decide k -SAT within the time bound excluded in the proof of Theorem 19. $\square$

As is well known, the length of the shortest non-accepted string bounds from below the number of states of an equivalent deterministic automaton: if w is a shortest string not in a language L , then the states reached by the prefixes of w in any DFA recognizing L are pairwise distinct, since two prefixes leading to the same state would yield a shorter string outside L . Consequently, for an unsatisfiable φ , no DFA with fewer than $2^n + 1 = 2^{(N-1)/2} + 1$ states recognizes $L(\mathcal{A}_\varphi)$.

The lower bound of Corollary 23 can be generalized as follows. Let $b > 2$ be a fixed integer and let φ be a k -CNF formula with n variables. Let $\mathcal{A}$ be the automaton obtained from the automaton $\mathcal{A}_\varphi$ of Proposition 18 by adding

($b-2$) $n-1$ isolated non-accepting states. Since $\mathcal{A}_\varphi$ has $2n+1$ states, the resulting automaton $\mathcal{A}$ has $N = bn$ states. If φ is not satisfiable, then the shortest non-accepted string of $\mathcal{A}$ has length $2^n = 2^{N/b}$. Consequently, unless SETH fails, there is no algorithm running in time $O^*(2^{N/(b+\epsilon)})$ that decides whether the shortest non-accepted string of an N -state rpoNFA has length $2^{N/b}$.

Observe the connection between the shortest non-accepted string and bounded universality. An integer ℓ is the length of the shortest non-accepted string of a language L if and only if ℓ is the maximum number such that $\Sigma^{\leq \ell-1} \subseteq L$. Since bounded universality is PSPACE-complete for NFAs [31], the decision problem for the shortest non-accepted string is also PSPACE-complete. Moreover, Fernau and Krebs [10] showed that, unless SETH fails, there is no algorithm running in time $O^*((|\Sigma| - \epsilon)^\ell)$, for any $\epsilon > 0$, that decides, given an NFA $\mathcal{A}$ and an integer ℓ (encoded in unary), whether there exists a string in $\Sigma^{\leq \ell}$ that is not accepted by $\mathcal{A}$.

Finally, we point out that it remains open whether analogous results hold for ptNFAs.

5. Automata with all states accepting

Kao et al. [30] studied NFAs in which all states are accepting and showed that deciding universality for such NFAs is PSPACE-complete. However, their reduction introduces cycles and therefore does not apply to weakly acyclic NFAs. For completeness, we show that requiring all states to be accepting does not affect the complexity of universality for weakly acyclic NFAs and for their self-loop deterministic subclass, rpoNFAs. This observation does not extend to ptNFAs: due to completeness in their definition, every ptNFA with all states accepting is necessarily universal.

Recall that universality is PSPACE-complete for weakly acyclic NFAs (even over a binary alphabet) and also for rpoNFAs; if the alphabet is fixed, the problem becomes coNP-complete.

Theorem 24. *The complexity of deciding universality for poNFAs (resp. rpoNFAs) with all states accepting coincides with the complexity of deciding universality for poNFAs (resp. rpoNFAs) without this restriction.*

Proof. Let $\mathcal{A}$ be an NFA over an alphabet Σ . We encode acceptance by transitions under a fresh letter. Construct an automaton $\mathcal{A}'$ by taking a copy of $\mathcal{A}$, adding a new state q and a new letter $\#$, and extending the transition function as follows:

- for every accepting state of $\mathcal{A}$, add a transition under $\#$ to q ,
- add self-loops under all letters in $\Sigma \cup \{\#\}$ at q ,
- declare all states of $\mathcal{A}'$ to be accepting.

We show that $\mathcal{A}$ is universal over Σ if and only if $\mathcal{A}'$ is universal over $\Sigma \cup \{\#\}$.

Assume first that $\mathcal{A}$ is universal. Then every computation of $\mathcal{A}'$ on a string $w \in \Sigma^*$ reaches only configurations containing at least one accepting state of $\mathcal{A}$. Hence, at any position of the computation, $\mathcal{A}'$ can read the letter $\#$ and move to q , from which it accepts any continuation. Therefore, $\mathcal{A}'$ accepts every string over $\Sigma \cup \{\#\}$.

Conversely, assume that $\mathcal{A}$ is not universal. Then there exists a string $w \in \Sigma^*$ such that $\mathcal{A}$ reaches a configuration with no accepting state. The same holds for $\mathcal{A}'$. From such a configuration, no $\#$ -transition is enabled, and hence $\mathcal{A}'$ does not accept $w\#$. Thus, $\mathcal{A}'$ is not universal.

The construction preserves weak acyclicity as well as self-loop determinism, which completes the proof. $\square$

On the other hand, the construction of $\mathcal{A}_\varphi$ in the previous section does not require all states to be accepting. In fact, it is sufficient to designate only three states as accepting, for example $\{x_1^0, x_1^1, q_+\}$. Indeed, every reachable configuration of $\mathcal{A}_\varphi$ either encodes a valuation of the input formula, in which case it contains exactly one of x_i^0 or x_i^1 for each $i = 1, \dots, n$, or it contains the state q_+ . Hence, at least one of these designated states is always present in every accepting computation.

6. Conclusions

We studied conditional lower bounds for the universality problem for several classes of weakly acyclic NFAs. We showed that, assuming ETH, there is no subexponential-time algorithm $2^{o(n)}$ deciding universality for any of the considered automata classes, even over a binary alphabet. Under SETH, we further strengthened this result for rpoNFAs by proving that there is no algorithm deciding universality in time $O^*(2^{n/c})$ for any $c > 2$; this holds even if all states of the automaton are accepting. Hence, the currently known algorithms are essentially optimal up to constant factors in the exponent.

In particular, our results rule out algorithms with running time $O^*(1.4^n)$ for universality of rpoNFAs. However, they do not preclude algorithms with running time $O^*(1.5^n)$. Whether such algorithms exist, or whether the lower bounds can be further tightened, remains an open problem. Another natural question is whether our SETH-based lower bounds for rpoNFAs can be extended to ptNFAs or to rpoNFAs over a fixed alphabet. We regard the latter as the main question left open here: the reduction of Proposition 18 is linear in the number of states but not in the size of the alphabet, and, as explained in Remark 20, encoding the alphabet in binary destroys the constant in the exponent.

As an immediate consequence, the same lower bounds apply to the language inclusion and equivalence problems for the considered automata classes. We also discussed implications for weakly acyclic automata with all states accepting, as well as for the problems of bounded universality and determining the length of the shortest non-accepted string.

References

- [1] A. V. Aho, J. E. Hopcroft, J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, 1974.
- [2] P. A. Abdulla, Y. Chen, L. Holík, R. Mayr, T. Vojnar, When simulation meets antichains, in: J. Esparza, R. Majumdar (Eds.), *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, volume 6015 of *LNCS*, Springer, 2010, pp. 158–174.
- [3] F. Bonchi, D. Pous, Checking NFA equivalence with bisimulations up to congruence, in: R. Giacobazzi, R. Cousot (Eds.), *Symposium on Principles of Programming Languages*, ACM, 2013, pp. 457–468.
- [4] P. Černý, E. M. Clarke, T. A. Henzinger, A. Radhakrishna, L. Ryzhyk, R. Samanta, T. Tarrach, From non-preemptive to preemptive scheduling using synchronization synthesis, *Formal Methods in System Design* 50 (2017) 97–139.
- [5] D. Chocholatý, T. Fiedor, V. Havlena, L. Holík, M. Hruska, O. Lengál, J. Síc, Mata: A fast and simple finite automata library, in: B. Finkbeiner, L. Kovács (Eds.), *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, volume 14571 of *LNCS*, Springer, 2024, pp. 130–151.
- [6] L. Clemente, R. Mayr, Efficient reduction of nondeterministic automata with application to language inclusion testing, *Logical Methods in Computer Science* 15 (2019).
- [7] D. L. Dill, A. J. Hu, H. Wong-Toi, Checking for language inclusion using simulation preorders, in: K. G. Larsen, A. Skou (Eds.), *Computer Aided Verification*, volume 575 of *LNCS*, Springer, 1991, pp. 255–265.
- [8] R. van Glabbeek, B. Ploeger, *Five Determinisation Algorithms*, Springer Berlin Heidelberg, 2008, p. 161–170.
- [9] I. Baburin, R. Cotterell, A close analysis of the subset construction, in: A. Malcher, L. Prigioniero (Eds.), *Descriptional Complexity of Formal Systems*, volume 15759 of *LNCS*, Springer, 2025, pp. 17–33.
- [10] H. Fernau, A. Krebs, Problems on finite automata and the exponential time hypothesis, *Algorithms* 10 (2017) 24.
- [11] J. Balun, T. Masopust, P. Osička, Speed me up if you can: Conditional lower bounds on opacity verification, in: J. Leroux, S. Lombardy, D. Peleg (Eds.), *Mathematical Foundations of Computer Science*, volume 272 of *LIPICs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023, pp. 16:1–16:15.

- [12] M. de Oliveira Oliveira, M. Wehar, On the fine grained complexity of finite automata non-emptiness of intersection, in: N. Jonoska, D. Savchuk (Eds.), *Developments in Language Theory*, volume 12086 of *Lecture Notes in Computer Science*, Springer, 2020, pp. 69–82.
- [13] M. Blondin, M. Cadilhac, X. Cui, P. Czerner, J. Esparza, J. Schulz, Weakly acyclic diagrams: A data structure for infinite-state symbolic verification, in: A. Gurfinkel, M. Heule (Eds.), *Tools and Algorithms for the Construction and Analysis of Systems*, volume 15698 of *LNCS*, Springer, 2025, pp. 23–42.
- [14] M. Krötzsch, T. Masopust, M. Thomazo, Complexity of universality and related problems for partially ordered NFAs, *Information and Computation* 255 (2017) 177–192.
- [15] A. Bouajjani, A. Muscholl, T. Touili, Permutation rewriting and algorithmic verification, *Information and Computation* 205 (2007) 199–224.
- [16] W. Czerwiński, W. Martens, T. Masopust, Efficient separability of regular languages by subsequences and suffixes, in: *International Colloquium on Automata, Languages and Programming*, volume 7966 of *LNCS*, 2013, pp. 150–161.
- [17] T. Place, L. van Rooijen, M. Zeitoun, Separating regular languages by piecewise testable and unambiguous languages, in: *Mathematical Foundations of Computer Science*, volume 8087 of *LNCS*, 2013, pp. 729–740.
- [18] T. Masopust, Complexity of deciding detectability in discrete event systems, *Automatica* 93 (2018) 257–261.
- [19] J. A. Brzozowski, F. E. Fich, Languages of R -trivial monoids, *Journal of Computer and System Sciences* 20 (1980) 32–49.
- [20] T. Schwentick, D. Thérien, H. Vollmer, Partially-ordered two-way automata: A new characterization of DA, in: *Developments in Language Theory*, volume 2295 of *LNCS*, 2001, pp. 239–250.
- [21] H. Straubing, A generalization of the Schützenberger product of finite monoids, *Theoretical Computer Science* 13 (1981) 137–150.
- [22] D. Thérien, Classification of finite monoids: The language approach, *Theoretical Computer Science* 14 (1981) 195–208.
- [23] J. A. Brzozowski, R. Knast, The dot-depth hierarchy of star-free languages is infinite, *Journal of Computer and System Sciences* 16 (1978) 37–55.
- [24] R. S. Cohen, J. A. Brzozowski, Dot-depth of star-free events, *Journal of Computer and System Sciences* 5 (1971) 1–16.
- [25] H. Straubing, Finite semigroup varieties of the form V^*D , *Journal of Pure and Applied Algebra* 36 (1985) 53–94.
- [26] W. Thomas, *Languages, Automata, and Logic*, Springer Berlin Heidelberg, 1997, p. 389–455.
- [27] K. W. Wagner, Leaf language classes, in: *Machines, Computations, and Universality*, volume 3354 of *LNCS*, 2004, pp. 60–81.
- [28] I. Simon, *Hierarchies of Events with Dot-Depth One*, Ph.D. thesis, University of Waterloo, Canada, 1972.
- [29] T. Masopust, M. Krötzsch, Partially ordered automata and piecewise testability, *Logical Methods in Computer Science* 17 (2021).
- [30] J. Kao, N. Rampersad, J. O. Shallit, On NFAs where all states are final, initial, or both, *Theoretical Computer Science* 410 (2009) 5010–5021.
- [31] P. Gawrychowski, M. Lange, N. Rampersad, J. O. Shallit, M. Szykula, Existential length universality, in: C. Paul, M. Bläser (Eds.), *Symposium on Theoretical Aspects of Computer Science*, volume 154 of *LIPICs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020, pp. 16:1–16:14.

- 632 [32] A. Potechin, J. O. Shallit, Lengths of words accepted by nondeterministic finite automata, *Information Processing*
633 *Letters* 162 (2020) 105993.
- 634 [33] D. Kozen, Lower bounds for natural proof systems, in: *Annual Symposium on Foundations of Computer Science*,
635 *IEEE Computer Society*, 1977, pp. 254–266.
- 636 [34] K. Ellul, B. Krawetz, J. Shallit, M. Wang, Regular expressions: New results and open problems, *Journal of*
637 *Automata, Languages and Combinatorics* 10 (2005) 407–437.
- 638 [35] J. E. Hopcroft, R. Motwani, J. D. Ullman, *Introduction to automata theory, languages, and computation*, Pearson
639 international edition, third ed., Addison-Wesley, 2007.
- 640 [36] R. Impagliazzo, R. Paturi, On the complexity of k -SAT, *Journal of Computer and System Sciences* 62 (2001)
641 367–375.
- 642 [37] R. Paturi, P. Pudlák, M. E. Saks, F. Zane, An improved exponential-time algorithm for k -SAT, *Journal of the*
643 *ACM* 52 (2005) 337–364.
- 644 [38] T. Hertli, 3-SAT faster and simpler - unique-sat bounds for PPSZ hold in general, *SIAM Journal on Computing*
645 43 (2014) 718–729.
- 646 [39] M. Cygan, F. Fomin, L. Kowalik, D. Marx, M. Pilipczuk, S. S., *Parametrized Algorithms*, Springer: Berlin,
647 Germany, 2015.
- 648 [40] T. Masopust, M. Thomazo, On boolean combinations forming piecewise testable languages, *Theoretical*
649 *Computer Science* 682 (2017) 165–179.
- 650 [41] A. I. Zimin, Blocking sets of terms, *Matematicheskii Sbornik* 119 (1982) 363–375. Translated in *Sbornik:*
651 *Mathematics* 47 (1984), 353–364.
- 652 [42] N. J. A. Sloan, The on-line encyclopedia of integer sequences (OEIS), 2023. Sequence A123121.
- 653 [43] N. J. A. Sloan, The on-line encyclopedia of integer sequences (OEIS), 2023. Sequence A001511.