International Journal of Foundations of Computer Science
Vol. 30, No. 1 (2019) 115–134

DOI: 10.1142/S0129054119400069

World Scientific
www.worldscientific.com

Descriptional Complexity of the Forever Operator

Michal Hospodár*, Galina Jirásková† and Peter Mlynárčik‡
Mathematical Institute, Slovak Academy of Sciences
Grešákova 6, Košice, 040 01, Slovakia
**hosmich@gmail.com*
†*jiraskov@saske.sk*
‡*mlynarcik1972@gmail.com*

Accepted 23 January 2019
Communicated by Michel Rigo

We examine the descriptional complexity of the forever operator, which assigns the language $\overline{\Sigma^*\overline{L}}$ to a regular language L, and we investigate the trade-offs between various models of finite automata. We consider complete and partial deterministic finite automata, nondeterministic finite automata with single or multiple initial states, alternating, and Boolean finite automata. We assume that the argument and the result of this operation are accepted by automata belonging to one of these six models. We investigate all possible trade-offs and provide a tight upper bound for 32 of 36 of them. The most interesting result is the trade-off from nondeterministic to deterministic automata given by the Dedekind number $\mathrm{M}(n-1)$. We also prove that the nondeterministic state complexity of $\overline{\Sigma^*\overline{L}}$ is 2^{n-1} which solves an open problem stated by Birget [The state complexity of $\overline{\Sigma^*\overline{L}}$ and its connection with temporal logic, *Inform. Process. Lett.* **58** (1996) 185–188].

Keywords: Regular languages; forever operator; deterministic automata; nondeterministic automata; Boolean automata; minimal automata; trade-off.

1. Introduction

Formal languages may be recognized by several kinds of formal systems. Different classes of formal systems can be compared either from the point of view of their computational power, or from the descriptional complexity point of view. As for computational power, for example, deterministic and nondeterministic finite automata recognize the same class of languages, while the class of languages recognized by deterministic pushdown automata is strictly included in the class of languages recognized by nondeterministic ones. However, from the descriptional complexity point of view, there is an exponential gap between the cost of description of regular languages by deterministic and nondeterministic finite automata [14, 16–18, 20].

Descriptional complexity, which measures the cost of description of languages by different formal systems, was deeply investigated in last three decades [1, 7, 15, 22] mostly in the class of regular languages. Several kinds of finite automata were proposed and the trade-offs between the costs of description in different classes of automata were examined. Let us mention at least the exact trade-off $\binom{2n}{n+1}$ for the conversion of two-way nondeterministic automata to one-way nondeterministic automata [12], and the exact trade-off for the conversion of self-verifying automata to deterministic automata given by the function that counts the maximal number of maximal cliques in a graph with n vertices [11].

In 1996, Jean-Camille Birget [2] answered the following question of Jean-Éric Pin. Let L be a regular language over an alphabet Σ recognized by a nondeterministic finite automaton (NFA) or a deterministic finite automaton (DFA) with n states. How many states are sufficient and necessary in the worst case for an NFA (DFA) to recognize the language $\overline{\Sigma^*\overline{L}}$? The notation $\overline{L}$ stands for the complement of L. Birget provided the exact trade-off from DFAs to NFAs, and lower and upper bounds for the nondeterministic state complexity of $\overline{\Sigma^*\overline{L}}$.

The motivation of Pin's question came from the word model of Propositional Temporal Logic [5]. The set of all models of a formula φ over a fixed alphabet Σ is a formal language $L(\varphi)$ over Σ which has the non-trivial property of being regular and aperiodic. Some of the temporal operators used in this logic are $\circ$ ("next") and $\diamond$ ("eventually", or "at some moment in the future"); there are also the usual Boolean operations $-$, $\wedge$, $\vee$. A natural dual to the "eventually" operator is the "forever" (or, "always in the future") operator $\Box$, defined to be $-\diamond-$ ("not eventually not"). Formulas and their models are related as follows: $L(\overline{\varphi}) = \overline{L(\varphi)}$, $L(\varphi\wedge\psi) = L(\varphi)\cap L(\psi)$, $L(\varphi\vee\psi) = L(\varphi)\cup L(\psi)$, $L(\circ\varphi) = \Sigma L(\varphi)$, $L(\diamond\varphi) = \Sigma^* L(\varphi)$. Thus $L(\Box\varphi) = L(\overline{\diamond\overline{\varphi}}) = \overline{\Sigma^*\overline{L}}$. Hence in [2], Birget studied the state complexity of the "forever" operator.

Here we continue this research by investigating the complexity of the forever operator for different models of finite automata. We consider complete and partial deterministic finite automata, nondeterministic automata with a single or multiple initial states, and Boolean automata with a single initial state, called *alternating* finite automata in [2, 6], or with an initial function [4]. Similarly as Jean-Éric Pin, we ask the following question: If a language L is represented by an n-state automaton of some model, how many states are sufficient and necessary in the worst case for an automaton of some other model to accept $\overline{\Sigma^*\overline{L}}$?

We study all the possible 36 trade-offs, and except for four cases, we always get tight upper bounds. In particular, we are able to prove that the upper bound on the nondeterministic state complexity of $\overline{\Sigma^*\overline{L}}$ is 2^{n-1}. This improves Birget's upper bound $2^{n+1}+1$ and meets his lower bound for DFA-to-NFA trade-off. The most interesting result of this paper is the tight upper bound for the NFA-to-DFA trade-off given by the Dedekind number $\mathrm{M}(n-1)$; recall that the Dedekind number $\mathrm{M}(n)$ counts the number of antichains of subsets of an n-element set.

2. Preliminaries

Let Σ be a finite alphabet of symbols. Then Σ^* denotes the set of strings over Σ including the empty string ε. A language is any subset of Σ^*. For a language L, the complement of L is the language $\overline{L} = \Sigma^* \setminus L$. The concatenation of languages K and L is the language $KL = \{uv \mid u \in K \text{ and } v \in L\}$. The cardinality of a finite set A is denoted by $|A|$, and its power-set by 2^A. By $\log n$, we denote the binary logarithm of the number n. For details, we refer to [19, 21].

A *nondeterministic finite automaton with multiple initial states* (NNFA) is a 5-tuple $A = (Q, \Sigma, \circ, I, F)$, where Q is a finite set of states, Σ is a finite non-empty alphabet, $\circ : Q \times \Sigma \to 2^Q$ is the transition function which is naturally extended to the domain $2^Q \times \Sigma^*$, $I \subseteq Q$ is the set of initial states, and $F \subseteq Q$ is the set of final states. The *language accepted by* A is $L(A) = \{w \in \Sigma^* \mid I \circ w \cap F \neq \emptyset\}$. *If* $|I| \geq 2$, we say that A is a nondeterministic finite automaton with nondeterministic choice of initial state (so we use the abbreviation NNFA, cf. [21]). Otherwise, if $|I| = 1$, we say that A is a *nondeterministic finite automaton* (NFA). Then, if $I = \{s\}$, we write $A = (Q, \Sigma, \circ, s, F)$ instead of $A = (Q, \Sigma, \circ, \{s\}, F)$. In an ε-NFA, we also allow the transitions on the empty string. It is known that the ε-transitions can be removed without increasing the number of states in the resulting NFA [21, Theorem 2.3].

An NFA A is a (complete) *deterministic finite automaton* (DFA) if $|q \circ a| = 1$ for each q in Q and each a in Σ. Next, A is a *partial deterministic finite automaton* (pDFA) if $|q \circ a| \leq 1$ for each q in Q and each a in Σ. We write $p \circ a = q$ in such cases. For DFAs, we use $\cdot$ to denote the transition function that maps $Q \times \Sigma$ to Q.

We call a state of an NNFA *sink state* if it has a loop on every input symbol. From every final sink state, every string is accepted, but from every non-final sink state in a DFA, no string is accepted. Notice that every minimal pDFA has no non-final sink states, and every minimal DFA has at most one non-final sink state.

For a symbol a and states p and q, we say that (p, a, q) is a transition in the NNFA A if $q \in p \cdot a$, and for a string w, we write $p \xrightarrow{w} q$ if $q \in p \cdot w$. We also say that the state q has an *in-transition* on a, and the state p has an *out-transition* on a.

To *omit* a state q of a DFA means to remove it from the state set and to remove also all its in-transitions and out-transitions. To *replace* the state q with a sink state means to remove each of its out-transitions and add a loop (q, a, q) for each a.

The reverse of a string is defined as $\varepsilon^R = \varepsilon$ and $(wa)^R = aw^R$ for each symbol a and string w. The reverse of a language L is the language $L^R = \{w^R \mid w \in L\}$. The reverse of an NNFA $A = (Q, \Sigma, \cdot, I, F)$ is an NNFA A^R obtained from A by reversing all the transitions and by swapping the roles of initial and final states. The NNFA A^R recognizes the reverse of $L(A)$.

Every NNFA $A = (Q, \Sigma, \cdot, I, F)$ can be converted to an equivalent DFA $\mathcal{D}(A) = (2^Q, \Sigma, \cdot, I, F')$ where $F' = \{S \in 2^Q \mid S \cap F \neq \emptyset\}$. We call the DFA $\mathcal{D}(A)$ the *subset automaton* of the NNFA A. We use the following proposition to prove reachability of states in a subset automaton in some cases.

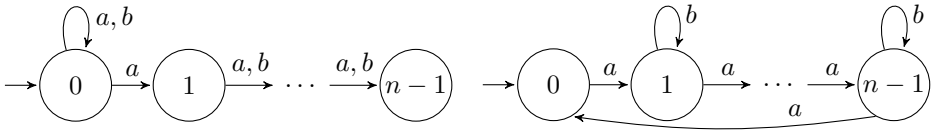


Fig. 1. The NFAs used in Proposition 1.

Proposition 1. *In the subset automaton of the* NFA *shown in Fig.* 1(left), *each subset containing* 0 *is reachable from* $\{0\}$, *and in the subset automaton of the* NFA *shown in Fig.* 1(right), *each subset is reachable from* $\{0, 1, \ldots, n-1\}$.

Proof. The proof of case (1) is by induction on the size of subsets. The subset $\{0\}$ is the initial subset. Each subset $\{0, i_1, i_2, \ldots, i_k\}$ of size $k+1$, where $1 \leq k \leq n-1$ and $1 \leq i_1 < i_2 < \cdots < i_k$, is reached from the subset $\{0, i_2 - i_1, \ldots, i_k - i_1\}$ of size k by the string ab^{i_1-1}. Notice that the proof works for arbitrary transitions on a, b in the state $n-1$. The proof of case (2) is also by induction on the size of subsets. The set $\{0, 1, \ldots, n-1\}$ of size n is the basis. Each subset S with $t \notin S$ of size $k-1$, where $1 \leq k \leq n$, is reachable from the subset $S \cup \{t\}$ of size k by $a^{n-t}ba^t$. □

To prove distinguishability, we use the following notions and observations. A state q of an NFA $A = (Q, \Sigma, \cdot, s, F)$ is called *uniquely distinguishable* (cf. [3]) if there is a string w which is accepted by A from and only from the state q. A transition (p, a, q) in the NFA A is called a *unique in-transition* if there is no state r of A such that $r \neq p$ and (r, a, q) is a transition in A. A state q is *uniquely reachable* from a state p, if there is a sequence of unique in-transitions (p_{i-1}, a_i, p_i) $(1 \leq i \leq k)$ such that $p_0 = p$ and $p_k = q$.

Proposition 2 (cf. [3, Propositions 14 and 15]). *Let* A *be an* NFA *and* $\mathcal{D}(A)$ *be the subset automaton of* A.

(a) *If two subsets of the state set of* A *differ in a uniquely distinguishable state, then the two subsets are distinguishable in* $\mathcal{D}(A)$.
(b) *If a uniquely distinguishable state* q *is uniquely reachable from a state* p, *then the state* p *is uniquely distinguishable as well.*
(c) *If there is a uniquely distinguishable state of an* NFA A *that is uniquely reachable from any other state of* A, *then every state of* A *is uniquely distinguishable.*
(d) *If every state of* A *is uniquely distinguishable, then the subset automaton* $\mathcal{D}(A)$ *does not have equivalent states.*

A set of pairs of strings $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ is called a *fooling set* for a language L if for all i, j in $\{1, 2, \ldots, n\}$, we have $x_i y_i \in L$, and if $i \neq j$, then $x_i y_j \notin L$ or $x_j y_i \notin L$. It is well-known [1, 8] that the size of a fooling set for L provides a lower bound on the number of states in any NNFA accepting the language L.

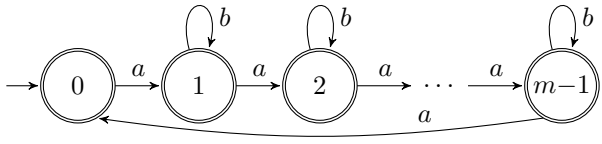


Fig. 2. The NFA for L such that every NFA for L^R has at least $m+1$ states and every DFA for L^R has at least 2^m states.

For NFAs, the following lemma from [10] is useful; notice that $\mathcal{A}$ and $\mathcal{B}$ in [10, Lemma 4] must be disjoint.

Lemma 3 (cf. [10, Lemma 4]). *Let $\mathcal{A}$ and $\mathcal{B}$ be disjoint sets of pairs of strings and let u and v be two strings such that $\mathcal{A} \cup \mathcal{B}$, $\mathcal{A} \cup \{(\varepsilon, u)\}$, and $\mathcal{B} \cup \{(\varepsilon, v)\}$ are fooling sets for a language L. Then every* NFA *for L has at least $|\mathcal{A}|+|\mathcal{B}|+1$ states.*

The next result is used later in our paper.

Proposition 4. *Let L be the language accepted by the* NFA *shown in Fig.* 2. *Then*

(a) *every* NFA *for L^R has at least $m+1$ states, and*
(b) *every* DFA *for L^R has at least 2^m states.*

Proof. We reverse the NFA in Fig. 2 to get the NNFA N for L^R.

(a) Let

$$\begin{aligned}
\mathcal{A} &= \{((ba)^{m-1-i}, (ba)^i) \mid 0 \le i \le m-2\},\\
\mathcal{B} &= \{(b, a(ba)^{m-2})\},\\
u &= a(ba)^{m-2}, \quad \text{and}\\
v &= a^m.
\end{aligned}$$

The reader may verify that $\mathcal{A} \cup \mathcal{B}$, $\mathcal{A} \cup \{(\varepsilon, u)\}$, and $\mathcal{B} \cup \{(\varepsilon, v)\}$ are fooling sets for the language L^R. By Lemma 3, every NFA for L^R needs at least $m+1$ states.

(b) By Proposition 1, the subset automaton $\mathcal{D}(N)$ has 2^m reachable states. To prove distinguishability, notice that the state 0 is uniquely distinguishable in N by ε, and it is uniquely reachable from any other state of N through unique in-transitions on symbol a. By Proposition 2(c-d), the subset automaton $\mathcal{D}(N)$ does not have equivalent states. □

A *Boolean finite automaton* (BFA, cf. [4]) is a quintuple $A = (Q, \Sigma, \delta, g_s, F)$, where Q is a finite non-empty set of states such that $Q = \{q_1, \ldots, q_n\}$, Σ is an input alphabet, δ is the transition function that maps $Q \times \Sigma$ into the set $\mathcal{B}_n$ of Boolean functions with variables $\{q_1, \ldots, q_n\}$, $g_s \in \mathcal{B}_n$ is the initial Boolean function, and $F \subseteq Q$ is the set of final states. The transition function δ is extended to the domain $\mathcal{B}_n \times \Sigma^*$ as follows: For all g in $\mathcal{B}_n$, a in Σ, and w in Σ^*, we have $\delta(g, \varepsilon) = g$; if $g = g(q_1, \ldots, q_n)$, then $\delta(g, a) = g(\delta(q_1, a), \ldots, \delta(q_n, a))$; $\delta(g, wa) = \delta(\delta(g, w), a)$.

Next, let $f = (f_1, \ldots, f_n)$ be the Boolean vector with $f_i = 1$ iff $q_i \in F$. The language accepted by the BFA A is the set of strings $L(A) = \{w \in \Sigma^* \mid \delta(g_s, w)(f) = 1\}$. A Boolean finite automaton is called *alternating* (AFA, cf. [6]) if the initial function is a projection $g(q_1, \ldots, q_n) = q_i$.

We use the following observations for trade-offs between various automata throughout this paper. We use the claim in Lemma 5(a) quite often in the paper without referring to Lemma 5(a) again and again.

Lemma 5 (Properties of Finite Automata). *Let L be a regular language.*

(a) *The language L is accepted by an n-state* BFA (AFA) *if and only if L^R is accepted by a* DFA *of 2^n states (of which 2^{n-1} are final, in case of an* AFA).
(b) *Let L^R be a regular language accepted by a minimal n-state* DFA. *Then every* BFA *for L has at least $\lceil \log n \rceil$ states.*
(c) *If the minimal DFA for L^R has more than 2^{n-1} final states, then every AFA for L has at least $n + 1$ states.*
(d) *Let L be unary. Then L is accepted by an n-state* BFA (AFA) *if and only if L is accepted by a* DFA *of 2^n states (of which 2^{n-1} are final).*
(e) *If L is accepted by an n-state* BFA (AFA), *then $\overline{L}$ is accepted by an n-state* BFA (AFA, *respectively*).
(f) *If L is accepted by an n-state* BFA, *then L is accepted by an* AFA *of at most $n + 1$ states, and by an* NNFA *of at most 2^n states.*
(g) *If L is accepted by an n-state* NNFA, *then L is accepted by an* NFA *of at most $n + 1$ states and by a* pDFA *of at most $2^n - 1$ states. If L is accepted by an n-state* pDFA, *then L is accepted by a* DFA *of at most $n + 1$ states.*

Proof. (a) ($\Rightarrow$; cf. [6, Theorem 4.1, Corollary 4.2] and [9, Lemma 1]).

Let $A = (\{q_1, q_2, \ldots, q_n\}, \Sigma, \delta, g_s, F)$ be an n-state BFA for L. Construct a 2^n-state NFA $A' = (\{0,1\}^n, \Sigma, \delta', S, \{f\})$, where

- for every $u = (u_1 \ldots, u_n) \in \{0,1\}^n$ and every $a \in \Sigma$, $\delta'(u, a) = \{u' \in \{0,1\}^n \mid \delta(q_i, a)(u') = u_i \text{ for } i = 1, \ldots, n\}$;
- $S = \{(b_1, \ldots, b_n) \in \{0,1\}^n \mid g_s(b_1, \ldots, b_n) = 1\}$;
- $f = (f_1, \ldots, f_n) \in \{0,1\}^n$ with $f_i = 1$ iff $q_i \in F$.

Then $L(A) = L(A')$ and $(A')^R$ is deterministic. Moreover if A is an AFA then A' has 2^{n-1} initial states. It follows that L^R is accepted by a DFA with 2^n states, of which 2^{n-1} are final if A is an AFA.

($\Leftarrow$; cf. [9, Lemma 2]) Consider 2^n-state NFA A^R for L which has exactly one final state and the set of initial states S (and $|S| = 2^{n-1}$). Let the state set Q of A^R be $\{0, 1, \ldots, 2^n - 1\}$ with final state k and the initial set S ($S = \{2^{n-1}, \ldots, 2^n - 1\}$). Let δ be the transition function of A^R. Moreover, for every $a \in \Sigma$ and for every $i \in Q$, there is exactly one state j such that j goes to i on a in A^R. For a state $i \in Q$, let $\text{bin}(i) = (b_1, \ldots, b_n)$ be the binary n-tuple such that $b_1 b_2 \cdots b_n$ is the binary notation of i on n digits with leading zeros if necessary.

Let us define an n-state BFA $A' = (Q', \Sigma, \delta', g_s, F')$, where $Q' = \{q_1, \ldots, q_n\}$, $F' = \{q_\ell \,|\, \mathrm{bin}(k)_\ell = 1\}$, and $g_s(\mathrm{bin}(i)) = 1$ iff $i \in S$ ($g_s = q_1$). We define δ' to satisfy the condition: for each i in Q and a in Σ, $(\delta'(q_1, a), \ldots, \delta'(q_n, a))(\mathrm{bin}(i)) = \mathrm{bin}(j)$ where $i \in \delta(j, a)$. Then $L(A') = L(A^R)$.

(b)–(d) These are corollaries of case (a).

(e) Let L be accepted by an n-state BFA (AFA). Then, by (a), the language L^R is accepted by a DFA of 2^n states (of which 2^{n-1} are final). Then the complement $\overline{L^R}$ is also accepted by a DFA of 2^n states (of which 2^{n-1} are final). Since $\overline{L^R} = \overline{L}^R$, the claim follows again by (a).

(f) Let L be accepted by an n-state BFA. Then L^R is accepted by a DFA of 2^n states by (a). Add some unreachable final and non-final sink states to get an equivalent DFA of 2^{n+1} states of which 2^n are final. Then, by (a), L is accepted by an $(n+1)$-state AFA. By reversing the 2^n-state DFA for L^R, we get a 2^n-state NNFA for L.

(g) These claims are well-known. □

If u, v, and w are strings over Σ such that $w = uv$, then u is a *prefix* of w and v is a *suffix* of w. A language L is *prefix-closed* (*suffix-closed*) if $w \in L$ implies that every prefix (suffix) of w is in L.

In 1996, Birget [2] studied the state complexity of the "forever" operator $\overline{\Sigma^* \overline{L}}$ on DFAs and NFAs. Here we continue this research and to simplify the exposition, we use the following notation:

$$f_L = \overline{\Sigma^* \overline{L}}. \tag{1}$$

3. Descriptional Complexity of the Forever Operator

We start with an investigation of some properties of the "forever" operator.

Lemma 6 (Properties of $\overline{\Sigma^* \overline{L}}$). *Let L be a regular language and $f_L = \overline{\Sigma^* \overline{L}}$. Then*

(a) $f_L = \{w \in L \,|\, \textit{every suffix of } w \textit{ is in } L\}$;
(b) $f_L = \emptyset$ *if and only if* $\varepsilon \notin L$;
(c) $f_L = L$ *if and only if L is suffix-closed.*
(d) *If L^R is accepted by a* DFA *A, then f_L^R is accepted by a* DFA *obtained from A by replacing each non-final state of A with a non-final sink state and by a* pDFA *obtained from A by omitting each non-final state of A.*

Proof. The claim (a) follows directly from the definition of f_L, and (b) and (c) follow directly from (a).

(d) We have $f_L^R = \overline{\overline{L^R}\Sigma^*}$. To get a DFA for $\overline{L^R}$, we interchange final and non-final states in A. Then, to get a DFA for $\overline{L^R}\Sigma^*$, we replace all the out-transitions in every final state with loops on every input symbol. Finally, to get a DFA for f_L^R,

we again interchange final and non-final states. Now, all non-final states are sink states. We can omit all of them to get a pDFA for f_L^R. □

In what follows we consider six models of finite automata: DFAs, pDFAs, NFAs, NNFAs, AFAs, and BFAs. We try to answer the following question. If a language L is represented by an n-state automaton of some model, how many states are sufficient and necessary in the worst case for an automaton of some other model to accept the language $f_L = \overline{\Sigma^*\overline{L}}$? We first consider upper bounds. Although we have 36 possible trade-offs, it is enough to prove only some of them. The remaining trade-offs follow either from inclusions of some models of finite automata or from Lemma 5. For the (N)NFA-to-(p)DFA trade-offs, we need the Dedekind number $\mathrm{M}(n)$ which counts the number of antichains of subsets of an n-element set. The number $\mathrm{M}(n)$ lies in the order of magnitude $2^{2^{\Theta(n)}}$ [13]:

$$2^{n-\log n} \leq \binom{n}{\lfloor n/2 \rfloor} \leq \log \mathrm{M}(n) \leq \binom{n}{\lfloor n/2 \rfloor}\left(1 + O\left(\frac{\log n}{n}\right)\right) \leq 2^{n+1-(\log n)/2}.$$

It follows that $\log \mathrm{M}(n)$ lies in the order of magnitude $2^{n-|\Theta(\log n)|}$. Moreover, we assume that $\varepsilon \in L$ and $L \neq \Sigma^*$ in the statement of the next theorem because otherwise f_L is empty or equals Σ^* by Lemma 6(b) and (c).

Theorem 7 (Upper Bounds). *Let $n \geq 3$ and $f_L = \overline{\Sigma^*\overline{L}}$. Let L be a regular language such that $\varepsilon \in L$ and $L \neq \Sigma^*$. Let L be accepted by a finite automaton A of n states.*

(1) *If A is a* DFA, *then f_L is accepted by a* DFA *of at most 2^{n-1} states.*
(2) *If A is a* pDFA, *then f_L is accepted by a* pDFA *of at most 2^{n-1} states.*
(3) *If A is an* NFA, *then f_L is accepted by*
   (a) *an* NFA *of at most 2^{n-1} states*;
   (b) *a* pDFA *of at most $\mathrm{M}(n-1)-1$ states.*
(4) *If A is an* NNFA, *then f_L is accepted by*
   (a) *an* NNFA *of at most 2^n-2 states.*
   (b) *a* pDFA *of at most $\mathrm{M}(n)-1$ states.*
(5) *If A is an* AFA, *then f_L is accepted by*
   (a) *an* AFA *of at most n states*;
   (b) *an* NNFA *of at most 2^{n-1} states.*
(6) *If A is a* BFA, *then f_L is accepted by*
   (a) *a* BFA *of at most n states*;
   (b) *an* NNFA *of at most 2^n-1 states.*

Proof. (1) We provide a simple alternative proof to [2, Theorem 1(b)]. We first interchange final and non-final states in A to get the DFA $\overline{A}$ for $\overline{L}$. Then we add a loop on every input symbol in the initial state of $\overline{A}$ to get an NFA N for $\Sigma^*\overline{L}$.

In $\mathcal{D}(N)$, only subsets containing the initial state are reachable. Finally, we again interchange the final and non-final states of $\mathcal{D}(N)$.

(2) Let $A = (Q, \Sigma, \cdot, s, F)$ be an n-state pDFA for L. It is enough to show that the language $\Sigma^*\overline{L}$ is accepted by a DFA of at most $2^{n-1}+1$ states, one of which is final sink state. To get an $(n+1)$-state DFA $\overline{A}$ for $\overline{L}$, we first add a new non-final sink state q_d to A. Then, for each transition which is undefined in A, we add the corresponding transition to q_d. Finally, we interchange final and non-final states of the resulting automaton. We construct an $(n+1)$-state NFA N for $\Sigma^*\overline{L}$ from DFA $\overline{A}$, by adding a loop on each input symbol in the initial state s. In the corresponding subset automaton, each reachable subset must contain s. Moreover, the state q_d is a final sink state. It follows that each string is accepted by N from q_d, and therefore each subset containing q_d, is equivalent to $\{q_d\}$. In total, we get at most $2^{n-1}+1$ reachable and pairwise distinguishable states.

(3a) Let $A = (Q, \Sigma, \cdot, s, F)$ be an n-state NFA for L. We reverse A to get an n-state NNFA A^R for L^R with a unique final state s. In the corresponding subset automaton $\mathcal{D}(A^R)$, using Lemma 6(d), we omit all the non-final subsets, that is, all subsets not containing s, to get a 2^{n-1}-state pDFA B for f_L^R. We have two cases. If there is a final subset which is not reachable in B, then we reverse B and add a new initial state to get an NFA for f_L of at most 2^{n-1} states. Otherwise, each final subset, that is, each subset containing s is reachable in B. We show that if a string w is accepted by B from the state $\{s\}$, then w is accepted by B from any other state. The claim holds for $w = \varepsilon$ since all states of B are final. Let $w = a_1 a_2 \cdots a_k$, where $a_i \in \Sigma$, be accepted by B from $\{s\}$. Then in B, we have the following computation:

$$\{s\} \xrightarrow{a_1} S_1 \xrightarrow{a_2} S_2 \xrightarrow{a_3} \cdots \xrightarrow{a_k} S_k,$$

where $s \in S_i$ since each state of B contains s. Notice that any other state S of B is a superset of $\{s\}$. Recall that B is derived from the subset automaton $\mathcal{D}(A^R)$ in which we must have

$$S \xrightarrow{a_1} S'_1 \xrightarrow{a_2} S'_2 \xrightarrow{a_3} \cdots \xrightarrow{a_k} S'_k \tag{2}$$

for some sets $S'_1, S'_2, \ldots, S'_k$ such that $S_i \subseteq S'_i$ $(1 \leq i \leq k)$. Since each S_i contains s, each S'_i must contain s as well. It follows that (2) is a computation in B, so w is accepted by B from S; notice that each set S_i is reachable in B.

We modify pDFA B as follows. We make all states of B non-final, except for $\{s\}$. Next, we add the ε-transition to $\{s\}$ from any other state in B. Denote the resulting NFA by B'. Then $L(B) \subseteq L(B')$. Let us show that $L(B') \subseteq L(B)$. Let w be accepted by B'. If $w = \varepsilon$, then $w \in L(B)$. Otherwise w can be partitioned as $w = w_1 w_2 \cdots w_k$ and in B' we have the following computation on w:

$$F \xrightarrow{w_1} S_1 \xrightarrow{\varepsilon} \{s\} \xrightarrow{w_2} S_2 \xrightarrow{\varepsilon} \{s\} \xrightarrow{w_3} \cdots \xrightarrow{\varepsilon} \{s\} \xrightarrow{w_k} S_k$$

in which while reading each w_i no added ε-transition is used, so $\{s\} \xrightarrow{w_i} S_i$ is a computation in B. As shown above, each w_i is accepted by B from each state of B. It follows that in B we have an accepting computation $F \xrightarrow{w_1} S_1 \xrightarrow{w_2} S'_2 \xrightarrow{w_3} \cdots \xrightarrow{w_k} S'_k$

for some states $S'_2, \ldots, S'_k$ of B. Hence w is accepted by B, so $L(B') \subseteq L(B)$. This means that B' is a 2^{n-1}-state NFA with one final state for f_L^R. By reversing B' and removing ε-transitions, we get a 2^{n-1}-state NFA for f_L.

(3b) It is enough to show that $\Sigma^*\overline{L}$ is accepted by a DFA of at most $\mathrm{M}(n-1)$ states, one of which is a final sink state. Let $A = (Q, \Sigma, \cdot, s, F)$ be an n-state NFA for L, and B be the 2^n-state subset automaton of A. We interchange the final and non-final states in B, to get a 2^n-state DFA $\overline{B}$ for $\overline{L}$. To get a 2^n-state NFA N for $\Sigma^*\overline{L}$, we add a loop on each input symbol in the initial state of the DFA $\overline{B}$. Finally, let C be the subset automaton of N. Then C is a DFA for $\Sigma^*\overline{L}$. Formally, we have

$$
\begin{aligned}
&B = \mathcal{D}(A) = (2^Q, \Sigma, \cdot, \{s\}, F_B) \text{ where } F_B = \{X \subseteq Q \mid X \cap F \neq \emptyset\});\\
&\overline{B} = (2^Q, \Sigma, \cdot, \{s\}, F_{\overline{B}}) \text{ where } F_{\overline{B}} = 2^Q \backslash F_B = \{X \subseteq Q \mid X \subseteq Q \backslash F\});\\
&N = (2^Q, \Sigma, \circ, \{s\}, F_{\overline{B}}) \text{ where for each } X \in 2^Q \text{ and each } a \text{ in } \Sigma,\\
&\qquad\qquad \{s\} \circ a = \{\{s\}, \{s\} \cdot a\} \text{ and } X \circ a = \{X \cdot a\} \text{ if } X \neq \{s\};\\
&C = \mathcal{D}(N) = (2^{2^Q}, \Sigma, \circ, \{\{s\}\}, F_C) \textit{ where } F_C = \{X \in 2^{2^Q} \mid X \cap F_{\overline{B}} \neq \emptyset\}.
\end{aligned}
$$

Thus, the states of C are sets of subsets of Q, and a state $\mathcal{S} = \{S_1, S_2, \ldots, S_k\}$ is final if there is an i such that $S_i \subseteq Q \backslash F$. Our aim is to show that C has at most $\mathrm{M}(n-1)$ reachable and pairwise distinguishable states. We first show that each state of C is equivalent to an antichain in 2^Q.

Let $S \subseteq T \subseteq Q$ and w be accepted by N from the state T. Let us show that w is accepted by N also from the state S. If w is accepted from T, then there is a computation in N on w starting in T and ending in a final state T' of N. If this computation does not use any transition which was added to get N from $\overline{B}$, then we have the same computation in $\overline{B}$. This means that $T \cdot w \subseteq Q \backslash F$, and since B is the subset automaton of A, we have $S \cdot w \subseteq T \cdot w \subseteq Q \backslash F$. Therefore w is accepted by $\overline{B}$ from S. Since in N we have the same computation from S on w, the string w is accepted by N from S. Now assume that w is accepted by N from T by a computation using an added transition. Then w can be partitioned as $w = uv$, where $T \xrightarrow{u} \{s\} \xrightarrow{v} T'$ and $T' \subseteq Q \backslash F$, and moreover, while reading u, no added transition is used. It follows that in $\overline{B}$, we have $T \cdot u = \{s\}$, and therefore also $S \cdot u \subseteq \{s\}$. If $S \cdot u = \emptyset$, then $S \cdot w = \emptyset$, and therefore also $S \circ w = \emptyset$, so w is accepted by N from S. If $S \cdot u = \{s\}$, then in N we have $S \xrightarrow{u} \{s\} \xrightarrow{v} T'$, which means that w is accepted by N from S.

Thus if in a state $\mathcal{S} = \{S_1, S_2, \ldots, S_k\}$ of C we have $S_i \subseteq S_j$ for some i and j, then $\mathcal{S}$ is equivalent to $\mathcal{S} \backslash \{S_j\}$. It follows that each state of C is equivalent to an antichain in 2^Q. Moreover, since N has a loop on each symbol in its initial state $\{s\}$, and C is the subset automaton of N, each reachable state of C must contain the set $\{s\}$, that is, each reachable antichain has a form $\{\{s\}, S_2, S_3, \ldots, S_k\}$, where $k \geq 1$, and $\{S_2, S_3, \ldots, S_k\}$ is an antichain in $2^{Q \backslash \{s\}}$. This gives the upper bound $\mathrm{M}(n-1)$. Notice that the empty antichain corresponds to the initial state $\{\{s\}\}$. We also have to count the antichain $\{\emptyset\}$ which is an unreachable final sink state, but it is equivalent to the reachable state $\{\{s\}, \emptyset\}$.

(4a) If all states of a given NNFA are initial, then L is suffix-closed and $f_L = L$ by Lemma 6(c). Otherwise, L^R is accepted by an 2^n-state DFA which has at least two non-final states. Omit all the non-final states to get a pDFA for f_L^R (cf. Lemma 6(3b)), and reverse the resulting pDFA to get the desired NNFA for f_L.

(4b) It is enough to show that $\Sigma^*\overline{L}$ is accepted by a DFA of at most $\mathrm{M}(n)$ states, one of which is a final sink state. Let $A = (Q, \Sigma, \cdot, I, F)$ be an n-state NNFA for L, and B be the 2^n-state subset automaton of A. In the same way as in the case (3b),

$$
\begin{aligned}
&B = \mathcal{D}(A) = (2^Q, \Sigma, \cdot, I, F_B) \text{ where } F_B = \{X \subseteq Q \mid X \cap F \neq \emptyset\});\\
&\overline{B} = (2^Q, \Sigma, \cdot, I, F_{\overline{B}}) \text{ where } F_{\overline{B}} = 2^Q \backslash F_B = \{X \subseteq Q \mid X \subseteq Q \backslash F\});\\
&N = (2^Q, \Sigma, \circ, I, F_{\overline{B}}) \text{ where for each } X \in 2^Q \text{ and each } a \text{ in } \Sigma,\\
&\qquad\qquad I \circ a = \{I, I \cdot a\}, \text{ and } X \circ a = \{X \cdot a\} \text{ if } X \neq I;\\
&C = \mathcal{D}(N) = (2^{2^Q}, \Sigma, \circ, \{I\}, F_C) \text{ where } F_C = \{X \in 2^{2^Q} \mid X \cap F_{\overline{B}} \neq \emptyset\}.
\end{aligned}
$$

The states of C are sets of subsets of Q, and a state $\mathcal{S} = \{S_1, S_2, \ldots, S_k\}$ is final if there is an i such that $S_i \subseteq Q \backslash F$. Our aim is to show that C has at most $\mathrm{M}(n)$ reachable and pairwise distinguishable states. We first show that each state of C is equivalent to an antichain in 2^Q. Let $S \subseteq T \subseteq Q$. We show that the state $\{I, S, T\}$ is equivalent to $\{I, S\}$ in C. To this aim, let w be accepted by N from T. If no added transition is used in this computation, then similarly as in the case (3b), w is accepted by C from S. Otherwise $w = uv$ where $T \xrightarrow{u} I \xrightarrow{v} T'$ with $T' \subseteq Q \backslash F$. Since the state I has a loop on each symbol, in C we have $I \xrightarrow{u} I \xrightarrow{v} T'$, so the string w is accepted by C from I. It follows that the state $\{I, S, T\}$ is equivalent to $\{I, S\}$ in C. Thus if in a state $\mathcal{S} = \{S_1, S_2, \ldots, S_k\}$ of C we have $S_i \subseteq S_j$ for some i and j, then $\mathcal{S}$ is equivalent to $\mathcal{S} \backslash \{S_j\}$. It follows that each state of C is equivalent to an antichain in 2^Q. Since the number of antichains in 2^Q is $\mathrm{M}(n)$, and the reachable state $\{I, \emptyset\}$ is equivalent to the unreachable antichain $\{\emptyset\}$.

(5a) If L is accepted by an n-state AFA, then L^R is accepted by a DFA of 2^n states of which 2^{n-1} are final. Replace each non-final state with a non-final sink state to get a DFA for f_L^R of 2^n states of which 2^{n-1} are final. Hence f_L is accepted by an n-state AFA.

(5b) In the DFA for f_L^R obtained as in case (5a), we omit the non-final sink states to get an equivalent pDFA of 2^{n-1} states. By reversing this pDFA, we get a 2^{n-1}-state NNFA for f_L.

(6a) If A is an n-state BFA, then L^R is accepted by a DFA of 2^n states. Replace each non-final state with a non-final sink state to get a DFA for f_L^R of 2^n states given by Lemma 5(f). Hence f_L is accepted by an n-state BFA.

(6b) In the DFA for f_L^R obtained as in case (6a), we omit the non-final sink states to get an equivalent pDFA of at most $2^n - 1$ states; recall that $L \neq \Sigma^*$. By reversing this pDFA, we get the desired NNFA for f_L. □

Now we turn our attention to lower bounds. We again need to prove only some of them. All the remaining bounds follow from the inclusions of models or from

Lemma 5. However, in some cases, we use witnesses over a smaller alphabet for the bound that follows from some other trade-off. In 32 of 36 cases, our lower bounds meet the upper bounds given by Theorem 7. The remaining four cases are the trade-offs from NNFA to DFA, pDFA, NFA, and NNFA. Except for four trade-offs, our witnesses are defined over a fixed alphabet of size one, two, three, or four. The binary case is always optimal in the sense that there is no unary witness language.

Theorem 8 (Lower Bounds). *Let $n \geq 3$ and $f_L = \overline{\Sigma^* \overline{L}}$. There exists a regular language L accepted by an n-state finite automaton A such that A is*

(1) *a ternary* DFA *and every* BFA *for f_L has at least n states*;
(2) *a ternary* DFA *and every* NNFA *for f_L has at least 2^{n-1} states*;
(3) *a binary* DFA *and every* pDFA *for f_L has at least 2^{n-1} states*;
(4) *a binary* pDFA *and every* BFA *for f_L has at least n states*;
(5) *a quaternary* pDFA *and every* DFA *for f_L has at least $2^{n-1}+1$ states*;
(6) *an* NFA *and every* DFA *for f_L has at least $\mathrm{M}(n-1)$ states*;
(7) *a binary* NNFA *and every* AFA *for f_L has at least $n+1$ states*;
(8) *a unary* AFA *and*
   (a) *every* BFA *for f_L has at least n states*;
   (b) *every* NNFA *for f_L has at least 2^{n-1} states*;
(9) *a binary* AFA *and*
   (a) *every* NFA *for f_L has at least $2^{n-1}+1$ states*;
   (b) *every* DFA *for f_L has at least $2^{2^{n-1}}$ states*;
(10) *a unary* BFA *and*
   (a) *every* AFA *for f_L has at least $n+1$ states*;
   (b) *every* NNFA *for f_L has at least 2^n-1 states*;
(11) *a binary* BFA *and*
   (a) *every* NFA *for f_L has at least 2^n states*;
   (b) *every* DFA *for f_L has at least 2^{2^n-1} states.*

Proof. (1) Let L be the language accepted by the DFA A shown in Fig. 3. We reverse A to get an NFA A^R for L^R. In the corresponding subset automaton, all (final) subsets containing 0 are reachable by Proposition 1, and the empty set is reached from $\{0\}$ by c. Notice that no other subset is reachable. Moreover, the subset automaton does not have equivalent states since the state 0 is uniquely

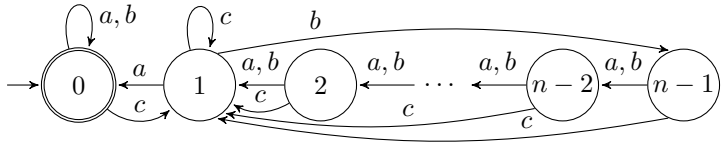


Fig. 3. The DFA for L such that every BFA for $\overline{\Sigma^* \overline{L}}$ has n states.

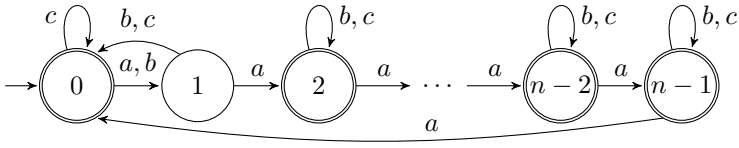


Fig. 4. The DFA from [2] for L such that every NNFA for $\overline{\Sigma^*\overline{L}}$ has 2^{n-1} states.

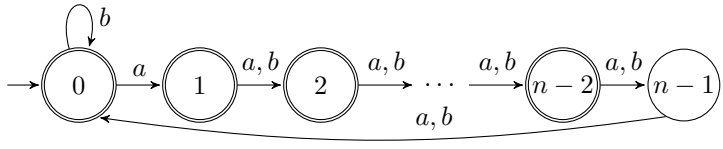


Fig. 5. The binary DFA for L such that every pDFA for $\overline{\Sigma^*\overline{L}}$ has 2^{n-1} states.

distinguishable in A^R by ε, and it is uniquely reachable from any other state of A^R through unique in-transitions $2 \xrightarrow{b} 3 \xrightarrow{b} \cdots \xrightarrow{b} n-1 \xrightarrow{b} 1 \xrightarrow{c} 0$. Since in the minimal DFA for L^R we have all states final but one non-final sink state, the language L^R is prefix-closed. Therefore L is suffix-closed, so $f_L = L$. Since the minimal DFA for L^R has $2^{n-1}+1$ states, every BFA for L, so for f_L, has at least n states.

(2) This case follows from [2, Proof of Theorem 2(a)].

(3) Let L be accepted by the n-state DFA A shown in Fig. 5. We construct an n-state NFA N for $\Sigma^*\overline{L}$ by interchanging final and non-final states in A and by adding the transition $(0, a, 0)$. It is enough to prove that the subset automaton $\mathcal{D}(N)$ has at least 2^{n-1} reachable and pairwise distinguishable states. We prove reachability by using Proposition 1. To prove distinguishability, notice that the state $n-1$ is uniquely distinguishable by ε in N and it is uniquely reachable from any other state through unique in-transitions on a. By Proposition 2, the subset automaton $\mathcal{D}(N)$ does not have equivalent states. Since $\mathcal{D}(N)$ has no non-final sink state, it is also a minimal pDFA. Notice that the lower bound 2^{n-1} for a DFA accepting f_L follows from the proof. In [2, Proof of Theorem 2(b)], it is claimed that this bound is met by the complement of binary language $a\{a,b\}^{n-2}$. However, the minimal DFA for this language has $n+1$ states.

(4) Let L be the language accepted by the pDFA A shown in Fig. 6. We reverse A to get an NNFA for L^R. In the corresponding subset automaton, we replace every non-final state with a single non-final sink state. By Lemma 6(d), we get a DFA B for f_L^R with 2^{n-1} final states. We reach all of them using induction on the size of subsets. The set $\{0, 1, \ldots, n-1\}$ of size n is the basis. Each subset S with $0 \in S$ and $t \notin S$ of size $k-1$, where $1 \leq k \leq n$, is reachable from the subset $S \cup \{t\}$ of size k by the string $a^{t-1}ba^{n-t}$. By doing this, we always keep the state 0 in the set since symbol a performs a loop on state 0 and symbol b moves state 1 to 0. This means that all sets with 0 are reachable in B through final states. The non-final empty set is reached from $\{0\}$ on b. For distinguishability, let S and T be two different subsets

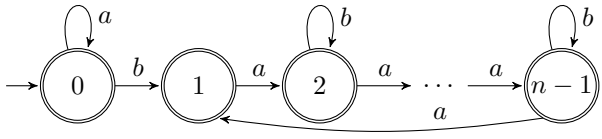


Fig. 6. The pDFA for L such that every BFA for $\overline{\Sigma^*\overline{L}}$ has n states.

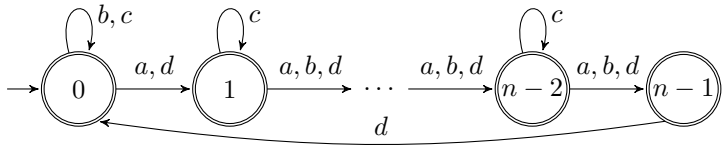


Fig. 7. The pDFA for L such that every DFA for $\overline{\Sigma^*\overline{L}}$ has $2^{n-1}+1$ states.

such that $t \notin S$ and $t \in T$. Then $a^{t-1}b$ is rejected from S and accepted from T. Since a performs a loop on 0 and b moves 1 to 0, we are always in final states. So we distinguish every pair of 2^{n-1} final states. By Lemma 5, every BFA for f_L has at least n states.

(5) Let L be the language accepted by the pDFA A shown in Fig. 7. We construct an $(n+1)$-state NFA N for $\Sigma^*\overline{L}$ as follows. First, we add a new non-final sink state n and the transitions on a, b, c from $n-1$ to n. Then we make state n final, and all the remaining states non-final. Finally, we add the transitions $(0, a, 0)$ and $(0, d, 0)$. By Proposition 1, in the subset automaton $\mathcal{D}(N)$, all the subsets containing 0 are reachable from the initial subset $\{0\}$ via strings over $\{a, b\}$. All the subsets containing n are final and equivalent to $\{n\}$. All the remaining subsets are non-final. Two distinct subsets of $\{0, 1, \ldots, n-1\}$ differ in a state i, and the string $d^{n-1-i}c$ distinguishes the two subsets; notice that as for the states in $\{0, 1, \ldots, n-1\}$, the string c is accepted only from $n-1$, the state $n-1$ is uniquely reachable from any other state through unique in-transitions $0 \xrightarrow{d} 1 \xrightarrow{d} \cdots \xrightarrow{d} n-1$, and therefore $d^{n-1-i}c$ is accepted only from the state i. Thus the subset automaton $\mathcal{D}(N)$ has at least $2^{n-1}+1$ reachable and pairwise distinguishable states. It is claimed in [2, Theorem 1(b)] that the upper bound in this case is 2^{n-1}. The proof of [2, Theorem 1(b)] does not work in the case of a partial automaton for L since in such a case we can reach an accepting state in the automaton $\mathbf{B}$ also by a string which is not in f_L. Our witness fixes this small inaccuracy.

(6) Let L be accepted by the n-state NFA $A = (Q, \Sigma, \cdot, 0, F)$, where $Q = \{0, 1, \ldots, n-1\}$, $\Sigma = \{a_X, b_X \mid X \subseteq Q\}$, $F = Q \setminus \{n-1\}$, and the transition function is defined as follows:

$$0 \cdot a_X = X \text{ and } i \cdot a_X = \{i\} \text{ if } i \neq 0,$$

$$i \cdot b_X = \begin{cases} \{n-1\}, & \text{if } i \in X; \\ \{0\}, & \text{if } i \notin X; \end{cases}$$

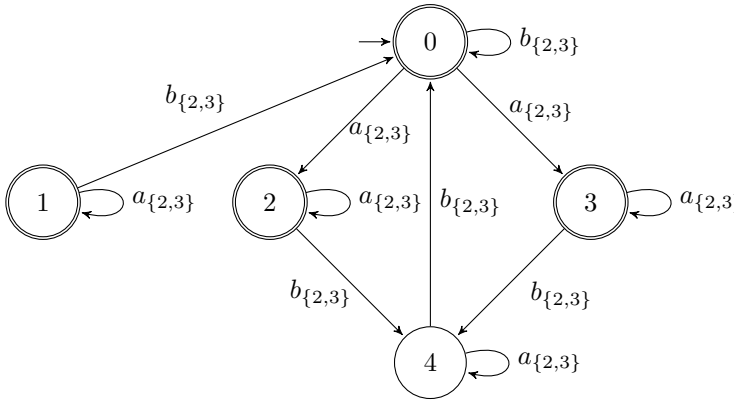


Fig. 8. The example of NFA for L such that every DFA for $\overline{\Sigma^* \overline{L}}$ has $\mathrm{M}(n-1)$ states. The alphabet is $\{a_X, b_X \mid X \subseteq Q\}$, only transitions on $a_{\{2,3\}}$ and $b_{\{2,3\}}$ are shown.

see Fig. 8 for an illustration. Then

$$
\begin{aligned}
B &= \mathcal{D}(A) = (2^Q, \Sigma, \cdot, \{0\}, 2^Q \backslash \{\{n-1\}, \emptyset\});\\
\overline{B} &= (2^Q, \Sigma, \cdot, \{0\}, \{\{n-1\}, \emptyset\});\\
N &= (2^Q, \Sigma, \circ, \{0\}, \{\{n-1\}, \emptyset\}) \text{ where } \{0\} \circ a = \{0\} \cup \{0\} \cdot a \text{ and}\\
&\qquad\qquad X \circ a = X \cdot a \text{ if } X \neq \{0\};\\
C &= \mathcal{D}(N) = (2^{2^Q}, \Sigma, \circ, \{\{0\}\}, \{X \in 2^{2^Q} \mid X \cap \{\{n-1\}, \emptyset\} \neq \emptyset\}).
\end{aligned}
$$

Our aim is to show that C has at least $\mathrm{M}(n-1)$ reachable and distinguishable states. Let $S_1, S_2, \ldots, S_k$ be subsets of Q such that $0 \notin S_i$ for every i. Then in C we have $\{\{0\}\} \xrightarrow{a_{S_1}} \{\{0\}, S_1\} \xrightarrow{a_{S_2}} \{\{0\}, S_1, S_2\} \xrightarrow{a_{S_3}} \cdots \xrightarrow{a_{S_k}} \{\{0\}, S_1, S_2, \ldots, S_k\}$. It follows that every state $\mathcal{S} = \{\{0\}, S_1, S_2, \ldots, S_k\}$ where $\{S_1, S_2, \ldots, S_k\}$ is an antichain of subsets of $\{1, 2, \ldots, n-1\}$ is reachable. To prove distinguishability, let $\mathcal{S} = \{\{0\}, S_1, S_2, \ldots, S_k\}$ and $\mathcal{T} = \{\{0\}, T_1, T_2, \ldots, T_\ell\}$ be two distinct reachable antichains in C. Then there exists a subset X of $\{1, 2, \ldots, n-1\}$ such that, without loss of generality, $X \in \mathcal{S} \backslash \mathcal{T}$. We have two cases. (i) No subset of X is in $\mathcal{T}$. Then b_X is accepted from $\mathcal{S}$ since $X \in \mathcal{S}$, $X \circ b_X = X \cdot b_X = \{n-1\}$, and therefore $\{n-1\} \in \mathcal{S} \circ b_X$. Hence b_X is accepted by C from $\mathcal{S}$. On the other hand, we have $0 \in \{0\} \circ b_X$. Next, since T_j $(1 \leq j \leq \ell)$ is not a subset of X, there is a state i such that $i \in T_j \backslash X$. Since $i \cdot b_X = \{0\}$, we must have $0 \in T_j \circ b_X$. Hence b_X is rejected by C from $\mathcal{T}$. (ii) There is a subset Y of X such that $Y \in \mathcal{T}$. Then no subset of Y is in $\mathcal{S}$ since $\mathcal{S}$ is an antichain and $X \in \mathcal{S}$. By the former case, b_Y is accepted from $\mathcal{T}$ and rejected from $\mathcal{S}$. Thus C has $\mathrm{M}(n-1)$ reachable and distinguishable states.

(7) Let L be accepted by the NNFA A from Fig. 9. Since each state of A is initial, L is suffix-closed, so $f_L = L$. To show that every AFA for f_L, so for L, has $n+1$ states, it is enough to show that the minimal DFA for L^R has more than 2^{n-1} final states. Since the reverse of A is isomorphic to A, we have $L^R = L$. In

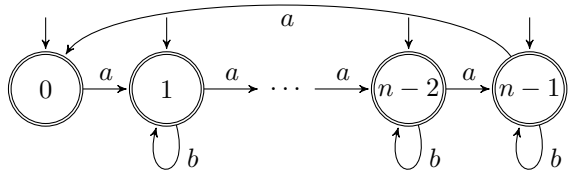


Fig. 9. The NNFA for L such that every AFA for $\overline{\Sigma^* \overline{L}}$ has $n+1$ states.

the subset automaton $\mathcal{D}(A)$, the initial subset is $\{0, 1, \ldots, n-1\}$. By Proposition 1, every subset is reachable in $\mathcal{D}(A)$. Next, the state 0 is uniquely distinguishable in A by the string $(ab)^{n-1}$, and it is uniquely reachable from any other state of A. Thus $\mathcal{D}(A)$ is minimal. Since $\mathcal{D}(A)$ has $2^n - 1$ final states, every AFA for L, so for f_L, has at least $n+1$ states.

(8) Let $L = \{a^i \mid 0 \leq i \leq 2^{n-1} - 1\}$. Then L is a unary language accepted by a 2^n-state DFA with 2^{n-1} final states. So L is accepted by an n-state AFA. Since L is suffix-closed, $f_L = L$. (a) Since the minimal DFA for L has $2^{n-1}+1$ states, every BFA for L has at least n states. (b) The longest string in L is of length $2^{n-1} - 1$, and therefore every NNFA for L has at least 2^{n-1} states.

(9) Set $m = 2^{n-1}$ in Fig. 2. Let K be accepted by the 2^n-state DFA A in which the transitions on final states $0, 1, \ldots, 2^{n-1} - 1$ are shown in Fig. 2. Moreover, A has 2^{n-1} non-final sink states. Set $L = K^R$. Then L is accepted by an n-state AFA. By Lemma 6(d), if we omit all non-final states of A, we get a pDFA for f_L^R. By Proposition 4, we get that (a) every NFA for f_L has at least $2^{n-1}+1$ states, and (b) every DFA for f_L has $2^{2^{n-1}}$ states.

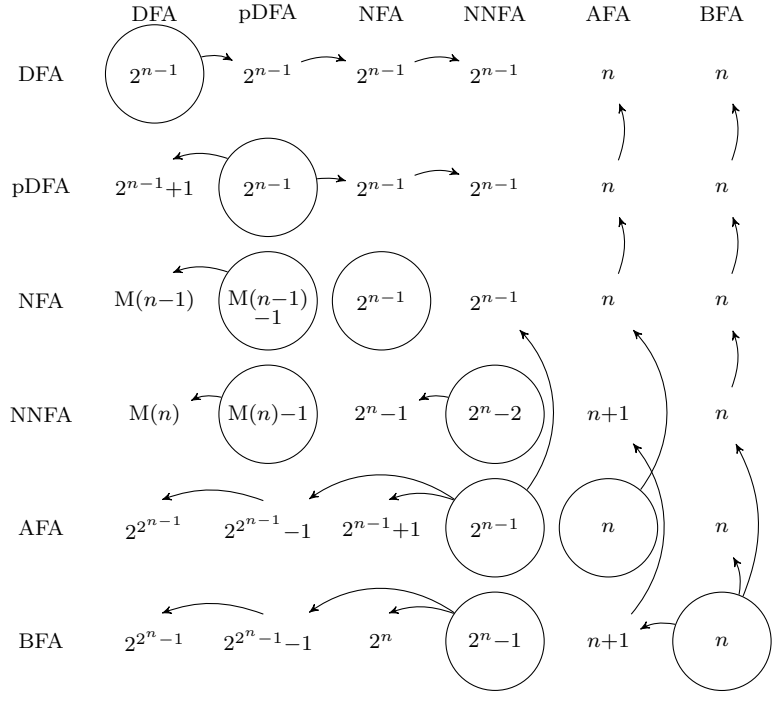
(10) Let $L = \{a^i \mid 0 \leq i \leq 2^n - 2\}$. Then L is a unary language accepted by a minimal 2^n-state DFA A, so L is accepted by an n-state BFA. Since L is suffix-closed, $f_L = L$. (a) Every AFA accepting L has at least $n+1$ states since the number of final states in A is greater than 2^{n-1}. (b) The longest string in L is of length $2^n - 2$, and therefore every NNFA for L has at least $2^n - 1$ states.

(11) Set $m = 2^n - 1$ in Fig. 2. Now the proof goes exactly the same way as in case (9). It results in the lower bound 2^n for NFAs and 2^{2^n-1} for DFAs. □

The upper and lower bounds from Theorems 7 and 8 are shown in Table 1 as circled. If an upper or lower bound in one cell follows from the bound in another cell, this is denoted by an arrow. Another witness for the same lower bound, but using a smaller alphabet, is circled dashed. As a corollary, we get the results that are displayed in Table 2. The table also shows the size of alphabets used for describing witness languages.

Table 3 shows the upper bounds on the complexity of the forever operator on unary regular languages; here $\mathrm{F}(n)$ denotes the Landau function defined as $\mathrm{F}(n) = \max\{\mathrm{lcm}(x_1, \ldots, x_k) \mid n = x_1 + \cdots + x_k\}$. In six cases, namely {AFA, BFA}-to-{NNFA, AFA, BFA}, the upper bounds are the same as for general languages and

Table 1. Upper (top) and lower bounds (bottom) on the complexity of $\overline{\Sigma^* \overline{L}}$.



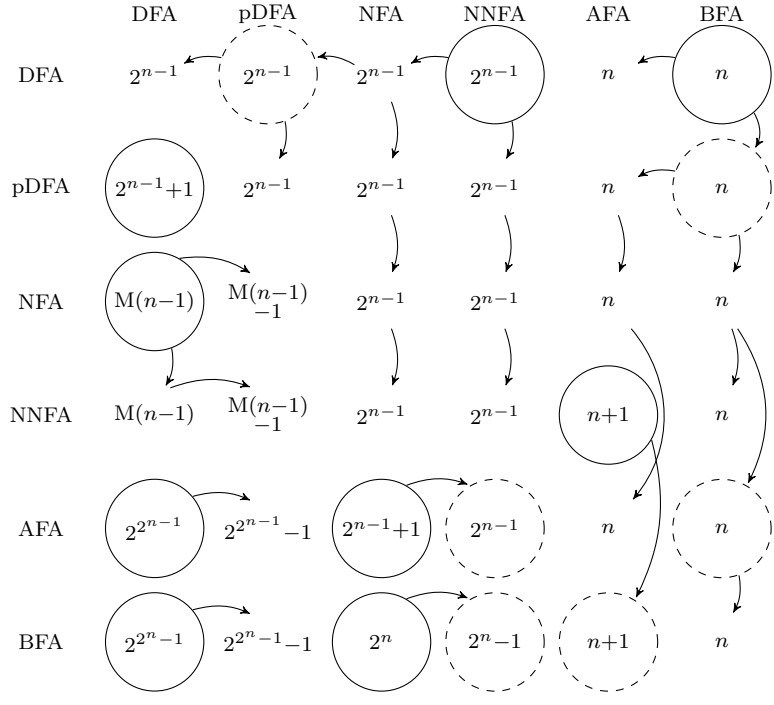


Table 2. The complexity of $\overline{\Sigma^*\overline{L}}$ for various types of finite automata. The DFA-NFA and DFA-NNFA trade-offs are from [2].

$L\backslash\overline{\Sigma^*\overline{L}}$	DFA	$\lvert\Sigma\rvert$	pDFA	$\lvert\Sigma\rvert$	NFA	$\lvert\Sigma\rvert$	NNFA	$\lvert\Sigma\rvert$	AFA	$\lvert\Sigma\rvert$	BFA	$\lvert\Sigma\rvert$
DFA	2^{n-1}	2	2^{n-1}	2	2^{n-1}	3	2^{n-1}	3	n	3	n	3
pDFA	$2^{n-1}+1$	4	2^{n-1}	2	2^{n-1}	3	2^{n-1}	3	n	2	n	2
NFA	$\mathrm{M}(n-1)$	2^{n+1}	$\mathrm{M}(n-1)-1$	2^{n+1}	2^{n-1}	3	2^{n-1}	3	n	2	n	2
NNFA	$\geq \mathrm{M}(n-1)2^{n+1}$ $\leq \mathrm{M}(n)$		$\geq \mathrm{M}(n-1)-12^{n+1}$ $\leq \mathrm{M}(n)-1$		$\geq 2^{n-1}$ $\leq 2^n-1$	3	$\geq 2^{n-1}$ $\leq 2^n-2$	3	$n+1$	2	n	2
AFA	$2^{2^{n-1}}$	2	$2^{2^{n-1}}-1$	2	$2^{n-1}+12$		2^{n-1}	1	n	1	n	1
BFA	2^{2^n-1}	2	$2^{2^n-1}-1$	2	2^n	2	2^n-1	1	$n+1$	1	n	1

Table 3. The upper bounds on the complexity of forever operator in the unary case. We have $\mathrm{D}(n)=\mathrm{F}(n-1)+n^2-2\in 2^{O(\sqrt{n\log n})}$ and $\lceil\log(\mathrm{D}(n))\rceil<n$.

$L\backslash\overline{\Sigma^*\overline{L}}$	DFA	pDFA	NFA	NNFA	AFA	BFA
DFA	n	$n-1$	$n-1$	$n-1$	$\lceil\log n\rceil+1$	$\lceil\log n\rceil$
pDFA	$n+1$	n	n	n	$\lceil\log n\rceil+1$	$\lceil\log(n+1)\rceil$
NFA	$\mathrm{D}(n)$	$\mathrm{D}(n)$	$\mathrm{D}(n)$	$\mathrm{D}(n)$	$\lceil\log(\mathrm{D}(n))\rceil+1$	$\lceil\log(\mathrm{D}(n))\rceil$
NNFA	$\mathrm{D}(n)$	$\mathrm{D}(n)$	$\mathrm{D}(n)$	$\mathrm{D}(n)$	$\lceil\log(\mathrm{D}(n))\rceil+1$	$\lceil\log(\mathrm{D}(n))\rceil$
AFA	$2^{n-1}+1$	2^{n-1}	2^{n-1}	2^{n-1}	n	n
BFA	2^n	2^n-1	2^n-1	2^n-1	$n+1$	n

they are met by unary witnesses. In the remaining cases, the upper bounds are smaller than those in the general case. It follows that a binary alphabet is optimal whenever it is used to describe witnesses in the general case.

4. Conclusions

We investigated the descriptional complexity of $\overline{\Sigma^*\overline{L}}$ over complete and partial deterministic, nondeterministic, alternating, and Boolean finite automata. For each trade-off, except for those starting with NNFAs, we provided tight upper bounds for complexity of $\overline{\Sigma^*\overline{L}}$ depending on the complexity of L. The most interesting result is the tight upper bound on NFA-to-DFA trade-off given by the Dedekind number $\mathrm{M}(n-1)$. However, we used a growing alphabet of size 2^{n+1} to get the lower bound in this case. Except for (N)FA-to-(p)DFA trade-offs, all witnesses are described over an alphabet of fixed size. Moreover, binary and unary alphabets are optimal for their respective cases. Whenever we have a larger alphabet, we do not know whether or not it is optimal. The precise complexity for NNFA-to-(p)DFA and NNFA-to-(N)FA trade-offs remains open as well.

Acknowledgements

A preliminary version of this paper was presented at the conference DLT 2017 and appeared in the proceedings. The research was supported by VEGA Grant 2/0084/15 and grant APVV-15-0091.

References

[1] J. Birget, Intersection and union of regular languages and state complexity, *Inf. Process. Lett.* **43**(4) (1992) 185–190.

[2] J. Birget, The state complexity of $\overline{\Sigma^* \overline{L}}$ and its connection with temporal logic, *Inf. Process. Lett.* **58**(4) (1996) 185–188.

[3] J. A. Brzozowski, G. Jirásková, B. Liu, A. Rajasekaran and M. Szykula, On the state complexity of the shuffle of regular languages, *DCFS 2016*, eds. C. Câmpeanu, F. Manea and J. Shallit, LNCS, Vol. 9777 (Springer, 2016), pp. 73–86.

[4] J. A. Brzozowski and E. L. Leiss, On equations for regular languages, finite automata, and sequential networks, *Theoret. Comput. Sci.* **10** (1980) 19–35.

[5] J. Cohen, D. Perrin and J. Pin, On the expressive power of temporal logic, *J. Comput. Syst. Sci.* **46**(3) (1993) 271–294.

[6] A. Fellah, H. Jürgensen and S. Yu, Constructions for alternating finite automata, *Intern. J. Computer Math.* **35**(1–4) (1990) 117–132.

[7] Y. Gao, N. Moreira, R. Reis and S. Yu, A survey on operational state complexity, *CoRR* abs/1509.03254 (2015).

[8] I. Glaister and J. Shallit, A lower bound technique for the size of nondeterministic finite automata, *Inf. Process. Lett.* **59**(2) (1996) 75–77.

[9] G. Jirásková, Descriptional complexity of operations on alternating and Boolean automata, *CSR 2012*, eds. E. A. Hirsch, J. Karhumäki, A. Lepistö and M. Prilutskii, LNCS, Vol. 7353 (Springer, 2012), pp. 196–204.

[10] G. Jirásková and T. Masopust, Complexity in union-free regular languages, *Internat. J. Found. Comput. Sci.* **22**(7) (2011) 1639–1653.

[11] G. Jirásková and G. Pighizzini, Optimal simulation of self-verifying automata by deterministic automata, *Inform. Comput.* **209**(3) (2011) 528–535.

[12] C. A. Kapoutsis, Removing bidirectionality from nondeterministic finite automata, *MFCS 2005*, eds. J. Jedrzejowicz and A. Szepietowski, LNCS, Vol. 3618 (Springer, 2005), pp. 544–555.

[13] D. Kleitman and G. Markowsky, On Dedekind's problem: The number of isotone Boolean functions. II, *Trans. Amer. Math. Soc.* **213** (1975) 373–390.

[14] O. B. Lupanov, A comparison of two types of finite automata, *Problemy Kibernetiki* **9** (1963); P. Kibernetiki, (in Russian); German translation: Über den Vergleich zweier Typen endlicher Quellen. Probleme der Kybernetik **6** (1966) 328–335.

[15] A. N. Maslov, Estimates of the number of states of finite automata, *Soviet Math. Doklady* **11** (1970) 1373–1375.

[16] A. R. Meyer and M. J. Fischer, Economy of description by automata, grammars, and formal systems, *Proceedings of the* 12*th Annual Symposium on Switching and Automata Theory* (IEEE Computer Society Press, 1971), pp. 188–191.

[17] F. R. Moore, On the bounds for state-set size in the proofs of equivalence between deterministic, nondeterministic, and two-way finite automata, *IEEE Transaction on Computing* **C-20** (1971) 1211–1219.

[18] M. O. Rabin and D. Scott, Finite automata and their decision problems, *IBM* **3** (1959) 114–125.
[19] M. Sipser, *Introduction to the Theory of Computation* (Cengage Learning, 2012).
[20] Yu. L. Yershov, On a conjecture of V. A. Uspenskii, *Algebra i Logika* (*Seminar*) **1** (1962) 45–48 (in Russian).
[21] S. Yu, *Handbook of Formal Languages*: *Volume* 1 *Word*, *Language*, *Grammar* (Springer, Heidelberg, 1997), Heidelberg, ch. Regular Languages, pp. 41–110.
[22] S. Yu, Q. Zhuang and K. Salomaa, The state complexities of some basic operations on regular languages, *Theoret. Comput. Sci.* **125**(2) (1994) 315–328.