On the Descriptive Complexity of $\overline{\Sigma^* \overline{L}}$

Michal Hospodár(✉), Galina Jirásková, and Peter Mlynárčik

Mathematical Institute, Slovak Academy of Sciences,
Grešákova 6, 040 01 Košice, Slovakia
hosmich@gmail.com, jiraskov@saske.sk, mlynarcik1972@gmail.com

Abstract. We examine the descriptive complexity of the combined unary operation $\overline{\Sigma^* \overline{L}}$ and 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 [1996, The state complexity of $\overline{\Sigma^* \overline{L}}$ and its connection with temporal logic, Inform. Process. Lett. 58, 185–188].

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 descriptive 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 descriptive 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].

Descriptive complexity, which measures the cost of description of languages by different formal systems, was deeply investigated in last three decades (cf. [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

M. Hospodár et al.—Research supported by grant VEGA 2/0084/15 and grant APVV-15-0091. This work was conducted as a part of PhD study of Michal Hospodár and Peter Mlynárčik at the Faculty of Mathematics, Physics and Informatics of the Comenius University.

É. Charlier et al. (Eds.): DLT 2017, LNCS 10396, pp. 222–234, 2017.
DOI: 10.1007/978-3-319-62809-7_16

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 [11], 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 [10].

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 $\square$, 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 $L(\square\varphi) = L(\overline{\diamond\overline{\varphi}}) = \overline{\Sigma^*\overline{L(\varphi)}}$. 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 [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. To get lower bounds, we describe languages over a fixed alphabet, except for four cases where the alphabet grows exponentially with n. In most cases our worst-case examples are binary and unary, and these alphabets are always optimal.

2 Preliminaries

We assume that the reader is familiar with basic notions in automata theory. For details, we refer to [19,21].

We use standard models of (complete) deterministic finite automata (DFAs), partial deterministic finite automata (PFAs), nondeterministic finite automata with a single initial state (NFAs), nondeterministic finite automata with multiple initial states (NNFAs), boolean finite automata (BFAs), and boolean finite automata with a single initial state (AFAs).

We call a state of an NNFA $A = (Q, \Sigma, \cdot, I, F)$ *sink state* if it has a loop on every input symbol. 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 symbol a, and the state p has an *out-transition* on symbol a.

Let q be a state of a DFA A. To *omit* the state q 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 its out-transition (q, a, p) 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.

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* $\{0, 1, \ldots, n-1\}$. □

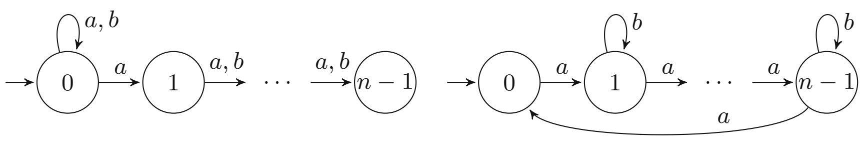

Fig. 1. The NFAs used in Proposition 1

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]). *If there is a uniquely distinguishable state of an* NFA A *that is uniquely reachable from any other state of* A*, then the subset automaton* $\mathcal{D}(A)$ *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 provide the proof of case (e) since all the remaining cases are either well known, or follow from [4,13], [6, Theorem 4.1 and Corollary 4.2], and [9, Lemmas 1 and 2]. We use the claim in Lemma 3(a) quite often in the paper without referring to Lemma 3(a) again and again.

Lemma 3 (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, respectively).*
(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* PFA *of at most* $2^n - 1$ *states. If* L *is accepted by an* n*-state* PFA*, then* L *is accepted by a* DFA *of at most* $n+1$ *states.*

Proof. (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). □

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:

$$b(L) = \overline{\Sigma^*\overline{L}}. \tag{1}$$

3 Results

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

Lemma 4 (Properties of $\overline{\Sigma^*\overline{L}}$). *Let L be a regular language and $b(L) = \overline{\Sigma^*\overline{L}}$.*

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

In what follows we consider six models of finite automata: DFAs, PFAs, 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 $b(L) = \overline{\Sigma^*\overline{L}}$? 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 3. 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 $2^{2^{\Theta(n)}}$ [12]:

$$2^{n-\log n} \leq \binom{n}{\lfloor n/2 \rfloor} \leq \log_2 \mathrm{M}(n) \leq \binom{n}{\lfloor n/2 \rfloor}\left(1 + O\left(\frac{\log n}{n}\right)\right) \leq 2^{n+1-(\log n)/2}.$$

It follows that $\log_2 \mathrm{M}(n)$ 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 $b(L)$ is empty or equals Σ^* by Lemma 4(b) and (c).

Theorem 5 (Upper Bounds). *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 $b(L)$ is accepted by a* DFA *of at most 2^{n-1} states.*
(2) If A is a PFA*, then $b(L)$ is accepted by a* PFA *of at most 2^{n-1} states.*
(3) If A is an NFA*, then $b(L)$ is accepted by*
(a) an NFA *of at most 2^{n-1} states;*
(b) a PFA *of at most $\mathrm{M}(n-1) - 1$ states.*

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

Proof (1). We 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 PFA 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 have $s \in F$ since $\varepsilon \in L$. We reverse A to get an n-state NNFA A^R for L^R with a unique final state s. In the subset automaton $\mathcal{D}(A^R)$, we omit all the non-final subsets, that is, all subsets not containing s, to get a 2^{n-1}-state PFA B with the initial state F. All states of B are final, and all of them contain s. We have two cases. If there is a final subset of $\mathcal{D}(A^R)$ which is not reachable in B, then we reverse B and add a new initial state to get an NFA for $b(L)$ of at most 2^{n-1} states. Otherwise, we modify PFA 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 ε-NFA by B'. We can show that $L(B') = L(B)$. This means that B' is a 2^{n-1}-state ε-NFA with one final state for $b(L)^R$. By reversing B' and removing ε-transitions, we get a 2^{n-1}-state NFA for $b(L)$.

(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

$B = \mathcal{D}(A) = (2^Q, \Sigma, \cdot, \{s\}, F_B)$ where $F_B = \{X \subseteq Q \mid X \cap F \neq \emptyset\}$);
$\overline{B} = (2^Q, \Sigma, \cdot, \{s\}, F_{\overline{B}})$ where $F_{\overline{B}} = 2^Q \setminus F_B = \{X \subseteq Q \mid X \subseteq Q \setminus F\}$);

$N = (2^Q, \Sigma, \circ, \{s\}, F_{\overline{B}})$ where for each X in 2^Q and each a in Σ,

$\{s\} \circ a = \{\{s\}, \{s\} \cdot a\}$, and

$X \circ a = \{X \cdot a\}$ if $X \neq \{s\}$;

$C = \mathcal{D}(N) = (2^{2^Q}, \Sigma, \circ, \{\{s\}\}, F_C)$ where $F_C = \{X \in 2^{2^Q} \mid X \cap F_{\overline{B}} \neq \emptyset\}$.

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 $S_i \subseteq Q \setminus F$. 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. We can show that w is accepted by N also 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 $\mathcal{S} \setminus \{S_j\}$. 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 $2^{Q \setminus \{s\}}$. 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 unreachable final sink state, but it is equivalent to the reachable state $\{\{s\}, \emptyset\}$.

(4a) If all the states of a given NNFA are initial, then L is suffix-closed, and therefore $b(L) = L$ by Lemma 4(c). Otherwise, L^R is accepted by a PFA which has $2^n - 1$ states, and at least one of them is non-final. Omit all the non-final states to get a PFA for $b(L)^R$ (cf. Lemma 4(d)), and reverse the resulting PFA to get the desired NNFA for $b(L)$.

(4b) Similarly as in (3b), we prove that only states $\mathcal{S} = \{I, S_1, S_2, \ldots, S_k\}$ where $\{S_1, S_2, \ldots, S_k\}$ is an antichain in 2^Q are pairwise distinguishable.

(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 $b(L)^R$ of 2^n states of which 2^{n-1} are final. Hence $b(L)$ is accepted by an n-state AFA.

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

(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 $b(L)^R$ of 2^n states given by Lemma 4(d). Hence $b(L)$ is accepted by an n-state BFA.

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

Now we turn our attention to lower bounds. We again need to prove only some of them and all the remaining bounds follow from the inclusions of models or from Lemma 3. 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 5. The remaining four cases are the trade-offs from NNFA to DFA, PFA, NFA, and NNFA. With the exception of four trade-offs, our witness languages 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 language meeting the upper bound (and the unary alphabet is always optimal :-).

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

Proof. (1) Let L be the language accepted by the DFA A shown in Fig. 2. We reverse A to get an NFA A^R for L^R. Using Propositions 1 and 2, we can prove that in the minimal DFA for L^R we have 2^{n-1} final states and one non-final sink state, so the language L^R is prefix-closed. Therefore L is suffix-closed, so $b(L)=L$. Since the minimal DFA for L^R has $2^{n-1}+1$ states, every BFA for L, so for $b(L)$, has at least n states.

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

(3) Let L be accepted by DFA $A=(\{0,\dots,n{-}1\},\{a,b\},\cdot,0,\{0,1,\dots,n{-}2\})$, where $i\cdot a=(i+1) \bmod n$, $0\cdot b=0$, and $i\cdot b=(i+1) \bmod n$ if $i\neq 0$.

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 PFA. Notice that the lower bound 2^{n-1} for a DFA accepting $b(L)$ follows from the proof. In [2, Proof of Theorem 2(b)], it is claimed that this bound is met by the 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 PFA A shown in Fig. 3. 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 transitions $(0, a, 0)$ and $(0, d, 0)$. By using Propositions 1 and 2 we can show that $\mathcal{D}(N)$ has $2^{n-1}+1$ reachable and pairwise distinguishable states.

(5) 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:

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

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

Then $B = \mathcal{D}(A) = (2^Q, \Sigma, \cdot, \{0\}, 2^Q \setminus \{\{n-1\}, \emptyset\})$;

$\overline{B} = (2^Q, \Sigma, \cdot, \{0\}, \{\{n-1\}, \emptyset\})$;

$N = (2^Q, \Sigma, \circ, \{0\}, \{\{n-1\}, \emptyset\})$ where

$\{0\} \circ a = \{0\} \cup \{0\} \cdot a$,

$X \circ a = X \cdot a$ if $X \neq \{0\}$;

$C = \mathcal{D}(N) = (2^{2^Q}, \Sigma, \circ, \{\{0\}\}, \{X \in 2^{2^Q} \mid X \cap \{\{n-1\}, \emptyset\} \neq \emptyset\})$. 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

$$\{\{0\}\} \xrightarrow{a_{S_1}} \{\{0\}, S_1\} \xrightarrow{a_{S_2}} \{\{0\}, S_1, S_2\} \xrightarrow{a_{S_3}} \ldots \xrightarrow{a_{S_k}} \{\{0\}, S_1, S_2, \ldots, S_k\}.$$

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. We can prove that two distinct antichains are distinguishable. It follows that C has at least $\mathrm{M}(n-1)$ reachable and distinguishable states.

(6) Let L be accepted by the NNFA A shown in Fig. 4. Since each state of A is initial, L is suffix-closed, so $b(L) = L$. We can show that the minimal DFA for L^R has more than 2^{n-1} final states. It follows that every AFA for L, so for $b(L)$, has at least $n+1$ states.

(7) 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, $b(L) = L$. (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.

(8) Let K be accepted by the 2^n-state DFA A shown in Fig. 5; notice that A has 2^{n-1} final states. Set $L = K^R$. Then L is accepted by an n-state AFA. By Lemma 4(d), if we omit all non-final states of A, we get a PFA C for $b(L)^R$ of 2^{n-1} states, all of them final. It is shown in [8, Theorem 2] that every NFA for $L(C)^R$ has at least $2^{n-1}+1$ states. Since $L(C)^R = b(L)$, the claim follows.

(9) Let K be accepted by the 2^n-state DFA A shown in Fig. 6; notice that A has 2^{n-1} final states. Set $L = K^R$. Then the language L is accepted by an n-state AFA. By Lemma 4(d), if we omit all non-final states of A, we get a PFA C for $b(L)^R$ of 2^{n-1} states, all of them final. Next, we reverse the PFA C to get an NNFA $N = (\{0, 1, \ldots, 2^{n-1}-1\}, \{a, b\}, \cdot^R, \{0, 1, \ldots, 2^{n-1}-1\}, \{0\})$ for $b(L)$.

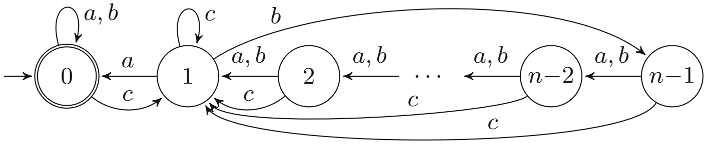

Fig. 2. The DFA for L such that every BFA for $\overline{\Sigma^* \overline{L}}$ has n states

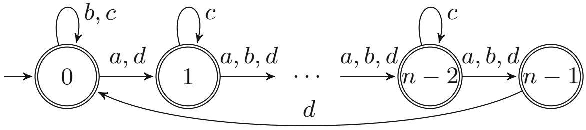

Fig. 3. The PFA for L such that every DFA for $\overline{\Sigma^* \overline{L}}$ has $2^{n-1} + 1$ states

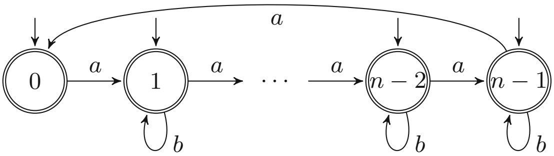

Fig. 4. The NNFA for L such that every AFA for $\overline{\Sigma^* \overline{L}}$ has $n + 1$ states

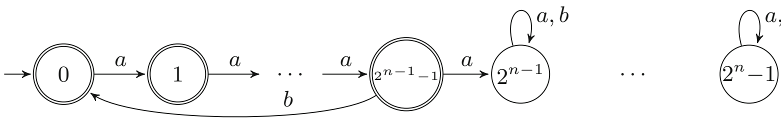

Fig. 5. The reverse of the witness for the AFA-to-NFA trade-off

By Proposition 1, the subset automaton $\mathcal{D}(N)$ has $2^{2^{n-1}}$ reachable states. To prove distinguishability, notice that the state 0 is uniquely distinguishable by ε, and it is uniquely reachable from any other state through unique in-transitions on symbol a. By Proposition 2, the subset automaton has no equivalent states.

(10) Let $L = \{a^i \mid 0 \le i \le 2^n - 2\}$. Then L is a unary language accepted by a minimal 2^n-state DFA A, so L is accepted by a n-state BFA. Since L is suffix-closed, $b(L) = L$. (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) Let K be accepted by the 2^n-state DFA A shown in Fig. 7. Set $L = K^R$. Then L is accepted by an n-state BFA. Now the proof goes exactly the same way as in the case (8) and it results in the lower bound 2^n.

(12) Let K be accepted by the 2^n-state DFA A shown in Fig. 8. Set $L = K^R$. Then L is accepted by an n-state BFA. Now the proof goes exactly the same way as in the case (9) and it results in the lower bound $2^{2^n - 1}$. $\square$

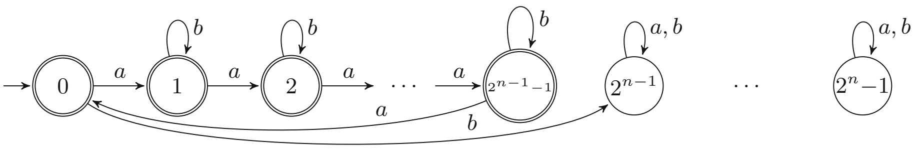

Fig. 6. The reverse of the witness for the AFA-to-DFA trade-off

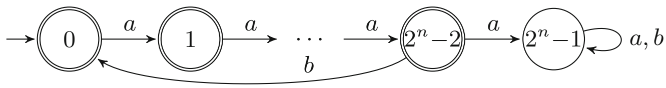

Fig. 7. The reverse of the witness for the BFA-to-NFA trade-off

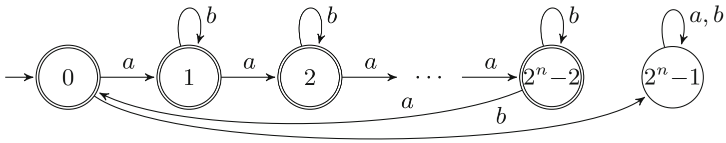

Fig. 8. The reverse of the witness for the BFA-to-DFA trade-off

Table 1. The complexity of $\overline{\Sigma^*\overline{L}}$ for various types of finite automata

$L\backslash b(L)$	DFA	$\|\Sigma\|$	PFA	$\|\Sigma\|$	NFA	$\|\Sigma\|$	NNFA	$\|\Sigma\|$	AFA	$\|\Sigma\|$	BFA	$\|\Sigma\|$
DFA	2^{n-1}	2	2^{n-1}	2	2^{n-1} [2]	3	2^{n-1} [2]	3	n	3	n	3
PFA	$2^{n-1}+1$	4	2^{n-1}	2	2^{n-1}	3	2^{n-1}	3	n	3	n	3
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	3	n	3
NNFA	$\geq \mathrm{M}(n-1)$	2^{n+1}	$\geq \mathrm{M}(n-1)-1$	2^{n+1}	$\geq 2^{n-1}$	3	$\geq 2^{n-1}$	3	$n+1$	2	n	2
	$\leq \mathrm{M}(n)$		$\leq \mathrm{M}(n)-1$		$\leq 2^{n}-1$		$\leq 2^{n}-2$					
AFA	$2^{2^{n-1}}$	2	$2^{2^{n-1}}-1$	2	$2^{n-1}+1$	2	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

4 Conclusions

We investigated the descriptive 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)NFA-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)NFA trade-offs remains open as well (Table 1).

References

1. Birget, J.: Intersection and union of regular languages and state complexity. Inform. Process. Lett. **43**(4), 185–190 (1992)
2. Birget, J.: The state complexity of $\overline{\Sigma^*\overline{L}}$ and its connection with temporal logic. Inform. Process. Lett. **58**(4), 185–188 (1996)
3. Brzozowski, J., Jirásková, G., Liu, B., Rajasekaran, A., Szykuła, M.: On the state complexity of the shuffle of regular languages. In: Câmpeanu, C., Manea, F., Shallit, J. (eds.) DCFS 2016. LNCS, vol. 9777, pp. 73–86. Springer, Cham (2016). doi:10.1007/978-3-319-41114-9_6
4. Brzozowski, J.A., Leiss, E.L.: On equations for regular languages, finite automata, and sequential networks. Theoret. Comput. Sci. **10**, 19–35 (1980)
5. Cohen, J., Perrin, D., Pin, J.: On the expressive power of temporal logic. J. Comput. Syst. Sci. **46**(3), 271–294 (1993)
6. Fellah, A., Jürgensen, H., Yu, S.: Constructions for alternating finite automata. Int. J. Comput. Math. **35**(1–4), 117–132 (1990)
7. Gao, Y., Moreira, N., Reis, R., Yu, S.: A survey on operational state complexity. CoRR abs/1509.03254 (2015). http://arxiv.org/abs/1509.03254
8. Jirásková, G.: State complexity of some operations on binary regular languages. Theoret. Comput. Sci. **330**(2), 287–298 (2005)
9. Jirásková, G.: Descriptional complexity of operations on alternating and boolean automata. In: Hirsch, E.A., Karhumäki, J., Lepistö, A., Prilutskii, M. (eds.) CSR 2012. LNCS, vol. 7353, pp. 196–204. Springer, Heidelberg (2012). doi:10.1007/978-3-642-30642-6_19
10. Jirásková, G., Pighizzini, G.: Optimal simulation of self-verifying automata by deterministic automata. Inform. Comput. **209**(3), 528–535 (2011)
11. Kapoutsis, C.: Removing bidirectionality from nondeterministic finite automata. In: Jędrzejowicz, J., Szepietowski, A. (eds.) MFCS 2005. LNCS, vol. 3618, pp. 544–555. Springer, Heidelberg (2005). doi:10.1007/11549345_47
12. Kleitman, D., Markowsky, G.: On Dedekind's problem: the number of isotone boolean functions. II. Trans. Amer. Math. Soc. **213**, 373–390 (1975)
13. Leiss, E.L.: Succint representation of regular languages by boolean automata. Theoret. Comput. Sci. **13**, 323–330 (1981)
14. Lupanov, O.B.: A comparison of two types of finite automata. Problemy Kibernetiki **9**, 321–326 (1963). (in Russian) German translation: Über den Vergleich zweier Typen endlicher Quellen. Probleme der Kybernetik **6**, 328–335 (1966)
15. Maslov, A.N.: Estimates of the number of states of finite automata. Soviet Math. Doklady **11**, 1373–1375 (1970)
16. Meyer, A.R., Fischer, M.J.: Economy of description by automata, grammars, and formal systems. In: Proceedings of the 12th Annual Symposium on Switching and Automata Theory, pp. 188–191. IEEE Computer Society Press (1971)
17. Moore, F.R.: On the bounds for state-set size in the proofs of equivalence between deterministic, nondeterministic, and two-way finite automata. IEEE Trans. Comput. **C-20**, 1211–1219 (1971)
18. Rabin, M.O., Scott, D.: Finite automata and their decision problems. IBM J. Res. Develop. **3**, 114–125 (1959)
19. Sipser, M.: Introduction to the Theory of Computation. Cengage Learning (2012)
20. Yershov, Yu.L.: On a conjecture of V. A. Uspenskii. Algebra i Logika (Seminar) **1**, 45–48 (1962). (in Russian)

21. Yu, S.: Chapter 2: Regular languages. In: Rozenberg, G., Salomaa, A. (eds.) Handbook of Formal Languages, vol. I, pp. 41–110. Springer, Heidelberg (1997)
22. Yu, S., Zhuang, Q., Salomaa, K.: The state complexities of some basic operations on regular languages. Theoret. Comput. Sci. **125**(2), 315–328 (1994)