International Journal of Foundations of Computer Science

COMPLEXITY IN UNION-FREE REGULAR LANGUAGES*

GALINA JIRÁSKOVÁ†
Mathematical Institute, Slovak Academy of Sciences, Grešákova 6, 040 01 Košice, Slovakia
jiraskov@saske.sk

TOMÁŠ MASOPUST‡
CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands
Mathematical Institute, Czech Academy of Sciences, Žižkova 22, 616 62 Brno, Czech Republic
masopust@math.cas.cz

Received (Day Month Year)
Accepted (Day Month Year)
Communicated by (xxxxxxxxxx)

We continue the investigation of union-free regular languages that are described by regular expressions without the union operation. We also define deterministic union-free languages as languages accepted by one-cycle-free-path deterministic finite automata, and show that they are properly included in the class of union-free languages. We prove that (deterministic) union-freeness of languages does not accelerate regular operations, except for the reversal in the nondeterministic case.

Keywords: Union-free regular language; finite automaton; one-cycle-free-path automaton; descriptional complexity; closure properties.

2010 Mathematics Subject Classification: 68Q45, 68Q19

1. Introduction

The class of regular languages is the simplest class of languages in the Chomsky hierarchy. Regular languages have been intensively investigated due to their practical applications in various areas of computer science, and for their importance in the theory as well. In recent years, several special subclasses have been deeply examined, such as finite languages described by expressions without the star operation [22], suffix- and prefix-free languages used in codes [11], star-free and locally testable languages, ideal, closed, and convex languages, etc. For a survey of descriptional and computational complexity of finite automata, we refer the reader to [12].

*This paper has been presented at the 14th Conference on Developments in Language Theory (DLT 2010) held in London, Ontario, Canada on August 17-20, 2010.
†Research supported by VEGA grant 2/0183/11, and by the Slovak Research and Development Agency under contract APVV-0035-10 "Algorithms, Automata, and Discrete Data Structures."
‡Research supported by the CAS, Institutional Research Plan no. AV0Z10190503.

In this paper, we continue this research and study union-free regular languages that are represented by regular expressions without the union operation. Nagy in [26] introduced one-cycle-free-path nondeterministic finite automata, in which from each state, there is exactly one cycle-free path to the final state. He proved that these automata characterize the class of union-free languages. First, we complement his closure-property results. Then, in Section 3, we investigate the nondeterministic state complexity of operations in the class of union-free languages. Surprisingly, we show that all the known upper bounds for regular languages are met by union-free languages, except for reversal, where the bound is n instead of $n+1$. In Section 4, we define deterministic union-free languages as languages accepted by deterministic one-cycle-free-path automata, and show that they are properly included in the class of union-free languages. We study the state complexity of a number of operations, and prove that deterministic union-freeness does not accelerate any of them.

To conclude this section, we mention several related works. Brzozowski [5] examined union-free regular expressions under the name star-dot expressions. Crvenković, Dolinka, Ésik [7] investigated algebraic properties of union-free languages. Afonin and Golomazov [1] studied union-free decompositions of regular languages, and Nagy [27] union-complexity of regular languages.

2. Preliminaries

We assume that the reader is familiar with basic concepts of finite automata and regular languages. For unexplained notions, we refer to [30, 31]. If Σ is an alphabet, that is, a finite non-empty set, then Σ^* denotes the set of all strings over the alphabet Σ including the empty string ε. A *language* over Σ is any subset of Σ^*. We denote the size of a finite set A by $|A|$ and its powerset by 2^A.

A *nondeterministic finite automaton* (nfa) is a quintuple $M = (Q, \Sigma, \delta, S, F)$, where Q is a finite non-empty set of states, Σ is an input alphabet, S is the set of initial states, F is the set of accepting states, and δ is the transition function that maps $Q \times (\Sigma \cup \{\varepsilon\})$ into 2^Q. The transition function is extended to the domain $2^Q \times \Sigma^*$ in a natural way. The nfa M accepts a string w in Σ^* if $\delta(S, w) \cap F \neq \emptyset$. The *language accepted* by M is the set of all strings accepted by M. The automaton M is *deterministic* (dfa) if it has a single initial state, no ε-transitions, and $|\delta(q, a)| = 1$ for all states q in Q and symbols a in Σ. In this case, we usually write $\delta : Q \times \Sigma \to Q$.

A language is *regular* if there exists an nfa (or a dfa) accepting the language. The *state complexity* of a regular language L, $\mathrm{sc}(L)$, is the minimal number of states in any dfa accepting L. The *nondeterministic state complexity* of a regular language L, $\mathrm{nsc}(L)$, is the minimal number of states in any ε-free nfa with a single initial state accepting language L.

A path from state p to state q in an nfa/dfa M is a sequence $p_0 a_1 p_1 a_2 \cdots a_n p_n$, where $p_0 = p$, $p_n = q$, and $p_i \in \delta(p_{i-1}, a_i)$ for $i = 1, 2, \ldots, n$. The path is called *accepting cycle-free* if p_n is an accepting state, and $p_i \neq p_j$ whenever $i \neq j$. An

nfa/dfa is a *one-cycle-free-path* (1cfp) nfa/dfa if there is a unique accepting cycle-free path from each of its states (but the dead state in the case of dfa's).

A *regular expression* over an alphabet Σ is defined inductively as follows: $\emptyset$, ε, and a, for a in Σ, are regular expressions. If r and t are regular expressions, then also $(s + t)$, $(s \cdot t)$, and $(s)^*$ are regular expressions.

A regular expression is *union-free* if no symbol $+$ occurs in it. A regular language is *union-free* if there exists a union-free regular expression describing the language.

Let K and L be languages over Σ. We denote by $K \cap L$, $K \cup L$, $K - L$, $K \oplus L$ the intersection, union, difference, and symmetric difference of languages K and L, respectively. To denote complement, Kleene star, and reversal of L, we use L^c, L^*, and L^R. The left and right quotient of L with respect to a string w is the set $w\backslash L = \{x \mid wx \in L\}$ and $L/w = \{x \mid xw \in L\}$, respectively. The cyclic shift of L is defined as $L^{shift} = \{uv \mid vu \in L\}$. The *shuffle* of languages K and L is $K \sqcup L = \{u_1v_1u_2v_2 \cdots u_mv_m \mid m \geqslant 1, u_i, v_i \in \Sigma^*, u_1 \cdots u_m \in K, v_1 \cdots v_m \in L\}$. For the definition of positional addition, $K + L$, we refer to [17]: informally, strings are considered as numbers encoded in a $|\Sigma|$-adic system, and automata read their inputs from the least significant digit.

3. Union-Free Regular Languages

A regular language is union-free if it is described by a union-free regular expression. Nagy [26] proved that the classes of union-free regular languages and languages accepted by one-cycle-free-path nfa's coincide, and that union-free languages are closed under concatenation, Kleene-star, and substitution by a union-free language. Using an observation that the shortest string of a union-free language is unique, he proved not closeness under union, complementation, intersection, and substitution by a regular language. Our first result complements the closure properties.

Theorem 1 (Closure Properties) *The class of union-free regular languages is closed under reversal, but is not closed under cyclic shift, shuffle, symmetric difference, difference, left and right quotients, and positional addition.*

Proof. We prove the closeness under reversal by induction on the structure of a regular expression r. If r is $\emptyset$, ε, or a, the reversal is described by the same expression. If $r = st$, or $r = s^*$, then the reversal is $L(t)^R L(s)^R$ or $(L(s)^R)^*$, respectively, which are union-free due to closeness under concatenation and star.

To prove the nonclosure properties, we give union-free languages with the shortest string of length two in the resulting language, and show that there are at least two such strings in all cases: $\{ab\}^{shift} = \{a\} \sqcup \{b\} = \{ab\} \oplus \{ba\} = \{ab, ba\}$; $a(b + c)^* - a^* = \{ab, ac, \ldots\}$; $g\backslash(ge + gf)^*b = \{eb, fb, \ldots\}$ and $a(eb + fb)^*/b = \{ae, af, \ldots\}$; $88^* + 33^* = \{11, 91, \ldots\}$. As the shortest strings are not unique, the resulting languages are not union-free. □

The subset construction insures that every nfa of n states is simulated by a dfa of at most 2^n states. The worst case binary examples are well known, see [20, 23, 25]. In addition, Domaratzki et al. [8] have shown that there are at least 2^{n-2} distinct binary languages accepted by nfa's of n states that require 2^n deterministic states. However, none of the above mentioned automata is a one-cycle-free-path nfa. The following theorem shows that the bound 2^n is also tight for union-free languages.

Theorem 2 (NFA to DFA Conversion) *For every n, there exists a binary one-cycle-free-path nfa of n states whose equivalent minimal dfa has 2^n states.*

Proof. Consider the binary 1cfp nfa with states $0, 1, \ldots, n-1$, where 0 is the initial state and $n-1$ is the sole accepting state. By a, each state i goes to $\{i+1\}$, except for state $n-1$, which goes to the empty set. By b, each state i goes to $\{0, i\}$. We show that the corresponding subset automaton has 2^n reachable and pairwise distinguishable states. Each singleton $\{i\}$ is reached from the initial state $\{0\}$ by a^i, and the empty set is reached by a^n. Each set $\{i_1, i_2, \ldots, i_k\}$, where $0 \leqslant i_1 < i_2 < \cdots < i_k \leqslant n-1$, of size k, $2 \leqslant k \leqslant n$, is reached from the set $\{i_2 - i_1, i_3 - i_1, \ldots, i_k - i_1\}$ of size $k-1$ by string ba^{i_1}. This proves the reachability of all subsets. For distinguishability, notice that the string a^{n-1-i} is accepted by the nfa only from state i. Two different subsets must differ in a state i, and so the string a^{n-1-i} distinguishes the two subsets. □

We next study the nondeterministic state complexity of regular operations in the class of union-free languages. Surprisingly, all the upper bounds on the nondeterministic state complexity of operations on regular languages are also met by union-free languages, except for reversal where the tight upper bound is n instead of $n+1$. We use a fooling set lower-bound technique, see [2, 3, 4, 10, 13].

Definition 3. *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*
(F1) *$x_i y_i \in L$ for $i = 1, 2, \ldots, n$, and*
(F2) *if $i \neq j$, then $x_i y_j \notin L$ or $x_j y_i \notin L$.*

It is well known that the size of a fooling set for a regular language provides a lower bound on the number of states in any nfa for the language. The argument is simple. Fix the accepting computations of any nfa on strings $x_i y_i$. Then, the states on these computations reached after reading x_i must be pairwise distinct, otherwise the nfa accepts both $x_i y_j$ and $x_j y_i$ for two distinct pairs. The next lemma shows that sometimes, if we insist on having just one initial state, one more state is necessary.

Lemma 4. *Let $\mathcal{A}$ and $\mathcal{B}$ be 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 with a single initial state for L has at least $|\mathcal{A}| + |\mathcal{B}| + 1$ states.*

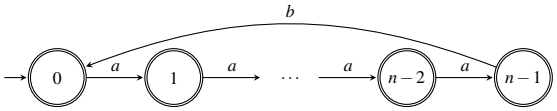

Fig. 1. The binary n-state nfa language meeting the bound $n + 1$ on reversal.

Proof. Consider an nfa for language L, and let $\mathcal{A} = \{(x_i, y_i) \mid i = 1, 2, \ldots, m\}$ and $\mathcal{B} = \{(x_{m+j}, y_{m+j}) \mid j = 1, 2, \ldots, n\}$. Since the strings $x_k y_k$ are in L, we fix an accepting computation of the nfa on each string $x_k y_k$. Let p_k be the state on this computation that is reached after reading x_k. As $\mathcal{A} \cup \mathcal{B}$ is a fooling set for L, the states p_1, p_2, …, p_{m+n} are pairwise distinct. As $\mathcal{A} \cup \{(\varepsilon, u)\}$ is a fooling set, the initial state is distinct from all the states p_1, p_2, …, p_m. As $\mathcal{B} \cup \{(\varepsilon, v)\}$ is a fooling set, the (sole) initial state is also distinct from all the states p_{m+1}, p_{m+2}, …, p_{m+n}. Thus the nfa has at least $m + n + 1$ states. □

Example 5. It is shown in [15] that there is a *binary* regular language L with $\mathrm{nsc}(L) = n$ and $\mathrm{nsc}(L^R) = n + 1$. The language is shown in Fig. 1, and the proof in [15] is by a counting argument. Notice that if $\mathcal{F}$ is a fooling set for language L^R, then $\{(y^R, x^R) \mid (x, y) \in \mathcal{F}\}$ is a fooling set for language L. Therefore, we cannot expect that we would be able to find a fooling set of size $n + 1$ for language L^R. However, Lemma 4 is applicable here with $\mathcal{A} = \{(ba^i, a^{n-1-i}) \mid i = 0, 1, \ldots, n-2\}$, $\mathcal{B} = \{(ba^{n-1}, \varepsilon)\}$, $u = \varepsilon$, *and* $v = a$.

Theorem 6 (Nondeterministic State Complexity) *Let K and L be union-free regular languages over an alphabet Σ accepted by an m-state and an n-state one-cycle-free-path nfa, respectively. Then,*

1. $\mathrm{nsc}(K \cup L) \leqslant m + n + 1$, *and the bound is tight if* $|\Sigma| \geqslant 2$;
2. $\mathrm{nsc}(K \cap L) \leqslant mn$, *and the bound is tight if* $|\Sigma| \geqslant 2$;
3. $\mathrm{nsc}(KL) \leqslant m + n$, *and the bound is tight if* $|\Sigma| \geqslant 2$;
4. $\mathrm{nsc}(K \sqcup L) \leqslant mn$, *and the bound is tight if* $|\Sigma| \geqslant 2$;
5. $\mathrm{nsc}(K + L) \leqslant 2mn + 2m + 2n + 1$, *and the bound is tight if* $|\Sigma| \geqslant 6$;
6. $\mathrm{nsc}(L^2) \leqslant 2n$, *and the bound is tight if* $|\Sigma| \geqslant 2$;
7. $\mathrm{nsc}(L^c) \leqslant 2^n$, *and the bound is tight if* $|\Sigma| \geqslant 3$;
8. $\mathrm{nsc}(L^R) \leqslant n$, *and the bound is tight if* $|\Sigma| \geqslant 1$;
9. $\mathrm{nsc}(L^*) \leqslant n + 1$, *and the bound is tight if* $|\Sigma| \geqslant 1$;
10. $\mathrm{nsc}(L^{shift}) \leqslant 2n^2 + 1$, *and the bound is tight if* $|\Sigma| \geqslant 2$.

Proof. 1. To get an nfa for union from two given nfa's, we add a new initial state that goes by the empty string to the initial states of the given automata. To prove tightness, consider the binary union-free languages $(a^m)^*$ and $(b^n)^*$, and let us give an alternative proof to that in [18] using Lemma 4. Consider the following sets of pairs of strings: $\mathcal{A} = \{(a^i, a^{m-i}) \mid i = 1, 2, \ldots, m-1\} \cup \{(a^m, a^m)\}$ and $\mathcal{B} = \{(b^j, b^{n-j}) \mid j = 1, 2, \ldots, n-1\} \cup \{(b^n, b^n)\}$.

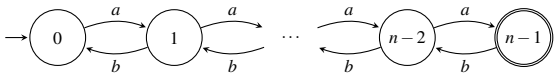

Fig. 2. One-cycle-free-path nfa meeting the bound $2n$ on square and $2n^2+1$ on cyclic shift.

Let $L = (a^m)^* \cup (b^n)^*$. We show that the set $\mathcal{A} \cup \mathcal{B}$ is a fooling set for L. The concatenation of the first and the second part of each pair results in a string in $\{a^m, a^{2m}, b^n, b^{2n}\}$, and so is in L. The concatenation of the first part of a pair and the second part of another pair results in a string in $\{a^r, a^{m+r}, b^s, b^{n+s}, a^r b^s, b^s a^r, a^m b^n, b^n a^m \mid 0 < r < m, 0 < s < n\}$, and so is not in L. Finally, both sets $\mathcal{A} \cup \{(\varepsilon, b^n)\}$ and $\mathcal{B} \cup \{(\varepsilon, a^m)\}$ are fooling sets for L as well. By Lemma 4, every nfa with a single initial state for L has at least $m+n+1$ states.

2. The cross-product construction provides the upper bound mn for intersection. To prove tightness, consider binary union-free languages $((b^*a)^m)^*$ and $((a^*b)^n)^*$ (see also [18]). The set $\{(a^i b^j, a^{m-i} b^{n-j}) \mid 0 \leqslant i \leqslant m-1,\ 0 \leqslant j \leqslant n-1\}$ is a fooling set of size mn for the intersection of the two languages.

3. To get an nfa for concatenation of languages given by two nfa's, we only add an ε-transition from all the final states in the first automaton to the initial state in the second automaton. For tightness, consider binary languages $(a^m)^*$ and $(b^n)^*$. The set $\{(a^i, a^{m-i}b^n) \mid i = 0, 1, \ldots, m-1\} \cup \{(a^m b^j, b^{n-j}) \mid j = 1, 2, \ldots, n\}$ is a fooling set of size $m+n$ for the concatenation of the two languages, and so every nfa for the concatenation has at least $m+n$ states.

4. The state set of an nfa for shuffle is the product of the state sets of the given nfa's, and its transition function δ is defined using transition functions δ_A and δ_B of the given automata by $\delta((p,q),a) = \{(\delta_A(p,a),q),(p,\delta_B(q,a))\}$, cf. [6]. This gives the upper bound mn. The bound is met by the shuffle of languages $(a^m)^*$ and $(b^n)^*$ because the set $\{(a^i b^j, a^{m-i} b^{n-j}) \mid 0 \leqslant i \leqslant m-1,\ 0 \leqslant j \leqslant n-1\}$ is a fooling set of size mn for the shuffle of the two languages.

5. An nfa of $2mn + 2m + 2n + 1$ states for positional addition is described in [17]: The group of $2mn$ states corresponds to the situation when both automata read their inputs without or with a carry. Then there is a group of $2m + 2n$ states simulating the situation when one of the two automata has already finished reading of its input. One more state is necessary if a carry eventually occurs. It was shown in [17] that the bound is met by the positional addition of union-free languages $((1^*5)^m)^*$ and $((2^*5)^n)^*$ over the alphabet $\{0,1,2,3,4,5\}$.

6. Since L^2 is the concatenation of the language L with itself, the upper bound $2n$ follows from part 3. The lower bound is shown in [9] for a union-free language $a^{n-1}(ba^{n-1})^*$. For the sake of completeness, we give a different (and simpler) proof for the lower bound. Moreover, our worst-case language is a witness not only for square but also for cyclic shift. Consider the 1cfp nfa shown in Fig. 2. Construct an nfa with the state set $Q = \{p_0, p_1, \ldots, p_{n-1}\} \cup \{q_0, q_1, \ldots, q_{n-1}\}$ for language L^2 from two copies of the nfa for L by adding an ε-transition from the final state of the

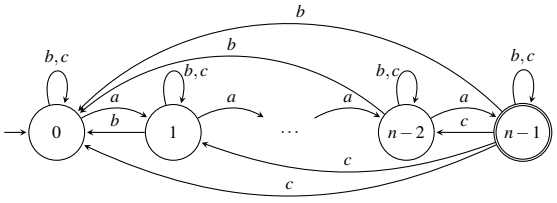

Fig. 3. One-cycle-free-path nfa meeting the 2^n bound on complement.

first copy to the initial state of the second copy. The initial state of the resulting nfa is p_0, and the only final state is q_{n-1}. For each state s in Q, define two strings x_s and y_s in such a way that the initial state p_0 goes to state s by string x_s, and each state s goes to the accepting state q_{n-1} by string y_s:

$$x_s = \begin{cases} a^i & \text{if } s = p_i, \\ a^{2n-2}b^{n-1-i} & \text{if } s = q_i, \end{cases} \qquad y_s = \begin{cases} a^{2n-2-i} & \text{if } s = p_i \text{ and } i \neq n-1, \\ b^{n-1}a^{2n-2} & \text{if } s = p_{n-1}, \\ a^{n-1-i} & \text{if } s = q_i. \end{cases}$$

Thus, each string $x_s y_s$ is in L^2. If $s \notin \{p_{n-1}, q_0\}$, then p_0 goes *only* to state s by string x_s, and string y_s is accepted *only* from state s. It follows that $x_s y_t$ is not in L^2 if s and t are two distinct states in $Q - \{p_{n-1}, q_0\}$. If $s \in Q - \{p_{n-1}, q_0\}$ and $t = q_0$, then string $x_s y_t$ is not in L^2 because string a^{n-1} is accepted only from states p_{n-1} and q_0. Finally, if $s \in Q - \{p_{n-1}\}$ and $t = p_{n-1}$, then string $x_s y_t$ is not in L^2 because string $y_t = b^{n-1}a^{2n-2}$ is accepted only from state p_{n-1}, and string x_s does not reach state p_{n-1} from state p_0. Hence $\{(x_s, y_s) \mid s \in Q\}$ is a fooling set for L^2 of size $2n$.

7. Subset construction applied to a given n-state nfa M followed by interchanging of accepting and rejecting states results in an nfa (even a dfa) for the complement of language $L(M)$ with at most 2^n states. The bound has been proved to be tight for a four-letter alphabet in [4], and for a binary alphabet in [15]. However, the binary witness nfa's in [15] are not 1cfp. We prove the tightness of the bound also for 1cfp automata.

Consider a ternary language L accepted by the 1cfp nfa in Fig. 3; denote the state set $\{0, 1, \ldots, n-1\}$ by Q. By c, state $n-1$ goes to $\{0, 1, \ldots, n-1\}$, and each other state i goes to $\{i\}$. Transitions by a and b are the same as in the automaton in the proof of Theorem 2. Therefore, in the corresponding subset automaton, each subset S of the state set Q is reached from the initial state $\{0\}$ by a string x_S in $\{a, b\}^*$. We now define strings y_S so that the set $\{(x_S, y_S) \mid S \subseteq Q\}$ would be a fooling set for L^c. Let S be a subset of Q. If $S = \{0, 1, \ldots, n-2\}$, let $y_S = c$; otherwise, let $y_S = y_1 y_2 \cdots y_n$, where for each i in Q,

$$y_{n-i} = \begin{cases} a & \text{if } i \in S, \\ ca & \text{if } i \notin S. \end{cases}$$

First, we show that for each subset S, the string y_S is not accepted by the nfa from any state in the set S, but is accepted from each state that is not in S. The claim holds if $S = \{0, 1, \ldots, n-2\}$ because c is not accepted from any state in $\{0, 1, \ldots, n-2\}$, but is accepted from state $n-1$. Let $S \neq \{0, 1, \ldots, n-2\}$. By a and ca, each state i goes to $\{i+1\}$, except for state $n-1$, which goes to the empty set by a, and to $\{1, 2, \ldots, n-1\}$ by ca. If i is in S, then $y_S = y_1 y_2 \cdots y_{n-i-1} a y_{n-i+1} \cdots y_n$. State i goes to $\{n-1\}$ by $y_1 y_2 \cdots y_{n-i-1}$, and the next symbol a of the string y_S cannot be read. Hence, the string y_S is not accepted from state i. On the other hand, if i is not in S, then $y_S = y_1 y_2 \cdots y_{n-i-1} c a y_{n-i+1} \cdots y_n$. In case $i < n-1$, state i goes to state $n-1$ by $y_1 y_2 \cdots y_{n-i-1}$, then it may go to state $n-i-1$ by ca, and, finally, to the accepting state $n-1$ by $y_{n-i+1} \cdots y_n$. In case $i = n-1$, since $S \neq \{0, 1, \ldots, n-2\}$, there is a state j with $j < n-1$, which is not in S. It follows that $y_S = c a y_2 \cdots y_{n-j-1} c a y_{n-j+1} \cdots y_n$. State $n-1$ may go to state $j+1$ by ca, then to state $n-1$ by $y_2 \cdots y_{n-j-1}$, then to state $n-j-1$ by ca, and, finally, to the accepting state $n-1$ by $y_{n-j+1} \cdots y_n$. This proves our claim.

Now, we show that the set $\{(x_S, y_S) \mid S \subseteq Q\}$ is a fooling set for the language L^c. To prove (F1), notice that the initial state $\{0\}$ goes to the set S by string x_S. As string y_S is not accepted from any state in S, string $x_S y_S$ is not accepted by the nfa, and thus is in L^c. To prove (F2), let S and T be two different subsets of state set Q. Then, there is a state i such that, without loss of generality, $i \in S$ and $i \notin T$. Consider the computation of the nfa on string $x_S y_T$. As state i is in S, the initial state $\{0\}$ goes to i by x_S. As i is not in T, the string y_T is accepted by the nfa from state i. It follows that string $x_S y_T$ is accepted by the nfa, and so is not in L^c. Hence, the set $\{(x_S, y_S) \mid S \subseteq Q\}$ is a fooling set for the complement of L, and, thus, every nfa for the complement needs at least 2^n states.

8. To get an nfa for the reversal of a language accepted by an n-state 1cfp nfa, reverse all the transitions, make the initial state final, and (the only) final state initial. The resulting nfa has n states (and a single initial state). The unary union-free language a^{n-1} meets the bound.

9. The standard construction of an nfa for Kleene star that adds a new initial (and accepting) state connected through an ε-transition to the initial state of the given nfa as well as ε-transitions from each final state to the initial state, provides the upper bound $n+1$. For tightness, consider the union-free language $a^{n-1}(a^n)^*$. The set $\{(\varepsilon, \varepsilon)\} \cup \{(a^i, a^{n-1-i}) \mid i = 1, 2, \ldots, n-2\} \cup \{(a^{n-1}, a^n), (a^n, a^{n-1})\}$ is a fooling set of size $n+1$ for the Kleene star of this language.

10. The nfa for cyclic shift in [16] consists of an initial state and $2n$ copies of a given nfa. The initial state goes by the empty string to the i-th state of each i-th copy, and all the final states in the i-th copy go by the empty string to the initial state in the $(n+i)$-th copy. The i-th state in each $(n+i)$-th copy is a final state of the resulting nfa. The one-cycle-free-path nfa in Fig. 2 meets the bound $2n^2+1$, cf. [16]. To prove the result, a fooling set of size $2n^2$ is described in [16], and then Lemma 4 is used to show that one more state is necessary. □

4. Deterministic Union-Free Regular Languages

We now turn our attention to deterministic union-free languages, that is, to languages accepted by one-cycle-free-path deterministic finite automata. We first show that deterministic union-free languages are properly included in the class of union-free languages. Then, we study the state complexity of regular operations.

Theorem 7 (1cfp DFAs vs. 1cfp NFAs) *The class of deterministic union-free regular languages is a proper subclass of the class of union-free regular languages.*

Proof. Let $n \geqslant 4$. Consider the regular language $L = \{\varepsilon, a^{n-3}\} \cup \{a^i \mid i \geqslant n-1\}$. Since every dfa recognizing language L has at least three final states, the language is not deterministic union-free. To prove that language L is union-free, we describe a 1cfp nfa for L. The only initial and final state is state 0, and the automaton consists of $n+2$ cycles. Each cycle starts and ends in state 0, but otherwise, the cycles are pairwise disjoint. The length of the cycles is consequently $n-3, n-1$, and then $n, n+1, \ldots, 2n-1$. The automaton is 1cfp nfa, accepts ε, a^{n-3}, and a^{n-1}, as well as all the strings of length at least n, but no other strings because going through more than one cycle results in a string of length at least n. □

The next theorem shows that deterministic union-freeness of languages does not accelerate basic regular operations. This contrasts with the results in previously studied subclasses of regular languages such as finite, unary, prefix-, suffix-, factor-, subword-free (or closed, or convex) etc. In the case of intersection and square, the known witness languages are deterministic union-free, see [32, 28]. Slightly changed Maslov's automata, cf. [21], provide lower bounds for star and concatenation, while a modification of the hardest dfa in [16] gives a lower bound for cyclic shift. In the case of reversal, the paper [29] claims that there is a binary n-state dfa language whose reversal requires 2^n deterministic states. Although the witness automaton is one-cycle-free-path dfa, the result cannot be used because the proof is not correct. For $n = 8$, the resulting dfa has only 252 states instead of 256. A similar problem arises here whenever n is divisible by 4.

Theorem 8 (State Complexity) *Let K and L be union-free regular languages over Σ accepted by an m-state and an n-state 1cfp dfa respectively. Then,*

1. $\mathrm{sc}(K \cup L) \leqslant mn$, *and the bound is tight if* $|\Sigma| \geqslant 2$;
2. $\mathrm{sc}(K \cap L) \leqslant mn$, *and the bound is tight if* $|\Sigma| \geqslant 2$;
3. $\mathrm{sc}(K - L) \leqslant mn$, *and the bound is tight if* $|\Sigma| \geqslant 2$;
4. $\mathrm{sc}(K \oplus L) \leqslant mn$, *and the bound is tight if* $|\Sigma| \geqslant 2$;
5. $\mathrm{sc}(KL) \leqslant m2^n - 2^{n-1}$, $m \geqslant 2, n \geqslant 3$, *and the bound is tight if* $|\Sigma| \geqslant 2$;
6. $\mathrm{sc}(L^2) \leqslant n2^n - 2^{n-1}$, *and the bound is tight if* $|\Sigma| \geqslant 2$;
7. $\mathrm{sc}(L^c) \leqslant n$, *and the bound is tight if* $|\Sigma| \geqslant 1$;
8. $\mathrm{sc}(L^*) \leqslant 2^{n-1} + 2^{n-2}$, $n \geqslant 2$, *and the bound is tight if* $|\Sigma| \geqslant 2$;
9. $\mathrm{sc}(L^R) \leqslant 2^n$, $n \geqslant 2$, *and the bound is tight if* $|\Sigma| \geqslant 3$;
10. $\mathrm{sc}(L^{shift}) \leqslant 2^{n^2+n\log n}$. *The bound* $2^{n^2+n\log n-5n}$ *is met if* $|\Sigma| \geqslant 4$.

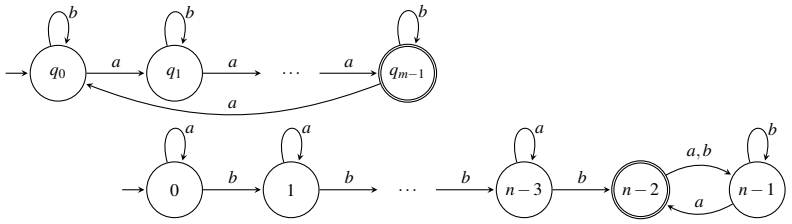

Fig. 4. One-cycle-free-path dfa's meeting the $m2^n - 2^{n-1}$ bound on concatenation.

Proof. 1.-4. The cross-product construction gives the upper bound mn. For all the four operations, the bound is met by deterministic union-free binary languages $((b^*a)^m)^*$ and $((a^*b)^n)^*$, see also [18], except for the case of union with $m = 1$, and the case of symmetric difference with $m = n = 2$. In all the other cases, the strings a^ib^j with $0 \leqslant i \leqslant m-1$ and $0 \leqslant j \leqslant n-1$ are pairwise distinct in the right equivalence defined by the intersection (union, difference, symmetric difference, respectively). For the union with $m = 1$, we take $K = \emptyset$. The bound 4 on the state complexity of symmetric difference in the case of $m = n = 2$ is met by deterministic union-free binary languages $b^*a(a+b)^*$ and $a^*b(a+b)^*$.

5. The upper bound is $m2^n - 2^{n-1}$, see [21, 32], because in the subset automaton corresponding to the standard nfa for concatenation, each reachable subset consists of exactly one state of the first automaton and some states of the second automaton. However, no subset containing an accepting state of the first automaton and not containing the initial state of the second automaton is reached. Note that neither the ternary witness automata in [32] nor the binary witnesses in [14] are 1cfp dfa's. However, Maslov [21] claimed the result for two binary languages accepted by automata, the first of which is a 1cfp dfa, while the other can be modified to become a 1cfp dfa by changing its accepting state from $n-1$ to $n-2$. As no proof is provided in [21], we recall the automata and show that they meet the upper bound.

Consider the languages accepted by the 1cfp dfa's shown in Fig. 4. Construct an nfa for the concatenation of the languages from these dfa's by adding an ε-transition from state q_{m-1} to state 0. The initial state of the resulting nfa is state q_0 and the sole accepting state is $n-2$. We show that the corresponding subset automaton has $(m-1)2^n + 2^{n-1} = m2^n - 2^{n-1}$ reachable and pairwise distinguishable states.

By induction on the size of subsets we first prove that each set $\{q_i\} \cup S$, where $0 \leqslant i \leqslant m-2$ and S is a subset of $\{0, 1, \ldots, n-1\}$, as well as each set $\{q_{m-1}\} \cup T$, where T is a subset of $\{0, 1, \ldots, n-1\}$ containing state 0, is reachable. Each singleton set $\{q_i\}$ with $i \leqslant m-2$ is reached from the initial state $\{q_0\}$ by a^i. Assume the reachability of all appropriate sets of size k, and let $S = \{q_i, j_1, j_2, \ldots, j_k\}$ be a subset of size $k+1$. First, let $i = m-1$, which means that $j_1 = 0$. As symbol a is a permutation symbol in the second dfa, we use $\delta^{-1}(j, a^r)$ to denote the state that goes to state j by a^r. Consider the set $S' = \{q_{m-2}, \delta^{-1}(j_2, a), \ldots, \delta^{-1}(j_k, a)\}$ of size k. Set S' is reachable by the induction hypothesis, and since S' goes to S by

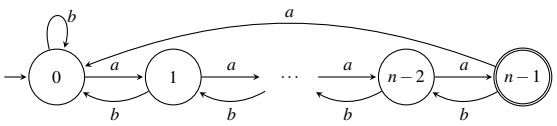

Fig. 5. One-cycle-free-path dfa meeting the $2^{n-1} + 2^{n-2}$ bound on star.

a, set S is reachable as well. Now, let $i \leqslant m-2$ and $j_1 = 0$. Then, set S is reached from the set $\{q_{m-1}, 0, \delta^{-1}(j_2, a^{i+1}), \ldots, \delta^{-1}(j_k, a^{i+1})\}$ by a^{i+1}. Finally, if $i \leqslant m-2$ and $j_1 > 0$, set S is reached from the set $\{q_i, 0, j_2 - j_1, j_3 - j_1, \ldots, j_k - j_1\}$ by b^{j_1}. This concludes the proof of reachability.

Let $\{q_i\} \cup S$ and $\{q_j\} \cup T$ be two distinct reachable sets. If $i < j$, then string $ba^{m-j-1}b^{n-2}$ distinguishes the two subsets. If $i = j$, then S and T differ in a state j, and, moreover, $j > 0$ if $i = m-1$. Then, either string b^{n-j-2} if $j \leqslant n-3$, or the empty string if $j = n-2$, or string a if $j = n-1$ distinguishes the two subsets.

6. The upper bound follows from the upper bound on concatenation, and, as shown in [28], is met by the binary language accepted by a 1cfp dfa with states $0, 1, \ldots, n-1$, where 0 is the initial state, and $n-1$ is the sole accepting state; by a, each state i goes to state $i+1 \bmod n$, and by b, each state i goes to itself except for state 1 that goes to state 0 by b.

7. To get a dfa for complement, we only exchange the accepting and rejecting states. The bound is met by the language $(a^n)^*$.

8. The upper bound is $2^{n-1} + 2^{n-2}$, cf. [32], because in the subset automaton corresponding to the standard nfa for star, the reachable states are as follows: a new initial and accepting state, all the subsets of the state set of a given dfa containing its initial state, and all the non-empty subsets containing neither its initial nor its final state. The witness language in [32] is not deterministic union-free, however, Maslov [21] provides a deterministic union-free witness example shown in Fig. 5. As there is no proof in [21], we give it here. Construct an nfa for the star of the language accepted by the 1cfp dfa in Fig. 5 by adding a new initial and accepting state q_0 that goes to state 1 by a and to state 0 by b, and by adding the transition by a from state $n-2$ to state 0. We prove the reachability of $2^{n-1} + 2^{n-2}$ subsets in the corresponding subset automaton by induction on the size of subsets.

The initial state $\{q_0\}$ and all the singleton sets $\{i\}$ are reachable. Assume that all the subsets of size $k-1$ containing 0, or containing neither 0 nor $n-1$ are reachable. Let $S = \{i_1, i_2, \ldots, i_k\}$ be a subset of size k with $0 \leqslant i_1 < i_2 < \cdots < i_k \leqslant n-1$ (and if $i_1 > 0$, then $i_k < n-1$). First, let $i_1 = 0$. Then, set S is reached from the set $\{i_2 + (n-1) - i_k - 1, i_3 + (n-1) - i_k - 1, \ldots, i_{k-1} + (n-1) - i_k - 1, n-2\}$ of size $k-1$, containing neither 0 nor $n-1$, by string ab^{n-1-i_k}. Now, let $i_1 > 0$. Then, $i_k < n-1$, and set S is reached from the set $\{0, i_2 - i_1, i_3 - i_1, \ldots, i_k - i_1\}$, which contains state 0, by a.

To prove distinguishability notice that the initial (and accepting) state $\{q_0\}$ is equivalent to any state not containing state $n-1$. However, string a^n is accepted

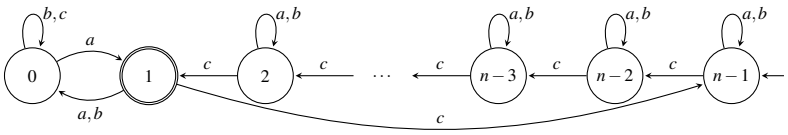

Fig. 6. One-cycle-free-path dfa meeting the 2^n bound on reversal.

by the nfa from state $n-1$ but not from state q_0. Two different subsets of the state set of the given dfa differ in a state i, and string a^{n-1-i} distinguishes them.

9. Reversal of a dfa language is accepted by an nfa obtained from the dfa by reversing all the transitions, and interchanging the role of accepting and initial states. The subset construction gives a dfa of at most 2^n states. As pointed out by Mirkin [24], Lupanov's ternary worst-case example for nfa-to-dfa conversion in [20] is, in fact, a reversed dfa. Leiss [19] presented a ternary and a binary dfa's that meet the upper bound.

As none of these automata is 1cfp dfa, consider the 1cfp dfa shown in Fig. 6. Construct the reversed nfa. Note that in this nfa each state i goes to state $(i+1) \bmod n$ by ca. It follows that, in the subset automaton, each subset not containing state 0 is reached from a subset containing state 0 by a string in $(ca)^*$. We show by induction on the size of subsets that each subset of the state set $\{0,1,\ldots,n-1\}$ containing state 0 is reachable in the subset automaton.

The set $\{0\}$ is reached from the initial state $\{1\}$ of the subset automaton by a. The subset $\{0,i_1,i_2,\ldots,i_k\}$, where $1 \leqslant i_1 < i_2 < \cdots < i_k \leqslant n-1$, of size $k+1$ is reached from the set $\{0,i_2-i_1+1,i_3-i_1+1,\ldots,i_k-i_1+1\}$ of size k by string bc^{i_1-1}. Finally, the empty set is reached from state $\{1\}$ by b. For distinguishability, notice that string c^{n-1-i} is accepted by the nfa only from state i for $i=1,2,\ldots,n-1$, and string ac^{n-2} is accepted only from state 0.

10. The upper bound follows from [16, 21]. The work [16] proves the lower bound $2^{n^2+n\log n-5n}$ for the language accepted by the dfa of Fig. 7 over the alphabet $\{a,b,c,d\}$. By a, states 0 and $n-1$ go to itself and there is a cycle $(1,2,\ldots,n-2)$; by b, state 0 goes to itself and there is a cycle $(1,2,\ldots,n-1)$; by c, all the states go to itself except for state 0 that goes to 1 and state 1 that goes to 0; by d, all the states go to state 0 except for state $n-1$ that goes to state 1. This automaton is not one-cycle-free-path dfa. Therefore, change transitions on symbol b, see Fig. 8, so that in the new dfa by b, all the states go to itself, except for state $n-2$ that goes to $n-1$ and state $n-1$ that goes to $n-2$. The resulting automaton is a 1cfp dfa, and, moreover, the transitions by old symbol b are now implemented by string ba. It follows that the proof in [16] works for the new 1cfp dfa if we replace all the occurrences of b in the proof by ba. □

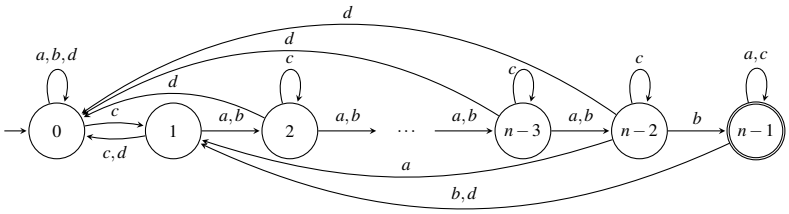

Fig. 7. The dfa meeting the $2^{n^2+n\log n-5n}$ bound on cyclic shift.

5. Conclusions

We have investigated union-free regular languages described by regular expressions without the operation of union. Using results of Nagy [26] on characterization of automata accepting those languages, we have proved additional closure properties, and studied the nondeterministic state complexity of regular operations. We have shown that all the known upper bounds for regular languages are met by union-free languages, except for reversal operation, where the tight bound is n instead of $n+1$. This gives rise to a question where is the breakpoint of this complexity.

Furthermore, we have defined deterministic union-free languages as languages accepted by deterministic one-cycle-free-path automata, and proved that they are properly included in the class of union-free languages. We have examined the state complexity of a number of regular operations, and have shown that deterministic union-freeness of languages accelerates none of them. This contrasts with results on complexity of operations in previously studied subclasses of regular languages.

Some questions remain open. We conjecture that for the difference of two union-free languages, nfa's need $m2^n$ states, and we do not now the result on the shuffle of deterministic union-free languages. A description of deterministic union-free regular languages in terms of regular expressions or grammars, as well as the case of unary union-free languages, is of interest, too.

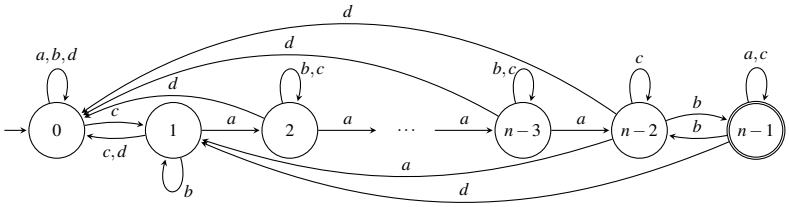

Fig. 8. One-cycle-free-path dfa meeting the $2^{n^2+n\log n-5n}$ bound on cyclic shift.

References

[1] S. Afonin and D. Golomazov. Minimal union-free decompositions of regular languages. In *Proc. of LATA 2009*, volume 5457 of *LNCS*, pages 83–92. Springer, 2009.
[2] A. V. Aho, J. D. Ullman, and M. Yannakakis. On notions of information transfer in VLSI circuits. In *Proc. of STOC 1983*, pages 133–139, 1983.
[3] J.-C. Birget. Intersection and union of regular languages and state complexity. *Inform. Process. Lett.*, 43:185–190, 1992.
[4] J.-C. Birget. Partial orders on words, minimal elements of regular languages, and state complexity. *Theoret. Comput. Sci.*, 119:267–291, 1993. Erratum available at `http://clam.rutgers.edu/∼birget/poWordsERR.ps`.
[5] J. Brzozowski. *Regular expression techniques for sequential circuits*. PhD thesis, Department of Electrical Engineering, Princeton University, Princeton, NJ, June 1962.
[6] C. Câmpeanu, K. Salomaa, and S. Yu. Tight lower bound for the state complexity of shuffle of regular languages. *J. Autom. Lang. Comb.*, 7(3):303–310, 2002.
[7] S. Crvenković, I. Dolinka, and Zoltán Ésik. On equations for union-free regular languages. *Inform. and Comput.*, 164(1):152–172, 2001.
[8] M. Domaratzki, D. Kisman, and J. Shallit. On the number of distinct languages accepted by finite automata with n states. *J. Autom. Lang. Comb.*, 7(4):469–486, 2002.
[9] M. Domaratzki and A. Okhotin. State complexity of power. *Theoret. Comput. Sci.*, 410(24-25):2377–2392, 2009.
[10] I. Glaister and J. Shallit. A lower bound technique for the size of nondeterministic finite automata. *Inform. Process. Lett.*, 59:75–77, 1996.
[11] Y.-S. Han and K. Salomaa. State complexity of basic operations on suffix-free regular languages. *Theoret. Comput. Sci.*, 410(27-29):2537–2548, 2009.
[12] M. Holzer and M. Kutrib. Descriptional and computational complexity of finite automata–a survey. *Inform. and Comput.*, 209(3):456–470, 2011.
[13] J. Hromkovič. *Communication complexity and parallel computing*. Springer, Heidelberg, 1997.
[14] J. Jirásek, G. Jirásková, and A. Szabari. State complexity of concatenation and complementation. *Int. J. Found. Comput. Sci.*, 16(3):511–529, 2005.
[15] G. Jirásková. State complexity of some operations on binary regular languages. *Theoret. Comput. Sci.*, 330:287–298, 2005.
[16] G. Jirásková and A. Okhotin. State complexity of cyclic shift. *Theor. Inform. Appl.*, 42(2):335–360, 2008.
[17] G. Jirásková and A. Okhotin. Nondeterministic state complexity of positional addition. In *Proc. of DCFS 2009*, pages 151–161. EPTCS vol. 3, 2009.
[18] M. Kutrib and M. Holzer. Nondeterministic descriptional complexity of regular languages. *Int. J. Found. Comput. Sci.*, 14(6):1087–1102, 2003.
[19] E. Leiss. Succint representation of regular languages by boolean automata. *Theoret. Comput. Sci.*, 13:323–330, 1981.
[20] O. B. Lupanov. Über den vergleich zweier typen endlicher quellen (German. Russian original). *Probl. Kybernetik*, 6:328–335, 1966. translation from Probl. Kibernetiki 9, 321-326 (1963).
[21] A. N. Maslov. Estimates of the number of states of finite automata. *Soviet Math. Dokl.*, 11(5):1373–1375, 1970.
[22] R. McNaughton and S. Papert. *Counter-Free Automata*. The MIT Press, 1971.
[23] A. R. Meyer and M. J. Fischer. Economy of description by automata, grammars, and formal systems. In *Proc. of FOCS 1971*, pages 188–191. IEEE, 1971.
[24] B. G. Mirkin. On dual automata. *Kibernetika*, 2(1):7–10, 1966.

[25] F. R. Moore. On the bounds for state-set size in the proofs of equivalence between deterministic, nondeterministic, and two-way finite automata. *IEEE Trans. Comput.*, 20(10):1211–1214, 1971.

[26] B. Nagy. Union-free regular languages and 1-cycle-free-path automata. *Publ. Math. Debrecen*, 68(1-2):183–197, 2006.

[27] B. Nagy. On union-complexity of regular languages. In *Proc. of CINTI 2010*, pages 177–182. IEEE, 2010.

[28] N. Rampersad. The state complexity of L^2 and L^k. *Inform. Process. Lett.*, 98(6):231–234, 2006.

[29] A. Salomaa, D. Wood, and S. Yu. On the state complexity of reversals of regular languages. *Theoret. Comput. Sci.*, 320:315–329, 2004.

[30] M. Sipser. *Introduction to the theory of computation.* PWS Publishing Company, Boston, 1997.

[31] S. Yu. Chapter 2: Regular languages. In *Handbook of Formal Languages – Vol. I*, pages 41–110. Springer, Heidelberg, 1997.

[32] S. Yu, Q. Zhuang, and K. Salomaa. The state complexities of some basic operations on regular languages. *Theoret. Comput. Sci.*, 125(2):315–328, 1994.