Operations on Permutation Automata

Michal Hospodár¹✉ and Peter Mlynářík^{1,2}

¹ Mathematical Institute, Slovak Academy of Sciences, Grešákova 6,
040 01 Košice, Slovakia

hosmich@gmail.com, mlynarcik1972@gmail.com

² Faculty of Humanities and Natural Sciences, University of Prešov,
Ul. 17. novembra 1, 081 16 Prešov, Slovakia

Abstract. We investigate the class of languages recognized by permutation deterministic finite automata. Using automata constructions and some properties of permutation automata, we show that this class is closed under Boolean operations, reversal, and quotients, and it is not closed under concatenation, power, Kleene closure, positive closure, cut, shuffle, cyclic shift, and permutation. We prove that the state complexity of Boolean operations, Kleene closure, positive closure, and right quotient on permutation languages is the same as in the general case of regular languages. Next, we get the tight upper bounds on the state complexity of concatenation ($m2^n - 2^{n-1} - m + 1$), square ($n2^{n-1} - 2^{n-2}$), reversal ($\binom{n}{\lceil n/2 \rceil}$), and left quotient ($\binom{m}{\lceil m/2 \rceil}$; tight if $m \leq n$). All our witnesses are unary or binary, and the binary alphabet is always optimal, except for Boolean operations in the case of $\gcd(m, n) = 1$. In the unary case, the state complexity of all considered operations is the same as for regular languages, except for quotients and cut. In case of quotients, it is $\min\{m, n\}$, and in case of cut, it is either $2m - 1$ or $2m - 2$, depending on whether there exists an integer ℓ with $2 \leq \ell \leq n$ such that $m \bmod \ell \neq 0$.

1 Introduction

A deterministic finite automaton (DFA) is said to be a permutation DFA if every input symbol induces a permutation on the state set. A language recognized by a permutation DFA is called a permutation language. The class of permutation languages has been introduced by Thierrin [22] who also proved some closure properties of this class using cancellative congruences. The aim of this paper is to study this class in detail, including the operational state complexity.

The state complexity of a regular operation is the number of states sufficient and necessary in the worst case for a DFA to accept the language resulting from the operation, considered as a function of the numbers of states in DFAs for operands. The state complexity of basic regular operations was given by Maslov [15] and Yu et al. [23].

Research supported by VEGA grant 2/0132/19 and grant APVV-15-0091.

© Springer Nature Switzerland AG 2020

N. Jonoska and D. Savchuk (Eds.): DLT 2020, LNCS 12086, pp. 122–136, 2020.

https://doi.org/10.1007/978-3-030-48516-0_10

If the operands of an operation belong to some subclass of regular languages, then the complexity of this operation may be significantly smaller than in the general case. The operational state complexity in several subclasses of regular languages has been studied in the literature. Cmpeanu et al. [6] considered finite languages, Han and Salomaa [8,9] investigated prefix-free and suffix-free languages, and Brzozowski et al. [3,4] examined ideal and closed languages. The classes of co-finite, star-free, union-free, and unary languages have been studied as well [1,5,13,18].

In this paper, we first study closure properties of the class of permutation languages. We provide alternative proofs to those in [22]; our proofs use automata constructions and properties of permutation DFAs. Moreover, we prove that permutation languages are not closed under power, positive closure, cut, shuffle, cyclic shift, and permutation. Then we study the operational state complexity in the class of permutation languages. For each considered operation, we obtain tight upper bounds on its state complexity. Our witnesses are defined over unary or binary alphabets. In the case of left and right quotients, we need $m \leq n$ to prove tightness.

2 Preliminaries

We assume that the reader is familiar with the backgrounds in formal languages and automata theory. For details, the reader may refer to [11,21].

For a rational number r , we denote the set $\{i \in \mathbb{Z} \mid 0 \leq i < r\}$ by $\mathbf{r}$. For example, we have $\mathbf{3} = \{0, 1, 2\}$, and for an integer n we have $\mathbf{n} = \{0, 1, \dots, n-1\}$. Next, we have $\frac{7}{2} = \{0, 1, 2, 3\}$ and $\frac{n}{2} = \{0, 1, \dots, \lceil \frac{n}{2} \rceil - 1\}$. For a finite set S , its size is denoted by $|S|$, and its power-set by 2^S .

Let Σ be a non-empty alphabet of symbols. Then Σ^* denotes the set of all strings over Σ including the empty string ε . A language is any subset of Σ^* .

Given two languages K and L over Σ , the *complement* of L is the language $L^c = \Sigma^* \setminus L$, and the operations of *intersection*, *union*, *difference*, and *symmetric difference* are defined as standard set operations. The *concatenation* of K and L is $KL = \{uv \mid u \in K \text{ and } v \in L\}$. The k -th power of L is defined as $L^0 = \{\varepsilon\}$ and $L^k = LL^{k-1}$ if $k \geq 1$. The second power is called *square*. The *Kleene closure* of L is $L^* = \bigcup_{i \geq 0} L^i$. The *positive closure* of L is $L^+ = \bigcup_{i \geq 1} L^i$. The *right quotient* of K by L is $KL^{-1} = \{w \mid wx \in K \text{ for some } x \text{ in } L\}$. The *left quotient* of K by L is $L^{-1}K = \{w \mid xw \in K \text{ for some } x \text{ in } L\}$. The *cut* of K and L is $K!L = \{uv \mid u \in K, v \in L, \text{ and } uv' \notin K \text{ for every non-empty prefix } v' \text{ of } v\}$. The *shuffle* of K and L is $K \sqcup L = \{u_0v_1u_1 \cdots v_ku_k \mid u_0u_1 \cdots u_k \in K \text{ and } v_1v_2 \cdots v_k \in L\}$. The *cyclic shift* of L is $\text{shift}(L) = \{uv \mid vu \in L\}$. The *permutation* of L is $\text{per}(L) = \{w \mid \psi(w) = \psi(v) \text{ for some } v \text{ in } L\}$ where $\psi(w)$ is the Parikh vector of w .

A *nondeterministic finite automaton with multiple initial states* (MNFA) is a quintuple $M = (Q, \Sigma, \cdot, I, F)$ where Q is a finite non-empty set of states, Σ is a non-empty alphabet of input symbols, $I \subseteq Q$ is the set of initial states, $F \subseteq Q$ is the set of final states, and $\cdot : Q \times \Sigma \rightarrow 2^Q$ is the transition function which is

naturally extended to the domain $2^Q \times \Sigma^*$. For a state q , a set S , and a string w , we write qw and Sw instead of $q \cdot w$ and $S \cdot w$ if it does not cause any confusion. The language accepted by M is the set $L(M) = \{w \in \Sigma^* \mid Iw \cap F \neq \emptyset\}$.

For states p and q and a symbol a , we write that M has a transition (p, a, q) if $q \in pa$. We also say that p has an *out-transition* on a and q has an *in-transition* on a . A state q is called a *sink state* if it has the transition (q, a, q) for each input symbol a and no other out-transitions. To omit a state means to remove it from the set of states and to remove all its in-transitions and out-transitions from the transition function.

The *reverse* of an MNFA M is the MNFA M^R obtained from M by reverting all transitions and swapping the role of initial and final states. A subset S of Q is *reachable* in M if $S = Iw$ for some string w , and it is *co-reachable* in M if it is reachable in M^R .

If $|I| = 1$, we say that M is a *nondeterministic finite automaton* (NFA) and write $(Q, \Sigma, \cdot, s, F)$ instead of $(Q, \Sigma, \cdot, \{s\}, F)$.

An NFA $(Q, \Sigma, \cdot, s, F)$ is called *deterministic* (DFA) if $|qa| = 1$ for each state q and each symbol a . Similarly, an MNFA with this property is said to be an MDFA. We write $pa = q$ instead of $pa = \{q\}$ and use the notation $p \xrightarrow{a} q$. A DFA is *minimal* if its language cannot be accepted by any smaller DFA (with respect to number of states). It is well known that a DFA is minimal if and only if all its states are reachable and pairwise distinguishable. The state complexity of a language L , $sc(L)$, is the number of states in a minimal DFA accepting L .

Every MNFA $M = (Q, \Sigma, \cdot, I, F)$ can be converted to an equivalent deterministic finite automaton $\mathcal{D}(M) = (2^Q, \Sigma, \cdot, I, \{S \mid S \cap F \neq \emptyset\})$ [19]. This DFA is called the *subset automaton* of M . The DFA $\mathcal{D}(M)$ may not be minimal since some of its states may be unreachable or equivalent to other states. To prove distinguishability of states in subset automata, we use the following observation.

Lemma 1 ([12, Lemma 1]). *If for each state q of an MNFA M , the singleton set $\{q\}$ is co-reachable in M , then all states of the subset automaton $\mathcal{D}(M)$ are pairwise distinguishable.*

To prove minimality of unary DFAs, we use the following lemma.

Lemma 2 ([16, Lemma 1]). *A unary DFA $(\mathbf{n}, \{a\}, \cdot, 0, F)$ with $i \cdot a = i + 1$ for $i = 0, 1, \dots, n - 2$ and $(n - 1) \cdot a = k$ for some k in $\mathbf{n}$ is minimal if and only if the two following conditions hold:*

- (1) *its loop (the DFA obtained by omitting states in $\mathbf{k}$ and making k the initial state) is minimal,*
- (2) *if $k \neq 0$, then the states $k - 1$ and $n - 1$ do not have the same finality.*

A DFA $A = (Q, \Sigma, \cdot, s, F)$ is said to be a *permutation DFA* if for each $p, q \in Q$ and each $a \in \Sigma$, $p \cdot a = q \cdot a$ implies that $p = q$. A language is said to be a *permutation language* if it is recognized by a permutation DFA.

In a permutation DFA, each input symbol a induces a permutation on Q , namely $q \mapsto q \cdot a$. To describe transitions in permutation automata, we use the following notation. The formula $a: (p_1, p_2, \dots, p_k)$ denotes that $p_i \cdot a = p_{i+1}$

for $i = 1, 2, \dots, k-1$, $p_k \cdot a = p_1$, and $q \cdot a = q$ if $q \neq p_i$ for $i = 1, 2, \dots, k$. For example, we denote the maximal cyclic permutation on $\mathbf{n}$ by $(0, 1, \dots, n-1)$, the transposition of 0 and 1 by $(0, 1)$, and the identity by (0) . The next observation shows that a language is a permutation language if and only if its minimal DFA is a permutation DFA.

Proposition 3. *Let A be a permutation DFA. Then the minimal DFA equivalent to A is a permutation DFA.*

Proof. Assume that a permutation DFA $A = (Q_A, \Sigma, \cdot, s_A, F_A)$ is equivalent to a minimal DFA $B = (Q_B, \Sigma, \circ, s_B, F_B)$ which is not a permutation DFA. Then there exist two distinct states p and q in B and an input symbol a in Σ such that $p \circ a = q \circ a$. Since B is minimal, the states p and q are reachable from s_B by some strings u and v , respectively. Let $p' = s_A \cdot u$ and $q' = s_A \cdot v$ be the corresponding states in A . Since B is minimal, the states p and q are not equivalent, and therefore also p' and q' are not equivalent. However, $p' \cdot a$ and $q' \cdot a$ are equivalent since $p \circ a = q \circ a$. Moreover, for every positive i , the states $p' \cdot a^i$ and $q' \cdot a^i$ are equivalent as well. Since A is a permutation DFA, there exist positive integers k and ℓ such that $p' \cdot a^k = p'$ and $q' \cdot a^\ell = q'$. Then $p' \cdot a^{k\ell} = p'$ and $q' \cdot a^{k\ell} = q'$. It follows that p' and q' are equivalent, a contradiction. $\square$

Notice that the converse does not hold; for example, the minimal unary DFA for a^* is a permutation DFA, however, this language is accepted by the two-state DFA $(\mathbf{2}, \{a\}, \cdot, 0, \mathbf{2})$ with $0 \cdot a = 1$ and $1 \cdot a = 0$ which is not a permutation DFA. We use the following corollary of Proposition 3 without citing it.

Corollary 4. *A language L is a permutation language if and only if the minimal DFA accepting L is a permutation DFA.*

Lemma 5. *If a DFA A has a reachable sink state and $L(A) \notin \{\emptyset, \Sigma^*\}$, then $L(A)$ is not a permutation language.*

Proof. Let A have a reachable sink state. Let B be the minimal DFA for $L(A)$. Then B must have a reachable sink state d . Since $L(A) \notin \{\emptyset, \Sigma^*\}$, the state d is reached from some other reachable state q by a symbol a . Then $q \cdot a = d \cdot a$, hence B is not a permutation automaton, and the lemma follows. $\square$

A MDFA $(Q, \Sigma, \cdot, I, F)$ is said to be a *permutation MDFA* if each input symbol a in Σ induces a permutation on Q .

Lemma 6. *Every language recognized by a permutation MDFA is a permutation language.*

Proof. Let $M = (Q, \Sigma, \cdot, I, F)$ be a permutation MDFA. Let us show that the subset automaton $\mathcal{D}(M)$ is a permutation DFA. Let S and T be subsets of Q and assume that $Sa = Ta$. Since M is a permutation automaton, we have $|S| =$

$|Sa| = |Ta| = |T|$. If Sa is empty, then S and T are empty as well. Otherwise, let $|Sa| = k$ and let $Sa = \{p_1, p_2, \dots, p_k\}$. For each p_i in Sa there is a state s_i in S and a state t_i in T such that $s_i \cdot a = p_i = t_i \cdot a$. This means that $s_i = t_i$. Moreover, if $i \neq j$, then $p_i \neq p_j$, and therefore $s_i \neq s_j$ and $t_i \neq t_j$ since M is a permutation MDFA. It follows that $S = \{s_1, s_2, \dots, s_k\} = \{t_1, t_2, \dots, t_k\} = T$, so the subset automaton $\mathcal{D}(M)$ is a permutation DFA. $\square$

Lemma 7. *Let M be an n -state permutation MDFA. Then $L(M)$ is accepted by a permutation DFA with at most $\binom{n}{\lceil n/2 \rceil}$ states. This bound is met by permutation MDFA $M = (\mathbf{n}, \{a, b\}, \cdot, \frac{n}{2}, \{0\})$ with $a: (0, 1, \dots, n-1)$ and $b: (0, 1)$.*

Proof. Let $M = (Q, \Sigma, \cdot, I, F)$ be a permutation MDFA. Let $|I| = k$. Then each reachable set in the subset automaton $\mathcal{D}(M)$ is of size k . Hence $\mathcal{D}(M)$ has at most $\binom{n}{k}$ states. Since $\binom{n}{k} \leq \binom{n}{\lceil n/2 \rceil}$, we get our upper bound. Let us show that this upper bound is met by the MDFA M from the statement of the lemma. Since a and b are generators of the symmetric group, and the symmetric group acts transitively on all subsets of given fixed size, each subset of $\mathbf{n}$ of size $\lceil n/2 \rceil$ is reachable in the subset automaton $\mathcal{D}(M)$ from the initial subset. Since each singleton set $\{i\}$ is co-reachable in M via a string in a^* , all states in $\mathcal{D}(M)$ are pairwise distinguishable by Lemma 1. This gives our lower bound. $\square$

3 Closure Properties of Permutation Languages

In this section, we study the closure properties of the class of permutation languages. The results for Boolean operations, concatenation, Kleene closure, reversal, and quotients have been already obtained by Thierrin [22] using cancellative congruences; here we provide alternative proofs using automata constructions and properties of permutation automata. Moreover, we consider the operations of power, positive closure, cut, shuffle, cyclic shift, and permutation.

Theorem 8 (Closure Properties). *The class of permutation languages is closed under complementation, intersection, union, difference, symmetric difference, reversal, right and left quotient, and it is not closed under concatenation, cut, shuffle, power, Kleene and positive closure, cyclic shift, and permutation.*

Proof. (a) Let K and L be accepted by permutation DFAs A and B , respectively. The language L^c is accepted by a DFA with the same transitions as in B , hence it is a permutation language. Next, intersection, union, difference, and symmetric difference are accepted by a product automaton with appropriately defined final states. The resulting product automaton is a permutation DFA. The language L^R is accepted by a permutation MDFA obtained from B by reversing all transitions and by swapping the roles of initial and final states, and it is a permutation language by Lemma 6. If $A = (Q, \Sigma, \cdot, s, F)$, then the right quotient KL^{-1} is accepted by a permutation DFA obtained from A by making all states in the

set $\{q \mid q \cdot w \in F \text{ for some } w \text{ in } L\}$ final. The left quotient $L^{-1}K$ is accepted by a permutation MDFA obtained from A by making all states in $\{s \cdot w \mid w \in L\}$ initial, so it is a permutation language by Lemma 6.

(b) Let $k \geq 2$ and $K = L = a(aa)^*$. Then K and L are unary permutation languages and we have $KL = K!L = K \sqcup L = aa(aa)^*$ and $L^k = a^k(aa)^*$, which are not permutation languages.

(c) Let $L = aa(aaa)^*$. Then L is a unary permutation language and we have $L^* = \{\varepsilon, aa\} \cup a^4a^*$ and $L^+ = \{aa\} \cup a^4a^*$, which are co-finite and different from a^* . By Lemma 5, they are not permutation languages.

(d) Let $A = (\mathbf{3}, \{a, b\}, \cdot, 0, \{0, 2\})$ be a permutation DFA with $a: (0, 1, 2)$ and $b: (0, 1)$. Construct an MNFA with ε -transitions for $\text{shift}(L(A))$ as described in [14, p. 340]. In the corresponding subset automaton, the initial subset is $I = \{q_{00}, q_{11}, q_{22}, p_{00}, p_{20}\}$, and we have $Ib = \{q_{01}, q_{10}, q_{22}, p_{01}, p_{10}, p_{20}, p_{21}\}$ and $Iba = \{q_{02}, q_{11}, q_{20}, p_{00}, p_{02}, p_{11}, p_{20}, p_{21}, p_{22}\}$. Notice that the set Ib is not final and $Iba \supseteq \{p_{20}, p_{21}, p_{22}\}$. The transitions on states p_{20}, p_{21}, p_{22} are the same as in A , so a and b perform permutations on these three states. This means that $Iba \cdot w$ is always a superset of $\{p_{20}, p_{21}, p_{22}\}$ and therefore each string is accepted from the final state Iba . Hence Iba is equivalent to a final sink state, which means that the minimal DFA for $\text{shift}(L(A))$ has a reachable final sink state. By Lemma 5, $\text{shift}(L(A))$ is not a permutation language.

(e) Let $A = (\mathbf{3}, \{a, b\}, \cdot, 0, \{2\})$ be a permutation DFA with $a: (0, 1, 2)$. The language $\text{per}(L(A))$ consists of strings which have at least one a and at least one b ; notice that A has loops $(0, b, 0)$ and $(2, a, 2)$. It is accepted by the DFA $(\mathbf{4}, \{a, b\}, \circ, 0, \{3\})$ with the transitions $(0, a, 1)$, $(0, b, 2)$, $(1, a, 1)$, $(1, b, 3)$, $(2, a, 3)$, $(2, b, 2)$, $(3, a, 3)$, and $(3, b, 3)$ which has a reachable sink state 3. Hence $\text{per}(L(A))$ is not a permutation language by Lemma 5. $\square$

4 State Complexity of Operations on Permutation Languages

In this section, we study the state complexity of operations on permutation languages. Let us start with Boolean operations.

Theorem 9 (Boolean Operations). *Let $m, n \geq 3$. Let K and L be languages accepted by permutation DFAs with m and n states, respectively. Then we have $\text{sc}(K \cup L), \text{sc}(K \cap L), \text{sc}(K \setminus L), \text{sc}(K \oplus L) \leq mn$, and these upper bounds are met by binary permutation languages. Moreover, if $\gcd(m, n) = 1$, then these bounds are met by unary permutation languages.*

Proof. The upper bound mn is the same as for regular languages and it is met by binary permutation languages $K = \{w \in \{a, b\}^* \mid |w|_a = 0 \bmod m\}$ and $L = \{w \in \{a, b\}^* \mid |w|_b = 0 \bmod n\}$ by [13, Theorem 8(1-4)]. In the unary case, the permutation languages $(a^m)^*$ and $(a^n)^*$ meet the upper bound for all four operations whenever $\gcd(m, n) = 1$; distinguishability in the mn -state cyclic DFA can be proved by using the Chinese Remainder Theorem. $\square$

We continue with concatenation, power, Kleene and positive closure, and reversal. We first consider the unary case, then we deal with a general alphabet.

Theorem 10 (Basic Operations: Unary Case). *Let $m, n \geq 2$. Let K and L be unary languages accepted by permutation DFAs with m and n states, respectively. Then*

- (a) $\text{sc}(KL) = \text{sc}(K \sqcup L) \leq mn$, and this bound is tight if $\gcd(m, n) = 1$;
- (b) $\text{sc}(L^k) \leq k(n-1) + 1$, and this bound is tight;
- (c) $\text{sc}(L^+), \text{sc}(L^*) \leq (n-1)^2 + 1$, and this bound is tight;
- (d) $\text{sc}(L^R) \leq n$, and this bound is tight.

Proof. (a) In the case of unary languages, we have $KL = K \sqcup L$, and the upper bound mn follows from [23, Theorem 5.5]. For tightness in the case when $\gcd(m, n) = 1$, consider the unary permutation languages $K = a^{m-1}(a^m)^*$ and $L = a^{n-1}(a^n)^*$. Then $\text{sc}(KL) = mn$ by [23, Theorem 5.4].

(b) The upper bound is the same as for regular languages [20, Theorem 3], and it is met by the permutation language $a^{n-1}(a^n)^*$ [20, Proof of Theorem 4].

(c) Let $A = (Q, \{a\}, \cdot, s, F)$ be a permutation DFA for L . If $F = \{s\}$, then $L^+ = L^* = L$, and $\text{sc}(L^+) = \text{sc}(L^*) \leq n$. Otherwise, if L is empty, then the upper bound follows since $n \geq 2$. Let L be non-empty and $F \neq \{s\}$. We construct an n -state NFA for L^+ by adding the transitions (q, a, s) whenever $q \cdot a \in F$. In the corresponding subset automaton, the initial subset is $\{s\}$ and there exists an integer t such that $t \leq n-1$ and $|\{s\} \cdot a^t| \geq 2$. Since A is a cyclic unary DFA, we cannot reach the singleton set $\{s\}$ from any other state of the subset automaton. It follows that to get a DFA for L^* , we only need to mark the initial state $\{s\}$ as final. It is shown in [23, Proof of Theorem 5.3] that the subset automaton for L^+ has at most $(n-1)^2 + 1$ reachable states, and that this upper bound is met by the permutation language $(aa)^*$ if $n = 2$ and $a^{n-1}(a^n)^*$ if $n \geq 3$.

(d) The upper bound n is met by the permutation language $(a^n)^*$. $\square$

Theorem 11 (Basic Operations: General Case). *Let $m, n \geq 2$. Let K and L be languages accepted by permutation DFAs with m and n states. Then*

- (a) $\text{sc}(KL) \leq m2^n - 2^{n-1} - m + 1$,
- (b) $\text{sc}(L^2) \leq n2^{n-1} - 2^{n-2}$,
- (c) $\text{sc}(L^+) \leq 2^{n-1} + 2^{n-2} - 1$ and $\text{sc}(L^*) \leq 2^{n-1} + 2^{n-2}$,
- (d) $\text{sc}(L^R) \leq \binom{n}{\lceil n/2 \rceil}$,

and these bounds are met by binary witnesses and the binary alphabet is optimal.

Proof. (a) If $K = \emptyset$ or $L = \emptyset$, then $KL = \emptyset$. Otherwise, let K and L be accepted by permutation DFAs $A = (Q_A, \Sigma, \cdot_A, s_A, F_A)$ and $B = (Q_B, \Sigma, \cdot_B, s_B, F_B)$ with $Q_A \cap Q_B = \emptyset$, $|Q_A| = m$, $|Q_B| = n$, $|F_A| = k \geq 1$, and $|F_B| \geq 1$. Then KL is accepted by an MNFA $(Q_A \cup Q_B, \Sigma, \cdot, I, F_B)$ where $I = \{s_A\}$ if $s_A \notin F_A$ and $I = \{s_A, s_B\}$ otherwise, and the transition function $\cdot$ contains all transitions

of $\cdot_A$ and $\cdot_B$ and moreover the transitions (q, a, s_B) whenever $q \cdot_A a \in F_A$. In the corresponding subset automaton, only the states of the form $\{q\} \cup S$ with $q \in Q_A$ and $S \subseteq Q_B$ are reachable; denote such a state by (q, S) . Moreover, if $q \in F_A$, then $s_B \in S$. Next, since B is a permutation automaton, we have $Q_B \cdot w = Q_B$ for every string w in Σ^* . It follows that every string is accepted from (p, Q_B) for each p in Q_A , and all these m states are equivalent to each other. This gives the upper bound $(m - k)2^n + k2^{n-1} - m + 1 \leq m2^n - 2^{n-1} - m + 1$.

For tightness, consider the permutation DFAs $A = (\mathbf{m}, \{a, b\}, \cdot_A, 0, \{m - 1\})$ with $a: (0, 1, \dots, m - 1)$ and $b: (0)$ and $B = (\mathbf{n}, \{a, b\}, \cdot_B, 0, \{n - 1\})$ with $a: (0)$ and $b: (0, 1, \dots, n - 1)$. Notice that in A , the symbol a induces the maximal cyclic permutation while b is an identity, and the transitions in B are symmetric. Construct an NFA for $L(A)L(B)$ as described above. Let us show by induction on $|S|$ that each state (i, S) with $i = 0, 1, \dots, m - 2$ and each state $(m - 1, S)$ with $0 \in S$ is reachable in the corresponding subset automaton. The base case, $|S| = 0$, holds true since for every i with $1 \leq i \leq m - 2$, the state $(i, \emptyset)$ is reached from the initial state $(0, \emptyset)$ by a^i . Now assume that our claim holds for each S' with $|S'| = k$. Let $|S| = k + 1$. Consider three cases:

- (1) $i = m - 1$. Then we must have $0 \in S$. Take $S' = S \setminus \{0\}$. Then $(m - 2, S')$ is reachable by the induction hypothesis and it is sent to $(m - 1, S)$ by a .
- (2) $0 \leq i \leq m - 2$ and $0 \in S$. Then the set $(m - 1, S)$ is reachable as shown in case (1) and it is sent to (i, S) by a^{i+1} .
- (3) $0 \leq i \leq m - 2$ and $0 \notin S$. Take $S' = \{s - \min S \mid s \in S\}$. Then $0 \in S'$, so the set (i, S') is reachable as shown in case (2) and it is sent to (i, S) by $b^{\min S}$.

Now we prove distinguishability. Each state $(i, \mathbf{n})$ is equivalent to a final sink state. Consider a state (i, S) with $S \neq \mathbf{n}$. Let $s \notin S$. Then b^{n-1-s} is rejected from (i, S) , so (i, S) is not equivalent to a final sink state. Let (i, S) and (j, T) be two distinct states of the subset automaton with $S, T \neq \mathbf{n}$. First let $S \neq T$, and without loss of generality, let $s \in S \setminus T$. Then the string b^{n-1-s} is accepted from S and it is rejected from T . Now let $S = T$, and without loss of generality, let $i < j$. Since $S \neq \mathbf{n}$, there exists a state s with $s \notin S$. Let $S' = \{(p - s) \bmod n \mid p \in S\}$. Then $0 \notin S'$. Consider the string $b^{n-s}a^{m-1-j}$. It leads (i, S) to $(i + m - 1 - j, S')$ and it leads (j, S) to $(m - 1, \{0\} \cup S')$. Since the second components of the resulting states differ in the state 0, the states (i, S) and (j, S) are distinguishable. This completes our proof for concatenation.

(b) If the DFA for L has a unique final state which is initial, then $L^2 = L$. Otherwise, let L be accepted by a permutation DFA A with the initial state s and at least one final state f with $f \neq s$. Construct the NFA N for $L^2 = LL$ as described in case (a). If the initial state s of A is final, then the initial state of the subset automaton $\mathcal{D}(N)$ is $(s, \{s\})$. It follows that for each reachable state (q, S) of $\mathcal{D}(N)$, we have $q \in S$. Moreover, if q is final, then $s \in S$. In total, we get at most $2^{n-1} + 2^{n-2} + (n - 2)2^{n-1} = n2^{n-1} - 2^{n-2}$ reachable states. Now assume that s is not final. Let us show that no state (q, S) with $q \in S$ is reachable in $\mathcal{D}(N)$. Since s is not final, the initial state of $\mathcal{D}(N)$ is $(s, \emptyset)$ with $s \notin \emptyset$. Now let (q, S) be any reachable state with $q \notin S$. Consider the state $(p, T) = (q, S)a$ for any input symbol a . Assume for a contradiction that $p \in T$. It follows

that $qa = p$ and either there is a state r in S such that $ra = p$, or $p = s$ and p is final. In the former case, we get a contradiction with the fact that A is a permutation DFA. The latter case cannot occur since s is not final. It follows by induction that for every reachable state (q, S) we have $q \notin S$. In total, taking into account that $s \in S$ whenever q is final, we get at most $(n-1)2^{n-1} + 2^{n-2}$ reachable states. This gives the upper bound.

Fig. 1. The witness DFA for square meeting the upper bound $n2^{n-1} - 2^{n-2}$.

To prove tightness, consider the permutation DFA $A = (\mathbf{n}, \{a, b\}, \cdot, 0, \{n-1\})$ with transitions $a: (0, 1, \dots, n-1)$ and $b: (0, 1)$ shown in Fig. 1. Construct an NFA for $L(A)^2$ as described above. Let us show by induction on $|S|$ that each state (i, S) with $i = 0, 1, \dots, n-2$ and $i \notin S$ and each state $(n-1, S)$ with $0 \in S$ and $n-1 \notin S$ is reachable. The base case, $|S| = 0$, holds true since for every i with $1 \leq i \leq n-2$, the state $(i, \emptyset)$ is reached from the initial state $(0, \emptyset)$ by a^i . Now assume that our claim holds for each S' with $|S'| = k$. Let $|S| = k+1$. Consider four cases:

- (1) $i = n-1$. Then $0 \in S$ and $n-1 \notin S$. Take $S' = (S \setminus \{0\}) \cdot a^{n-1}$. Then $n-2 \notin S'$, so $(n-2, S')$ is reachable by induction and it is sent to $(n-1, S)$ by a .
- (2) $i = 0$ and $1 \in S$. Take $S' = S \cdot a^{n-1}$. Since $0 \notin S$, we have $n-1 \notin S'$, so $(n-1, S')$ is reachable as shown in case (1) and it is sent to $(0, S)$ by a .
- (3) $i = 0$ and $\min S > 1$. Take $S' = S \cdot a^{n-\min S+1}$. Then $0 \notin S'$ and $1 \in S'$, so $(0, S')$ is reachable as shown in case (2) and it is sent to $(0, S)$ by $(ab)^{\min S-1}$.
- (4) $1 \leq i \leq n-2$. Take $S' = S \cdot a^{n-i}$. Since $i \notin S$, we have $0 \notin S'$, so the set $(0, S')$ is reachable as shown in case (1) or (2) and it is sent to (i, S) by a^i .

Now we prove distinguishability. Let (i, S) and (j, T) be two distinct states of the subset automaton with $i \notin S$ and $j \notin T$. First let $S \neq T$, and without loss of generality, assume that $s \in S \setminus T$. Then the string a^{n-1-s} is accepted from S and rejected from T . Now let $S = T$, and without loss of generality, let $i < j$. We have $i, j \notin S$. Consider the string $a^{n-j}(ab)^{j-i-1}$. Let $S' = S \cdot a^{n-j}$. Since $j \notin S$, we have $0 \notin S'$. Next,

$$\begin{aligned} (i, S) &\xrightarrow{a^{n-j}} (n-j+i, S') \xrightarrow{(ab)^{j-i-1}} (n-1, S' \cdot (ab)^{j-i-1} \cup \{0\}), \\ (j, S) &\xrightarrow{a^{n-j}} (0, S' \cup \{1\}) \xrightarrow{(ab)^{j-i-1}} (j-i-1, (S' \cup \{1\}) \cdot (ab)^{j-i-1}). \end{aligned}$$

Since $0 \notin S'$, we have $0 \notin (S' \cup \{1\}) \cdot (ab)^{j-i-1}$; notice that the only state which is set to 0 by a string in $(ab)^*$ is 0. This means that the resulting states differ in the second component, therefore (i, S) and (j, S) are distinguishable. This completes our proof for square.

(c) The upper bound for Kleene closure is the same as for regular languages. For tightness, consider the n -state DFA $A = (\mathbf{n}, \{a, b\}, \cdot, 0, \{n-1\})$ with $a: (0, 1, \dots, n-1)$ and $b: (1, 2, \dots, n-2)$. Notice that this DFA, shown in Fig. 2, differs from the DFA in [17, Figure 4] just by having the unique final state $n-1$ and by having the transition $(n-2, b, 1)$ instead of $(n-2, b, 0)$. The language $L(A)^*$ is accepted by MNFA $(\{q_0\} \cup \mathbf{n}, \{a, b\}, \circ, \{q_0, 0\}, \{q_0, n-1\})$ where $\circ$ has the same transitions as $\cdot$ and moreover we have $0 \in (n-2) \circ a$ and $0 \in (n-1) \circ b$. Let $\mathcal{R} = \{q_0, 0\} \cup \{S \subseteq \mathbf{n} \mid S \neq \emptyset \text{ and if } n-1 \in S, \text{ then } 0 \in S\}$. We can show by induction on the size of sets that each set in $\mathcal{R}$ is reachable in the corresponding subset automaton. We also can prove that the sets in $\mathcal{R}$ are pairwise distinguishable.

Fig. 2. The witness DFA for Kleene closure meeting the upper bound $(3/4)2^n$.

To get an NFA for L^+ , we only remove the state q_0 from the MNFA for L^* described above. The corresponding subset automaton has at most $(3/4)2^n - 1$ reachable states, and this upper bound is met by our witness for Kleene closure.

(d) If L is accepted by an n -state permutation DFA A , then L^R is accepted by a permutation MDFA A^R obtained from A by reversing all transitions and by swapping the roles of initial and final states. We have $\text{sc}(L^R) \leq \binom{n}{\lceil n/2 \rceil}$ and this upper bound is met by the language accepted by the permutation DFA $(\mathbf{n}, \{a, b\}, \cdot, 0, \frac{n}{2})$ with $a: (n-1, n-2, \dots, 0)$ and $b: (0, 1)$ by Lemma 7. $\square$

Notice that for concatenation, the upper bound is almost the same as for regular languages, while for square, the bound is exactly one half of the upper bound in the general case. In the next theorem, we consider quotients.

Theorem 12 (Left and Right Quotient). *Let $m, n \geq 2$. Let K and L be languages accepted by permutation DFAs with m and n states, respectively. Then $\text{sc}(KL^{-1}) \leq m$, and this bound is met by unary languages if $m \leq n$. Next, $\text{sc}(L^{-1}K) \leq \binom{m}{\lceil m/2 \rceil}$, and this bound is met by binary languages if $m \leq n$. In the unary case, the tight upper bound for both quotients is $\min\{m, n\}$.*

Proof. We have $\text{sc}(KL^{-1}) \leq m$ by the construction of a DFA for the right quotient. To get tightness in the case of $m \leq n$, let $K = L = (a^m)^*$. Then $KL^{-1} = K$, so $\text{sc}(KL^{-1}) = m$.

The language $L^{-1}K$ is accepted by an m -state permutation MDFA. We have $\text{sc}(L^{-1}K) \leq \binom{m}{\lceil m/2 \rceil}$ by Lemma 7. For tightness, let $m \leq n$. Let K be the language accepted by DFA $A = (\mathbf{m}, \{a, b\}, \cdot, 0, \{0\})$ with $a: (0, 1, \dots, m-1)$ and $b: (0, 1)$. Let L be the language accepted by DFA $B = (\mathbf{n}, \{a, b\}, \cdot, 0, \frac{m}{2})$ with $a: (0, 1, \dots, m-1)$ and $b: (0, 1)$; notice that states $m, m+1, \dots, n-1$ are unreachable and the transitions on a and b in states $0, 1, \dots, m-1$ are the same as in DFA A . The language $L^{-1}K$ is accepted by an MDFA M obtained from A by making all states in $\frac{m}{2}$ initial; notice that after reading every input string, both A and B are in the same state. Then $\text{sc}(L^{-1}K) = \binom{m}{\lceil m/2 \rceil}$ by Lemma 7.

In the unary case, we have $L^{-1}K = KL^{-1}$. Therefore $\text{sc}(L^{-1}K) \leq m$. Let us show that $\text{sc}(KL^{-1}) \leq n$. The languages K and L are periodic; let their periods be k and ℓ , respectively. By construction, the quotient KL^{-1} has period k . If $a^{i+\ell} \in KL^{-1}$, then $a^i \in KL^{-1}$ since L has period ℓ . Let $a^i \in KL^{-1}$. Then $a^i w \in K$ for some $w \in L$. It follows that $a^i a^\ell a^{(k-1)\ell} w \in K$ since K has period k . Next, $a^{(k-1)\ell} w \in L$ since L has period ℓ . It follows that $a^{i+\ell} \in KL^{-1}$. Thus KL^{-1} has period ℓ , so $\text{sc}(KL^{-1}) \leq \ell \leq n$. This gives the upper bound $\min\{m, n\}$ which is met by $K = L = (a^{\min\{m, n\}})^*$. $\square$

Finally, we consider the cut operation on permutation languages. We briefly recall the construction of a DFA for the cut operation, cf. [2, p. 74], [7, p. 91] or [10, p. 193]. For two languages accepted by DFAs A and B with m and n states, respectively, we can construct the *cut automaton* $A!B$ as follows. The cut automaton has states in a grid with a row for every state of A and a column for every state of B , and one additional column which we denote by $\perp$. The column $\perp$ corresponds to the situation that we have not read a string in $L(A)$ yet. When we reach a final state in A for the first time, we leave the column $\perp$ and enter the product part of $A!B$ in the corresponding state. Unless we reach a final state of A again, the transitions in $A!B$ are the same as in the product automaton $A \times B$. When we reach a final state in A again, we reset to a state in the column corresponding to the initial state of B . The final states of $A!B$ are all states in columns corresponding to the final states of B .

Formally, let $A = (Q_A, \Sigma, \cdot_A, s_A, F_A)$ and $B = (Q_B, \Sigma, \cdot_B, s_B, F_B)$ be two DFAs. Let $\perp \notin Q_B$. Define the cut automaton $A!B = (Q, \Sigma, \cdot, s, Q_A \times F_B)$ where $Q = (Q_A \times \{\perp\}) \cup (Q_A \times Q_B)$, $s = (s_A, \perp)$ if $\varepsilon \notin L(A)$ and $s = (s_A, s_B)$ otherwise, and for each state (p, q) in Q and each input symbol a in Σ , we have

$$(p, \perp) \cdot a = \begin{cases} (p \cdot_A a, \perp), & \text{if } p \cdot_A a \notin F_A; \\ (p \cdot_A a, s_B), & \text{otherwise;} \end{cases}$$

and

$$(p, q) \cdot a = \begin{cases} (p \cdot_A a, q \cdot_B a), & \text{if } p \cdot_A a \notin F_A; \\ (p \cdot_A a, s_B), & \text{otherwise.} \end{cases}$$

The state complexity of cut in the general case is already solved in [7] where binary permutation languages were used as witnesses. The unary case is more interesting, as shown in the next theorem.

Theorem 13 (Cut Operation). *Let $m \geq 1$ and $n \geq 3$. Let K and L be languages accepted by permutation DFAs with m and n states, respectively. Then $\text{sc}(K!L) \leq (m-1)n + m$, and this upper bound is met by binary languages. If K and L are unary, then $\text{sc}(K!L) \leq 2m-1$. This bound is tight if $m \bmod \ell \neq 0$ for some ℓ with $2 \leq \ell \leq n$, otherwise the tight upper bound is $2m-2$.*

Proof. If $K = \emptyset$ or $L = \emptyset$, then $K!L = \emptyset$, so $\text{sc}(K!L) = 1$. Let K and L be non-empty. In the general case, the witness languages from [7, Proof of Theorem 3.1] are accepted by permutation DFAs and they meet the upper bound $(m-1)n + m$.

In the unary case, since each non-empty permutation language is infinite, the upper bound is $2m-1$ by [7, Proof of Theorem 3.2].

First assume that there exists ℓ with $2 \leq \ell \leq n$ such that $m \bmod \ell \neq 0$. Let $K = a^{m-1}(a^m)^*$ and $L = a^{(m-1) \bmod \ell}(a^\ell)^*$ be unary languages recognized by permutation DFAs A and B of m and ℓ states, respectively. The cut automaton $A!B$ has a tail of length $m-1$ and a loop of length m starting in the state $(m-1, 0)$ which has in-transitions from the non-final state $(m-2, \perp)$ and the final state $(m-2, (m-1) \bmod \ell)$. Let us show that the loop is minimal. If $m \leq \ell$, then the loop has just one final state, so it is minimal. Let $m > \ell$. If $m \bmod \ell = 1$, then $(m-1) \bmod \ell = 0$, hence the two states $(m-2, 0)$ and $(m-1, 0)$ are final, while every other contiguous segment of final states is of length 1, so the loop is minimal again. If $2 \leq m \bmod \ell \leq \ell-1$, then $1 \leq (m-1) \bmod \ell \leq \ell-2$. Then every contiguous segment of non-final states is of length $\ell-1$, except for the segment containing $(m-1, 0)$ which is of length $(m-1) \bmod \ell < \ell-1$, so the loop is minimal.

Let $m \bmod \ell = 0$ for each ℓ with $2 \leq \ell \leq n$. Then $m \bmod 2 = m \bmod 3 = 0$, so $m \geq 6$. Consider the permutation DFAs $A = (\mathbf{m}, \{a\}, \cdot_A, 0, \{m-2, m-1\})$ with $a: (0, 1, \dots, m-1)$ and $B = (\mathbf{n}, \{a\}, \cdot_B, 0, \{0\})$ with $a: (0, 1)$. The DFA $A!B$ has a tail of length $m-2$ and a loop of size m starting in the state $(m-2, 0)$ which has in-transitions from the non-final state $(m-3, \perp)$ and the final state $(m-3, 0)$. In the loop, each contiguous segment of final states is of length 1, except for that consisting of $(m-3, 0)$, $(m-2, 0)$, and $(m-1, 0)$. Therefore the loop is minimal, and so we have $\text{sc}(L(A)!L(B)) = 2m-2$.

Let us show that this bound is tight if $m \bmod \ell = 0$ for each ℓ with $2 \leq \ell \leq n$. Since $m \bmod n = 0$ and $m \bmod (n-1) = 0$, we must have $m > n$. Let K and L be accepted by unary permutation DFAs A and B ; these DFAs are cyclic, that is, they do not have any tail. To meet the upper bound $2m-1$, the cut automaton $A!B$ must have $m-1$ states of the form $(i, \perp)$. This is possible only if $A = (\mathbf{m}, \{a\}, \cdot_A, 0, \{m-1\})$ with $a: (0, 1, \dots, m-1)$. Let $B = (\mathbf{n}, \{a\}, \cdot_B, 0, F)$ with $a: (0, 1, \dots, \ell-1)$ for some ℓ with $1 \leq \ell \leq n$. If $\ell = 1$, then $L = \emptyset$ or $L = a^*$, and $K!L$ is either empty or equal to $a^{m-1}a^*$, so it is of state complexity 1 or m . Let $\ell \geq 2$. Then $m \bmod \ell = 0$. In the loop of the cut automaton $A!B$, we have

$$(m-1, 0) \xrightarrow{a^\ell} (\ell-1, 0) \xrightarrow{a^\ell} (2\ell-1, 0) \xrightarrow{a^\ell} \dots \xrightarrow{a^\ell} (m-\ell-1, 0) \xrightarrow{a^\ell} (m-1, 0).$$

Since the loop in B is of length ℓ , we have $(i, j) \xrightarrow{a^\ell} (i+\ell, j)$ for each state (i, j) in the loop; here $i+\ell$ is modulo m . The states (i, j) and $(i+\ell, j)$ have the same finality since all states with j in the second component have the same

Table 1. Closure properties and state complexity of operations on the class of permutation languages. ($\star$): there exists ℓ with $2 \leq \ell \leq n$ such that $m \bmod \ell \neq 0$.

Operation	Closed?	State complexity	$ \Sigma $	State complexity, $ \Sigma = 1$
L^c	Yes ([22])	n	1	n
$\cap, \cup, \setminus, \oplus$	Yes ([22])	mn	2	mn ; $\gcd(m, n) = 1$
KL	No ([22])	$m2^n - 2^{n-1} - m + 1$	2	mn ; $\gcd(m, n) = 1$
L^2	No	$n2^{n-1} - 2^{n-2}$	2	$2n - 1$
L^k	No	?		$k(n - 1) + 1$
L^+	No	$(3/4)2^n - 1$	2	$(n - 1)^2 + 1$
L^*	No ([22])	$(3/4)2^n$	2	$(n - 1)^2 + 1$
L^R	Yes ([22])	$\binom{n}{\lfloor n/2 \rfloor}$	2	n
$L^{-1}K$	Yes ([22])	$\binom{m}{\lfloor m/2 \rfloor}$; $m \leq n$	2	$\min\{m, n\}$
KL^{-1}	Yes ([22])	m ; $m \leq n$	1	$\min\{m, n\}$
$K!L$	No	$(m - 1)n + m$	2	$2m - 1$ if ($\star$); $2m - 2$ otherwise
$K \sqcup L$	No	?		mn ; $\gcd(m, n) = 1$
$\text{shift}(L)$	No	?		n
$\text{per}(L)$	No	?		n

finality as j in B . It follows that the loop can be replaced by a loop of length ℓ , so $\text{sc}(K!L) \leq m - 1 + \ell < 2m - 1$. $\square$

5 Conclusions

We examined the class of permutation languages, that is, the languages that are recognized by deterministic finite automata in which every input symbol induces a permutation on the state set. We used automata constructions and properties of permutation automata to show that the class of permutation languages is closed under Boolean operations, reversal, and left and right quotient, and it is not closed under concatenation, power, positive closure, Kleene closure, cut, shuffle, cyclic shift, and permutation.

We also studied the state complexity of operations on permutation languages. Our results are summarized in Table 1. The table also displays the size of alphabet used to describe witnesses. All our witnesses are described over a unary or binary alphabet, and the binary alphabet is always optimal except for the Boolean operations in case of $\gcd(m, n) = 1$.

We did not consider the state complexity of shuffle and cyclic shift and leave them for the future work. Although the class of regular languages is not closed under the permutation operation, we conjecture that the permutation of a permutation language is always regular. If this is the case, then the state complexity of this operation is of great interest to us as well.

References

1. Bassino, F., Giambruno, L., Nicaud, C.: Complexity of operations on cofinite languages. In: López-Ortiz, A. (ed.) LATIN 2010. LNCS, vol. 6034, pp. 222–233. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12200-2_21
2. Berglund, M., Björklund, H., Drewes, F., van der Merwe, B., Watson, B.: Cuts in regular expressions. In: Béal, M.-P., Carton, O. (eds.) DLT 2013. LNCS, vol. 7907, pp. 70–81. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38771-5_8
3. Brzozowski, J.A., Jirásková, G., Li, B.: Quotient complexity of ideal languages. Theor. Comput. Sci. **470**, 36–52 (2013). <https://doi.org/10.1016/j.tcs.2012.10.055>
4. Brzozowski, J.A., Jirásková, G., Zou, C.: Quotient complexity of closed languages. Theory Comput. Syst. **54**(2), 277–292 (2014). <https://doi.org/10.1007/s00224-013-9515-7>
5. Brzozowski, J.A., Liu, B.: Quotient complexity of star-free languages. Int. J. Found. Comput. Sci. **23**(6), 1261–1276 (2012). <https://doi.org/10.1142/S0129054112400515>
6. Câmpeanu, C., Culik, K., Salomaa, K., Yu, S.: State complexity of basic operations on finite languages. In: Boldt, O., Jürgensen, H. (eds.) WIA 1999. LNCS, vol. 2214, pp. 60–70. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45526-4_6
7. Drewes, F., Holzer, M., Jakobi, S., van der Merwe, B.: Tight bounds for cut-operations on deterministic finite automata. Fund. Inform. **155**(1–2), 89–110 (2017). <https://doi.org/10.3233/FI-2017-1577>
8. Han, Y., Salomaa, K.: State complexity of basic operations on suffix-free regular languages. Theor. Comput. Sci. **410**(27–29), 2537–2548 (2009). <https://doi.org/10.1016/j.tcs.2008.12.054>
9. Han, Y., Salomaa, K., Wood, D.: State complexity of prefix-free regular languages. In: Leung, H., Pighizzini, G. (eds.) DCFS 2006, pp. 165–176. New Mexico State University, Las Cruces (2006)
10. Holzer, M., Hospodár, M.: The range of state complexities of languages resulting from the cut operation. In: Martín-Vide, C., Okhotin, A., Shapira, D. (eds.) LATA 2019. LNCS, vol. 11417, pp. 190–202. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-13435-8_14
11. Hopcroft, J.E., Ullman, J.D.: Introduction to Automata Theory, Languages, and Computation. Addison-Wesley, Boston (1979)
12. Jirásková, G., Krajňáková, I.: NFA-to-DFA trade-off for regular operations. In: Hospodár, M., Jirásková, G., Konstantinidis, S. (eds.) DCFS 2019. LNCS, vol. 11612, pp. 184–196. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-23247-4_14
13. Jirásková, G., Masopust, T.: Complexity in union-free regular languages. Int. J. Found. Comput. Sci. **22**(7), 1639–1653 (2011). <https://doi.org/10.1142/S0129054111008933>
14. Jirásková, G., Okhotin, A.: State complexity of cyclic shift. RAIRO Theor. Inf. Appl. **42**(2), 335–360 (2008). <https://doi.org/10.1051/ita:2007038>
15. Maslov, A.N.: Estimates of the number of states of finite automata. Soviet Math. Doklady **11**, 1373–1375 (1970)
16. Nicaud, C.: Average state complexity of operations on unary automata. In: Kutylowski, M., Pacholski, L., Wierzbicki, T. (eds.) MFCS 1999. LNCS, vol. 1672, pp. 231–240. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48340-3_21

17. Palmovský, M.: Kleene closure and state complexity. *RAIRO Theor. Inform. Appl.* **50**(3), 251–261 (2016). <https://doi.org/10.1051/ita/2016024>
18. Pighizzini, G., Shallit, J.: Unary language operations, state complexity and Jacobsthal's function. *Int. J. Found. Comput. Sci.* **13**(1), 145–159 (2002). <https://doi.org/10.1142/S012905410200100X>
19. Rabin, M.O., Scott, D.: Finite automata and their decision problems. *IBM J. Res. Dev.* **3**, 114–125 (1959). <https://doi.org/10.1147/rd.32.0114>
20. Rampersad, N.: The state complexity of L^2 and L^k . *Inf. Process. Lett.* **98**(6), 231–234 (2006). <https://doi.org/10.1016/j.ipl.2005.06.011>
21. Sipser, M.: *Introduction to the Theory of Computation*. Cengage Learning, Boston (2012)
22. Thierrin, G.: Permutation automata. *Math. Syst. Theory* **2**(1), 83–90 (1968). <https://doi.org/10.1007/BF01691347>
23. Yu, S., Zhuang, Q., Salomaa, K.: The state complexities of some basic operations on regular languages. *Theor. Comput. Sci.* **125**(2), 315–328 (1994). [https://doi.org/10.1016/0304-3975\(92\)00011-F](https://doi.org/10.1016/0304-3975(92)00011-F)