Theoretical Computer Science 450 (2012) 68–80

Contents lists available at SciVerse ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Nondeterministic state complexity of star-free languages

Markus Holzer *, Martin Kutrib, Katja Meckel

Institut für Informatik, Universität Giessen, Arndtstr. 2, 35392 Giessen, Germany

ARTICLE INFO

Keywords:
Star-free languages
Nondeterministic finite automata
Operational state complexity
Descriptional complexity

ABSTRACT

We investigate the nondeterministic state complexity of several operations on finite automata accepting star-free and unary star-free languages. It turns out that in most cases exactly the same tight bounds as for general regular languages are reached. This nicely complements the results recently obtained by Brzozowski and Liu (2011) [8] for the operation problem of star-free and unary star-free languages accepted by deterministic finite automata.

1. Introduction

The operation problem on a language family is the question of cost (in terms of states) of operations on languages from this family with respect to their representations. More than a decade ago, the operation problem for regular languages represented by deterministic finite automata (DFAs) as studied in [38,39] renewed the interest in descriptional complexity issues of finite automata in general. Although the research area of finite automata dates back to the beginning of the 1950s, their (descriptional) complexity with respect to the operation problem had attracted surprisingly less attention in the early days. This lack of interest may be one reason for the prevailing view on regular languages during the late seventies [38]:

> Since the late seventies, many believed that everything of interest about regular languages is known except for a few very hard problems, [· · ·]. It appeared that not much further work could be done on regular languages.

Nowadays descriptional complexity of finite automata and related structures is a vivid area of research, for which the (recent) surveys on this area give evidence [17–19,38].

It is well known that nondeterministic and deterministic finite automata are computationally equivalent. More precisely, given some n-state NFA one can always construct a language equivalent DFA with at most 2^n states [33] and, therefore, NFAs can offer exponential savings in space compared with DFAs. In fact, later it was shown independently in [26,30,31] that this exponential upper bound is the best possible, that is, for every n there is an n-state NFA which cannot be simulated by any DFA with strictly less than 2^n states. Recently, in [2], it was shown that this exponential tight bound for the determinization of NFAs also holds when restricting the NFAs to accept only subregular language families such as star languages [3], (two-sided) comet languages [5], ordered languages [36], star-free languages [27], power-separating languages [37], prefix-closed languages, etc. On the other hand, there are also subregular language families known, where this exponential bound is not met. Prominent examples are the family of unary regular languages, where an asymptotic bound of $e^{\Theta(\sqrt{n \cdot \ln n})}$ states for determinization has been shown in [9,10], and the family of finite languages with a tight bound of $O(k^{\frac{n}{\log_2(k)+1}})$, where k is the size of the alphabet [34]. The significantly different behavior with respect to the relative succinctness of NFAs compared to DFAs is also reflected in the operation problem for these devices. The operation problem for NFAs was first investigated in [15] and later continued by a study on the operation problem for NFAs accepting unary languages [16]. Several more

* Corresponding author.
E-mail addresses: holzer@informatik.uni-giessen.de (M. Holzer), kutrib@informatik.uni-giessen.de (M. Kutrib), meckel@informatik.uni-giessen.de (K. Meckel).

0304-3975/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2012.04.028

Table 1
Deterministic and nondeterministic state complexities for the operation problem on star-free languages summarized. The results for DFAs are from [8].

Operation	Star-free language accepted by …	
	DFA	NFA
$\cup$	mn	$m+n+1$
$\cap$	mn	mn
$\sim$		2^n
$\cdot$	$(m-1)2^n+2^{n-1}$	$m+n$
$*$	$2^{n-1}+2^{n-2}$	$n+1$
R	$2^n-1 \leq \cdot \leq 2^n$	$n+1$

Table 2
Deterministic and nondeterministic state complexities for the operation problem on *unary* star-free languages summarized. The results for DFAs are again from [8].

Operation	*Unary* star-free language accepted by …	
	DFA	NFA
$\cup$	$\max\{m,n\}$	$m+n \leq \cdot \leq m+n+1$
$\cap$	$\max\{m,n\}$	$\Theta(mn)$
$\sim$		$\Theta(n^2)$
$\cdot$	$m+n-1$	$m+n-1 \leq \cdot \leq m+n$
$*$	$n^2-7n+13$	$n+1$
R	n	n

results on unary languages can be found, for example, in [28,29,32]. It turned out that in most cases when an operation is cheap for DFAs it is costly for NFAs and *vice versa*. We give two examples: (i) the complementation operation applied to a language accepted by an n-state DFA results in a DFA of exactly the same number of states, while complementing NFAs gives an exponential tight bound of 2^n states [20], and conversely (ii) for two languages accepted by m- and n-state DFAs we have a tight bound of $m \cdot 2^n - t \cdot 2^{n-1}$ states for concatenation [38,39], where t is the number of accepting states of the "left" automaton, and $m+n+1$ states when considering NFAs [15]. All these results are for general regular languages. So, the question arises what happens to these bounds if the operation problem is restricted to subregular language families.

In fact, for some subregular language families this question was recently studied in the literature [6–8,12,13,21–23] mostly for DFAs. To this end, the notion of quotient complexity [4], which has been studied in a series of papers [6–8], is a useful tool for exploring the deterministic state complexity. An example for a subregular language family whose DFA operation problems meet the general bounds for most operations is the family of star-free languages [8], while prefix-, infix-, and suffix-closed languages [7], bifix-, factor-, and subword-free languages [6] show a diverse behavior mostly not reaching the general bounds. For a few language families, in particular prefix- and suffix-free regular languages, also the operation problem for NFAs was considered [12,13,21,23], but for the exhaustively studied family of star-free languages it is still open.

The family of star-free (or regular non-counting) languages is an important subfamily of the regular languages, which can be obtained from the elementary languages $\{a\}$, for $a \in \Sigma$, and the empty set $\emptyset$ by applying the Boolean operations union, complementation, and concatenation finitely often. They obey nice characterizations in terms of aperiodic monoids and permutation-free DFAs [27]. Here we investigate their operation problem for NFAs and NFAs accepting unary languages with respect to the basic operations union, intersection, complementation, concatenation, Kleene star, and reversal. It turns out that in most cases exactly the same tight bounds as in the general case are reached. This nicely complements the results recently obtained for the operation problem of star-free and unary star-free languages accepted by DFAs [8]. We summarize our results in Tables 1 and 2, where we also list the results for DFAs accepting star-free and unary star-free languages [8], for comparison reasons.

2. Preliminaries

For $n \geq 0$ we write $\Sigma^{\leq n}$ for the set of all words whose lengths are at most n and Σ^n for the set of all words of length n. The empty word is denoted by λ. For the length of w we write $|w|$. Set inclusion is denoted by $\subseteq$ and strict set inclusion by $\subset$. We write 2^S for the powerset and $|S|$ for the cardinality of a set S.

A *nondeterministic finite automaton* (NFA) is a 5-tuple $\mathcal{A} = (S, \Sigma, \delta, s_0, F)$, where S is the finite set of *states*, Σ is the finite set of *input symbols*, $s_0 \in S$ is the *initial state*, $F \subseteq S$ is the set of *accepting states*, and $\delta : S \times \Sigma \to 2^S$ is the *transition function*. As usual the transition function is extended to $\delta : S \times \Sigma^* \to 2^S$ reflecting sequences of inputs: $\delta(s, \lambda) = \{s\}$ and

$\delta(s, aw) = \bigcup_{s' \in \delta(s,a)} \delta(s', w)$, for $s \in S$, $a \in \Sigma$, and $w \in \Sigma^*$. A word $w \in \Sigma^*$ is *accepted* by $\mathcal{A}$ if $\delta(s_0, w) \cap F \neq \emptyset$. The *language accepted* by $\mathcal{A}$ is $L(\mathcal{A}) = \{ w \in \Sigma^* \mid w \text{ is accepted by } \mathcal{A} \}$.

A finite automaton is *deterministic* (DFA) if and only if $|\delta(s, a)| = 1$, for all $s \in S$ and $a \in \Sigma$. In this case we simply write $\delta(s, a) = s'$ for $\delta(s, a) = \{s'\}$ assuming that the transition function is a mapping $\delta : S \times \Sigma \to S$. So, any DFA is complete, that is, the transition function is total, whereas for NFAs it is possible that δ maps to the empty set. A finite automaton is called *unary* if its set of input symbols is a singleton. In this case we use $\Sigma = \{a\}$ throughout the paper. A state s is *reachable* in $\mathcal{A}$ if there is an input word w with $s \in \delta(s_0, w)$. Without loss of generality we assume that any state of a nondeterministic finite automaton is reachable. A finite automaton is said to be *minimal* if there is no finite automaton of the same type with fewer states, accepting the same language. Note that a sink state is counted for DFAs, since they are always complete, whereas it is not counted for NFAs, since their transition function may map to the empty set.

Next, we briefly recall the so-called (extended) *fooling set* technique (see, for example, [1,11,17]) that is widely used for proving lower bounds on the number of states necessary for an NFA to accept a given language.

Theorem 1 (*Extended Fooling Set Technique*)**.** *Let $L \subseteq \Sigma^*$ be a regular language and suppose there exists a set of pairs $P = \{ (x_i, y_i) \mid 1 \leq i \leq n \}$ such that (1) $x_i y_i \in L$, for $1 \leq i \leq n$, and (2) $i \neq j$ implies $x_i y_j \notin L$ or $x_j y_i \notin L$, for $1 \leq i, j \leq n$. Then any nondeterministic finite automaton accepting L has at least n states. Here P is called an (extended)* fooling set *for L.*

Now we turn to the subregular language family of interest. A language $L \subseteq \Sigma^*$ is *star-free* (or regular *non-counting*) if and only if it can be obtained from the elementary languages $\{a\}$, for $a \in \Sigma$, and the empty set $\emptyset$ by applying the Boolean operations union, complementation, and concatenation finitely often. These languages are exhaustively studied in, for example, [27] and [35]. Since regular languages are closed under Boolean operations and concatenation, every star-free language is regular. On the other hand, not every regular language is star free. Here we sometimes utilize an alternative characterization of star-free languages by so called permutation-free automata [27]: a regular language $L \subseteq \Sigma^*$ is star-free if and only if the minimal DFA accepting L is *permutation-free*, that is, there is *no* word $w \in \Sigma^*$ that induces a non-trivial permutation on any subset of the set of states. Here a trivial permutation is simply the identity permutation. Note that word uw induces a non-trivial permutation $\{s_1, s_2, \ldots, s_n\} \subseteq S$ in a DFA with state set S and transition function δ if and only if wu induces a non-trivial permutation $\{\delta(s_1, u), \delta(s_2, u), \ldots, \delta(s_n, u)\}$ in the same automaton. With this characterization at hand, it is easy to see that any *unary* star-free language is either finite or co-finite, since a deterministic finite automaton accepting a unary language cannot have a non-trivial cycle, because otherwise it would not be permutation free. Moreover, any unary finite or unary co-finite language is in fact star-free. Thus, we have: a language $L \subseteq \{a\}^*$ is star-free if and only if L is finite or co-finite.

Finally, we recall another useful fact on unary (star-free) languages, which is related to number theory: given two non-negative integers $x, y \geq 0$ which are relatively prime, that is, the greatest common divisor $\gcd(x, y)$ equals 1. Then the biggest integer that cannot be written as a linear combination of these two integers is $(x-1)(y-1) - 1 = xy - x - y$. A direct application is that for two relatively prime numbers x and y the unary regular language $\{a^x, a^y\}^*$ is in fact star free, because it is co-finite. In the section on the operation problem on NFAs accepting unary star-free languages we will often meet this language or a variant thereof.

3. Operations on general star-free languages

3.1. Boolean operations

We start our investigations with Boolean operations. For deterministic finite automata it was recently shown that in the worst case the Boolean operations union, intersection, and complementation have state complexity $m \cdot n$, $m \cdot n$, and n not only for general regular languages, but also for star-free languages. However, the state complexity of NFA operations for general regular languages is essentially different [15]. Namely, union, intersection, and complementation have nondeterministic state complexity $m+n+1$, $m \cdot n$, and 2^n. It is worth mentioning that the exponential bound of 2^n states for complementation was shown to be tight in [20]. Here we prove that this is also the case for star-free languages. Note, that all the upper bounds are from [15]. Thus, we only have to give star-free witness languages meeting these bounds. At first we consider the union operation.

Theorem 2. *For any integers $m, n \geq 2$ let $\mathcal{A}$ be an m-state and $\mathcal{B}$ be an n-state NFA that accept star-free languages. Then $m + n + 1$ states are sufficient for an NFA to accept the language $L(\mathcal{A}) \cup L(\mathcal{B})$. The bound is tight for binary alphabets.*

Proof. As already mentioned, the upper bound of $m + n + 1$ states is that for arbitrary regular languages shown in [15]. For the lower bound we argue as follows: consider the NFA $\mathcal{A} = (S, \{a, b\}, \delta, s_0, F)$ with state set $S = \{0, 1, \ldots, m-1\}$, for $m \geq 2$. State 0 is the initial state s_0, and state $m-1$ is the only final state. The transition function is given by (cf. Fig. 1):

- $\delta(i, a) = \{i + 1\}$, for $0 \leq i < m - 1$, and
- $\delta(m - 1, b) = \{0\}$.

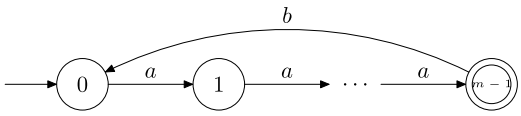

Fig. 1. The m-state NFA $\mathcal{A}$, $m \geq 2$, accepting a star-free language. The automaton $\mathcal{B}$ has n states and letters a and b are interchanged.

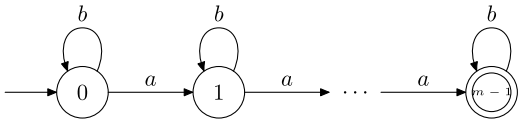

Fig. 2. The m-state NFA $\mathcal{A}$, $m \geq 2$, accepting the star-free language $b^*(ab^*)^{m-1}$. The automaton $\mathcal{B}$ has n states and letters a and b are interchanged.

The language accepted by $\mathcal{A}$ is $a^{m-1}(ba^{m-1})^*$. Observe, that the automaton $\mathcal{A}$ is actually a *partial* DFA, where the sink state is missing. The corresponding *complete* DFA is minimal and does not obey a non-trivial permutation on the state set. Therefore, the language $L(\mathcal{A})$ is a star-free language. Similarly, we define the automaton $\mathcal{B}$ by taking $\mathcal{A}$ with n states and interchanging the letters a and b. Hence, we obtain the star-free language $L(\mathcal{B}) = b^{n-1}(ab^{n-1})^*$.

It remains to be shown that $m+n+1$ states are needed by any NFA to accept $L(\mathcal{A}) \cup L(\mathcal{B})$. To this end, we construct the following set of pairs

$$P = \{ (a^{m-1}ba^i, a^{m-1-i}ba^{m-1}) \mid 0 \leq i \leq m-1 \} \cup \{ (b^{n-1}ab^i, b^{n-1-i}ab^{n-1}) \mid 0 \leq i \leq n-1 \}.$$

First consider the pairs of the form $(a^{m-1}ba^i, a^{m-1-i}ba^{m-1})$ in P. Clearly, the word $a^{m-1}ba^i \cdot a^{m-1-j}ba^{m-1}$, for $0 \leq i, j \leq m-1$, is in the union of $L(\mathcal{A})$ and $L(\mathcal{B})$ if and only if $i = j$. Thus, the pair $(a^{m-1}ba^i, a^{m-1-i}ba^{m-1})$ induces a word that belongs to the union under consideration, but any word induced by crossing different pairs of the above form results in two words not in $L(\mathcal{A}) \cup L(\mathcal{B})$. Symmetrically we can argue for the pairs of the form $(b^{n-1}ab^i, b^{n-1-i}ab^{n-1})$. Finally, we have to compare pairs $(a^{m-1}ba^i, a^{m-1-i}ba^{m-1})$, for $0 \leq i \leq m-1$, with pairs $(b^{n-1}ab^j, b^{n-1-j}ab^{n-1})$, for $0 \leq j \leq n-1$. In this case we obtain the words $a^{m-1}ba^i \cdot b^{n-1-j}ab^{n-1}$ and $b^{n-1}ab^j \cdot a^{m-1-i}ba^{m-1}$, where the start and end blocks of a's and b's of the words do not correspond. Thus, both words do not belong to the union of $L(\mathcal{A})$ and $L(\mathcal{B})$. Hence, P is a fooling set for the language $L(\mathcal{A}) \cup L(\mathcal{B})$ of size $m+n$. To the upper bound one state is missing. We argue that the initial state of the automaton that accepts the language $L(\mathcal{A}) \cup L(\mathcal{B})$ is not one of the states induced by P.

Assume to the contrary that the initial state of the automaton accepting the language $L(\mathcal{A}) \cup L(\mathcal{B})$ is one of the states induced by P. If the initial state is equal to the state referenced by the pair $(a^{m-1}ba^i, a^{m-1-i}ba^{m-1})$, $0 \leq i \leq m-1$, then the word $a^{m-1}ba^i \cdot b^{n-1}$ is also accepted, because b^{n-1} is in $L(\mathcal{A}) \cup L(\mathcal{B})$ and must be accepted from the initial state. This contradicts the definition of $L(\mathcal{A}) \cup L(\mathcal{B})$. Symmetrically, we argue for the pairs $(b^{n-1}ab^i, b^{n-1-i}ab^{n-1})$, for $0 \leq i \leq n-1$. In all cases we obtain a contradiction to our assumption. This shows that an additional state is needed, which gives the $m+n+1$ lower bound for the language $L(\mathcal{A}) \cup L(\mathcal{B})$. □

Now we turn to the intersection of NFAs. Again, we make use of the upper bound already proven.

Theorem 3. *For any integers $m, n \geq 2$ let $\mathcal{A}$ be an m-state and $\mathcal{B}$ be an n-state NFA that accept star-free languages. Then $m \cdot n$ states are sufficient for an NFA to accept the language $L(\mathcal{A}) \cap L(\mathcal{B})$. This bound is tight for binary alphabets.*

Proof. The upper bound of $m \cdot n$ states follows from the construction presented in [15] for the intersection of general regular languages accepted by NFAs. In order to show a matching lower bound we apply the DFA used in [8] to prove the corresponding result for deterministic finite automata. Clearly, the DFA is also an NFA. However, here we have to show that the resulting NFA is minimal. So, let $\mathcal{A} = (S, \{a, b\}, \delta, s_0, F)$ with state set $S = \{0, 1, \ldots, m-1\}$, for $m \geq 2$. State 0 is the initial state s_0, and state $m-1$ is the only final state. The transition function is given by (cf. Fig. 2):

- $\delta(i, a) = \{i+1\}$, for $0 \leq i < m-1$, and
- $\delta(i, b) = \{i\}$, for $0 \leq i \leq m-1$.

The language accepted by $\mathcal{A}$ is $b^*(ab^*)^{m-1}$, which can easily be shown to be star free. Similarly, we define the automaton $\mathcal{B}$ by taking the automaton defined above but now with n states, and interchange the letters a and b. Hence we obtain the language $L(\mathcal{B}) = a^*(ba^*)^{n-1}$, which is star-free, too.

It is not hard to verify that

$$L(\mathcal{A}) \cap L(\mathcal{B}) = \{ u \in \{a, b\}^* \mid |u|_a = m-1 \text{ and } |u|_b = n-1 \}.$$

It remains to be shown that this language needs at least mn states if accepted by an NFA. To this end, consider the following set of pairs

$$P = \{ (a^i b^j, a^{m-1-i}b^{n-1-j}) \mid 0 \leq i \leq m-1 \text{ and } 0 \leq j \leq n-1 \}.$$

For each pair $(a^i b^j, a^{m-1-i}b^{n-1-j})$ in S the word $a^i b^j a^{m-1-i}b^{n-1-j}$ has $m-1$ symbols a and $n-1$ symbols b. So, it belongs to $L(\mathcal{A}) \cap L(\mathcal{B})$. Next, consider different pairs $(a^i b^j, a^{m-1-i}b^{n-1-j})$ and $(a^{i'} b^{j'}, a^{m-1-i'}b^{n-1-j'})$ from P with $i \neq i'$ or $j \neq j'$. At least one of the words induced by crossed pairs is not in the intersection of the languages accepted by $\mathcal{A}$ and $\mathcal{B}$. Thus, the set P is a fooling set for $L(\mathcal{A}) \cap L(\mathcal{B})$ of size $m \cdot n$, which proves the stated claim. □

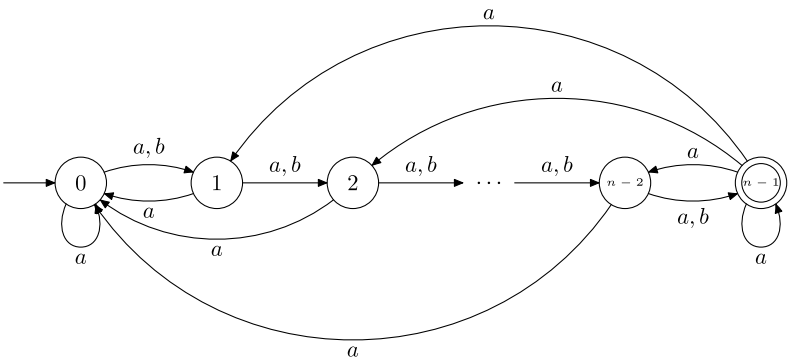

Fig. 3. The n-state NFA $\mathcal{A}$, $n \geq 3$, used for the lower bound on the complementation problem for star-free languages. Any NFA accepting the complement of $L(\mathcal{A})$ needs at least 2^n states.

Next we come to the complementation operation. Here the situation is a little bit more involved to come up with an NFA accepting a star-free language that meets the tight exponential bound of 2^n states [15,20]. The following easy example already gives an exponential lower bound. Consider the languages $L_n = \{a, b\}^* a \{a, b\}^n b \{a, b\}^*$, for $n \geq 0$, accepted by $(n + 3)$-state NFAs. Obviously, these languages are star free. Moreover, from [15] it is known that any NFA that accepts the complement of L_n needs at least 2^{n-2} states. Thus, we have proven a tight bound in order of magnitude. The question arises, whether one can do better. We answer the question in the affirmative by showing that the language used in [20] (accepted by the NFA depicted in Fig. 3) is in fact star-free.

Before we can start with our investigation we need some additional notation that gives some insights on permutation-free automata that are built by the powerset construction from NFAs [14].

Lemma 4. *Let $\mathcal{A}$ be an NFA with state set S over alphabet Σ, and assume that $\mathcal{A}'$ is the equivalent minimal DFA obtained by the powerset construction, which is non-permutation-free. If the word w in Σ^* induces a non-trivial permutation on the state set $\{P_1, P_2, \ldots, P_k\} \subseteq 2^S$ of $\mathcal{A}'$ such that $\delta'(P_i, w) = P_{i+1}$, for $1 \leq i < k$, and $\delta'(P_k, w) = P_1$, then there are no two states P_i and P_j with $i \neq j$ such that $P_i \subseteq P_j$.*

Now we are prepared for the next theorem.

Theorem 5. *For any integer $n \geq 2$ let $\mathcal{A}$ be an n-state NFA that accepts a star-free language. Then 2^n states are sufficient for an NFA to accept the complement of the language $L(\mathcal{A})$. The bound is tight for binary alphabets.*

Proof. It suffices to prove that the NFA $\mathcal{A} = (S, \Sigma, \delta, 0, F)$, with alphabet $\Sigma = \{a, b\}$, state set $S = \{0, 1, \ldots, n - 1\}$, set of final states $F = \{n - 1\}$, and

$$\delta(i, x) = \begin{cases} \{i + 1\}, & \text{if } i < n - 1 \text{ and } x = b \\ \{0, i + 1\}, & \text{if } i < n - 1 \text{ and } x = a \\ \{1, 2, \ldots, n - 1\}, & \text{if } i = n - 1 \text{ and } x = a \end{cases}$$

that is depicted in Fig. 3 accepts a star-free language. To this end, we consider the equivalent minimal DFA $\mathcal{A}' = (S', \Sigma, \delta', \{0\}, F')$ obtained by the powerset construction where $S' \subseteq 2^S$.

The cardinality of the symmetric difference of two states R and T of $\mathcal{A}'$ is denoted by $\langle R, T \rangle = |R \setminus T| + |T \setminus R|$. The outline of the proof is as follows. We assume contrarily that the language accepted by $\mathcal{A}'$ is not star free. Then there exists a word $w \in \{a, b\}^*$ that induces a non-trivial permutation on a subset $P = \{P_1, P_2 \ldots, P_k\}$ of the states of $\mathcal{A}'$ such that $\delta'(P_i, w) = P_{i+1}$, $1 \leq i < k$, and $\delta'(P_k, w) = P_1$. Next we consider arbitrary pairs $P_i \neq P_j$ and distinguish whether the state $n - 1$ of $\mathcal{A}$ belongs to none of P_i and P_j, to both, or to exactly one of them. In all cases we will derive either a contradiction or a decrease of the cardinality of the symmetric difference. In particular, this shows that the cardinality can never increase. By $\langle P_i, P_j \rangle = \langle \delta'(P_i, w^k), \delta'(P_j, w^k) \rangle = \langle \delta'(P_i, w^{k \cdot m}), \delta'(P_j, w^{k \cdot m}) \rangle$, for $m \geq 0$, this implies a contradiction also in the case of decreasing cardinality.

We first show that w must be at least two letters long in order to induce a non-trivial permutation. If w would be equal to a, then at most $n - 1$ applications of δ' to any P_i give a state P_i' which includes $n - 1$. At most two more applications of δ' to P_i' give the set $\{0, 1, \ldots, n - 1\}$ which includes any state from P. This contradicts Lemma 4. If w would be equal to b, then at most $n - 1$ applications of δ' to any P_i give the empty set, which is a rejecting sink state that can never be part of P.

Now we turn to distinguish the three cases for the occurrence of the state $n - 1$. Let $P_i = \{i_1, i_2, \ldots, i_{\ell_i}\}$ and $P_j = \{j_1, j_2, \ldots, j_{\ell_j}\}$ be two arbitrary but different states from P, where $i_1 < i_2 < \cdots < i_{\ell_i}$ and $j_1 < j_2 < \cdots < j_{\ell_j}$.

First we consider the case that the state $n - 1$ belongs to exactly one of P_i or P_j. Without loss of generality we assume $n - 1 \in P_i \setminus P_j$. If, in this case, the first letter of w is an a we have

$$\delta'(P_i, a) = \begin{cases} \{1, 2, \ldots, n - 1\} & \text{if } P_i = \{n - 1\} \\ \{0, 1, \ldots, n - 1\} & \text{otherwise.} \end{cases}$$

So, a contradiction to Lemma 4 follows if either $P_i \neq \{n-1\}$ or the prefixes of multiples of w start with a and include two a's that are not separated by exactly $n-2$ letters b, because in both cases P_i is transformed into $\{0, 1, \ldots, n-1\}$ which, in turn, includes any state from P. Therefore, we conclude $P_i = \{n-1\}$ which is transformed into $\{1, 2, \ldots, n-1\}$, and in order to avoid a contradiction, P_j must include at least one state from $\{0, 1, \ldots, n-2\}$. If 0 belongs to P_j, then $\delta'(P_i, ab^{n-2}) = \{n-1\}$ and $\delta'(P_j, ab^{n-2}) = \{n-2, n-1\}$, which is a contradiction to Lemma 4. If 0 does not belong to P_j, we consider the evolutions of P_i and P_j under input word ab^{n-2}. Then $\delta'(\{n-1\}, ab^{n-2}) = \{n-1\}$ and $\delta'(\{j_1, \ldots, j_{\ell_j}\}, ab^{n-2}) = \{n-2\}$ and $\delta'(\{n-2\}, ab^{n-2}) = \{n-2\}$. So, the evolution runs into cycles, and there is no way to reach state P_j from P_i and *vice versa*. Thus, they cannot be part of a non-trivial permutation.

We conclude that in the present case the first letter of w is a b and have

$$\delta'(P_i, b) = \begin{cases} \emptyset, & \text{if } P_i = \{n-1\} \\ \{i_1 + 1, \ldots, i_{\ell_i - 1} + 1\} & \text{otherwise} \end{cases}$$

as well as $\delta'(P_j, b) = \{j_1 + 1, \ldots, j_{\ell_j} + 1\}$. Furthermore, P_i must be different from $\{n-1\}$ since otherwise it would be transformed into the empty set. Together, this implies $\langle \delta'(P_i, b), \delta'(P_j, b) \rangle = \langle P_i, P_j \rangle - 1$. Thus, the cardinality of the symmetric difference is properly decreased. This concludes the case $n-1 \in P_i \setminus P_j$.

For the next case, we assume that state $n-1$ is not included in P_i and P_j, that is, $n-1 \notin P_i \cup P_j$. If the first letter of w is an a, we obtain $\delta'(P_i, a) = \{0, i_1+1, \ldots, i_{\ell_i}+1\}$ and $\delta'(P_j, a) = \{0, j_1+1, \ldots, j_{\ell_j}+1\}$. If the first letter of w is a b, we obtain $\delta'(P_i, b) = \{i_1 + 1, \ldots, i_{\ell_i} + 1\}$ and $\delta'(P_j, b) = \{j_1 + 1, \ldots, j_{\ell_j} + 1\}$. So, the single states belonging to P_i and P_j are "shifted" towards $n-1$. Since P_i and P_j are different, whatever the input is, applications of δ' evolve to a situation where one or both of the new states include $n-1$. Thus, to the case $n-1 \in P_i \setminus P_j$ covered before or to the following case $n-1 \in P_i \cap P_j$.

For the final case, we assume that state $n-1$ belongs to both P_i and P_j, that is, $n-1 \in P_i \cap P_j$. Clearly, now P_i as well as P_j must be different from $\{n-1\}$. Otherwise, one of both would be a subset of the other, which is a contradiction to Lemma 4. If, in the present case, the first letter of w is an a, then P_i and P_j are immediately transformed into $\{0, 1, 2, \ldots, n-1\}$ which causes again a contradiction to Lemma 4. So, we know that the first letter of w is a b. After consuming the letter b, that is, after one transition we have $\delta'(P_i, b) = \{i_1 + 1, \ldots, i_{\ell_i - 1} + 1\}$ and $\delta'(P_j, b) = \{j_1 + 1, \ldots, j_{\ell_j - 1} + 1\}$. If both new states $\delta'(P_i, b)$ and $\delta'(P_j, b)$ contain $n-1$, we repeat the argumentation of the present case. This means that we will be concerned with another application of δ' on input b, resulting in $\delta'(P_i, bb) = \{i_1 + 2, \ldots, i_{\ell_i - 2} + 2\}$ and $\delta'(P_j, bb) = \{j_1 + 2, \ldots, j_{\ell_j - 2} + 2\}$. Since P_i and P_j are different and the single states belonging to it are shifted towards $n-1$ during an application of δ' on input b, the argumentation can be repeated until we end up with either both new states do not contain $n-1$, or $n-1$ belongs to exactly one of them. The latter situation has completely been covered by the case $n-1 \in P_i \setminus P_j$ before. The former situation brings us to the case $n-1 \notin P_i \cup P_j$ which, in turn, may end up in the present case again. However, in every possible step of this cycle between both cases, the single states are shifted towards $n-1$. Moreover, only a 0 may additionally be included. So, since P_i and P_j are different, the cycle appears finitely often only. This concludes the case $n-1 \in P_i \cap P_j$ and, hence, the proof. □

3.2. Catenation operations

Next we investigate the concatenation operation and its iteration, the Kleene star. In general, these operations have deterministic state complexity $m \cdot 2^n - 2^{n-1}$ and $2^{n-1} + 2^{n-2}$ in the worst case, which is also met for star-free languages [8]. For NFAs all these operations are cheap, in the sense that $m+n$ and $n+1$ states are sufficient and necessary in the worst case. We show that for star-free languages exactly the same bounds apply. For concatenation we find the following situation.

Theorem 6. *For any integers $m, n \geq 2$ let $\mathcal{A}$ be an m-state and $\mathcal{B}$ be an n-state NFA that accept star-free languages. Then $m+n$ states are sufficient for an NFA to accept the language $L(\mathcal{A}) \cdot L(\mathcal{B})$. The bound is tight for binary alphabets.*

Proof. Again, the upper bound is that for general regular languages [15]. For the lower bound we use the NFAs $\mathcal{A}$ and $\mathcal{B}$ introduced in the proof of Theorem 2. Recall that $L(\mathcal{A}) = a^{m-1}(ba^{m-1})^*$ and $L(\mathcal{B}) = b^{n-1}(ab^{n-1})^*$. In order to show that $m+n$ states are necessary for any NFA to accept the language $L(\mathcal{A}) \cdot L(\mathcal{B})$ we construct the set

$$P = \{ (a^i, a^{m-1-i}ba^{m-1}b^{n-1}) \mid 0 \leq i \leq m-1 \} \cup \{ (a^{m-1}b^{n-1}ab^j, b^{n-1-j}) \mid 0 \leq j \leq n-1 \},$$

whose fooling set property is verified as follows: consider the word pairs of the following form $(a^i, a^{m-1-i}ba^{m-1}b^{n-1})$. The word $a^i \cdot a^{m-1-j}ba^{m-1}b^{n-1}$, for $0 \leq i, j \leq m-1$, is in $L(\mathcal{A}) \cdot L(\mathcal{B})$ if and only if $i = j$. Thus, the pair $(a^i, a^{m-1-i}ba^{m-1}b^{n-1})$ induces a word that belongs to the concatenation of the languages under consideration, but the crossing of different pairs of this form gives two words that are not in $L(\mathcal{A}) \cdot L(\mathcal{B})$. Similarly, we can argue for the pairs $(a^{m-1}b^{n-1}ab^j, b^{n-1-j})$, for $0 \leq j \leq n-1$. Finally, we have to compare pairs of the form $(a^i, a^{m-1-i}ba^{m-1}b^{n-1})$, for $0 \leq i \leq m-1$, with pairs $(a^{m-1}b^{n-1}ab^j, b^{n-1-j})$, for $0 \leq j \leq n-1$. Since $a^i \cdot b^{n-1-j}$ belongs to $L(\mathcal{A}) \cdot L(\mathcal{B})$ if and only if $i = m-1$ and $j = 0$, for the cases $0 \leq i < m-1$ and $0 < j \leq n-1$ at least one word induced by crossing the corresponding pairs is not in the concatenation of the languages accepted by the automata $\mathcal{A}$ and $\mathcal{B}$. For the remaining case $i = m-1$ and $j = 0$ we find that the other induced word $a^{m-1}b^{n-1}ab^j \cdot a^{m-1-i}ba^{m-1}b^{n-1}$ does not belong to $L(\mathcal{A}) \cdot L(\mathcal{B})$. Therefore, P is a fooling set for the language $L(\mathcal{A}) \cdot L(\mathcal{B})$ of size $m+n$. Hence, the stated claim follows. □

The star-free languages are not closed under Kleene star, which is seen by the finite language a^2. Since the minimal DFA accepting $(a^2)^*$ reads as $\mathcal{A} = (\{0, 1\}, \{a\}, \delta, 0, \{0\})$ with $\delta(0, a) = 1$ and $\delta(1, a) = 0$ and contains a non-trivial permutation on the state set $\{0, 1\}$ by reading the word a, this language is *not* star-free. Nevertheless, one can consider the corresponding operation problem (leaving the family of star-free languages).

Theorem 7. *For any integer $n \geq 2$ let $\mathcal{A}$ be an n-state NFA that accepts a star-free language. Then $n + 1$ states are sufficient for an NFA to accept the Kleene star of the language of $\mathcal{A}$. This bound is tight for binary alphabets.*

Proof. The upper bound can be found in [15]. For the lower bound we again use the automaton $\mathcal{A}$ introduced in the proof of Theorem 2. Recall, that $L(\mathcal{A}) = a^{n-1}(ba^{n-1})^*$. We claim that

$$P = \{ (a^{n-1}ba^i, a^{n-1-i}ba^{n-1}) \mid 0 \leq i \leq n-1 \} \cup \{(\lambda, \lambda)\}$$

is a fooling set for $L(\mathcal{A})^*$. In fact, the word $a^{n-1}ba^i \cdot a^{n-1-j}ba^{n-1}$, for $0 \leq i, j \leq n-1$, is in $L(\mathcal{A})^*$ if and only if $i = j$. Hence, the pair $(a^{n-1}ba^i, a^{n-1-i}ba^{n-1})$ gives a word that belongs to $L(\mathcal{A})^*$, but for different pairs of this form none of the induced words (by crossing) is a member of $L(\mathcal{A})^*$. Obviously, the remaining pair (λ, λ) gives the empty word that is a member of $L(\mathcal{A})^*$ by definition. Finally, for the words $a^{n-1}ba^i \cdot \lambda$ and $\lambda \cdot a^{n-1-i}ba^{n-1}$, for $0 \leq i \leq n-1$, at least one is not in $L(\mathcal{A})^*$. Thus, the pairs $(a^{n-1}ba^i, a^{n-1-i}ba^{n-1})$ from P with the pair (λ, λ) obey the properties required for being a fooling set. Therefore, the claim follows since P is of size $n + 1$. □

3.3. Reversal operation

Finally we turn our attention to the reversal operation. This operation has deterministic state complexity 2^n in the worst case, which is also met up to one state off in the case of star-free languages [8]. For NFAs this operation is cheap, leading to a tight bound of $n + 1$ states [20]. Our result on the reversal operation follows from the general case by a slight modification of the automaton depicted in Fig. 1. This automaton also was used to show the tight bound of $n + 1$ states for the reversal operation on general NFAs. The modification is simply to make all states accepting. Since this does not effect the existence of non-trivial permutations on the state set of the equivalent minimal DFA, we may conclude that the language accepted is star free. Thus, we obtain the following result.

Theorem 8. *For any integer $n \geq 2$ let $\mathcal{A}$ be an n-state NFA that accepts a star-free language. Then $n + 1$ states are sufficient for an NFA to accept the reversal of the language $L(\mathcal{A})$. This bound is tight for binary alphabets.* □

4. Operations on unary star-free languages

Before we continue with the operation problem for *unary* star-free languages accepted by NFAs, recall the following fact, which was already mentioned earlier. A language $L \subseteq \{a\}^*$ is star free if and only if L is finite or co-finite. We start with the Boolean operations as in the previous section.

4.1. Boolean operations

For deterministic finite automata accepting unary star-free languages the worst case state complexities of the Boolean operations union, intersection, and complement are $\max\{m, n\}$, $\max\{m, n\}$, and n [8]. These bounds do not reach the worst state complexity of union and intersection for general unary languages [38], which is in both cases mn. The situation is different for NFAs. The upper bound of $m + n + 1$ for the general unary union of NFAs has been shown in [15]. For unary *star-free* languages the following lower bound misses the upper bound by one state.

Theorem 9. *For any integers m, n with $n \geq 6$ and $3 < m \leq n - 2$ there exist an m-state NFA $\mathcal{A}$ and an n-state NFA $\mathcal{B}$ accepting unary star-free languages, such that $m + n$ states are necessary for any NFA to accept the union $L(\mathcal{A}) \cup L(\mathcal{B})$.*

Proof. Let $\mathcal{B} = (S_{\mathcal{B}}, \{a\}, \delta_{\mathcal{B}}, s_{0,\mathcal{B}}, F_{\mathcal{B}})$ be the NFA with state set $S_{\mathcal{B}} = \{0, 1, \ldots, n-1\}$, where state 0 is the initial state $s_{0,\mathcal{B}}$ and the single final state. The transition function is given by

- $\delta_{\mathcal{B}}(i, a) = \{i + 1\}$, for $0 \leq i \leq n - 3$,
- $\delta_{\mathcal{B}}(n - 2, a) = \{0, n - 1\}$, and
- $\delta_{\mathcal{B}}(n - 1, a) = \{0\}$.

The language accepted by $\mathcal{B}$ is $\{a^{n-1}, a^n\}^*$. Since $n - 1$ and n are relatively prime, this language includes all words longer than $(n - 1) \cdot n - (n - 1) - n$ and, thus, is co-finite and hence star free. Next, let $\mathcal{A} = (S_{\mathcal{A}}, \{a\}, \delta_{\mathcal{A}}, s_{0,\mathcal{A}}, F_{\mathcal{A}})$ be an NFA with state set $S_{\mathcal{A}} = \{0', 1', \ldots, (m-1)'\}$, for $3 < m \leq n - 2$. State $0'$ is the initial state $s_{0,\mathcal{A}}$ and state $(m - 1)'$ is the only accepting state. The transition function is given by

- $\delta_{\mathcal{A}}(i', a) = \{(i + 1)'\}$, for $0 \leq i < m - 1$.

The language accepted by $\mathcal{A}$ is $\{a^{m-1}\}$. Since it is finite, it is star free.

We now consider an NFA $\mathcal{C} = (S, \{a\}, \delta, q, F)$ that accepts $L(\mathcal{A}) \cup L(\mathcal{B})$. The two shortest words belonging to the union of $L(\mathcal{A})$ and $L(\mathcal{B})$ are λ and a^{m-1} since $3 < m-1 < n-2$. To accept these two words we provide m states $q_0, q_1, \ldots, q_{m-1}$. Let q_0 be the initial state accepting λ, and q_{m-1} be the state accepting a^{m-1}. So, we have $q_{m-1} \in \delta(q_0, a^{m-1})$.

The states q_0 and q_{m-1} must be different since, otherwise, the words of $a^{m-1}(a^{m-1})^*$ are also accepted. But these words do not belong to the union of the languages considered as follows. For $3 < m \leq n-2$ and $k \geq 1$ we have $(a^{m-1})^k = a^{k \cdot m-k}$. The only possibility that such a word belongs to the union $L(\mathcal{A}) \cup L(\mathcal{B})$ is that the word can be obtained by combining multiples of a^{n-1} and a^n. Consider the word a^{2m-2}. Since $3 < m \leq n-2$ we have $6 \leq 2m-2 \leq 2n-6$. So the word a^{2m-2} needs to be equal to either a^n or a^{n-1}. This especially means $2m-2 = n$ or $2m-2 = n-1$. The next words belonging to $\{a^{n-1}, a^n\}^*$ are a^{2n-2}, a^{2n-1} and a^{2n}. Since either $2m-2 = n$ or $2m-2 = n-1$ we obtain $4m-4 = 2n$ or $4m-4 = 2n-2$. Thus, the word $3m-3$ cannot belong the union of our witness languages. Therefore, states q_0 and q_{m-1} must be different.

Assume there is a cycle on the path from q_0 to q_{m-1}. The length x of this cycle is at most $m \leq n-2$. But then the word a^{m-1+x} is also accepted by $\mathcal{C}$. However, a^{m-1+x} does not belong to either $L(\mathcal{A})$ or $L(\mathcal{B})$. Therefore, there cannot exist a cycle on the path from q_0 to q_{m-1}.

So far, we need m states and a^{m-1} and λ are accepted. Furthermore, all positive multiples of a^{n-1} and a^n and all combinations of those words have to be accepted. To accept the words of $\{a^{n-1}\}^*$ we need a cycle of length $n-1$ between not necessarily different states of the NFA that is reachable from the initial state q_0. Assume q_0 is the only accepting state within this cycle, that is, $\delta(q_0, a^{i(n-1)}) \cap F = \{q_0\}$, for $i \geq 0$. Then the word $a^{n-1}a^{m-1}$ is also accepted but does not belong to $L(\mathcal{A}) \cup L(\mathcal{B})$. Therefore, q_0 cannot be the only accepting state of the cycle. Similarly, we obtain a contradiction when q_{m-1} is the only accepting state of that cycle. Thus, we consider the case that both accepting states (but no more) belong to the cycle. To avoid the previous contradiction we must have $\{q_0\} \in \delta(q_{m-1}, a^{n-1})$ and $\{q_{m-1}\} \in \delta(q_0, a^{n-1})$. But then the word $a^{m-1}a^{n-1}$ not belonging to the union is also accepted. It follows that there has to exist a new accepting state $q_{n-1} \in \delta(q_0, a^{n-1})$ that accepts a^{n-1} and all its multiples since none of the already existing non-accepting states can be accepting. Otherwise, the automaton $\mathcal{C}$ would accept a word a^x with $0 < x < n-1$ and $x \neq m-1$.

So far, we have $m+1$ states from which q_0, q_{m-1}, and q_{n-1} are accepting. In order to accept all multiples of a^{n-1} we also need $n-2$ non-accepting states building a path from q_{n-1} to q_{n-1} while reading a^{n-1}. We prove these states not to be equivalent to any of the already existing states. Let $q_1, q_2, \ldots, q_{m-2}$ be the non-accepting states on the path from q_0 to q_{m-1} while reading a^{m-1}. Let q_i be a state of $\{q_1, q_2, \ldots, q_{m-2}\}$ and let q_i also be on the path from q_{n-1} to q_{n-1}. For all choices of q_i it is possible to take the detour of $n-1$ states from state q_i to q_{n-1} and back to q_i on a path from q_0 to q_{m-1}. Because of this it is possible to accept the word $a^{m-1}a^{n-1}$ that does not belong to the union. Thus, there exists a path of length $n-1$ from state q_{n-1} to itself that does not include any state from $\{q_0, q_1, \ldots, q_{m-1}\}$. This path also cannot contain a loop of length $1 < x < n-1$ since then a word $a^{2n-2}a^x$ would be accepted that does not belong to the union. There also cannot exist a loop of length 1 since then the word a^{n+1} is accepted that does not belong to the union, either. Hence, there exist another $n-2$ non-accepting states. So we already need $m+n-1$ states to accept the subset $\{a^{m-1}\} \cup \{a^{n-1}\}^*$ of the language $L(\mathcal{A}) \cup L(\mathcal{B})$.

The only words still not accepted belong to the sets $\{a^n\}^*$ and $\{a^{n-1}, a^n\}^+$. To accept all words of $\{a^n, a^{n-1}\}^*$ using as many existing states as possible, we need to extend the cycle from q_{n-1} to q_{n-1}. We cannot add a state of the set $\{q_0, q_1, \ldots, q_{m-1}\}$ since otherwise the word $a^n a^{m-1}$ would be accepted. We also cannot add a cycle of length x with $3 \leq x \leq n-2$ on the path from q_{n-1} to q_{n-1} while reading a^{n-1} since then words of the form $a^{2n-2}a^x$ would be accepted that do not belong to the considered language. A cycle of length 2 is also impossible since then the word $a^{2n-2}a^4$ would be accepted. The only possibility left is to add a single new state that allows to take a detour on the path from q_{n-1} to q_{n-1}. Now the automaton also accepts all words of $\{a^{n-1}, a^n\}^*$ and, thus, all words of the union $L(\mathcal{A})$ and $L(\mathcal{B})$. We conclude that $m+n$ states are necessary to accept this language. □

Next we turn to the intersection. The upper bound of mn states for unary regular languages has been shown in [15]. The following theorem shows that this bound is tight in the order of magnitude for an infinite sequence of unary *star-free* languages.

Theorem 10. *For any even integer $n \geq 4$ there exist an n-state NFA $\mathcal{A}$ and an $(n+1)$-state NFA $\mathcal{B}$ accepting unary star-free languages, such that $\Omega(n^2)$ states are necessary for any NFA to accept the intersection $L(\mathcal{A}) \cap L(\mathcal{B})$.*

Proof. Consider the NFA $\mathcal{A}$ introduced in the proof of Theorem 9 and recall that $L(\mathcal{A}) = \{a^{n-1}, a^n\}^*$ which includes all words longer than $(n-1) \cdot n - (n-1) - n$. Similarly, we define the automaton $\mathcal{B}$ by taking $\mathcal{A}$ with $n+1$ states. Hence, we obtain the star-free language $L(\mathcal{B}) = \{a^n, a^{n+1}\}^*$.

Since any three consecutive natural numbers starting with an odd number not smaller than three are pairwise relatively prime, we have $\gcd(n-1, n) = 1$, $\gcd(n, n+1) = 1$, and $\gcd(n-1, n+1) = 1$. We now show that $\Omega(n^2)$ states are necessary for any NFA to accept the intersection $L(\mathcal{A}) \cap L(\mathcal{B})$. At first we determine which words do not belong to the intersection under consideration. We know that all words of length greater than $n(n+1) - n - (n+1) = n^2 - n - 1$ belong to the considered intersection since $n(n+1) - n - (n+1)$ is the biggest number that cannot be obtained by a linear combination of n and $n+1$. All words having a length that is a multiple of n belong to the intersection as well. Now we are looking for the smallest number which is not a multiple of n and which fulfills the equation

$$b(n-1) + cn = dn + e(n+1), \tag{1}$$

for some $b, c, d, e \geq 0$. Assuming for a moment $d \geq c$, this is equivalent to the following equation:

$$b(n-1) = (d-c)n + e(n+1) \iff x(n-1) = yn + z(n+1), \tag{2}$$

for some integers $x, y, z \geq 0$. Since also in this equation the coefficients need to be non-negative, we can concentrate on the smallest coefficient x to find the smallest multiple of $n-1$ that belongs to the intersection. We do not need to consider the multiples of n since all of these already belong to $L(\mathcal{A}) \cap L(\mathcal{B})$.

It is obvious that the equation

$$n \cdot (n-1) = 0 \cdot (n+1) + (n-1) \cdot n \tag{3}$$

is true for every n. We also have the relation $n-1 = 2n-(n+1)$. For our Eqs. (2) and (3) this means to obtain $(x-1)(n-1)$ we need to increase y by 1 and decrease z by 2.

We now can compute some lengths of words that belong to the intersection of $L(\mathcal{A})$ and $L(\mathcal{B})$. We especially can compute the length of the shortest non-empty word belonging to the intersection which is not a multiple of n. We get the following equations by transforming Eq. (3) using the above relation:

$$\begin{aligned} n \cdot (n-1) &= 0 \cdot (n+1) + (n-1) \cdot n \\ (n-1) \cdot (n-1) &= 1 \cdot (n+1) + (n-3) \cdot n \\ (n-2) \cdot (n-1) &= 2 \cdot (n+1) + (n-5) \cdot n \\ &\vdots \\ \left(\frac{n}{2}+1\right) \cdot (n-1) &= \left(\frac{n}{2}-1\right) \cdot (n+1) + 1 \cdot n. \end{aligned}$$

The last computed number $\frac{n(n+1)}{2} - 1$ gives the length of the shortest non-empty word belonging to the intersection that is not a multiple of n. This is due to the fact that all other possible linear combinations we obtain by continuing this computation, will only provide a bigger number or a number for that at least one coefficient is negative. Modifications by adding multiples of n on both sides of the last equation lead to bigger numbers, and subtracting multiples of n on both sides lead to numbers that cannot represent lengths of words belonging to $L(\mathcal{A})$.

So far we considered the case $d \geq c$. However, also when $c \geq d$ Eq. (1) provides an appropriate equation:

$$b(n-1) + (c-d)n = e(n+1) \iff x'(n+1) = y'n + z'(n-1),$$

for some $x', y', z' \geq 0$.

The relation $n+1 = 2n-(n-1)$ gives us the possibility to do a similar calculation as above. We start with the obvious equation

$$n \cdot (n+1) = 0 \cdot (n-1) + (n+1) \cdot n \tag{4}$$

to get the equation

$$\left(\frac{n}{2}\right) \cdot (n+1) = \left(\frac{n}{2}\right) \cdot (n-1) + n$$

that provides the length $\frac{n(n+1)}{2}$ of another word that belongs to the intersection, which is not a multiple of n. With the arguments from above we also cannot go on with this computation. Thus, the intersection of $L(\mathcal{A})$ and $L(\mathcal{B})$ contains no words shorter than $\frac{n(n+1)}{2} - 1$ except λ and words having a length that is a multiple of n.

We now consider a minimal NFA accepting the intersection of the languages of the chosen automata. This needs to reject all other words shorter than $\frac{n(n+1)}{2} - 1$ except for all words whose length is a multiple of n (including λ). To accept the word $a^{\frac{n(n+1)}{2}-1}$ there needs to exist a path in the NFA of that length. This path may contain cycles. If it does not contain a cycle there must exist at least $\frac{n(n+1)}{2} - 1$ states in the NFA and we are done. Now we consider the case that there exist cycles on the path that accepts $a^{\frac{n(n+1)}{2}-1}$, that is, we see at least one state at least twice. First we assume that there exists only one cycle on our path that is used only once completely. Skipping this cycle in the computation results in accepting the word $a^{\frac{n(n+1)}{2}-1-k}$, where k denotes the length of the cycle skipped. If this word is λ then the cycle length is $\frac{n(n+1)}{2} - 1$, thus, we have that number of states and we are done. Otherwise, its length needs to be a multiple of n. Therefore, k needs to be of the form

$$i \cdot n + \frac{n}{2} - 1 \quad \text{or} \quad i \cdot n - \frac{n}{2} - 1,$$

for some $i \geq 1$, since only in these cases the length of the word is a multiple of n. This holds because of the following considerations: The number

$$\left(\frac{n(n+1)}{2} - 1 - k\right) \cdot \frac{1}{n} = \frac{n+1}{2} - \frac{1+k}{n}$$

needs to be a non-negative integer. Since $n+1$ is odd we get a remainder of $\frac{1}{2}$ in the left fraction of the right part. To let the total sum become an integer there must be also a remainder of $\frac{1}{2}$ in the right fraction of the right part. Thus, $\frac{1+k}{n}$ needs to

fulfill one of the following equations:

$$\frac{1+k}{n} = i + \frac{1}{2} \quad \text{or} \quad \frac{1+k}{n} = i - \frac{1}{2}.$$

By applying some simple transformations these equations provide the forms from above.

Recall that at the moment we assume that there exists only one cycle on our path which is used only once completely. Then the computation can use at most all but one state twice. This means the NFA already has at least $(\frac{n(n+1)}{2} - 1) \cdot \frac{1}{2}$ states and we are done.

We next assume that there still exists only one cycle on the path accepting $a^{\frac{n(n+1)}{2}-1}$ but this cycle is used at least twice. Skipping this cycle in the computation once, results in accepting $a^{\frac{n(n+1)}{2}-1-k}$, where k denotes the length of the cycle skipped, similarly as above. Obviously, this word cannot be λ and is shorter than $a^{\frac{n(n+1)}{2}-1}$. So, its length needs to be a multiple of n, and k has the form as above. Next, skipping this cycle in the computation twice, results in accepting $a^{\frac{n(n+1)}{2}-1-2k}$. If this word is λ then the cycle length is at least $(\frac{n(n+1)}{2} - 1) \cdot \frac{1}{2}$, thus, we have that number of states and we are done. Otherwise, its length needs to be a multiple of n, but this is impossible because the numbers

$$\begin{aligned}\frac{n+1}{2} - \frac{1+2k}{n} &= \frac{n+1}{2} - \left(1 + 2i \cdot n + \frac{2n}{2} - 2\right) \cdot \frac{1}{n} \\ &= \frac{n}{2} + \frac{1}{2} - (2i \cdot n + n - 1)\frac{1}{n} \\ &= \frac{n}{2} + \frac{1}{2} - 2i - 1 + \frac{1}{n}\end{aligned}$$

and

$$\begin{aligned}\frac{n+1}{2} - \frac{1+2k}{n} &= \frac{n+1}{2} - \left(1 + 2i \cdot n - \frac{2n}{2} - 2\right) \cdot \frac{1}{n} \\ &= \frac{n}{2} + \frac{1}{2} - (2i \cdot n - n - 1)\frac{1}{n} \\ &= \frac{n}{2} + \frac{1}{2} - 2i + 1 + \frac{1}{n}\end{aligned}$$

are no integers, since $\frac{1}{n}$ is not $\frac{1}{2}$ for $n \geq 3$. Thus, there cannot exist a cycle that is used at least twice.

Now we assume that there exist several cycles on the path accepting $a^{\frac{n(n+1)}{2}-1}$. Skipping one of the cycles results in accepting a word whose length must be a multiple of n. Thus, we can use the same reasoning as before and conclude that every cycle has a length of the form $i \cdot n \pm \frac{n}{2} - 1$, for some $i \geq 1$. So, skipping altogether arbitrary two of the cycles leads to the same contradiction as in the case where one cycle has been skipped twice. This implies that the accepting computation on $a^{\frac{n(n+1)}{2}-1}$ uses at most one cycle once. Therefore, the NFA has at least $(\frac{n(n+1)}{2} - 1) \cdot \frac{1}{2}$ states. This shows that $\Omega(n^2)$ states are necessary for any NFA to accept the intersection. □

Next we come to the complementation operation. For this operation on general unary NFAs a crucial role is played by the function

$$F(n) = \max\{\operatorname{lcm}(x_1, \ldots, x_k) \mid x_1, \ldots, x_k \geq 1 \wedge x_1 + \cdots + x_k = n\}$$

which gives the maximal order of the cyclic subgroups of the symmetric group of n symbols—here $\operatorname{lcm}(x_1, x_2, \ldots, x_k)$ refers to the least common multiple of $x_1, x_2, \ldots, x_k$. The function F has been investigated in [24,25] where the asymptotic growth rate $\lim_{n\to\infty} \frac{\ln(F(n))}{\sqrt{n \cdot \ln(n)}} = 1$ has been proven. A bound immediately derived from this result is $\ln(F(n)) \in \Theta(\sqrt{n \cdot \ln(n)})$. For our purposes the implied rough estimation $F(n) \in e^{\Theta(\sqrt{n \cdot \ln(n)})}$ suffices. In [9] it has been shown that for any unary n-state NFA there exists an equivalent $O(F(n))$-state deterministic finite automaton. So, $O(F(n))$ is an upper bound for the unary NFA complementation. In [15] is has been proven that this bound is tight in the order of magnitude. However, the situation is different for unary star-free languages, where the bound is much cheaper.

Theorem 11. *For any integer $n \geq 3$ let $\mathcal{A}$ be an n-state NFA accepting a unary star-free language. Then $O(n^2)$ states are sufficient for an NFA to accept the complement of $L(\mathcal{A})$. This bound is tight in the order of magnitude.*

Proof. In [9] it is shown that any n-state NFA that accepts a unary language L can effectively be transformed into an equivalent DFA, such that the initial sequence of states has a length at most of order $O(n^2)$ which is followed by a (possibly large) single cycle. Since a star-free language is either finite or co-finite all states in the cycle are either accepting or rejecting. So, minimizing this DFA results in a permutation-free automaton, which collapses at least the aforementioned (possibly

large) cycle into a *trivial* loop at the end of the deterministic tail. Thus, the minimal DFA has at most $O(n^2)$ states. Since complementing DFAs does not increase the number of states, the upper bound follows.

For the lower bound we use the NFA $\mathcal{A}$ introduced in the proof of Theorem 9. Recall that $L(\mathcal{A}) = \{a^{n-1}, a^n\}^*$ and that it includes all words longer than $(n-1)\cdot n-(n-1)-n$. The complement of $L(\mathcal{A})$ is the finite language that includes all non-empty words which cannot be obtained by concatenating the words a^{n-1} and a^n finitely often. So, the longest word in the complement is $a^{(n-1)n-(n-1)-n}$. Since an NFA that accepts a finite language needs at least one state more than the length of the longest word of the language, any NFA accepting the complement of $L(\mathcal{A})$ needs at least $(n-1)n-(n-1)-n+1 = n^2-3n+2$ states. Thus, any NFA accepting the complement of $L(\mathcal{A})$ needs $\Omega(n^2)$ states. □

4.2. Catenation operations

Now we turn to the catenation operation and its iteration. For unary languages the deterministic state complexity of concatenation and Kleene star reads as mn and $(n-1)^2+1$, which cannot be reached by unary star-free languages as shown in [8]. There, a bound of $m+n-1$ and $n^2-7n+13$ were deduced for concatenation and Kleene star. It is worth mentioning that these bounds are much closer to the bound for the corresponding operation problems on DFAs accepting unary *finite* languages. Again, the state bounds for NFAs accepting unary languages are different from those bounds for DFAs [15]. The upper bound for the concatenation of general unary NFAs is $m+n$ [15]. For unary star-free languages the following lower bound misses this upper bound by one state.

Theorem 12. *For any integers $m, n \geq 2$ there exist an m-state NFA $\mathcal{A}$ and an n-state NFA $\mathcal{B}$ accepting unary star-free languages, such that $m+n-1$ states are necessary for any NFA to accept the concatenation $L(\mathcal{A})\cdot L(\mathcal{B})$.*

Proof. Let $\mathcal{A} = (S, \{a\}, \delta, s_0, F)$ be an NFA with state set $S = \{0, 1, \ldots, m-1\}$, for $m \geq 2$. State 0 is the initial state s_0, and state $m-1$ is the single final state. The transition function is given by

- $\delta(i, a) = \{i+1\}$, for $0 \leq i \leq m-2$, and
- $\delta(m-1, a) = \{m-1\}$.

The language accepted by $\mathcal{A}$ is $a^{m-1}a^*$. Since this language is co-finite it is star-free. Similarly we define the automaton $\mathcal{B}$ by taking $\mathcal{A}$ with n states. Hence, we obtain the star-free language $L(\mathcal{B}) = a^{n-1}a^*$. For the concatenation we have $L(\mathcal{A})\cdot L(\mathcal{B}) = a^{m+n-2}a^*$.

It remains to be shown that this language needs at least $m+n-1$ states if accepted by an NFA. To this end, we construct the following set of pairs

$$P = \{\, (a^i, a^{m+n-2-i}) \mid 0 \leq i \leq m+n-2 \,\}.$$

For each pair $(a^i, a^{m+n-2-i})$ in P the word $a^i a^{m+n-2-i}$ belongs to the concatenation. However, for each two different pairs $(a^i, a^{m+n-2-i})$ and $(a^j, a^{m+n-2-j})$ exactly one of the words $a^i a^{m+n-2-j}$ and $a^i a^{m+n-2-j}$ does and one does not belong to $L(\mathcal{A})\cdot L(\mathcal{B})$. So, the set P is a fooling set for $L(\mathcal{A})\cdot L(\mathcal{B})$ of size $m+n-1$. This proves the stated claim. □

The following theorem deals with the Kleene star operations and shows that the obtained bound is tight.

Theorem 13. *For any integer $n \geq 3$ let $\mathcal{A}$ be an n-state NFA that accepts a unary star-free language. Then $n+1$ states are sufficient for an NFA to accept the Kleene star of the language $L(\mathcal{A})$. This bound is tight.*

Proof. The upper bound of $n+1$ states for general unary languages can be found in [15]. For the lower bound we use the NFA $\mathcal{A} = (S, \{a\}, \delta, s_0, F\})$ with state set $S = \{0, 1, \ldots, n-1\}$. State 0 is the initial state s_0 and state $n-1$ is the single final state. The transition function δ is given by

- $\delta(i, a) = \{i+1\}$, for $0 \leq i \leq n-3$,
- $\delta(n-2, a) = \{0, n-1\}$, and
- $\delta(n-1, a) = \{0\}$.

The language accepted by $\mathcal{A}$ is $\{a^{n-1}, a^n\}^*\{a^{n-1}\}$, which can be shown to be star free by using similar arguments as in the proof of Theorem 9.

For the lower bound we argue as follows. Assume to the contrary that there is an NFA $\mathcal{C} = (S_{\mathcal{C}}, \{a\}, \delta_{\mathcal{C}}, q_0, F_{\mathcal{C}})$ with at most n states accepting the language $L(\mathcal{A})^*$. The shortest four words belonging to $L(\mathcal{A})^*$ are λ, a^{n-1}, a^{2n-2}, and a^{2n-1}. It follows $q_0 \in F_{\mathcal{C}}$. While accepting the word a^{n-1} automaton $\mathcal{C}$ passes through a sequence of states, say, $q_0, q_1, \ldots, q_{n-1}$, with $q_{i+1} \in \delta_{\mathcal{C}}(q_i, a)$, for $0 \leq i \leq n-2$ and $q_{n-1} \in F_{\mathcal{C}}$. Moreover, every state q_i, for $1 \leq i \leq n-2$, is non-accepting. Otherwise the automaton $\mathcal{C}$ would accept a word that does not belong to the language under consideration. Furthermore, it is easily seen that all states q_i, with $0 \leq i \leq n-2$, are different. If not, a non-empty word that is strictly shorter in length than $n-1$ would be accepted.

Next we exclude certain transitions between the states $\{q_0, q_1, \ldots, q_{n-1}\}$. First consider q_0. There is no transition from q_0 to q_0 because the input a has to be rejected. Moreover, there cannot be any transition from q_0 to q_j, for $2 \leq j \leq n-1$. Otherwise word $a^{n-1-(j-1)}$ would be accepted, and since $1 \leq n-1-(j-1) \leq n-2$, a word not belonging to $L(\mathcal{A})^*$

is accepted. Furthermore, there is no transition from state q_i to q_j, for $1 \leq i, j \leq n-2$ except for the transitions from q_i to $q_j = q_{i+1}$. We consider three cases: (i) if $i = j$, then there is a loop on state q_i and the computation on a^{n-1} can be extended by running through this loop once resulting in accepting the word $a^{n-1+1} = a^n$, which is not a member of $L(\mathcal{A})^*$. (ii) If $i < j$, then we have a forward transition in the state sequence $q_1, q_2, \ldots, q_{n-2}$. In this case $i + 1 < j$. Then automaton $\mathcal{C}$ accepts the word $a^{n-1-(j-i-1)}$, by taking this transition instead of running through the states in between. Since $1 \leq j - i - 1 \leq n - 4$ none of the words $a^{n-1-(j-i-1)}$ are in $L(\mathcal{A})^*$ and thus these transitions cannot exist. (iii) Finally, if $j < i$, then we have a backward transition which can be used to accept the word $a^{n-1+(i-j+1)}$. But since $2 \leq i - j + 1 \leq n - 2$ one can accept a word whose length is at least $n + 1$ but at most $2n - 3$. Since none of these words belong to $L(\mathcal{A})^*$ automaton $\mathcal{C}$ cannot contain any of these transitions.

Next we take a closer look on the number of accepting states of $\mathcal{C}$. Assume for a moment that $\mathcal{C}$ has only one accepting state, that is, state q_0, in addition to the $n-2$ non-accepting states $q_1, q_2, \ldots, q_{n-2}$. Then $q_0 = q_{n-1}$ and since the number of states is n there may be another state, say q'. This new state should be reachable and useful, that is, there is at least one path from q_0 via q' to q_0. Any path of this kind has clearly a length of at least $n - 1$. Moreover, since there are only $n - 1$ states different from q', any path of this kind without cycles cannot be longer than $2n - 2$. Therefore, we derive that the length of such a path is either exactly $n - 1$ or exactly $2n - 2$. Similarly as above we conclude that any cycle on such a path has a length of $n-1$, if it exists at all. So, any processing of the input a^{2n-2} drives automaton $\mathcal{C}$ into state q_0. This implies that a^{2n-1} cannot be accepted. From the contradiction and the assumption $|S_{\mathcal{C}}| \leq n$ we now have $|F_{\mathcal{C}}| = 2$, that is $F_{\mathcal{C}} = \{q_0, q_{n-1}\}$ and $q_0 \neq q_{n-1}$. Since a^n does not belong to $L(\mathcal{A})^*$, there is no transition from q_{n-1} to q_{n-1}. Thus, since a^{2n-2} as well as a^{2n-1} have to be accepted, there must exist a transition from q_{n-1} to q_0 (a transition from q_0 to q_{n-1} has been excluded above). This implies that a^n is accepted, which is a contradiction also for the case $|F_{\mathcal{C}}| = 2$. Together we conclude that n states are not enough. □

4.3. Reversal operation

The reversal operation is cheap for both DFA and NFA if they accept unary languages only. Here cheap means a tight bound of n states in both cases. This bound is also met by DFA accepting unary star-free languages [8]. We find a similar situation when considering NFA, because applying the operation reversal on a unary (star-free) language does not affect the language at all. Thus, the number of states of an NFA accepting the reversal of a language is the same as the one of the NFA accepting the language. For the sake of completeness we include the following theorem.

Theorem 14. *For any integer $n \geq 1$ let $\mathcal{A}$ be an n-state NFA accepting a unary star-free language. Then n states are sufficient for an NFA to accept the reversal of $L(\mathcal{A})$. This bound is tight.* □

References

[1] J.-C. Birget, Intersection and union of regular languages and state complexity, Inform. Process. Lett. 43 (1992) 185–190.
[2] H. Bordihn, M. Holzer, M. Kutrib, Determinization of finite automata accepting subregular languages, Theoret. Comput. Sci. 410 (2009) 3209–3222.
[3] J.A. Brzozowski, Roots of star events, J. ACM 14 (1967) 466–477.
[4] J.A. Brzozowski, Quotient complexity of regular languages, J. Autom. Lang. Comb. 15 (2010) 71–89.
[5] J.A. Brzozowski, R.S. Cohen, On decompositions of regular events, J. ACM 16 (1969) 132–144.
[6] J.A. Brzozowski, G. Jirásková, B. Li, J. Smith, Quotient complexity of bifix-, factor-, and subword-free languages, in: Automata and Formal Languages (AFL 2011), Institute of Mathematics and Informatics, College of Nyíregyháza, Debrecen, 2011, pp. 123–137.
[7] J.A. Brzozowski, G. Jirásková, C. Zou, Quotient complexity of closed languages, in: Computer Science Symposium in Russia (CSR 2010), in: LNCS, vol. 6072, Springer, 2010, pp. 84–95.
[8] J.A. Brzozowski, B. Liu, Quotient complexity of star-free languages, in: Automata and Formal Languages (AFL 2011), Institute of Mathematics and Informatics, College of Nyíregyháza Debrecen, 2011, pp. 138–152.
[9] M. Chrobak, Finite automata and unary languages, Theoret. Comput. Sci. 47 (1986) 149–158.
[10] M. Chrobak, Errata to finite automata and unary languages, Theoret. Comput. Sci. 302 (2003) 497–498.
[11] I. Glaister, J. Shallit, A lower bound technique for the size of nondeterministic finite automata, Inform. Process. Lett. 59 (1996) 75–77.
[12] Y.-S. Han, K. Salomaa, Nondeterministic state complexity for suffix-free regular languages, in: Descriptional Complexity of Formal Systems (DCFS 2010), in: EPTCS, vol. 31, 2010, pp. 189–204.
[13] Y.-S. Han, K. Salomaa, D. Wood, Nondeterministic state complexity of basic operations for prefix-free regular languages, Fund. Inform. 90 (2009) 93–106.
[14] M. Holzer, S. Jakobi, M. Kutrib, The magic number problem for subregular language families, in: Descriptional Complexity of Formal Systems (DCFS 2010), in: EPTCS, vol. 31, 2010, pp. 110–119.
[15] M. Holzer, M. Kutrib, Nondeterministic descriptional complexity of regular languages, Internat. J. Found. Comput. Sci. 14 (2003) 1087–1102.
[16] M. Holzer, M. Kutrib, Unary language operations and their nondeterministic state complexity, in: M. Ito, M. Toyama (Eds.), Developments in Language Theory (DLT 2002), in: LNCS, vol. 2450, Springer, 2003, pp. 162–172.
[17] M. Holzer, M. Kutrib, Nondeterministic finite automata – Recent results on the descriptional and computational complexity, Internat. J. Found. Comput. Sci. 20 (2009) 563–580.
[18] M. Holzer, M. Kutrib, Descriptional complexity – an introductory survey, in: Scientific Applications of Language Methods, Imperial College Press, 2010, pp. 1–58.
[19] M. Holzer, M. Kutrib, Descriptional and computational complexity of finite automata – a survey, Inform. Comput. 209 (2011) 456–470.
[20] G. Jirásková, State complexity of some operations on binary regular languages, Theoret. Comput. Sci. 330 (2005) 287–298.
[21] G. Jirásková, M. Krausová, Complexity in prefix-free regular languages, in: Descriptional Complexity of Formal Systems (DCFS 2010), in: EPTCS, vol. 31, 2010, pp. 197–204.
[22] G. Jirásková, T. Masopust, Complexity in union-free regular languages, in: Developments in Language Theory (DLT 2010), in: LNCS, vol. 6224, Springer, 2010, pp. 255–266.

[23] G. Jirásková, P. Olejár, State complexity of intersection and union of suffix-free languages and descriptional complexity, in: Non-Classical Models of Automata and Applications (NCMA 2009), in: books@ocg.at, vol. 256, Austrian Computer Society, 2009, pp. 151–166.
[24] E. Landau, Über die Maximalordnung der Permutationen gegebenen Grades, Archiv der Math. und Phys. 3 (1903) 92–103.
[25] E. Landau, Handbuch der Lehre von der Verteilung der Primzahlen, Teubner, 1909.
[26] O.B. Lupanov, A comparison of two types of finite sources, Problemy Kybernetiki 9 (1963) 321–326. (in Russian), German translation: Über den Vergleich zweier Typen endlicher Quellen. Probleme der Kybernetik 6 (1966) 328–335.
[27] R. McNaughton, S. Papert, Counter-Free Automata, in: Research Monographs, No. 65, MIT Press, 1971.
[28] C. Mereghetti, G. Pighizzini, Two-way automata simulations and unary languages, J. Autom. Lang. Comb. 5 (2000) 287–300.
[29] C. Mereghetti, G. Pighizzini, Optimal simulations between unary automata, SIAM J. Comput. 30 (2001) 1976–1992.
[30] A.R. Meyer, M.J. Fischer, Economy of description by automata, grammars, and formal systems, in: Symposium on Switching and Automata Theory (SWAT 1971), IEEE, 1971, pp. 188–191.
[31] 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 (1971) 1211–1214.
[32] G. Pighizzini, J.O. Shallit, Unary language operations, state complexity and Jacobsthal's function, Internat. J. Found. Comput. Sci. 13 (2002) 145–159.
[33] M.O. Rabin, D. Scott, Finite automata and their decision problems, IBM J. Res. Dev. 3 (1959) 114–125.
[34] K. Salomaa, S. Yu, NFA to DFA transformation for finite languages over arbitrary alphabets, J. Autom. Lang. Comb. 2 (1997) 177–186.
[35] M.P. Schützenberger, On finite monoids having only trivial subgroups, Inform. Control 8 (1965) 190–194.
[36] H.-J. Shyr, G. Thierrin, Ordered automata and associated languages, Tamkang J. Math. 5 (1974) 9–20.
[37] H.-J. Shyr, G. Thierrin, Power-separating regular languages, Math. Syst. Theory 8 (1974) 90–95.
[38] S. Yu, Regular languages, in: Handbook of Formal Languages, vol. 1, Springer, 1997, pp. 41–110. chap. 2.
[39] S. Yu, State complexity of regular languages, J. Autom. Lang. Comb. 6 (2001) 221–234.