Acta Informatica (2022) 59:163–181
https://doi.org/10.1007/s00236-021-00402-0

ORIGINAL ARTICLE

Finite automata with undirected state graphs

Martin Kutrib[1] · **Andreas Malcher**[1] · **Christian Schneider**[1]

Received: 13 February 2019 / Accepted: 11 May 2021 / Published online: 24 May 2021

Abstract
We investigate finite automata whose state graphs are undirected. This means that for any transition from state p to q consuming some letter a from the input there exists a symmetric transition from state q to p consuming a letter a as well. So, the corresponding language families are subregular, and in particular in the deterministic case, subreversible. In detail, we study the operational descriptional complexity of deterministic and nondeterministic undirected finite automata. To this end, the different types of automata on alphabets with few letters are characterized. Then, the operational state complexity of the Boolean operations as well as the operations concatenation and iteration is investigated, where tight upper and lower bounds are derived for unary as well as arbitrary alphabets under the condition that the corresponding language classes are closed under the operation considered.

1 Introduction

The operation problem for a language family is the question of costs (in terms of states) of operations on languages from this family with respect to their representations. More than two decades ago, the operation problem for regular languages represented by deterministic finite automata as studied in [14,15] renewed the interest in descriptional complexity issues of finite automata in general. In the meantime, impressively many results have been obtained for a large number of language families. It seems that the recent studies of operational state complexity focus on subregular languages. A recent survey of the several branches and details can be found in [4], which is also a valuable and comprehensive source of references. Already since the early days of automata theory, a significant theory on subfamilies of regular languages has been developed in the literature. Examples of early studied classes are finite languages, definite languages and variants, star-free languages, etc. Some of these regular subfamilies were motivated by particular issues such as, for instance, neural nets or circuit

Some of the results of this paper have been announced at DCFS 2018 in Halifax, July 2018. An extended abstract appeared in the proceedings of that conference [10].

✉ Martin Kutrib
kutrib@informatik.uni-giessen.de

Andreas Malcher
andreas.malcher@informatik.uni-giessen.de

[1] Institut für Informatik, Universität Giessen, Arndtstr. 2, 35392 Giessen, Germany

design. A list of some of these early regular subfamilies, their structure, and their properties was already given by Havel [5,6].

Since then, new developments in the theory of computer science triggered the study of new subregular language families. For instance, deterministic expression languages or one-unambiguous regular languages [2] are motivated from document type definitions (DTDs) used in standard generalized markup language (SGML) and extensible markup language (XML) schemes. In particular, the determinization of nondeterministic finite automata that accept some subregular languages has been investigated in detail [1]. In most cases, it turned out that the conversion problem is nearly as costly (in terms of the number of states) as in the general case.

Subregular language families of particular interest are the families of languages accepted by types of reversible finite automata. Reversibility is a fundamental principle in physics. Since abstract computational models with discrete internal states may serve as prototypes of computing devices which can physically be constructed, it is interesting to know whether these abstract models are able to obey physical laws. The observation that loss of information results in heat dissipation [12] strongly suggests to study computations without loss of information. Recent results on reversible finite automata can be found, for example, in [7–9,13].

Here, we are interested in a strict form of reversible finite automata, namely we do not only require that every state of the automaton has a unique predecessor for a given input letter, but that this predecessor can already be reached by a forward transition with the same input letter. These automata can be seen as finite automata whose state graphs are undirected. So, this notion is even stronger than the concept of time-symmetry studied in [3,11]. Time-symmetry appears in physics when a system can go back in time by applying the same transition function as for forward computations after a weak transformation of the phase-space. For example, in Newtonian mechanics one can go back in time by applying the same dynamics after a transformation that leaves masses and positions unchanged but reverses the sign of the momenta. While time-symmetric machines themselves cannot distinguish whether they run forward or backward in time, for undirected automata the time directions fade away since they are both available in the transition.

In the next section, we present the necessary notations and give an introductory example. Since the definition of undirected finite automata implies strong restrictions on the possible state graphs and, thus, the possible automata themselves, it is possible to characterize the different types of undirected automata with small alphabets in Sect. 3. These characterizations are a powerful tool to derive tight bounds on the operational state complexity of deterministic (Sect. 4) and nondeterministic (Sect. 5) undirected finite automata. All bounds obtained are summarized in Table 1. Finally, in Sect. 6, we discuss open and untouched problems for future work.

2 Preliminaries

Let Σ^* denote the set of all words over the finite alphabet Σ. The *empty word* is denoted by λ, and $\Sigma^+ = \Sigma^* \setminus \{\lambda\}$. The *reversal* of a word w is denoted by w^R. For the *length* of w we write $|w|$. For the number of occurrences of a symbol a in w, we use the notation $|w|_a$. Set inclusion is denoted by $\subseteq$ and strict set inclusion by $\subset$. We write 2^S for the power set and $|S|$ for the cardinality of a set S.

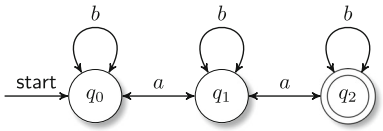


Fig. 1 State graph of an nondeterministic undirected finite automaton accepting the language $\{ w \in \{a, b\}^* \mid |w|_a \geq 1 \text{ and } |w|_a \bmod 2 = 0 \}$

A *nondeterministic finite automaton* (NFA) is a system $\langle Q, \Sigma, \delta, q_0, F \rangle$, where Q is the finite set of *internal states*, Σ is the finite set of *input symbols*, $q_0 \in Q$ is the *initial state*, $F \subseteq Q$ is the set of *accepting states*, and $\delta : Q \times \Sigma \to 2^Q$ is the *transition function*.

A finite automaton is *deterministic* (DFA) if and only if $|\delta(q, a)| = 1$, for all states $q \in Q$ and letters $a \in \Sigma$. In this case, we simply write $\delta(q, a) = p$ instead of $\delta(q, a) = \{p\}$ assuming that the transition function is a mapping $\delta : Q \times \Sigma \to Q$.

So, by definition, any DFA is *complete*, that is, the transition function is total, whereas it may be a partial function for NFAs in the sense that the transition function of nondeterministic machines may map to the empty set.

If the state graph induced by some finite automaton is undirected, then we obtain the subclasses of *nondeterministic undirected finite automata* (NUFA) and *deterministic undirected finite automata* (DUFA). Formally, for undirected finite automata, it is required that $q \in \delta(p, a)$ if and only if $p \in \delta(q, a)$, for all $p, q \in Q$ and $a \in \Sigma$. The *language accepted* by a finite automaton M is

$$L(M) = \{ w \in \Sigma^* \mid \delta(q_0, w) \cap F \neq \emptyset \},$$

where the transition function is recursively extended to $\delta : Q \times \Sigma^* \to 2^Q$.

In order to illustrate the definitions, we continue with an example.

Example 1 The NUFA $M = \langle \{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_2\} \rangle$ whose transition function is given through the state graph shown in Fig. 1 accepts the language $\{ w \in \{a, b\}^* \mid |w|_a \geq 1 \text{ and } |w|_a \bmod 2 = 0 \}$. □

3 Characterization of undirected finite automata with small alphabets

The definition of undirected finite automata implies strong restrictions on the possible state graphs and, thus, the possible automata themselves. These restrictions allow only a very few different state graphs for *minimal* deterministic undirected finite automata with a unary or binary input alphabet. The situation for nondeterministic automata is still rather restricted, but there are more types of distinguishable state graphs. The following characterizations of possible types of state graphs is used for later results on the closure properties of the families of accepted languages as well as their operational state complexity. In the rest of the section, we tacitly assume $\Sigma = \{a\}$ for unary languages and $\Sigma = \{a, b\}$ for binary languages.

3.1 Unary and binary deterministic undirected finite automata

The situation for unary deterministic undirected finite automata is straightforward. The only two possible state graphs are depicted in Fig. 2. Any of the states can be made accepting or rejecting which yields four different languages that are accepted by unary DUFAs.

Corollary 1 *Let M be a unary DUFA. Then, language $L(M)$ is one of the following:*

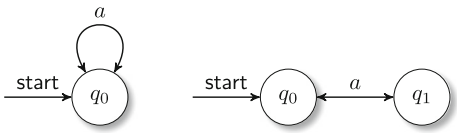


Fig. 2 Possible state graphs of unary deterministic undirected finite automata

1. $L_1 = \emptyset$,
2. $L_2 = a^*$,
3. $L_3 = \{ a^n \mid n \text{ is even} \}$,
4. $L_4 = \{ a^n \mid n \text{ is odd} \}$.

We continue with the analysis of possible state graphs of deterministic undirected finite automata accepting binary languages.

Theorem 1 *Let M be a minimal binary DUFA. Then, its state graph is of one of the five types shown in Fig. 3.*

Proof Let $M = \langle Q, \{a, b\}, \delta, q_0, F \rangle$ be a binary DUFA. First, we consider the cases where $\delta(q_0, a) = \delta(q_0, b)$. If $\delta(q_0, a) = \delta(q_0, b) = q_0$ then the transition function is fully specified and the state graph of M is of type 1 in Fig. 3. □ □

If $\delta(q_0, a) = \delta(q_0, b) \neq q_0$, then there is a new state, say state q_1, such that $\delta(q_0, a) = \delta(q_0, b) = q_1$. Since M is symmetric, again the transition function is fully specified. Now, the state graph of M is of type 2 in Fig. 3.

For the remaining cases, we assume $\delta(q_0, a) \neq \delta(q_0, b)$. Now, we can distinguish two subcases. The first subcase is given when one transition is a loop from q_0 to q_0, while the other goes to some new state q_1. So, let us first assume $\delta(q_0, a) = q_0$ and $\delta(q_0, b) = q_1$. Then, all transitions from state q_0 are specified. Due to the symmetry, the only transition missing is $\delta(q_1, a)$. Here a loop from q_1 to q_1 is possible or a transition to another new state q_2. For the case $\delta(q_1, a) = q_1$, the transition function is fully specified and we have a state graph of type 3 with $n = 1$. For the case $\delta(q_1, a) = q_2$, the only missing transition is $\delta(q_2, b)$. Now, the argumentation used for q_1 applies. Since the number of states is finite, eventually we end up with a loop transition and obtain a state graph of type 3.

The second subcase is given when both transitions $\delta(q_0, a)$ and $\delta(q_0, b)$ go to different new states, say q_1 and p_1. Then all transitions from state q_0 are specified. The transitions missing are $\delta(q_1, b)$ and $\delta(p_1, a)$. For both branches of the state graph we can argue as before for type 3. If one branch ends in a loop from a state to itself, the other branch necessarily ends also in a loop from a state to itself. The lengths of the branches determine the numbers $m, n \geq 1$ and the state graph is of type 4. Finally, if one branch does not end in a loop, but with a transition to some state of the other branch, then the other branch does not end in a loop as well (due to the symmetry). In this case, both branches are connected such that the state graph becomes a cycle, that is, it is of type 5. Since no more cases exist, the assertion follows. □

3.2 Unary nondeterministic undirected finite automata

In this subsection, we turn to the possible state graphs of unary NUFAs.

Theorem 2 *Let M be a unary NUFA. Then, language $L(M)$ is of one of the following types:*

1. $L_1 = \emptyset$,

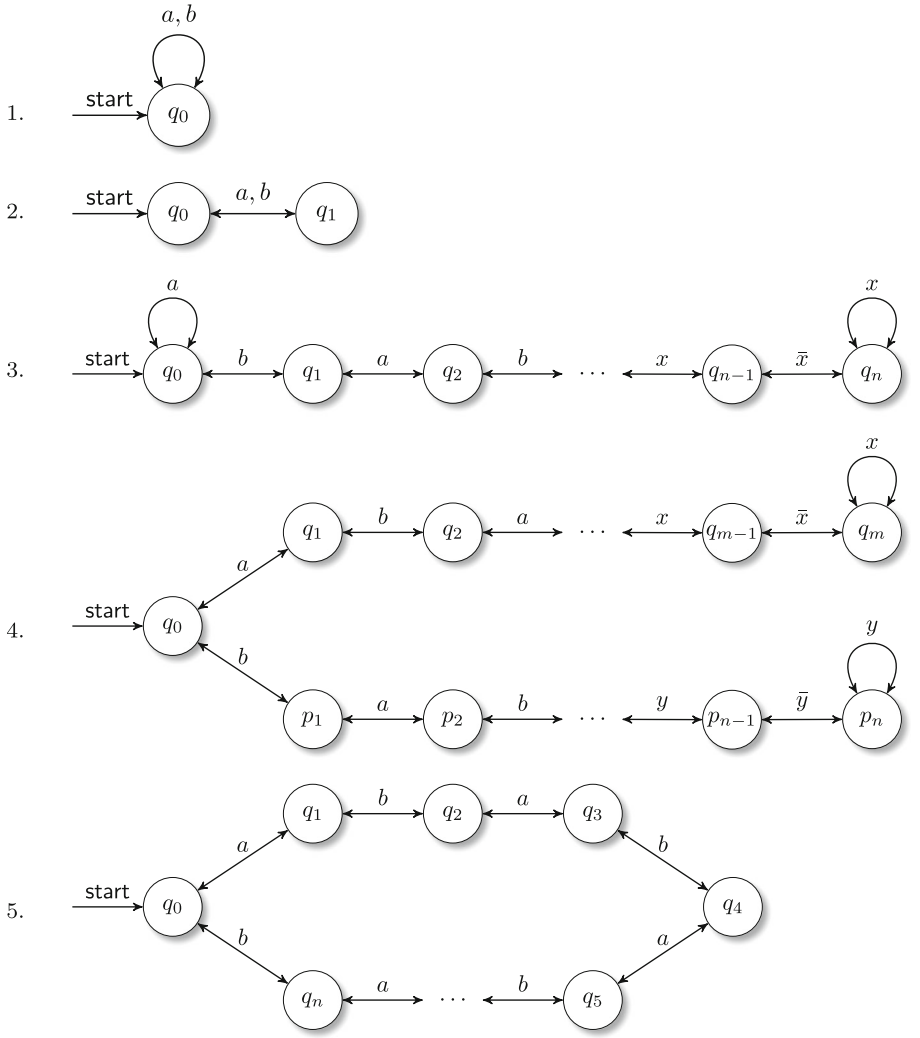


Fig. 3 Possible state graphs of binary deterministic undirected finite automata, where $m, n \geq 1$, the letters a and b are interchangeable, $x, y \in \{a, b\}$, $\bar{x} = a \iff x = b$, and $\bar{y} = a \iff y = b$. Accepting states can be anywhere

2. $L_2 = \{\lambda\}$,
3. $L_{3,k} = \{\, a^n \mid n \geq k \text{ and } n \text{ even} \,\}$, *for some* $k \geq 0$,
4. $L_{4,k} = \{\, a^n \mid n \geq k \text{ and } n \text{ odd} \,\}$, *for some* $k \geq 0$,
5. $L_{5,k,l} = \{\, a^n \mid n \geq k \text{ and } n \text{ is even if } k \leq n < l \,\}$, *for some* $k \geq 0$ *and* $l > k$,
6. $L_{6,k,l} = \{\, a^n \mid n \geq k \text{ and } n \text{ is odd if } k \leq n < l \,\}$, *for some* $k \geq 0$ *and* $l > k$.

Proof Let $M = \langle Q, \{a\}, \delta, q_0, F \rangle$ be an NUFA with unary input alphabet. Basically, the type of the accepted language is determined by two numbers (if they exist) $k \geq 0$ and $l > k$. The number k is defined to be the shortest path from the initial state q_0 to some accepting state $q_+ \in F$. If k exists, that is, if $L(M)$ is non-empty, then l is defined dependent on whether k is even or odd. If k is even, l is defined to be the shortest path longer than k that leads from

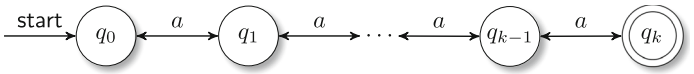


Fig. 4 State graph of a minimal nondeterministic undirected finite automaton accepting the language $L_{3,k}$ or $L_{4,k}$

the initial state q_0 to an accepting state q'_+ and has odd length. If such a path does not exist, l remains undefined. Similarly, if k is odd then l must be even.

Now, we can determine the type of $L(M)$. If there is no path from q_0 to some accepting state then $L(M) = L_1 = \emptyset$. Otherwise the number k is defined.

If $k = 0$ and, thus, $q_0 \in F$, then $\lambda \in L(M)$. If additionally $\delta(q_0, a)$ is undefined, then $L(M) = L_2 = \{\lambda\}$.

Next, assume that k is defined, $\delta(q_0, a)$ is defined, and l is undefined. Since M is undirected we obtain $q_0 \in \delta(q_0, a^2)$. We conclude that a^k belongs to $L(M)$ and, for all $i \geq 0$, the input a^{k+2i} belongs to $L(M)$ as well. Since l is undefined there are no words whose parity is different from a^k in $L(M)$. Therefore, if k is even, we derive $L(M) = L_{3,k} = \{ a^n \mid n \geq k \text{ and } n \text{ even} \}$. Similarly, if k is odd, we have $L(M) = L_{4,k} = \{ a^n \mid n \geq k \text{ and } n \text{ odd} \}$.

Finally, let k, $\delta(q_0, a)$, and l be defined. As before we may conclude that the input a^{k+2i} belongs to $L(M)$, for all $i \geq 0$. On the other hand, since l is defined and $l > k$, we also have $a^{l+2i} \in L(M)$, for all $i \geq 0$. The parities of k and l are different. Therefore, all words a^{l+i}, for $i \geq 0$, belong to $L(M)$. This implies $L(M) = L_{5,k,l} = \{ a^n \mid n \geq k \text{ and } n \text{ even for } k \leq n < l \}$ if k is even, and $L(M) = L_{6,k,l} = \{ a^n \mid n \geq k \text{ and } n \text{ odd for } k \leq n < l \}$ if k is odd.

Note that all language types are different and are, in fact, accepted by some NUFA. □

Note that the languages $\{ a^n \mid n \geq k \}$ are equal to $L_{5,k,k+1}$, if k is even, and equal to $L_{6,k,k+1}$, if k is odd. Next, we turn to the minimal numbers of states that are necessary for some NUFA to accept languages of different type.

Lemma 1 *Let M be a unary NUFA that accepts a language of type L_1 or L_2. Then, one state is sufficient and necessary for M.*

Proof Clearly, one non-accepting initial state for which the transition function is undefined is a witness when M accepts the empty set.

Similarly, if the initial state whose transition function is undefined is accepting, then M accepts the language $\{\lambda\}$. □

Lemma 2 *Let $k \geq 1$ and M be a unary NUFA that accepts a language of type $L_{3,k}$ or $L_{4,k}$. Then, $k + 1$ states are sufficient and necessary for M.*

Proof Since the shortest word accepted has length k, automaton M must have at least $k + 1$ states. Otherwise, $L(M)$ would be empty or a shorter word would be accepted. The NUFA depicted in Fig. 4 accepts $L_{3,k}$ or $L_{4,k}$ with $k + 1$ states. □

Lemma 3 *Let $k \geq 0$, $l > k$, and M be a unary NUFA that accepts a language of type $L_{5,k,l}$ or $L_{6,k,l}$. Then, l states are sufficient and necessary for M.*

Proof Since the shortest word accepted has length k, automaton M must have at least $k + 1$ states. If $l = k + 1$, the assertion follows.

In contrast to the assertion, assume now that M has $k+1 \leq j < l$ states. Since $a^l \in L(M)$, there is a loop passed through in an accepting computation on input a^l. Let $m \geq 1$ be the

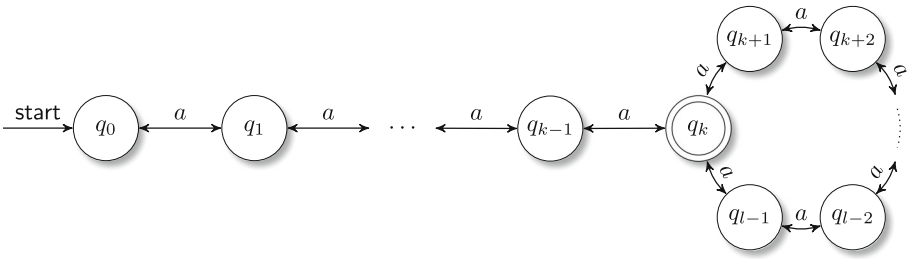


Fig. 5 State graph of a minimal nondeterministic undirected finite automaton accepting the language $L_{5,k,l}$ or $L_{6,k,l}$

length of the loop. So, there is an accepting computation on input a^{l-m}. Since k and l have different parities, m must be odd.

Consider the sequence of $l+1$ states $p_0, p_1, \ldots, p_x$, with p_x accepting, passed through in an accepting computation of a^l. Then, we let M accept a^{l-m} along the same sequence with leaving out the loop.

First assume that at least two of the states in the sequence belong to the loop left out, say p_i and p_j with $i < j$. While M runs through the loop from p_i to p_i via p_j, it reads an odd number of input symbols. So, either the computation from p_i to p_j or the computation from p_j on the loop back to p_i consumes an odd number of input symbols, but not both since the total length of the loop is odd. Since M is undirected, once in state p_i, M can reach p_j by following the loop or following the loop backward. On one path, it reads an even, and on the other path, it reads an odd number of input symbols. Continuing the computation from p_j to p_x gives in both cases a computation with less than l steps. However, only words of even or words of odd lengths less than l belong to $L(M)$. We derive from the contradiction that exactly one state in the sequence $p_0, p_1, \ldots, p_x$ belongs to the loop left out.

If there are more than one of such loops, the sum of their lengths must be odd. This implies that at least one has an even and at least one of them has an odd length. So, running through this one or that one gives accepted inputs of different parities shorter than l, a contradiction.

This implies that there is exactly one such loop that involves exactly one of the states, say p_i. We conclude that all states p_i, $0 \leq i \leq x$ are different. Moreover, $x+m$ must be l, and all states on the loop are different. Therefore, altogether $x+1+m-1 = x+m = l$ states are necessary.

The NUFA depicted in Fig. 5 accepts $L_{5,k,l}$ or $L_{6,k,l}$ with l states. □

4 Deterministic operational state complexity

In this section, we investigate the operational state complexity of DUFAs. Let $\circ$ be a binary operation under which the class of languages accepted by DUFAs is closed. The $\circ$-operation problem for DUFAs can be defined as follows:

- Given some m-state DUFA A and some an n-state DUFA B.
- How many states are sufficient and necessary (in terms of m and n) for a DUFA to accept the language $L(A) \circ L(B)$?

Table 1 Summary of the state complexities of the operations studied in this paper

	unary DUFA	DUFA	unary NUFA	NUFA
$L_1 \cup L_2$	1 or 2 [Th. 3]	mn [Th. 7, Th. 8]	—	—
$L_1 \cap L_2$	1 or 2 [Th. 3]	mn [Th. 5, Th. 6]	$\max(m,n)+1$ [Th. 9]	mn [Th. 12]
$\overline{L}$	1 or 2 [Th. 3]	n [Th. 4]	—	—
$L_1 \cdot L_2$	—	—	$m+n-1$ [Th. 10]	—
L^*	—	—	—	—
L^R	1 or 2 [Th. 3]	?	n [Th. 11]	—

It is marked by — if an automata class is not closed under the corresponding operation, ? means that the closure property is unknown. The complexities depicted are upper bounds which are tight in the sense that for all m, n there are automata of size m and n for which the upper bound is also necessary. The only exceptions are the intersection and union of DUFAs, where the upper bounds are shown to be tight for infinitely many m, n only

Table 2 Summary of the operational state complexities of DFAs and NFAs, where t is the number of accepting states of the "left" automaton

	unary DFA	DFA	unary NFA	NFA
$L_1 \cup L_2$	mn	mn	$m+n+1$	$m+n+1$
$L_1 \cap L_2$	mn	mn	mn	mn
$\overline{L}$	n	n	$2^{\Theta(\sqrt{n \cdot \log n})}$	2^n
$L_1 \cdot L_2$	mn	$m2^n - t2^{n-1}$	$m+n-1 \leq \cdot \leq m+n$	$m+n$
L^*	$(n-1)^2+1$	$3 \cdot 2^{n-2}$	$n+1$	$n+1$
L^R	n	2^n	n	$n+1$

The tight lower bounds for union, intersection, and concatenation of unary DFAs require m and n to be relatively prime

Obviously, this problem generalizes as well to unary language operations like, for example, complementation. Moreover, it also generalizes to other devices. The notion *state complexity* is used when the size of the finite automata is measured by their number of states.

Let L_1 and L_2 be languages over some alphabet Σ. In the following, we are particularly interested in the Boolean operations *complementation* ($\overline{L_1} = \Sigma^* \setminus L_1$), *union* ($L_1 \cup L_2 = \{ w \mid w \in L_1 \text{ or } w \in L_2 \}$), *intersection* ($L_1 \cap L_2 = \{ w \mid w \in L_1 \text{ and } w \in L_2 \}$) as well as in the operations *concatenation* ($L_1 \cdot L_2 = \{ uv \mid u \in L_1 \text{ and } v \in L_2 \}$), *iteration* ($L_1^* = \bigcup_{i \geq 0} L_1^i$ where $L_1^0 = \{\lambda\}$ and $L_1^{i+1} = L_1^i \cdot L_1$), and *reversal* ($L_1^R = \{ w^R \mid w \in L_1 \}$).

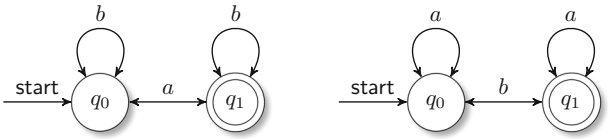
The results on the deterministic state complexity obtained in this section, and the results on the nondeterministic state complexity that is obtained in Sect. 5 are summarized in Table 1.

In order to compare the operational state complexities of DUFAs and NUFAs with classical deterministic and nondeterministic finite automata, we summarize the latter complexities in Table 2 (see, for example, [4] for details and references).

Before we can turn to study the operational state complexity of DUFAs under the above-mentioned operations, we have to give evidence under which operations their language class is *not* closed. The evidence of the closure under the remaining operations follows from the subsequent considerations of their state complexities.

The family of languages accepted by DUFAs is neither closed under concatenation nor under iteration. In the unary case, we have just the four languages from Corollary 1. Con-

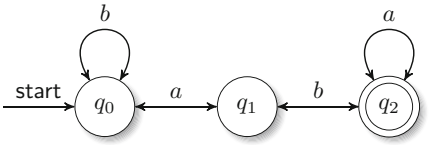
catenating $L_2 = a^*$ with $L_4 = \{ a^n \mid n \text{ is odd} \}$ yields a unary language that contains words of even as well as odd length, but does not contain the empty word. Thus, this language is not accepted by any DUFA. The iteration of $L_1 = \emptyset$ yields the language $\{\lambda\}$ that is not accepted by any DUFA either. This is already sufficient to show the non-closure under concatenation and iteration also in the general case. For the sake of completeness, we give non-unary examples witnessing the non-closure as well. Let A and B be the following DUFAs:



The shortest word in $L(A) \cdot L(B)$ is ab. So, a DUFA accepting $L(A) \cdot L(B)$ must have the following state sub-graph:

However, we have $ab \in L(A)$ and $b \in L(B)$ but abb cannot be accepted by any DUFA with a state sub-graph as depicted.

Concerning the iteration, we consider the DUFA A with the following state graph:



Assume there is a DUFA B that accepts $L(A)^*$. Since $\lambda \in L(A)^*$, the initial state of B is accepting. Moreover, the word ab belongs to $L(A)^*$. This implies that aa is accepted by B as well, but $aa \notin L(A)^*$.

The question whether the family of languages accepted by DUFAs is closed under reversal in the non-unary case is currently open. For the special case that the DUFAs have only one accepting state the closure follows by running the DUFAs backward, that is, the unique accepting state and the initial state switch their roles. However, this construction fails for more than one accepting state and, due to the symmetry, the usual trick to add an extra state and appropriate transitions fails as well.

We continue with the operational state complexity of DUFAs on unary alphabets. It has been shown in Corollary 1 that in this case, the automata have an easy form which makes it possible to show closure under the Boolean operations and reversal, and to establish upper and lower bounds for each of the operations. In the second part of the section, we investigate the state complexity of the Boolean operations for DUFAs over arbitrary alphabets and can establish tight bounds as well.

Theorem 3 *Let $m, n \in \{1, 2\}$ be integers and A be a minimal m-state and B be a minimal n-state DUFA over the same unary alphabet. Then one state is sufficient for a DUFA to accept $\overline{L(A)}$, $L(A) \cap L(B)$, $L(A) \cup L(B)$, or $L(A)^R$ if $m + n = 2$. Two states are sufficient for a DUFA to accept $\overline{L(A)}$, $L(A) \cap L(B)$, $L(A) \cup L(B)$, or $L(A)^R$ if $m = 2$.*

Table 3 Summary of the results for the union and intersection of languages accepted by unary DUFAs

$\cup$	L_1	L_2	L_3	L_4
L_4	L_4	L_2	L_2	L_4
L_3	L_3	L_2	L_3	L_2
L_2	L_2	L_2	L_2	L_2
L_1	L_1	L_2	L_3	L_4

$\cap$	L_1	L_2	L_3	L_4
L_4	L_1	L_4	L_1	L_4
L_3	L_1	L_3	L_3	L_1
L_2	L_1	L_2	L_3	L_4
L_1	L_1	L_1	L_1	L_1

Moreover, there are minimal m-state DUFAs and minimal n-state DUFAs such that these sufficient numbers of states are necessary.

Proof Owing to Corollary 1, we know that every language that is accepted by a unary DUFA has one of the following forms, namely $L_1 = \emptyset$, $L_2 = a^*$, $L_3 = \{ a^n \mid n \text{ is even} \}$, or $L_4 = \{ a^n \mid n \text{ is odd} \}$. For L_1 and L_2 one state and for L_3 and L_4 two states are necessary.

Now, we have to show that any application of the operations complementation, intersection, union, or reversal gives again one of the four unary languages which implies the upper bounds for the operations. For complementation, we have $\overline{L_1} = L_2$, $\overline{L_2} = L_1$, $\overline{L_3} = L_4$, and $\overline{L_4} = L_3$. For reversal, we have $L_i^R = L_i$ for $1 \leq i \leq 4$. For the operations union and intersection, we obtain the results which are summarized in Table 3 from which the lower bounds follow. □

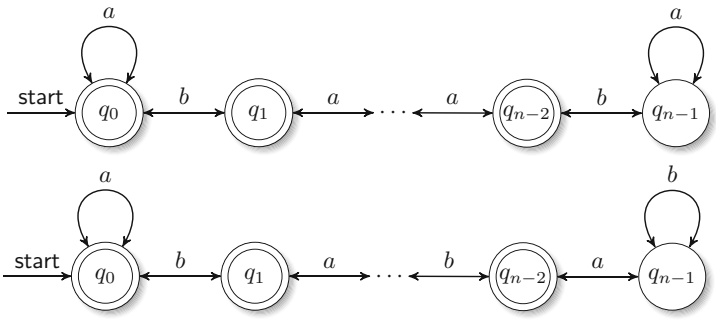
Now, we turn to arbitrary alphabets and consider first the complementation of DUFAs.

Theorem 4 *Let $n \geq 1$ be an integer. Then, n states are sufficient for a DUFA to accept the complement of any language accepted by an n-state DUFA. Moreover, for any minimal n-state DUFA A_n, any DUFA accepting the language $\overline{L(A_n)}$ has at least n states.*

Proof For the upper bound, it is sufficient to observe that the classical construction for the complementation of deterministic machines, that is, the interchanging of accepting and non-accepting states, works for DUFAs as well.

For the lower bound, let A_n be a minimal n-state DUFA. Assume that the complement $\overline{L(A_n)}$ is accepted by some DUFA B with $m < n$ states. Then, by the upper bound, $\overline{L(B)} = L(A_n)$ is accepted by a DUFA with m states which contradicts the minimality of A_n. □

Example 2 For every $n \geq 1$, the lower bound of the operational state complexity of DUFA complementation is witnessed by the following n-state DUFA $A_n = \langle \{q_0, q_1, \ldots, q_{n-1}\}, \{a, b\}, \delta_n, q_0, \{q_0, q_1, \ldots, q_{n-2}\} \rangle$ which is depicted for even n in the upper figure and for odd n in the lower figure.



Let us assume that $B = \langle Q, \{a, b\}, \delta, q_I, F\rangle$ is a DUFA accepting the complement $\overline{L(A_n)}$ with less than n states. Then, we consider $w_n = (ba)^{(n-1)/2}$ for odd n and $w_n = (ba)^{(n-2)/2}b$ for even n. In both cases, we have $|w_n| = n-1$ and $w_n \notin L(A_n)$ which implies that $w_n \in \overline{L(A_n)}$. Hence, $L(B) \neq \emptyset$ and there must be some $f \in F$ that is reachable from the initial state q_I. Since B has less than n states, there is some v with $|v| < n-1$ such that $\delta(q_I, v) = f$. Thus, $v \in L(B)$ and $|v| < n-1 = |w_n|$. On the other hand, w_n is the shortest word in $\overline{L(A_n)} = L(B)$ which gives the contradiction. □

Next, we continue with the intersection operation. To this end, we provide the following preparatory lemma which is needed to prove a tight lower bound.

Lemma 4 *Let $m \geq 4$ be an even integer. There are natural numbers x_1, x_2, y_1, and y_2 such that the following equations hold simultaneously. Moreover, one can choose x_1, x_2, y_1, and y_2 such that both equations additionally assume their minimum.*

$$m - 1 + 2mx_1 = 1 + (2m-2)y_1, \tag{1}$$
$$m + 1 + 2mx_2 = 2m - 3 + (2m-2)y_2. \tag{2}$$

Proof First, we show that Eq. (1) holds with a minimal value if we set $x_1 = m/2 = y_1 - 1$. Let $x_1 = y_1 - 1$, then $y_1 = m/2 + 1$ and we have

$$\begin{aligned} 1 + (2m-2)y_1 &= 1 + (2m-2)(m/2+1) = 1 + 2m \cdot m/2 + 2m - m - 2 = \\ &= m - 1 + 2mx_1. \end{aligned}$$

Thus, Eq. (1) holds if $x_1 = y_1 - 1 = m/2$, and assumes the value $m^2 + m - 1$. Next, we consider the case when $x_1 \neq y_1 - 1$ and show that in this case either Eq. (1) is not solvable or a value larger than $m^2 + m - 1$ is assumed. Let $x_1 \geq y_1$. Then, we have

$$\begin{aligned} m - 1 + 2mx_1 &= 1 + (2m-2)y_1 \Rightarrow \\ m - 1 + 2mx_1 &\leq 1 + (2m-2)x_1 \Rightarrow \\ x_1 &\leq 1 - m/2. \end{aligned}$$

This gives that the equation is not solvable, since $x_1 \geq 0$, but $1 - m/2 < 0$. Now, consider the remaining case $x_1 < y_1 - 1$. Then, we have

$$\begin{aligned} m - 1 + 2mx_1 &= 1 + (2m-2)y_1 \Rightarrow \\ m - 1 + 2mx_1 - (2m-2) &= 1 + (2m-2)(y_1 - 1) \Rightarrow \\ m - 1 + 2mx_1 - (2m-2) &> 1 + (2m-2)x_1 \Rightarrow \\ x_1 &> m/2. \end{aligned}$$

In this case, the value of the equation becomes larger than $m^2 + m - 1$, since $m - 1 + 2mx_1$ is a strictly increasing function of x_1. It follows that Eq. (1) holds with a minimal value, if $x_1 = m/2$ and $y_1 = x_1 + 1 = m/2 + 1$.

Second, we show that setting $x_2 = m/2 - 2 = y_2$ implies that Eq. (2) holds with a minimal value as well. Let $x_2 = y_2$, then we have

$$\begin{aligned} 2m - 3 + (2m-2)y_2 &= 2m - 3 + (2m-2)(m/2-2) = \\ &= 2m - 3 + 2m(m/2-2) - m + 4 = m + 1 + 2mx_2. \end{aligned}$$

Thus, Eq. (2) holds if $x_2 = y_2 = m/2 - 2$, and assumes the value $m^2 - 3m + 1$. Next, we consider the case when $x_2 \neq y_2$ and show that in this case either Eq. (2) is not solvable or a

value larger than $m^2 - 3m + 1$ is assumed. Let $x_2 < y_2$. Then, we have

$$\begin{aligned} m + 1 + 2mx_2 &= 2m - 3 + (2m - 2)y_2 \Rightarrow \\ m + 1 + 2mx_2 &> 2m - 3 + (2m - 2)x_2 \Rightarrow \\ x_2 &> m/2 - 2. \end{aligned}$$

Since $m + 1 + 2mx_2$ is a strictly increasing function of x_2, Eq. (2) assumes a value larger than $m^2 - 3m + 1$. Now, we consider the remaining case $x_2 > y_2$. Then, we have

$$\begin{aligned} m + 1 + 2mx_2 &= 2m - 3 + (2m - 2)y_2 \Rightarrow \\ m + 1 + 2my_2 &< 2m - 3 + (2m - 2)y_2 \Rightarrow \\ y_2 &< m/2 - 2. \end{aligned}$$

The left-hand side of Eq. (2) can be written as $(1 + 2x_2)m + 1$ and the right-hand side can be written as $(2 + 2y_2)m - (2y_2 + 3)$. Setting $g(x_2) = 1 + 2x_2$ and $h(y_2) = 2 + 2y_2$, we can write Eq. (2) as $g(x_2)m + 1 = h(y_2)m - (2y_2 + 3)$. We observe that at both sides of the equation there are multiples of m which are added by 1 and $-(2y_2 + 3)$, respectively. Hence, $1 + 2y_2 + 3$ has to be a multiple of m which gives that $2y_2 + 3 = km - 1$ for some integer $k \geq 1$. Thus, we obtain $y_2 = (km)/2 - 2$ which is a contradiction to the shown fact that $y_2 < m/2 - 2$. We conclude that Eq. (2) is not solvable if $x_2 > y_2$. Altogether, we have shown that Eq. (2) holds with a minimal value, if $x_2 = y_2 = m/2 - 2$. □

Theorem 5 *For any integers $m, n \geq 1$ let A be an m-state and B be an n-state DUFA. Then, $m \cdot n$ states are sufficient for a DUFA to accept the language $L(A) \cap L(B)$.*

Proof For the upper bound, we use the usual cross-product construction. Let $A = \langle Q_A, \Sigma, \delta_A, q_{0,A}, F_A \rangle$ be an m-state and $B = \langle Q_B, \Sigma, \delta_B, q_{0,B}, F_B \rangle$ be an n-state DUFA. Then define $C = \langle Q_A \times Q_B, \Sigma, \delta, (q_{0,A}, q_{0,B}), F_A \times F_B \rangle$, where $\delta((q_1, q_2), a) = (\delta_A(q_1, a), \delta_B(q_2, a))$ for all $q_1 \in Q_A$, $q_2 \in Q_B$, and $a \in \Sigma$. Clearly, C is an $(m \cdot n)$-state DUFA that accepts $L(A) \cap L(B)$. □

Theorem 6 *There are infinitely many integers $m, n \geq 1$ with $n = 2m - 2$ such that A is an m-state DUFA, B is an n-state DUFA, and $m \cdot n$ states are necessary for any DUFA to accept the language $L(A) \cap L(B)$.*

Proof For the lower bound, we consider two minimal DUFAs A and B, where A has m states for even $m \geq 4$ and B has $n = 2m - 2$ states. Both DUFAs have the form depicted in Fig. 6. □

Now, let C be a minimal DUFA accepting $L(A) \cap L(B)$. First, consider words in $L(A) \cap L(B)$ that have the form $(ab)^\ell a$ for some $\ell \geq 1$. Then, the left-hand side of Eq. (1) indicates the length of such words that are accepted by A, whereas the right-hand side of Eq. (1) indicates the length of such words that are accepted by B. According to Lemma 4, there are integers $x_1 = m/2$, $x_2 = m/2 - 2$, $y_1 = m/2 + 1$, and $y_2 = m/2 - 2$ such that Eqs. (1) and (2) hold simultaneously while assuming a minimal value. Hence, due to the equality and the minimality, we obtain a word $(ab)^{(m^2+m-2)/2}a$ of minimal length $m^2 + m - 1$ which belongs to $L(A) \cap L(B)$. Analogously, considering words in $L(A) \cap L(B)$ of the form $(ba)^\ell b$ for some $\ell \geq 1$ the left-hand side and right-hand side of Eq. (2) provides a word $(ba)^{(m^2-3m)/2}b$ of minimal length $m^2 - 3m + 1$ which belongs to $L(A) \cap L(B)$ as well. Since both minimal lengths are different, automaton C has to have two different paths from the initial state to an accepting state which implies, according to Theorem 1, that C is of

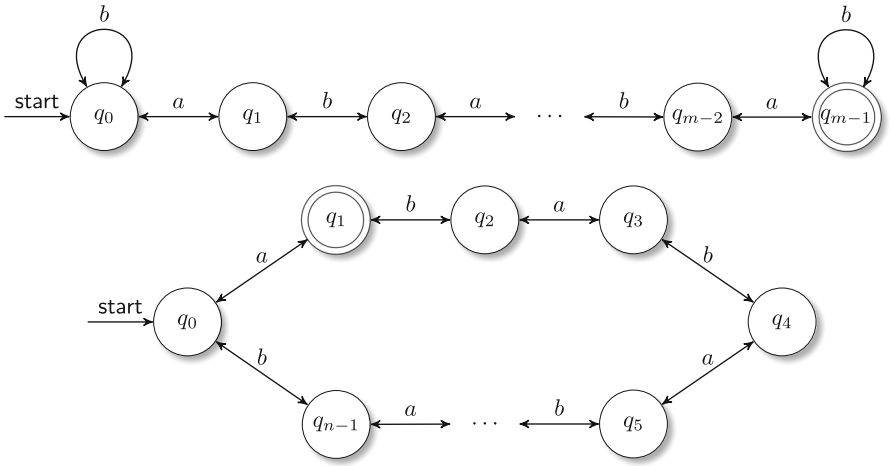


Fig. 6 The DUFA A (upper automaton) and B (lower automaton) used for proof of the lower bound

type 4 or type 5. Let us first assume that C is of type 4. Then, in particular the upper path contains an accepting state f that is entered after reading $w = (ab)^{(m^2+m-2)/2}a$ and the upper path ends in some state q that ends in a loop. If $f = q$, then there is a transition from f to f on input a which implies that input wa is accepted by C. This is a contradiction, since wa is not accepted by B. If $f \neq q$, then there are two possibilities, namely the remaining path from f to q processes inputs of the form $(ba)^k b^*$ or of the form $(ba)^{k-1}ba^*$ for some $k \geq 1$. In either case, we can construct a word w' which is accepted by C, but not by B which gives the contradiction. In the first case, we consider $w' = w(ba)^k b(ab)^k$, and in the second case, we set $w' = w(ba)^{k-1}ba(ba)^{k-1}b$. Hence, we conclude that C is of type 5. This means that C has two different paths which start in the initial state and end in the same state. Thus, C has at least $m^2 + m + m^2 - 3m + 2 - 2 = 2m^2 - 2m = m(2m - 2) = m \cdot n$ states. □

Finally, we study the union operation and obtain the same tight bound as for intersection. For the upper bound we can again use the usual cross-product construction as in the proof of Theorem 5.

Theorem 7 *For any integers $m, n \geq 1$ let A be an m-state and B be an n-state DUFA. Then, $m \cdot n$ states are sufficient for a DUFA to accept the language $L(A) \cup L(B)$.*

Theorem 8 *There are infinitely many integers $m, n \geq 1$ with $n = 2m - 2$ such that A is an m-state DUFA, B is an n-state DUFA, and $m \cdot n$ states are necessary for any DUFA to accept the language $L(A) \cup L(B)$.*

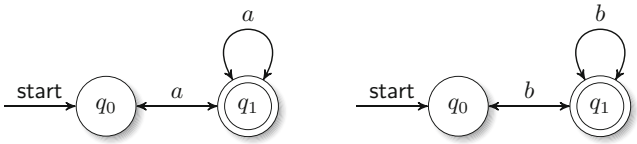
Proof For the lower bound, we consider two DUFAs A' and B' accepting the complements of the languages accepted by A and B used in the proof of Theorem 6. Both DUFAs can be obtained due to Theorem 4 by interchanging accepting and non-accepting states. The resulting DUFAs are minimal, have m and n states, respectively, $L(A') = \overline{L(A)}$ and $L(B') = \overline{L(B)}$. Now, let C' be a minimal DUFA accepting $L(A') \cup L(B')$ and assume that C' has $\ell < m \cdot n$ states. By applying Theorem 4, we can then construct an ℓ-state DUFA C such that $L(C) = \overline{L(C')} = \overline{L(A') \cup L(B')} = \overline{L(A')} \cap \overline{L(B')} = L(A) \cap L(B)$. This is a contradiction to the proof of Theorem 6 where it is shown that every DUFA accepting $L(A) \cap L(B)$ needs at least $m \cdot n$ states. □

5 Nondeterministic operational state complexity

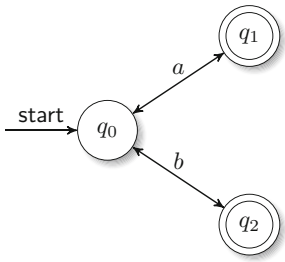
In this section, we complement the operational state complexity results by investigating NUFAs. As in the deterministic case, we first give evidence under which operations the family of languages accepted by NUFAs is *not* closed. By Theorem 2, unary languages from this family are of one of the six types mentioned in the theorem.

To show the non-closure under union, it is sufficient to consider language $L_2 = \{\lambda\}$ and $L_{4,3} = \{ a^n \mid n \geq 3 \text{ and } n \text{ odd} \}$. Their union is not of type 1 or 2 (of Theorem 2). Assume that the union is of one of the other types. Then k must be 0. So, either a or aa or both do belong to the language as well. However, both words do not belong to the union. The same reasoning applies to language $L_{4,3}^*$, which shows the non-closure under iteration. We turn to complementation. The complement of language $L_{3,4} = \{ a^n \mid n \geq 4 \text{ and } n \text{ even} \}$ includes the words a^i for $0 \leq i \leq 3$. Therefore, $\overline{L_{3,4}}$ is not of type 1, 2, 3, or 4. Assume that it is of type 5 or 6. Then, k must be 0 and l must be 1 which implies that the language is a^*, a contradiction.

In the non-unary case, we have additionally the non-closures under concatenation and reversal. As in the deterministic case, for the sake of completeness, we give next non-unary examples witnessing some non-closures. Let A and B be the following NUFAs:

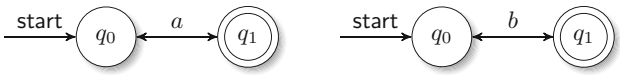


We have $L(A) = a^+$ and $L(B) = b^+$. So, an NUFA accepting $L(A) \cup L(B)$ must have either the following state sub-graph or the state sub-graph obtained by merging states q_1 and q_2.

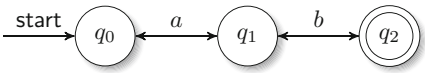


So, the word aab is accepted but neither belongs to $L(A)$ nor to $L(B)$. This shows the non-closure under union. It will turn out that the family of languages accepted by NUFAs is closed under intersection. So, by its non-closure under union and De Morgan's law, it follows that it is not closed under complementation.

The non-closure under concatenation is witnessed by the following NUFAs A and B:

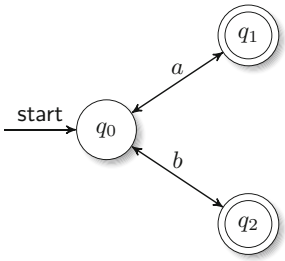


The shortest word in $L(A) \cdot L(B)$ is ab. So, an NUFA accepting $L(A) \cdot L(B)$ must have the following state sub-graph:



Therefore, the word $abbaab$ is accepted but cannot be factorized as uv with $u \in L(A)$ and $v \in L(B)$. This shows the non-closure under concatenation. Let this state sub-graph be the state graph of an NUFA A. We consider the iteration $L(A)^*$. Since $\lambda \in L(A)^*$, the initial state of an NUFA accepting $L(A)^*$ is accepting. Since $ab \in L(A)^*$, an a-transition from the initial state is defined. Therefore, $aa \notin L(A)^*$ is accepted as well. This shows the non-closure under iteration.

Finally, we consider the operation reversal. The non-closure under reversal is witnessed by the following NUFA A:



Assume that there is an NUFA B that accepts the reversal $L(A)^R$. Since $a \in L(A)^R$ and $b \in L(A)^R$, there is an a-transition and a b-transition from B's initial state to some accepting state. So, due to the symmetry, B accepts aab as well. However, the word baa does not belong to $L(A)$ and, thus, the non-closure under reversal follows.

We continue with the operational state complexity of unary NUFAs. Owing to the characterization given in Theorem 2, we can show tight bounds by a detailed case analysis. If the underlying alphabet is at least binary, then NUFAs are only closed under intersection. However, it is possible to obtain tight bounds as well. We start with the unary case.

Theorem 9 *For any integers $m, n \geq 1$ let A be an m-state and B be an n-state NUFA over the same unary alphabet. Then,* $\max(m, n) + 1$ *states are sufficient for an NUFA to accept $L(A) \cap L(B)$.*

Moreover, there are minimal m-state NUFAs and minimal n-state NUFAs such that this sufficient number of states is necessary.

Proof According to Theorem 2, $L(A)$ and $L(B)$ belong to one of the types L_1, L_2, $L_{3,k}$, $L_{4,k}$, $L_{5,k,l}$, or $L_{6,k,l}$ for some $k \geq 0$ and $l > k$. Hence, we have to show that each combination leads to an NUFA with at most $\max(m, n)+1$ states. The results of all combinations are summarized in the following table, where $k = \max(k_1, k_2), l = \max(l_1, l_2), L_{7,k_1,l_1,k_2,l_2} = L_{5,\max(k_1,l_2),l_1}$, if $l_1 > l_2$, $L_{7,k_1,l_1,k_2,l_2} = L_{6,\max(k_2,l_1),l_2}$, if $l_1 \leq l_2$, and $L_{8,k_2} = L_2$, if $k_2 = 0$, and $L_{8,k_2} = L_1$ otherwise.

$\cap$	L_1	L_2	L_{3,k_1}	L_{4,k_1}	L_{5,k_1,l_1}	L_{6,k_1,l_1}
L_{6,k_2,l_2}	L_1	L_1	$L_{3,\max(k_1,l_2)}$	$L_{4,k}$	L_{7,k_1,l_1,k_2,l_2}	$L_{6,k,l}$
L_{5,k_2,l_2}	L_1	L_{8,k_2}	$L_{3,k}$	$L_{4,\max(k_1,l_2)}$	$L_{5,k,l}$	L_{7,k_2,l_2,k_1,l_1}
L_{4,k_2}	L_1	L_1	L_1	$L_{4,k}$	$L_{4,\max(k_2,l_1)}$	$L_{4,k}$
L_{3,k_2}	L_1	L_{8,k_2}	$L_{3,k}$	L_1	$L_{3,k}$	$L_{3,\max(l_1,k_2)}$
L_2	L_1	L_2	L_{8,k_1}	L_1	L_{8,k_1}	L_1
L_1	L_1	L_1	L_1	L_1	L_1	L_1

We will not consider all combinations in detail, but exemplarily discuss two intersections. First, we consider $L_{3,k_1} \cap L_{6,k_2,l_2}$ which is

$$\{a^n \mid n \geq k_1 \text{ and } n \text{ even } \} \cap \{a^n \mid n \geq k_2 \text{ and } n \text{ odd for } k_2 \leq n < l_2\} = \{a^n \mid n \geq k_1 \text{ and } n \text{ even and } n \geq l_2\} = L_{3,\max(k_1,l_2)}.$$

Second, we consider $L_{5,k_1,l_1} \cap L_{6,k_2,l_2}$ where necessarily l_1 is odd and l_2 is even. If $l_1 > l_2$,

$$\begin{aligned} L_{5,k_1,l_1} \cap L_{6,k_2,l_2} &= \{a^n \mid n \geq k_1 \text{ and } n \text{ even for } k_1 \leq n < l_1\} \cap \{a^n \mid n \geq l_2\} \\ &= \{a^n \mid n \geq \max(k_1, l_2) \text{ and } n \text{ even for } k_1 \leq n < l_1\} \\ &= L_{5,\max(k_1,l_2),l_1}. \end{aligned}$$

Otherwise, if $l_1 < l_2$,

$$\begin{aligned} L_{5,k_1,l_1} \cap L_{6,k_2,l_2} &= \{a^n \mid n \geq l_1\} \cap \{a^n \mid n \geq k_2 \text{ and } n \text{ odd for } k_2 \leq n < l_2\} \\ &= \{a^n \mid n \geq \max(k_2, l_1) \text{ and } n \text{ odd for } k_2 \leq n < l_2\} \\ &= L_{6,\max(k_2,l_1),l_2}. \end{aligned}$$

For the upper bound, we can read off the table the following cases. If L_1, L_2, or $L_{8,t}$ is obtained, we know that one state is sufficient to accept the languages. Hence, $1 \leq \max(m, n)+1$ is an upper bound. If $L_{3,t}$ or $L_{4,t}$ is obtained, we know that $t + 1$ states are sufficient to accept the languages. Now, we have $t \in \{k_1, k_2, l_1, l_2\}$. Since $k_1 < l_1 = m$ and $k_2 < l_2 = n$, we obtain that $k_1 + 1 \leq m - 1 + 1 \leq \max(m, n) + 1$, $k_2 + 1 \leq n - 1 + 1 \leq \max(m, n) + 1$, $l_1 + 1 = m + 1 \leq \max(m, n) + 1$, and $l_2 + 1 = n + 1 \leq \max(m, n) + 1$. Thus, $\max(m, n) + 1$ is an upper bound.

Finally, if L_{5,t_1,t_2} or L_{6,t_1,t_2} is obtained, we know that t_2 states are sufficient for an NUFA to accept the languages. Now, we have $t_2 \in \{l_1, l_2, \max(l_1, l_2)\}$. Since $l_1 = m$ and $l_2 = n$, we obtain that $l_1 = m \leq \max(m, n) + 1$, $l_2 = n \leq \max(m, n) + 1$, and $\max(l_1, l_2) = \max(m, n) \leq \max(m, n) + 1$. Thus, $\max(m, n) + 1$ is an upper bound also for these cases.

For the lower bound, we first consider $L_{3,k_1} \cap L_{6,k_2,l_2} = L_{3,\max(k_1,l_2)}$. We know that L_{3,k_1} is accepted with $k_1 + 1 = m$ states due to Lemma 2 and L_{6,k_2,l_2} is accepted with $l_2 = n$ states due to Lemma 3. Moreover, to accept $L_{3,\max(k_1,l_2)}$ at least $\max(k_1, l_2) + 1 = \max(m - 1, n) + 1$ states are necessary due to Lemma 2. Second, consider the symmetric case $L_{6,k_1,l_1} \cap L_{3,k_2} = L_{3,\max(l_1,k_2)}$ with $m = l_1$ and $n = k_2 + 1$. Then $\max(m, n - 1) + 1$ states are necessary to accept $L_{3,\max(l_1,k_2)}$. Altogether,

$$\max(\max(m - 1, n) + 1, \max(m, n - 1) + 1) = \max(m, n) + 1$$

is the lower bound. □

Theorem 10 *For any integers $m, n \geq 1$ let A be an m-state and B be an n-state NUFA over the same unary alphabet. Then, $m + n - 1$ states are sufficient for an NUFA to accept $L(A) \cdot L(B)$.*

Moreover, there are minimal m-state NUFAs and minimal n-state NUFAs such that this sufficient number of states is necessary.

Proof We show that each combination leads to an NUFA with no more than $m+n-1$ states. The results of all combinations are summarized in the following table, where $k = k_1 + k_2$ and $l = k_1 + k_2 + \min(l_1 - k_1, l_2 - k_2)$.

$\cdot$	L_1	L_2	L_{3,k_1}	L_{4,k_1}	L_{5,k_1,l_1}	L_{6,k_1,l_1}
L_{6,k_2,l_2}	L_1	L_{6,k_2,l_2}	L_{6,k,k_1+l_2}	L_{5,k,k_1+l_2}	$L_{6,k,l}$	$L_{5,k,l}$
L_{5,k_2,l_2}	L_1	L_{5,k_2,l_2}	L_{5,k,k_1+l_2}	L_{6,k,k_1+l_2}	$L_{5,k,l}$	$L_{6,k,l}$
L_{4,k_2}	L_1	L_{4,k_2}	$L_{4,k}$	$L_{3,k}$	L_{6,k,l_1+k_2}	L_{5,k,l_1+k_2}
L_{3,k_2}	L_1	L_{3,k_2}	$L_{3,k}$	$L_{4,k}$	L_{5,k,l_1+k_2}	L_{6,k,l_1+k_2}
L_2	L_1	L_2	L_{3,k_1}	L_{4,k_1}	L_{5,k_1,l_1}	L_{6,k_1,l_1}
L_1	L_1	L_1	L_1	L_1	L_1	L_1

We show exemplarily that $L_{5,k_1,l_1} \cdot L_{6,k_2,l_2} = L_{6,k,l}$, where necessarily l_1 is odd and l_2 is even. All words in $L_{5,k_1,l_1} \cdot L_{6,k_2,l_2}$ have a length of at least $k_1 + k_2$. Let $t_1 = l_1 - k_1$ and $t_2 = l_2 - k_2$. If $t_1 < t_2$, then all words of length at least $l_1 + k_2$ belong to the concatenation and all words of length at least $k_1 + k_2$ and less than $l_1 + k_2$ have to have odd lengths, since $k_1 + k_2$ is odd. If $t_2 < t_1$, then all words of length at least $k_1 + l_2$ belong to the concatenation and all words of length at least $k_1 + k_2$ and less than $k_1 + l_2$ have to have odd lengths, since $k_1 + k_2$ is odd. Hence, the concatenation yields $L_{6,k,l}$ by setting $k = k_1 + k_2$ and $l = k_1 + k_2 + \min(t_1, t_2) = k_1 + k_2 + \min(l_1 - k_1, l_2 - k_2)$.

For the upper bound, we can read off the table the following cases. If L_1 or L_2 is obtained, we know that one state is sufficient to accept the languages. Hence, $1 \leq m + n - 1$ is an upper bound. If $L_{3,t}$ or $L_{4,t}$ is obtained, we know that $t + 1$ states are sufficient to accept the languages. Now, we have $t \in \{k_1, k_2, k_1 + k_2\}$. Since $m = k_1 + 1$ and $n = k_2 + 1$, we obtain that $k_1 + 1 = m - 1 + 1 \leq m + n - 1$, $k_2 + 1 = n - 1 + 1 \leq m + n - 1$, and $k_1 + k_2 + 1 = m - 1 + n - 1 + 1 = m + n - 1$. Thus, $m + n - 1$ is an upper bound. Finally, if L_{5,t_1,t_2} or L_{6,t_1,t_2} is obtained, we know that t_2 states are sufficient to accept the languages. Now, we have $t_2 \in \{l_1, k_2 + l_1, l_2, k_1 + l_2\}$. Since $k_1 < l_1 = m$ and $k_2 < l_2 = n$, we obtain that $l_1 = m \leq m + n - 1$, $k_2 + l_1 \leq m + n - 1$, $l_2 \leq n \leq m + n - 1$, and $k_1 + l_2 \leq m + n - 1$. Thus, $m + n - 1$ is an upper bound also for these cases as well.

For the lower bound, we consider $L_{3,k_1} \cdot L_{5,k_2,l_2} = L_{5,k_1+k_2,k_1+l_2}$. We know that L_{3,k_1} is accepted with $k_1 + 1 = m$ states owing to Lemma 2 and L_{5,k_2,l_2} is accepted with $l_2 = n$ states owing to Lemma 3. Moreover, to accept L_{5,k_1+k_2,k_1+l_2} at least $k_1 + l_2 = m - 1 + n$ states are necessary due to Lemma 2. □

Since the reversal of a unary language L is the language L again, the following statement is obvious.

Theorem 11 *Let $n \geq 1$ be an integer. Then, n states are sufficient for an NUFA to accept the reversal of any language accepted by an n-state NUFA.*

Moreover, for any minimal n-state NUFA A, any NUFA accepting the language $L(A)^R$ has at least n states.

Finally, we consider the general intersection operation for NUFAs.

Theorem 12 *For any integers $m, n \geq 1$ let A be an m-state and B be an n-state NUFA. Then, $m \cdot n$ states are sufficient for an NUFA to accept the language $L(A) \cap L(B)$.*

Moreover, there are minimal m-state NUFAs and minimal n-state NUFAs such that this sufficient number of states is necessary.

Proof For the upper bound, we can use the usual cross-product construction as in the proof of Theorem 5. Clearly, the resulting automaton is an $(m \cdot n)$-state NUFA that accepts $L(A) \cap L(B)$.

For the lower bound, we consider $L_m = \{ w \in \{a, b\}^* \mid |w|_a \geq m - 1 \}$ and $L'_n = \{ w \in \{a, b\}^* \mid |w|_b \geq n - 1 \}$ that are each accepted by an m-state NUFA and n-state NUFA, respectively. Next, we show that any NUFA for $L_m \cap L'_n$ needs at least $m \cdot n$ states. By way of contradiction, we assume that there is an NUFA $C = \langle Q, \{a, b\}, \delta, q_0, F \rangle$ accepting $L_m \cap L'_n$ with less than $m \cdot n$ states. Then, consider $a^{i_1}b^{j_1}$ and $a^{i_2}b^{j_2}$ such that $a^{i_1}b^{j_1} \neq a^{i_2}b^{j_2}$ for $0 \leq i_1, i_2 \leq m - 1$ and $0 \leq j_1, j_2 \leq n - 1$. Without loss of generality, we may assume that $i_1 > i_2$ or $j_1 > j_2$. Since there are $m \cdot n$ such words and $|Q| < m \cdot n$, we can conclude that the intersection

$$\{ q \in Q \mid q \in \delta(q_0, a^{i_1}b^{j_1}) \text{ and } \delta(q, a^{m-1-i_1}b^{n-1-j_1}) \cap F \neq \emptyset \} \cap \delta(q_0, a^{i_2}b^{j_2})$$

is not empty for at least two words $a^{i_1}b^{j_1}$ and $a^{i_2}b^{j_2}$. Thus, we obtain for $w = a^{i_2}b^{j_2}a^{m-1-i_1}b^{n-1-j_1}$ that $\delta(q_0, w) \cap F \neq \emptyset$, hence, $w \in L_m \cap L'_n$. Now, if $i_1 > i_2$, we obtain that $|w|_a = i_2 + m - 1 - i_1 < m - 1$. On the other hand, if $j_1 > j_2$, we obtain that $|w|_b = j_2 + n - 1 - j_1 < n - 1$. In both cases, we conclude that $w \notin L_m \cap L'_n$ which gives the contradiction and concludes the proof of the lower bound. □

The results on the deterministic and nondeterministic state complexity obtained are summarized in Table 1.

6 Conclusions

In this paper, we have introduced deterministic and nondeterministic finite automata with undirected state graphs. It was possible to characterize the languages accepted by such automata in the unary case for deterministic and nondeterministic automata as well as in the binary case for deterministic automata. This characterization enabled us to study the deterministic and nondeterministic operational state complexity in depth and an almost complete picture with tight bounds could be obtained. The deterministic state complexity of the reversal operation as well as the question of whether the language class is closed under reversal are currently open questions. For DUFAs with one accepting state, the construction of a DUFA accepting the reversal is straightforward and needs no additional states. However, the construction cannot directly be generalized to DUFAs with more than one accepting state.

Concerning the operational state complexity, we have investigated so far only those operations under which the corresponding language classes are closed. However, even in the case of non-closure, we still obtain regular languages. Thus, it would clearly be of interest to determine upper and lower bounds for the remaining operations from this point of view.

Since the language classes accepted by DUFAs and NUFAs are subregular, it would be interesting to devise an effective algorithm that decides, given an arbitrary finite automaton A, whether or not $L(A)$ could be accepted by an NUFA or DUFA as well. Naturally, the complexity of such an algorithm is of great interest. In the positive case, such an algorithm should construct an equivalent NUFA or DUFA. Then, the determination of the exact trade-

off concerning the number of states between NUFAs and DUFAs as well as arbitrary NFAs and DFAs becomes a challenging task.

Acknowledgements The authors wish to thank the anonymous referees for useful and kind comments.

Funding Open Access funding enabled and organized by Projekt DEAL.

References

1. Bordihn, H., Holzer, M., Kutrib, M.: Determinization of finite automata accepting subregular languages. Theoret. Comput. Sci. **410**, 3209–3222 (2009)
2. Brüggemann-Klein, A., Wood, D.: One-unambiguous regular languages. Inform. Comput. **140**, 229–253 (1998)
3. Gajardo, A., Kari, J., Moreira, A.: On time-symmetry in cellular automata. J. Comput. Syst. Sci. **78**, 1115–1126 (2012)
4. Gao, Y., Moreira, N., Reis, R., Yu, S.: A survey on operational state complexity. J. Autom. Lang. Comb. **21**, 251–310 (2016)
5. Havel, I.M.: The theory of regular events I. Kybernetica **5**, 400–419 (1969)
6. Havel, I.M.: The theory of regular events II. Kybernetica **6**, 520–544 (1969)
7. Holzer, M., Jakobi, S., Kutrib, M.: Minimal reversible deterministic finite automata. Int. J. Found. Comput. Sci. **29**, 251–270 (2018)
8. Holzer, M., Kutrib, M.: Reversible nondeterministic finite automata. In: Phillips, I., Rahaman, H. (eds.) Reversible Computation (RC 2017). LNCS, vol. 10301, pp. 35–51. Springer, Berlin (2017)
9. Kutrib, M.: Reversible and irreversible computations of deterministic finite-state devices. In: Italiano, G.F., Pighizzini, G., Sannella, D. (eds.) Mathematical Foundations of Computer Science (MFCS 2015). LNCS, vol. 9234, pp. 38–52. Springer, Berlin (2015)
10. Kutrib, M., Malcher, A., Schneider, C.: Finite automata with undirected state graphs. In: Konstantinidis, S., Pighizzini, G. (eds.) Descriptional Complexity of Formal Systems (DCFS 2018). LNCS, vol. 10952, pp. 212–223. Springer, Berlin (2018)
11. Kutrib, M., Worsch, T.: Time-symmetric machines. In: Dueck, G.W., Miller, D.M. (eds.) Reversible Computation (RC 2013), LNCS, vol. 7948, pp. 168–181. Springer, Berlin (2013)
12. Landauer, R.: Irreversibility and heat generation in the computing process. IBM J. Res. Dev. **5**, 183–191 (1961)
13. Lavado, G.J., Pighizzini, G., Prigioniero, L.: Weakly and strongly irreversible regular languages. In: Csuhaj-Varjú, E., Dömösi, P., Vaszil, G. (eds.) Automata and Formal Languages (AFL 2017). EPTCS, vol. 252, pp. 143–156 (2017)
14. Yu, S.: Regular languages. In: Rozenberg, G., Salomaa, A. (eds.) Handbook of Formal Languages, vol. 1, Chapter 2, pp. 41–110. Springer, Berlin (1997)
15. Yu, S.: State complexity of regular languages. J. Autom. Lang. Comb. **6**, 221–234 (2001)