Finite Automata with Undirected State Graphs

Martin Kutrib, Andreas Malcher, Christian Schneider

To cite this version:

Martin Kutrib, Andreas Malcher, Christian Schneider. Finite Automata with Undirected State Graphs. 20th International Conference on Descriptional Complexity of Formal Systems (DCFS), Jul 2018, Halifax, NS, Canada. pp.212-223, 10.1007/978-3-319-94631-3_18. hal-01905634

HAL Id: hal-01905634

https://hal.inria.fr/hal-01905634

Submitted on 26 Oct 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons `Attribution| 4.0 International License`

Finite Automata with Undirected State Graphs

Martin Kutrib, Andreas Malcher, and Christian Schneider

Institut für Informatik, Universität Giessen
Arndtstr. 2, 35392 Giessen, Germany
{kutrib,andreas.malcher}@informatik.uni-giessen.de
Christian.Schneider@math.uni-giessen.de

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 [9, 10] 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. A recent survey of the several branches and details can be found in [2], which is also a valuable and comprehensive source of references. It seems that the recent studies of operational state complexity focus on subregular languages. 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 [7] strongly suggests to study computations without loss of information. Recent results on reversible finite automata can be found, for example, in [3–5, 8].

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 [1,6]. Time-symmetry appears in physical reality 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 Section 3. These characterizations are a powerful tool to derive tight bounds on the operational state complexity of deterministic (Section 4) and nondeterministic (Section 5) undirected finite automata. All bounds obtained are summarized in Table 1. Finally, in Section 6 we discuss open and untouched problems for future work. We remark that some proofs are omitted due to space constraints.

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.

A *nondeterministic finite automaton* (NFA) is a system $M = \langle Q, \Sigma, \delta, q_0, F\rangle$, 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* (NUDFA) and *deterministic undirected finite automata* (DUDFA). 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 $\delta\colon Q \times \Sigma^* \to 2^Q$.

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

Example 1. The NUDFA $M = \langle \{q_0, q_1, q_3\}, \{a, b\}, \delta, q_0, \{q_2\} \rangle$ whose transition function is given through the state graph depicted in Figure 1 accepts the language $\{\, w \in \{a, b\}^+ \mid |w|_a \bmod 2 = 0 \,\}$. ■

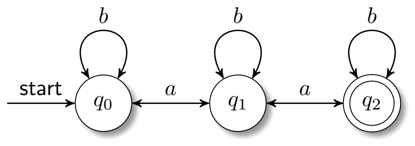

Fig. 1. State graph of an nondeterministic undirected finite automaton accepting the language $\{\, w \in \{a, b\}^+ \mid |w|_a \bmod 2 = 0 \,\}$.

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. So, there are only a very few different types of *minimal* deterministic undirected finite automata with a unary or binary input alphabet. The situation for nondeterministic devices is still rather restricted, but there are more types distinguishable. 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 Nondeterministic Undirected Finite Automata

We start with unary NUDFAs.

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

1. $L_1 = \emptyset$,
2. $L_2 = \{\lambda\}$,

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

Proof. Let $M = \langle Q, \{a\}, \delta, q_0, F\rangle$ be an NUDFA 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 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 $L(M) = L_{3,k} = \{ a^n \mid n \geq k$ and n even$\}$. Similarly, if k is odd, we have $L(M) = L_{4,k} = \{ a^n \mid n \geq k$ and n 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 $L(M) = L_{5,k,l} = \{ a^n \mid n \geq k$ and n even for $k \leq n < l\}$ if k is even, and $L(M) = L_{6,k,l} = \{ a^n \mid n \geq k$ and n odd for $k \leq n < l\}$ if k is odd. □

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 NUDFA to accept the languages of different type.

Lemma 3. *Let M be a unary NUDFA that accepts a language of type L_1 or L_2. Then one state is sufficient and necessary for M.*

Lemma 4. *Let $k \geq 1$ and M be a unary NUDFA 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 NUDFA depicted in Figure 2 accepts $L_{3,k}$ or $L_{4,k}$ with $k+1$ states. □

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

The NUDFA depicted in Figure 3 accepts $L_{5,k,l}$ or $L_{6,k,l}$ with l states.

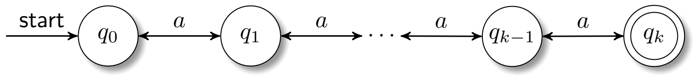

Fig. 2. State graph of a minimal nondeterministic undirected finite automaton accepting the language $L_{3,k}$ or $L_{4,k}$.

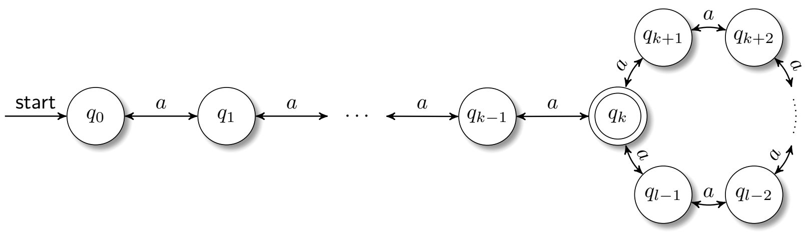

Fig. 3. State graph of a minimal nondeterministic undirected finite automaton accepting the language $L_{5,k,l}$ or $L_{6,k,l}$.

3.2 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 Figure 4. Any of the states can be made accepting or rejecting which yields four different languages that are accepted by unary DUDFAs.

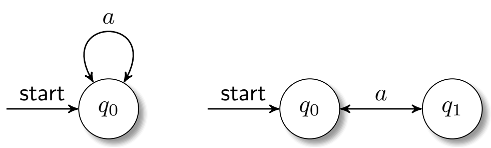

Fig. 4. Possible state graphs of unary deterministic undirected finite automata.

Corollary 6. *Let M be a unary DUDFA. Then language $L(M)$ is one of the following:*

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

While the family of binary languages accepted by NUDFAs is fairly rich, in the following the types of possible state graphs of deterministic undirected finite automata accepting binary languages are analyzed.

Theorem 7. *Let M be a minimal binary DUDFA. Then its state graph is of one of the five types depicted in Figure 5, where $m, n \geq 1$, the letters a and b are interchangeable, $x, y \in \{a, b\}$, $\bar{x} = a \iff x = b$, and $\bar{y} = a \iff y = b$.*

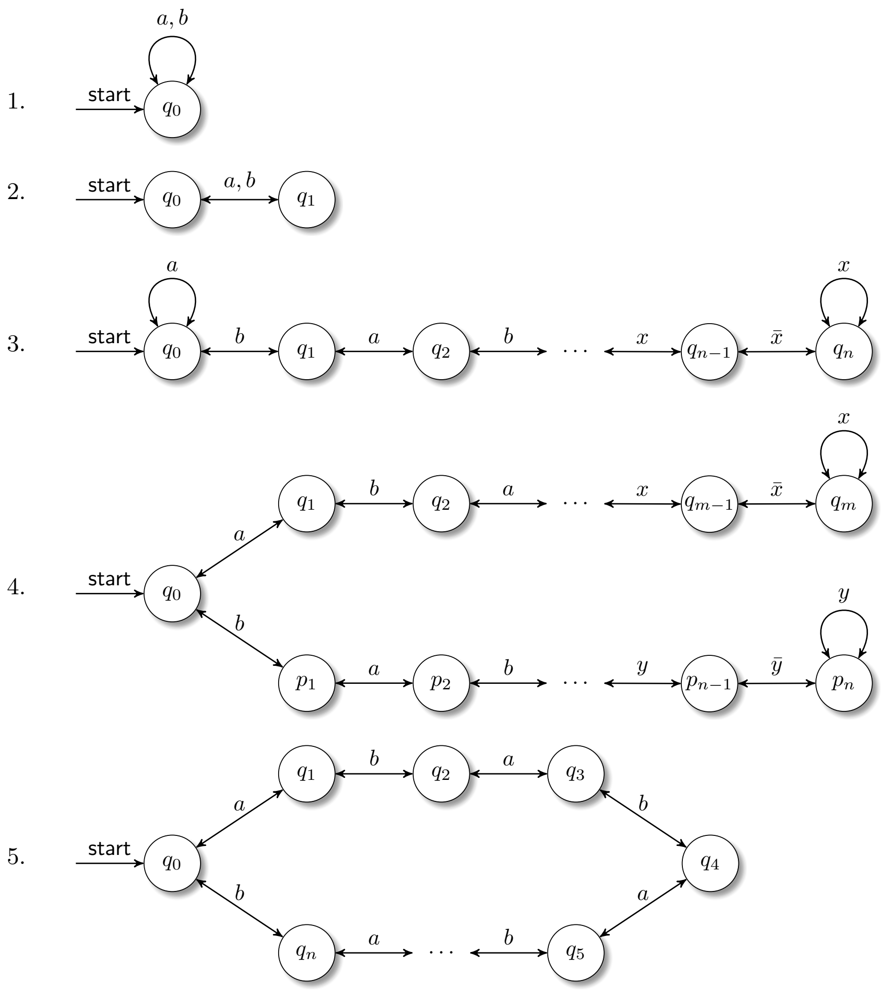

Fig. 5. Possible state graphs of binary deterministic undirected finite automata, where $m, n \geq 1$, the letters a and b are interchangeable, $x, y \in \{a, b\}$, $\bar{x} = a \iff x = b$, and $\bar{y} = a \iff y = b$.

4 Deterministic State Complexity

In this section, we investigate the state complexity of DUDFAs. We start with such automata on unary alphabets. It has been shown in Corollary 6 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 of two states for each of the operations. In the second part of the section, we investigate the state complexity of the Boolean operations for DUDFAs over arbitrary alphabets and can establish tight bounds as well.

Theorem 8. *For any integers $m, n \geq 1$ let A be an m-state and B be an n-state DUDFA over the same unary alphabet. Then two states are sufficient and necessary in the worst case for a DUDFA to accept $\overline{L(A)}$, $L(A) \cap L(B)$, $L(A) \cup L(B)$, or $L(A)^R$.*

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

Theorem 9. *For any integer $n \geq 1$ let A be an n-state DUDFA. Then n states are sufficient and necessary in the worst case for a DUDFA to accept the language $\overline{L(A)}$.*

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 10. *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, both equations take on the minimal value for x_1, x_2, y_1, and y_2.*

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

Theorem 11. *For any integers $m, n \geq 1$ let A be an m-state and B be an n-state DUDFA. Then $m \cdot n$ states are sufficient for a DUDFA 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 DUDFA and $B = \langle Q_B, \Sigma, \delta_B, q_{0,B}, F_B\rangle$ be an n-state DUDFA. 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 DUDFA that accepts $L(A) \cap L(B)$. □

Theorem 12. *There are infinitely many integers $m, n \geq 1$ with $n = 2m - 2$ such that A is an m-state DUDFA, B is an n-state DUDFA, and $m \cdot n$ states are necessary for any DUDFA to accept the language $L(A) \cap L(B)$.*

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

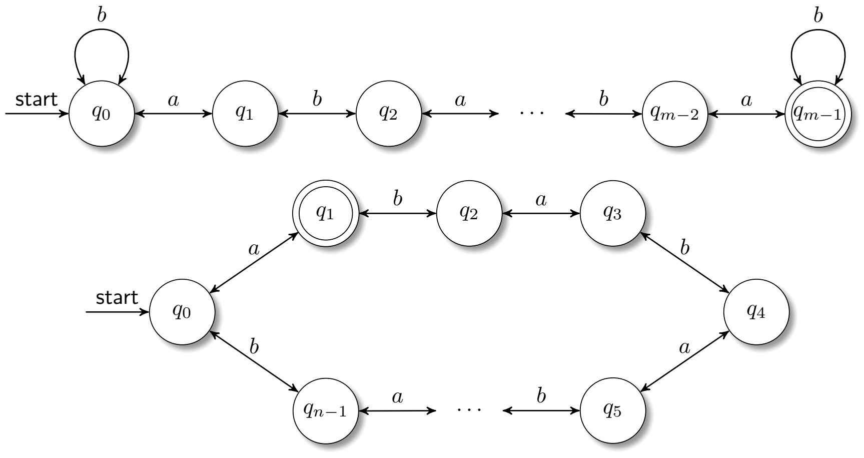

Fig. 6. The DUDFA A (upper automaton) and B (lower automaton) used for proof of the lower bound.

Now, let C be a minimal DUDFA 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 Equation (1) indicates the length of such words that are accepted by A, whereas the right hand side of Equation (1) indicates the length of such words that are accepted by B. According to Lemma 10 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 Equations (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 Equation (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 7, that C is of 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. $\square$

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 11.

Theorem 13. *For any integers $m, n \geq 1$ let A be an m-state and B be an n-state DUDFA. Then $m \cdot n$ states are sufficient for a DUDFA to accept the language $L(A) \cup L(B)$.*

Theorem 14. *There are infinitely many integers $m, n \geq 1$ with $n = 2m - 2$ such that A is an m-state DUDFA, B is an n-state DUDFA, and $m \cdot n$ states are necessary for any DUDFA to accept the language $L(A) \cup L(B)$.*

Proof. For the lower bound we consider two DUDFAs A' and B' accepting the complements of the languages accepted by A and B used in the proof of Theorem 12. Both DUDFAs can be obtained due to Theorem 9 by interchanging accepting and non-accepting states. The resulting DUDFAs 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 DUDFA accepting $L(A') \cup L(B')$ and assume that C' has $\ell < m \cdot n$ states. By applying Theorem 9 we can then construct an ℓ-state DUDFA 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 12 where it is shown that every DUDFA accepting $L(A) \cap L(B)$ needs at least $m \cdot n$ states. $\square$

5 Nondeterministic State Complexity

In this section, we complement the state complexity results by investigating NUDFAs. We start again with such automata on unary alphabets. 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 NUDFAs are only closed under intersection. However, it is possible to obtain tight bounds as well. We start with the unary case.

Theorem 15. *For any integers $m, n \geq 1$ let A be an m-state and B be an n-state NUDFA over the same unary alphabet. Then $\max(m,n)+1$ states are sufficient and necessary in the worst case for an NUDFA to accept $L(A) \cap L(B)$.*

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 NUDFA 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

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

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 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 as well.

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 owing to Lemma 4 and L_{6,k_2,l_2} is accepted with $l_2 = n$ states owing to Lemma 5. 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 4. 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 16. *For any integers $m, n \geq 1$ let A be an m-state and B be an n-state NUDFA over the same unary alphabet. Then $m+n-1$ states are sufficient and necessary in the worst case for an NUDFA to accept $L(A) \cdot L(B)$.*

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

Theorem 17. *For any integer $m \geq 1$, let A be an m-state unary NUDFA. Then m states are sufficient and necessary in the worst case for an NUDFA to accept $L(A)^R$.*

Finally, we consider the general intersection operation for NUDFAs.

Theorem 18. *For any integers $m, n \geq 1$ let A be an m-state and B be an n-state NUDFA. Then $m \cdot n$ states are sufficient and necessary in the worst case for an NUDFA to accept the language $L(A) \cap L(B)$.*

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

	unary DUDFA	DUDFA	unary NUDFA	NUDFA
$L_1 \cup L_2$	2	mn	—	—
$L_1 \cap L_2$	2	mn	$\max(m, n)+1$	mn
$\overline{L}$	2	m	—	—
L_1L_2	—	—	$m+n-1$	—
L^*	—	—	—	—
L^R	2	?	m	—

Table 1. Summary of the state complexities of the operations studied in this paper. It is marked by — if an automata class is not closed under the corresponding operation.

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 DUDFAs with one accepting state the construction of a DUDFA accepting the reversal is straightforward and needs no additional states. However, the construction cannot directly be generalized to DUDFAs 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 DUDFAs and NUDFAs 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 NUDFA or DUDFA as well. In the positive case, such an algorithm should construct an equivalent NUDFA or DUDFA. Then the determination of the exact trade-off concerning the number of states between NUDFAs and DUDFAs as well as arbitrary NFAs and DFAs becomes a challenging task.

References

1. Gajardo, A., Kari, J., Moreira, A.: On time-symmetry in cellular automata. J. Comput. System Sci. 78, 1115–1126 (2012)
2. Gao, Y., Moreira, N., Reis, R., Yu, S.: A survey on operational state complexity. J. Autom. Lang. Comb. 21, 251–310 (2016)
3. Holzer, M., Jakobi, S., Kutrib, M.: Minimal reversible deterministic finite automata. In: Potapov, I. (ed.) Developments in Language Theory (DLT 2015). LNCS, vol. 9168, pp. 276–287. Springer (2015)
4. 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 (2017)
5. 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 (2015)
6. 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 (2013)
7. Landauer, R.: Irreversibility and heat generation in the computing process. IBM J. Res. Dev. 5, 183–191 (1961)
8. 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)
9. Yu, S.: Regular languages. In: Rozenberg, G., Salomaa, A. (eds.) Handbook of Formal Languages, vol. 1, chap. 2, pp. 41–110. Springer, Berlin (1997)
10. Yu, S.: State complexity of regular languages. J. Autom. Lang. Comb. 6, 221–234 (2001)