State Complexity of Unambiguous Operations on Deterministic Finite Automata

Galina Jirásková, Alexander Okhotin

State complexity of unambiguous operations on deterministic finite automata

Galina Jirásková [1,⋆] and Alexander Okhotin[2,]

[1] Mathematical Institute, Slovak Academy of Sciences
Grešákova 6, 040 01 Košice, Slovak Republic
jiraskov@saske.sk
[2] St. Petersburg State University, 7/9 Universitetskaya nab.,
Saint Petersburg 199034, Russia,
alexander.okhotin@spbu.ru

Abstract. The paper determines the number of states in a deterministic finite automaton (DFA) necessary to represent "unambiguous" variants of the union, concatenation, and Kleene star operations on formal languages. For the disjoint union of languages represented by an m-state and an n-state DFA, the state complexity is $mn - 1$; for the unambiguous concatenation, it is known to be $m2^{n-1} - 2^{n-2}$ (Daley et al., "Orthogonal concatenation: Language equations and state complexity", *J. UCS*, 2010), and this paper shows that this number of states is necessary already over a binary alphabet; for the unambiguous star, the state complexity function is determined to be $\frac{3}{8}2^n + 1$. In the case of a unary alphabet, disjoint union requires up to $\frac{1}{2}mn$ states, unambiguous concatenation has state complexity $m + n - 2$, and unambiguous star requires $n - 2$ states in the worst case.

1 Introduction

The basic operations on formal languages are union, concatenation and Kleene star. The main models for language description, namely, regular expressions and formal grammars, use these operations to express the structure of strings. An important special case of concatenation is *unambiguous concatenation*. Concatenation of two formal languages, K and L, is said to be *unambiguous*, if every string w in KL has a unique representation as $w = uv$, with $u \in K$ and $v \in L$. The union operation also has its unambiguous special case: the *disjoint union*. These two operations are important, in particular, for giving rise to *unambiguous grammars*. One can also define the *unambiguous Kleene star*: the Kleene star of any language L with the property that every string in L^* has a unique decomposition as a concatenation of zero or more strings in L.

This paper investigates the succinctness of description of these three *unambiguous operations* by deterministic finite automata (DFA). The state complexity of union, concatenation and Kleene star in their unrestricted form has long been

⋆ Research supported by VEGA grant 2/0084/15 and grant APVV-15-0091.

known: Maslov [17] was the first to determine their state complexity as mn states for the union, $m2^n - 2^{n-1}$ states for the concatenation and $\frac{3}{4}2^n$ states for the Kleene star, where m and n is the number of states in the DFA recognizing the arguments. These results were further elaborated by Yu et al. [21] and by Jirásek et al. [10]. For nondeterministic finite automata (NFA), the state complexity of these operations was determined by Holzer and Kutrib [9], a similar study for two-way finite automata (2DFA) was carried out by Jirásková and Okhotin [14], whereas Jirásek Jr. et al. [11] determined the state complexity of basic operations for unambiguous finite automata (UFA). In the literature, special consideration was given to the case of the unary alphabet, where state complexity results are different: for DFA, as established by Yu et al. [21] and by Pighizzini and Shallit [19], both union and concatenation require up to mn states for relatively prime m, n, and fewer states for other values of m, n; the star requires $(n-1)^2 + 1$ states. The state complexity of basic operations for unary two-way automata was established by Kunc and Okhotin [15,16], and by Okhotin [18] for unary UFA.

The question addressed in this paper is, do these state complexity results essentially depend on using ambiguity, that is, on taking a union of overlapping languages, on concatenating languages that allow some strings to be obtained in multiple ways, and on taking the star of languages with multiple partitions? If union, concatenation or star is restricted to be unambiguous, will it become substantially easier to express?

In the case of the **disjoint union** operation, investigated in Section 3, it turns out that it can be represented with one state less than the union in general: that is, $mn-1$ states are sufficient to represesent disjoint union of a pair of DFA with m and with n states. It is also proved that this number of states is in the worst case necessary, with witness languages defined over a binary alphabet. For the unary alphabet, the state complexity is $\frac{1}{2}mn$, under certain assumptions on m and n.

For the **unambiguous concatenation**, it is already known from Daley at al. [4] that it requires exactly half as many states as concatenation in general: its state complexity is $m2^{n-1} - 2^{n-2}$. The witness languages of Daley at al. [4] are defined over a four-symbol alphabet. In this paper, new binary witness languages for this operation are constructed in Section 4. In the unary case, the state complexity of this operation is only $m + n - 2$.

Finally, as established in Section 5, representing the **unambiguous star** takes half as many states as star in general, plus one extra state: its state complexity is $\frac{3}{8}2^n + 1$, with witness languages over a binary alphabet. In the unary case, its state complexity is $n - 2$.

2 Basic notions

As usual, a *deterministic finite automaton* (DFA) is defined as a quintuple $A = (\Sigma, Q, q_0, \delta, F)$ where Σ is an input alphabet; Q is a finite non-empty set of states; $q_0 \in Q$ is the initial state; $\delta\colon Q \times \Sigma \to Q$ is the transition function; $F \subseteq Q$ is

the set of accepting states. The computation of A on a string $w = a_1 \dots a_n$, with $a_1, \dots, a_n \in \Sigma$, is the uniquely defined sequence of states $r_0, \dots, r_n \in Q$, in which $r_0 = q_0$, $r_i = \delta(r_{i-1}, a_i)$ for all i. If $r_n \in F$, the DFA is said to *accept* the string w. The language recognized by a DFA, denoted by $L(A)$, is the set of all strings it accepts.

It is convenient to extend the transition function to act on strings in Σ^*, as $\delta\colon Q \times \Sigma^* \to Q$, with $\delta(q, \varepsilon) = q$ and $\delta(q, aw) = \delta(\delta(q, a), w)$. In these terms, $L(A) = \{\, w \mid \delta(q_0, w) \in F \,\}$. Also, the function is sometimes extended to act on sets of states, as $\delta(S, w) = \{\, \delta(q, w) \mid q \in S \,\}$.

A state q of a DFA is called *dead*, if no string is accepted from q. In a minimal DFA, there can be at most one dead state. A state q of a DFA is called *cyclic* if there is a non-empty string v with $\delta(q, v) = q$.

A DFA over a unary alphabet $\Sigma = \{a\}$ is a chain of transitions by a that eventually turns back to one of the previous states. The repetitive part is called the *cycle*, and the earlier states outside of the cycle form the *tail*. Thus, the language is *ultimately periodic*, beginning from ℓ (the length of the tail), with the length of the cycle as the period.

A *nondeterministic finite automaton* (NFA) is a quintuple $A = (\Sigma, Q, Q_0, \delta, F)$, where $Q_0 \subseteq Q$ is the set of possible initial states and the transition function $\delta\colon Q \times \Sigma \to 2^Q$ may define multiple next states. An NFA accepts a string $w = a_1 \dots a_n$ if there exists a sequence of states $r_0, \dots, r_n \in Q$, with $r_0 \in Q_0$, $r_i \in \delta(r_{i-1}, a_i)$ for all i, and $r_n \in F$.

In some literature, NFAs are defined with a unique initial state, that is, with $Q_0 = \{q_0\}$. Every NFA can be converted to an NFA with a unique initial state by adding a new initial state.

An NFA $A = (\Sigma, Q, Q_0, \delta, F)$ can be transformed to an equivalent DFA with states corresponding to subsets of Q. The subset Q_0 is then the initial state of the DFA, its set of final states is $\{S \subseteq Q \mid S \cap F \neq \varnothing\}$, and its transition function $\delta'\colon 2^Q \times \Sigma \to 2^Q$ is defined by $\delta'(S, a) = \bigcup_{q \in S} \delta(q, a)$.

3 Union

In general, the union of an m-state DFA $A = (\Sigma, P, p_0, \gamma, E)$ and an n-state DFA $B = (\Sigma, Q, q_0, \delta, F)$ can be recognized by a DFA with mn states, known as the *direct product DFA*. Its set of states is the set of all pairs $P \times Q$, the initial state is (p_0, q_0), the transition function simulates A on the first component and B on the second component, thus mapping (p, q) by a to $(\gamma(p, a), \delta(q, a))$; and a pair (p, q) is accepting if $p \in E$ or $q \in F$.

For automata without the disjointness restriction, mn states are in the worst case necessary [17]. It turns out that if the union is disjoint, then the construction can be improved by one state.

Lemma 1. *For every m-state DFA A and for every n-state DFA B, with $m, n \geqslant 2$ and with $L(A) \cap L(B) = \varnothing$, there is a DFA with $mn - 1$ states that recognizes the disjoint union $L(A) \cup L(B)$.*

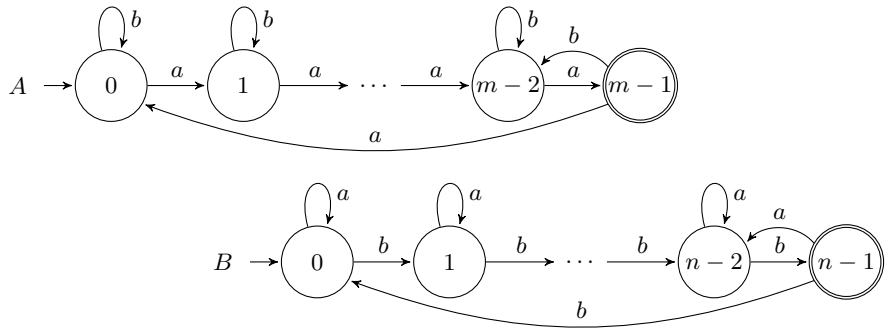

Fig. 1. Binary witnesses for disjoint union meeting the upper bound $mn - 1$.

Indeed, if any pair (p, q), with p accepting in A and q accepting in B, were reachable in the direct product automaton, then the union would not be disjoint. Thus, all such pairs can be excluded from the mn-state construction. Putting aside the trivial case of either automaton having no accepting states, at least one such pair exists.

It remains to show that this number of states, $mn - 1$, is necessary in the worst case.

Lemma 2. *For all $m, n \geqslant 2$, there exist languages $K, L \subseteq \{a, b\}^*$, recognized by an m-state and an n-state DFA, respectively, with $K \cap L = \varnothing$, for which every DFA recognizing $K \cup L$ has at least $mn - 1$ states.*

Proof (a sketch). The desired witness languages are recognized by the pair of automata shown in Fig. 1. Every string in K ends with a, while every string in L ends with b, and therefore, K and L are disjoint. In the direct product automaton, each pair (i, j) is reachable by $a^i b^j$ if $i \leqslant m-2$, and by $a^{m-1} b^j a$ for $i = m - 2$.

For every two pairs (i, j) and (k, ℓ), if $i < k$, then the string a^{m-1-k} is accepted from (i, j), but not from (k, ℓ). The case of $j < \ell$ is symmetric. □

The next theorem summarizes the results of the two lemmata above.

Theorem 3 (Disjoint Union). *Let A and B be an m-state and n-state DFA, respectively, such that $L(A) \cap L(B) = \varnothing$. Then the language $L(A) \cup L(B)$ is recognized by a DFA of at most $mn - 1$ states. This upper bound is tight, and it is met by the binary witness languages recognized by DFAs shown in Fig. 1.* □

The union operation in the case of a unary alphabet still requires all mn states, as long as m and n are relatively prime; the reduction of state complexity in the case when m and n have common divisors was studied by Pighizzini and Shallit [19]. However, if the union is disjoint, then it can be represented using half as many states.

Theorem 4 (Unary Disjoint Union). *Let $m, n \geqslant 4$. Let A be an m-state DFA and B an n-state DFA, both defined over a unary alphabet $\Sigma = \{a\}$, with $L(A) \cap L(B) = \varnothing$. Then the disjoint union $L(A) \cup L(B)$ is recognized by a DFA with at most $\lfloor \frac{1}{2}mn \rfloor$ states. This upper bound is tight whenever m and n are even numbers with $\frac{m}{2}$ and $\frac{n}{2}$ relatively prime, and it is met by the languages $(a^m)^*$ and $a(a^n)^*$.*

Proof. If either of the automata defines a finite language, then the union is recognized by a DFA with $m + n$ states, which cannot exceed $\frac{1}{2}mn$: indeed, $m + n \leqslant 2\max(m, n) \leqslant \frac{1}{2}\min(m, n)\max(m, n) = \frac{1}{2}mn$. Assume that both languages are infinite. Let k be the length of the cycle in A, and let ℓ be the length of the cycle in B.

If the cycle lengths in A and in B are relatively prime, then the union is ambiguous. Let d be the greatest common divisor of their cycle lengths. Then the union $L(A) \cup L(B)$ is recognized by a DFA with a cycle of length $\mathrm{lcm}(k, \ell) = \frac{1}{d}k\ell$, which is at most $\frac{1}{2}k\ell$. In addition, there are $\max(m - k, n - \ell)$ non-cyclic states. Assume that $m - k \geqslant n - \ell$. Then the number of states in the DFA is estimated as follows.

$$\frac{1}{2}k\ell + m - k \leqslant \frac{1}{2}k\ell + \frac{1}{2}(m - k)\ell = \frac{1}{2}m\ell \leqslant \frac{1}{2}mn$$

Now, consider the tightness. Since the first language consists of even-length strings and the other one of odd-length strings, their union is disjoint. Since $\gcd(m, n) = \frac{1}{2}mn$, the union is periodic with this period, and therefore every DFA recognizing the union must have at least this many states. □

4 Concatenation

The state compliexity of unambiguous concatenation has been investigated before, and the following result is known.

Theorem A (Daley et al. [20]) *Let A by an m-state DFA, and B an n-state DFA. If the concatenation $L(A)L(B)$ is unambiguous, then it can be represented by a DFA with $m2^{n-1} - 2^{n-2}$ states. The bound is tight for a four-symbol alphabet.*

This paper improves the tightness result by showing that the upper bound cannot be reduced already for a binary alphabet.

Lemma 5. *Let $m, n \geqslant 3$ and A and B be the DFAs shown in Fig. 2. Then the concatenation $L(A)L(B)$ is unambiguous and every DFA recognizing $L(A)L(B)$ must have at least $m2^{n-1} - 2^{n-2}$ states.*

Proof. The concatenation is unambiguous, because every non-empty string in $L(A)$ ends with an a, whereas every string in $L(B)$ contains exactly $n - 2$ occurrences of a. Thus, the only possible partition of a string into $L(A) \cdot L(B)$ is to split it right after the $(n - 1)$-th last occurrence of a.

Let $P = \{p_0, p_1, \ldots, p_{m-1}\}$ be the state set of the DFA A, and $Q = \{0, 1, \ldots, n - 1\}$ the state set of B. An NFA for $L(A)L(B)$ is constructed by

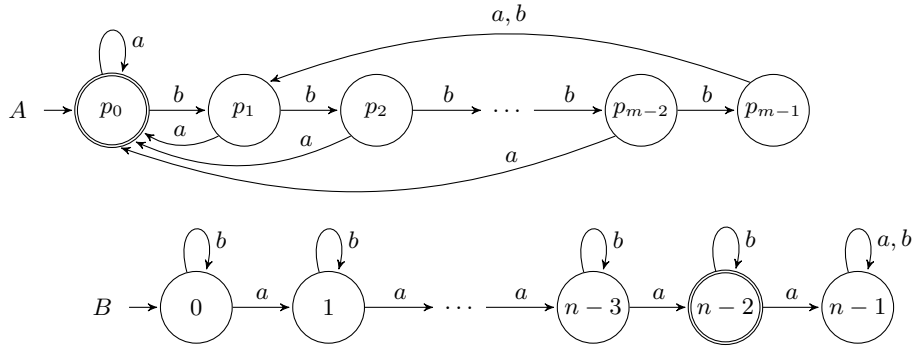

Fig. 2. Binary witnesses for unambiguous concatenation meeting the upper bound $m2^{n-1} - 2^{n-2}$.

omitting the dead state $n-1$ of B, by adding the transitions from p_i to 0 by a, for all $p_i \in P \setminus \{p_{m-1}\}$, and by making both p_0 and 0 initial.

Each reachable subset is represented as (p, S) with $p \in P$ and $S \subseteq Q \setminus \{n-1\}$.

The first goal is to show that for each set $S \subseteq Q$, the state $(p_0, \{0\} \cup S)$ is reachable, and so is each state (p_i, S), with $1 \leqslant i \leqslant m-1$. The proof is by induction on $|S|$. In the base case, $|S| = 0$, each pair $(p_i, \varnothing)$, with $i \in \{1, 2, \ldots, m-1\}$, is reached as follows.

$$(p_0, \{0\}) \xrightarrow{b} (p_1, \{0\}) \xrightarrow{(b^{m-2}a)^{n-2}} (p_1, \{n-2\}) \xrightarrow{b^{m-2}a} (p_1, \varnothing) \xrightarrow{b^{i-1}} (p_i, \varnothing)$$

Let $2 \leqslant k \leqslant n$ and assume that for each set S' with $|S'| = k-1$, each state (p_i, S) with $1 \leqslant i \leqslant m-1$ and the state $(p_0, \{0\} \cup S)$ is reachable. Let $S \subseteq Q$ and $|S| = k$. Consider several cases:

(1) $i = 1$. Let $j = \min S$. Take $S' = \{q - j - 1 \mid q \in S \setminus \{j\}\}$. Then $S' \subseteq Q$ and $|S'| = k-1$, so the state (p_1, S') is reachable by the induction assumption. The state (p_1, S) is reachable from it as follows.

$$\begin{aligned}(p_1, S') \xrightarrow{ab} &(p_1, \{0\} \cup \{q - j \mid q \in S \setminus \{j\}\}) \xrightarrow{(b^{m-2}a)^j} \\ &(p_1, \{j\} \cup \{q \mid q \in S \setminus \{j\}\}) = (p_1, S)\end{aligned}$$

(2) $2 \leqslant i \leqslant m-1$. Then (p_1, S) was shown to be reachable in case (1), and (p_i, S) is reached from it by the string b^{i-1}.

(3) $i = 0$. Then $(p_1, \{q - 1 \mid q \in S \setminus \{0\}\})$ is reachable as in case (1), and $(p_0, \{0\} \cup S)$ is reached from it by the symbol a.

This proves the reachability of $(m-1)2^n + 2^{n-1} = m2^n - 2^{n-1}$ states.

To prove distinguishability, let (p_i, S) and (p_j, T) be two distinct states. If $S \neq T$, let $j \in Q$ be in their symmetric difference, and assume, without loss of generality, that $j \in S$ and $j \notin T$. Then the string a^{n-2-j} is accepted from

(p_i, S), but not from (p_j, T). Let $S = T$, and, without loss of generality, let $i < j$. Then the string $b^{m-1-j}a$ distinguishes (p_i, S) and (p_j, S) since

$$(p_j, S) \xrightarrow{b^{m-1-j}} (p_{m-1}, S) \xrightarrow{a} \big(p_1, \{q+1 \mid q \in S\}\big),$$

$$(p_i, S) \xrightarrow{b^{m-1-j}} (p_{m-1-(j-i)}, S) \xrightarrow{a} \big(p_0, \{0\} \cup \{q+1 \mid q \in S\}\big),$$

so, the resulting states differ in the second component, and therefore are distinguishable. □

All the above results are summarized in the next theorem. Then, the unary case is discussed.

Theorem 6 (Unambiguous Concatenation). *Let A and B be an m-state and n-state DFA, respectively, such that the concatenation $L(A)L(B)$ is unambiguous. Then the language $L(A)L(B)$ is recognized by a DFA of at most $m2^{n-1} - 2^{n-2}$ states. This upper bound is tight, and it is met by the binary witness languages recognized by the DFAs shown in Fig. 2.*

Theorem 7 (Unary Unambiguous Concatenation). *Let $m, n \geqslant 2$, let A be an m-state and B an n-state DFA over a unary alphabet $\Sigma = \{a\}$, and let the concatenation $L(A)L(B)$ be unambiguous. Then $L(A)L(B)$ is recognized by a DFA with at most $m + n - 2$ states, and this upper bound is tight.*

Proof. If both languages are infinite, then the concatenation is ambiguous. Therefore, one of the automata defines a finite language; since concatenation is commutative, there is no loss of generality in the assumption that this is A. Then A recognizes some subset of $\{\varepsilon, a, a^2, \ldots, a^{m-2}\}$.

Let $L(B)$ be periodic with period k, beginning from ℓ; then, $k + \ell \leqslant n$. Then, the concatenation $L(A)L(B)$ is periodic with period k, beginning from $m + \ell - 2$, and is therefore recognized by a DFA with $m + n - 2$ states.

In the worst case, $m+n-2$ states are necessary to represent the unambiguous concatenation of two unary languages represented by an m-state and an n-state DFA, which is witnessed by the singleton languages $\{a^{m-2}\}$ and $\{a^{n-2}\}$. Their concatenation $\{a^{m+n-4}\}$ requires a DFA with $m + n - 2$ states. □

5 Star

The star of an n-state DFA is representable by a DFA with $\frac{3}{4}2^n$ states, and this number is in the worst case necessary [17]. However, if the star is unambiguous, then the necessary number of states can be reduced. The proof is based on the property that for the star to be unambiguous, the automaton has to have a dead state. This property is in turn based on the following auxiliary result.

Lemma 8. *If a DFA $A = (\Sigma, Q, q_0, \delta, F)$, with $L(A) \neq \varnothing$, has no dead states, then there is an accepting state $q \in F$ which is in a cycle, that is, $\delta(q, u) = q$ for some non-empty string $u \in \Sigma^*$.*

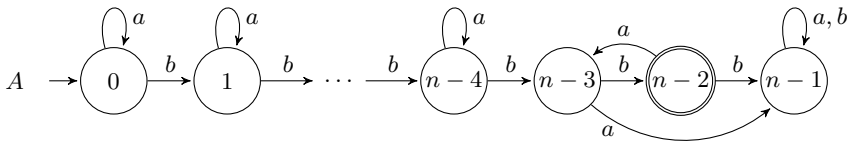

Fig. 3. A binary witness for unambiguous star meeting the upper bound $\frac{3}{8}2^n+1$.

Proof. Let $u_0 \in L(A)$. Then $\delta(q_0, u_0) = q_1$ for some accepting state q_1. Since A does not have any dead state, there is a non-empty string u_1 accepted from q_1. Let $q_2 = \delta(q_1, u_1)$ be the state, in which it is accepted. This yields a sequence of accepting states q_i and non-empty strings u_i, with $i \geqslant 1$. Since A has finitely many accepting states, this sequence eventually revisits some state, that is, $q_i = q_j$ for some $i < j$. Then the state q_i is cyclic, with $q_i = \delta(q_i, u_1 \dots u_{j-1})$. □

Lemma 9. *If A is a DFA and the star $L(A)^*$ is unambiguous, then A has a dead state.*

Proof. Let $A = (\Sigma, Q, q_0, \delta, F)$ be a minimal DFA for L. Suppose for a contradiction that A does not have any dead state. Then, by Lemma 8, there is an accepting state $p \in F$ and a nonempty string $v \in \Sigma^*$, with $\delta(p, v) = p$. Consider the sequence of states q_i, with $i \geqslant 0$, defined by $q_i = \delta(q_0, v^i)$. Since the number of states in A is finite, there are numbers $j \geqslant 0$ and $k \geqslant 1$ with $q_j = q_{j+k}$, and thus $\delta(q_j, v^k) = q_j$. Now let u be a string, by which p is reached from q_0, and let w be any string accepted from q_j, Then the string $uv^{j+k}w$ can be partitioned as $u \cdot v^{j+k}w$ or $uv^k \cdot v^j w$, which is a contradiction with the unambiguity of L^*. Therefore, A must have a dead state. □

Lemma 10. *For every $n \geqslant 4$, the language $L_n = (a^*b)^{n-3}b(ab)^*$ is recognized by an n-state DFA, the star L_n^* is unambiguous, and every DFA recognizing L_n^* must have at least $\frac{3}{8}2^n + 1$ states.*

Proof. The language L_n is recognized by the DFA A shown in Fig. 3, Construct an NFA A^* from A by omitting the dead state $n-1$, by adding the transition $(n-3, b, 0)$, and by adding one more initial and final state s; see Fig 4. The NFA A^* is unambiguous, since the intersection of every reachable and every co-reachable set of A^* is of size at most one.

Let us show that the subset automaton $\mathcal{D}(A^*)$ has $\frac{3}{8}2^n + 1$ reachable and pairwise distinguishable states. The initial subset is $\{s, 0\}$ and it is sent to the state $[0, n-3]$ by $b^{n-3}(ba)^n$. Next, every subset S of $[0, n-3]$ can be shifted cyclically by one, that is, it can be sent to $\{(s+1) \bmod (n-2) \mid s \in S\}$, by reading b if $n-3 \notin S$, by reading ba if $\{n-4, n-3\} \subseteq S$, and by reading baa if $n-3 \in S$ and $n-4 \notin S$. Moreover, the state $n-3$ can be eliminated from every subset of $[0, n-3]$ containing the state $n-3$ by reading a. It follows that every subset of $[0, n-3]$ is reachable from $[0, n-3]$.

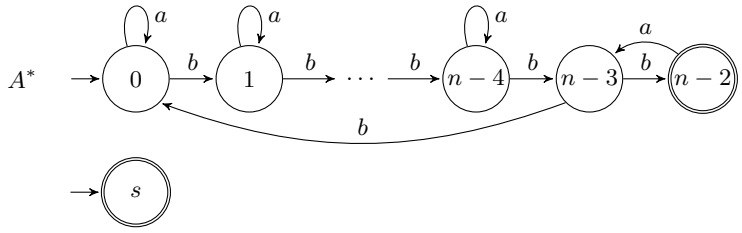

Fig. 4. NFA A^* for the star of the language accepted by DFA A from Fig. 3.

Now let $n-2 \in S$. Then also $0 \in S$. Let $S' = \{s-1 \mid s \in S \setminus \{0\}\}$. Then $S' \subseteq [0, n-3]$, so S' is reachable as shown above. Since $n-3 \in S'$, the set S' is sent to S by b. This proves reachability.

To prove distinguishability, let S and S' be any two distinct reachable subsets of the subset automaton $\mathcal{D}(A^*)$. Then they must differ in some state $i \in [0, n-2]$. Assume, without loss of generality, that $i \in S$ and $i \notin S'$ Then, the string b^{n-2-i} is accepted from S, but not from S'. □

The next theorem summarizes the results on the star operation. Then, the unary case is discussed.

Theorem 11 (Unambiguous Star). *Let $n \geqslant 4$ and A be an n-state DFA such that $L(A)^*$ is unambiguous. Then the language $L(A)^*$ is accepted by a DFA of at most $\frac{3}{8}2^n + 1$ states. This upper bound is tight, and it is met by the binary witness language $(a^*b)^{n-3}b(ab)^*$.*

Proof. To get an NFA A^* for L^* from the DFA A, first omit the dead state. Then add the transition (q, a, q_0) whenever $\delta(q, a) \in F$. Finally, add one more initial and final state s with no transitions going from it. In the subset automaton $\mathcal{D}(A^*)$, the initial subset is $\{s, q_0\}$, and no other reachable subset contains q_0. Moreover, no subset which contains a final state of A but does not contain state q_0 is reachable. In total, the number of reachable subsets in $\mathcal{D}(A^*)$ is at most $1 + 2^{n-1} - 2^{n-2} = \frac{3}{8}2^n + 1$. The tightness of this upper bound follows from Lemma 10. □

Theorem 12 (Unary Unambiguous Star). *Let A be an n-state unary DFA such that $L(A)^*$ si unambiguous. Then $L(A)^*$ is accepted by a DFA of at most $n-2$ states. This upper bound is tight, and it is met by the unary language $\{a^{n-2}\}$.*

Proof. In the unary case, for L^* to be unambiguous, L must be a singleton. Furthermore, if an n-state DFA recognizes a singleton $\{a^\ell\}$, then $\ell \leqslant n-2$, whereas the star of this language, $(a^\ell)^*$, is recognized by a DFA with ℓ states. Thus, unambiguous star of a unary DFA is representable using $n-2$ states. This number of states is necessary, witnessed by the language $\{a^{n-2}\}$. □

	DFA	UFA	NFA
$\cup$	mn [17]	$\leqslant m + O(n2^{0.79m})$ [11]	$m+n$ [9]
$\uplus$	$\boldsymbol{mn-1}$	$\leqslant m+n$	$\leqslant m+n$
$\cdot$	$m2^n - 2^{n-1}$ [17]	$\frac{3}{4}2^{m+n}-1$ [11]	$m+n$ [9]
UNAMB$\cdot$	$\boldsymbol{m2^{n-1}-2^{n-2}}$ [20]	$\leqslant m+n$	$\leqslant m+n$
$*$	$\frac{3}{4}2^n$ [17]	$\frac{3}{4}2^n$ [11]	$n+1$ [9]
UNAMB$*$	$\boldsymbol{\frac{3}{8}2^n+1}$	$\leqslant n+1$	$\leqslant n+1$

Table 1. State complexity of standard and unambiguous operations for DFA, UFA and NFA: union ($\cup$), disjoint union ($\uplus$), concatenation ($\cdot$), unambiguous concatenation (UNAMB$\cdot$), Kleene star (*), unambiguous Kleene star (UNAMB*).

6 Summary of results

State complexity of basic operations on regular languages and of their unambiguous variants studied in this paper is compared in Table 1 for three automata models: DFA, UFA and NFA.

For the three unambiguous operations on DFA, their state complexity has been established in this paper. These operations are easy to apply to UFA using the standard constructions for the union, concatenation and star of NFA: indeed, since the operations are unambiguous, they preserve the unambiguity of the automata involved; however, it remains to establish matching lower bounds.

Another line of related state complexity research is concerned with basic operations on *prefix-free languages*, that is, those with the property that $uv \in L$, with $v \in \Sigma^+$, implies that $u \notin L$. *Suffix-free languages* are defined similarly. Notably, for prefix-free and for suffix-free languages, both concatenation and star are unambiguous, and hence the state complexity results on these subcases are natural tight upper bounds for these two operations. For prefix-free languages and for the DFA model, union has state complexity $mn-2$ [12], the state complexity of concatenation is $m+n-1$ [7,12], whereas the star has state complexity n [7,12]. For suffix-free languages, the results are completely different: union has state complexity $mn-m-n+2$ [5,13], concatenation has $(m-1)2^{n-2}+1$ [5,3], and for the star it is $2^{n-2}+1$ [5,3].

One more related research direction is the recent study of variants of the basic operations on languages defined over the field GF(2) instead of the standard Boolean logic [1]. The union operation turns into the symmetric difference, whereas concatenation gives rise to the following new *GF(2)-concatenation* operation.

$$K \odot L = \{\, w \mid \#\text{ of partitions } w = uv, \text{ with } u \in K \text{ and } v \in L, \text{ is odd} \,\}$$

The *GF(2)-star* is defined similarly, so as to preserve only the strings with an odd number of partitions. Notably, the unambiguous operations studied in this

	DFA	UFA	NFA
$\cup$	$\leqslant mn$ [19]		$m+n$ [9]
$\uplus$	$\boldsymbol{\leqslant \frac{1}{2}mn}$	$\leqslant m+n$	$\leqslant m+n$
$\cdot$	$\leqslant mn$ [21]		$m+n$ [9]
UNAMB$\cdot$	$\boldsymbol{m+n-2}$	$\leqslant m+n$	$\leqslant m+n$
$*$	$(n-1)^2+1$ [21]	$(n-1)^2+1$ [18]	$n+1$ [9]
UNAMB$*$	$\boldsymbol{n-2}$	$\leqslant n+1$	$\leqslant n+1$

Table 2. State complexity of operations in the case of a unary alphabet.

paper are a special case of the GF(2) operations in the same way as they are a special case of classical operations: indeed, the differences between classical and GF(2) operations are in the treatment of ambiguity. As the GF(2) operations preserve regularity, their state complexity is worth being compared to the unambiguous and the classical cases. So far, it has been proved that for DFA, GF(2)-concatenation has state complexity $m \cdot 2^n$, while the state complexity of the GF(2)-star is $2^n + 1$ [1].

Another incomparable extension of unambiguous concatenation and star are the *unique concatenation* and the *unique star* [20], defined similarly, using the uniqueness of partition as the condition of membership. The state complexity of unique concatenation is at most $m3^n - 3^{n-1}$, and for the unique star the upper bound is $2 \cdot 3^{n-1} - \frac{3}{4}2^n + 2$ [20], their tightness remains open.

In the next Table 2, the state complexity of all the same operations is compared in the case of a unary alphabet. Again, for DFA, the state complexity of unambiguous operations has been established, whereas for UFA and for NFA there are only obvious upper bounds. For UFA, the state complexity of standard operations remains to be investigated, with only a few results known [18].

References

1. E. Bakinova, A. Basharin, I. Batmanov, K. Lyubort, A. Okhotin, E. Sazhneva, "Formal languages over GF(2)", *Language and Automata Theory and Applications* (LATA 2018, Bar-Ilan near Tel Aviv, Israel, 9–11 April 2018), LNCS 10792, 68–79.
2. J. A. Brzozowski, M. Szykuła, "Complexity of suffix-free regular languages", *Journal of Computer and System Sciences*, 89 (2017), 270–287.
3. R. Cmorik, G. Jirásková, "Basic operations on binary suffix-free languages", *MEMICS 2011*, LNCS 7119, 94–102.
4. M. Daley, M. Domaratzki, K. Salomaa, "Orthogonal concatenation: Language equations and state complexity", *Journal of Universal Computer Science*, 16:5 (2010), 653-675.
5. Y.-S. Han, K. Salomaa, "State complexity of basic operations on suffix-free regular languages", *Theoretical Computer Science*, 410 (2009), 2537–2548.
6. Y.-S. Han, K. Salomaa, "Nondeterministic State Complexity for Suffix-Free Regular Languages", *DCFS 2010*, EPTCS 31, 189–196.

7. Y.-S. Han, K. Salomaa, D. Wood, "Operational state complexity of prefix-free regular languages", *Automata, Formal Languages, and Related Topics*, 2009, 99–115.
8. Y.-S. Han, K. Salomaa, D. Wood, "Nondeterministic State Complexity of Basic Operations for Prefix-Free Regular Languages", *Fundamenta Informaticae*, 90:1–2 (2009), 93–106.
9. M. Holzer, M. Kutrib, "Nondeterministic descriptional complexity of regular languages", *International Journal of Foundations of Computer Science*, 14 (2003), 1087–1102.
10. J. Jirásek, G. Jirásková, A. Szabari, "State complexity of concatenation and complementation", *International Journal of Foundations of Computer Science*, 16:3 (2005), 511–529.
11. J. Jirásek Jr., G. Jirásková, J. Šebej, "Operations on unambiguous finite automata", *Developments in Language Theory* (DLT 2016, Montréal, Canada, July 25-28, 2016), LNCS 9840, 243–255.
12. G. Jirásková, M. Krausová, "Complexity in prefix-free regular languages", *DCFS 2010*, EPTCS 31, 197–204.
13. G. Jirásková, P. Olejár, "State complexity of intersection and union of suffix-free languages and descriptional complexity", *NCMA 2009*, books@ocg.at vol. 256, 151–166.
14. G. Jirásková, A. Okhotin, "On the state complexity of operations on two-way finite automata", *Information and Computation*, 253:1 (2017), 36–63.
15. M. Kunc, A. Okhotin, "State complexity of union and intersection for two-way nondeterministic finite automata", *Fundamenta Informaticae*, 110:1–4 (2011), 231–239.
16. M. Kunc, A. Okhotin, "State complexity of operations on two-way deterministic finite automata over a unary alphabet", *Theoretical Computer Science*, 449 (2012), 106–118.
17. A. N. Maslov, "Estimates of the number of states of finite automata", *Soviet Mathematics Doklady*, 11 (1970), 1373–1375.
18. A. Okhotin, "Unambiguous finite automata over a unary alphabet", *Information and Computation*, 212 (2012), 15–36.
19. G. Pighizzini, J. Shallit, "Unary language operations, state complexity and Jacobsthal's function", *International Journal of Foundations of Computer Science*, 13:1 (2002), 145–159.
20. N. Rampersad, B. Ravikumar, N. Santean, J. Shallit, "State complexity of unique rational operations", *Theoretical Computer Science*, 410 (2009), 2431–2441.
21. S. Yu, Q. Zhuang, K. Salomaa, "The state complexity of some basic operations on regular languages", *Theoretical Computer Science*, 125 (1994), 315–328.