Operations on Boolean and Alternating Finite Automata

Galina Jirásková*
Mathematical Institute, Slovak Academy of Sciences, Grešákova 6, 040 01 Košice, Slovakia
jiraskov@saske.sk

We examine the complexity of basic regular operations on languages represented by Boolean and alternating finite automata. We get tight upper bounds $m+n$ and $m+n+1$ for union, intersection, and difference, 2^m+n and 2^m+n+1 for concatenation, 2^n+n and 2^n+n+1 for square, m and $m+1$ for left quotient, 2^m and 2^m+1 for right quotient. We also show that in both models, the complexity of complementation and symmetric difference is n and $m+n$, respectively, while the complexity of star and reversal is 2^n. All our witnesses are described over a unary or binary alphabets, and whenever we use a binary alphabet, it is always optimal.

1 Introduction

Boolean and alternating finite automata [1, 2, 6, 7, 10, 11, 12] are generalizations of nondeterministic finite automata. They recognize regular languages, however, they may be exponentially smaller, with respect to the number of states, than equivalent nondeterministic finite automata (NFAs). While in an NFA the transition function maps any pair of a state and input symbol to a set of states that can be viewed as a disjunction of the states, in a Boolean finite automaton (BFA) the result of the transition function is given by any Boolean function with variables in the state set.

Fellah et al. [3] examined alternating finite automata (AFAs), that is, Boolean automata in which the initial Boolean function is given by a projection. They proved that every n-state AFA can be simulated by a (2^n+1)-state nondeterministic finite automaton with a unique initial state, and left as an open problem the tightness of this upper bound. An answer to this problem was given in [7, Lemma 1, Theorem 1] by describing an n-state binary AFA whose equivalent NFA with a unique initial state has at least 2^n+1 states. Here we present a different example in which the reachability and co-reachability of all singleton sets immediately implies the result.

In [3] it was also shown that given an m-state and n-state AFAs for languages K and L, the languages L^c, $K\cup L$, $K\cap L$, KL, and L^* are recognized by AFAs of at most $n, m+n+1, m+n+1, 2^m+n+1$, and 2^n+1 states, respectively, and the tightness of these upper bounds was left open as well.

Here we present the results obtained in [5, 6, 7, 8, 11] that provide the exact complexity of basic regular operations on languages represented by Boolean and alternating finite automata. Table 1 summarizes these results. It also displays the sizes of alphabet used to describe witness languages.

2 Preliminaries

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

*This research was supported by the Slovak Grant Agency for Science (VEGA) under contract 2/0096/23 "Automata and Formal Languages: Descriptional and Computational Complexity".

Zs. Gazdag, Sz. Iván, G. Kovásznai (Eds.): 16th International Conference on Automata and Formal Languages (AFL 2023)
EPTCS 386, 2023, pp. 3–10, doi:10.4204/EPTCS.386.1

Table 1: The complexity of basic regular operations on Boolean and alternating finite automata.

operation	BFA	$\lvert\Sigma\rvert$	AFA	$\lvert\Sigma\rvert$	source
complementation	n	1	n	1	[6, Thm. 1]
union	$m+n$	1	$m+n+1$	1	[7, Thm. 2(1) and 3(1)], [11, Thm. 4.3 and 4.4]
intersection	$m+n$	1	$m+n+1$	1	[7, Thm. 2(2) and 3(2)], [11, Thm. 4.3 and 4.4]
difference	$m+n$	1	$m+n+1$	1	[6, Thm. 13(a) and 14(a)], [11, Thm. 4.3 and 4.4]
symm. difference	$m+n$	1	$m+n$	1	[6, Thm. 13(b) and 14(b)], [11, Thm. 4.3 and 4.4]
star	2^n	2	2^n	2	[6, Thm. 12]
reversal	2^n	2	2^n	2	[6, Thm. 13(c) and 14(c)]
right quotient	2^m	2	2^m+1	2	[6, Thm. 13(d) and 14(d)]
left quotient	m	1	$m+1$	1	[6, Thm. 13(e) and 14(e)]
concatenation	2^m+n	2	2^m+n+1	2	[7, Thm. 4 and 5],[5, Thm. 6.4]
square	2^n+n	2	2^n+n+1	2	[8, Thm. 13 and 14]

A *Boolean finite automaton* (BFA) is a quintuple $A=(Q,\Sigma,\cdot,g_s,F)$ where $Q=\{q_1,q_2,\ldots,q_n\}$ is a finite non-empty set of states, Σ is a finite input alphabet, $\cdot$ is a transition function that maps $Q\times\Sigma$ into the set $\mathscr{B}_n$ of Boolean functions with variables $\{q_1,q_2,\ldots,q_n\}$, $g_s\in\mathscr{B}_n$ is the initial Boolean function, and $F\subseteq Q$ is the set of final states. The transition function $\cdot$ is extended to the domain $\mathscr{B}_n\times\Sigma^*$ as follows: For each $g\in\mathscr{B}_n$, each $a\in\Sigma$, and each $w\in\Sigma^*$, we have

- $g\cdot\varepsilon=g$,
- if $g=g(q_1,q_2,\ldots,q_n)$, then $g\cdot a=g(q_1\cdot a,q_2\cdot a,\ldots,q_n\cdot a)$,
- $g\cdot(wa)=(q\cdot w)\cdot a$.

Let $f=(f_1,f_2,\ldots,f_n)$ be a Boolean vector (*finality vector*) such that $f_i=1$ if and only if $q_i\in F$. The *language accepted by* a BFA A is the set of strings $L(A)=\{w\in\Sigma^*\mid(g_s\cdot w)(f)=1\}$. We illustrate the above mentioned notions in the following example.

Example 1. Consider the 2-state binary Boolean finite automaton $A=(\{q_1,q_2\},\{a,b\},\cdot,q_1\wedge q_2,\{q_1\})$ where the transition function $\cdot$ is defined in Table 2.

Table 2: The transition function of the BFA A.

$\cdot$	a	b
q_1	$q_1\vee q_2$	q_1
q_2	q_2	$q_1\wedge\neg q_2$

Then the string ab is accepted by A since we have

$$g_s\cdot ab=(q_1\wedge q_2)\cdot ab=((q_1\vee q_2)\wedge q_2)\cdot b=(q_1\vee(q_1\wedge\neg q_2))\wedge(q_1\wedge\neg q_2),$$

and the resulting function evaluates to 1 in the finality vector $(1,0)$. $\square$

A BFA A is called *alternating* (AFA) if its initial function is a projection $g_s(q_1,q_2,\dots,q_n)=q_1$; cf. [2, 3, 15]. It is *nondeterministic with multiple initial states* (MNFA) if g_s and all $q_i\cdot a$ are of the form $q_{i_1}\vee q_{i_2}\vee\cdots\vee q_{i_\ell}$. If moreover $g_s=q_1$, then A is *nondeterministic* (with a unique initial state) (NFA). If moreover all $q_i\cdot a$ are of the form q_j, then A is *deterministic* (DFA).

3 Simulations of BFAs and AFAs by MNFAs, NFAs, and DFAs

In this section we recall the trade-offs between different models of finite automata. Let us start with the simulation of BFAs by MNFAs.

Proposition 2 ([3, Theorem 4.1], [7, Lemma 1])**.** *Let L be a language accepted by an n-state BFA. Then L is accepted by a 2^n-state MNFA whose reverse is a DFA.*

Proof Idea. Let $A=(Q,\Sigma,\cdot,g_s,F)$ be a BFA with $Q=\{q_1,q_2,\dots,q_n\}$. Let $f=(f_1,f_2,\dots,f_n)$ be the Boolean finality vector with $f_i=1$ iff $q_i\in F$. Construct a MNFA $A'=(Q',\Sigma,\circ,I,\{f\})$ where

- $Q'=\{0,1\}^n$,
- $I=\{u\in Q'\mid g_s(u)=1\}$,
- for each $u\in Q'$ and each $a\in\Sigma$, we set $u\circ a=\{u'\in Q'\mid (q_1\cdot a,q_2\cdot a,\dots,q_n\cdot a)(u')=u\}$.

Then $L(A)=L(A')$. □

Since the reverse of the MNFA in the proof above is a DFA, we get the next result.

Corollary 3. *If L is accepted by an n-state BFA, then L^R is accepted by a 2^n-state DFA.* □

Notice that if A is an AFA, then the MNFA A' constructed in the proof of Proposition 2 has 2^{n-1} initial states, and we get the following observation.

Corollary 4. *If L is accepted by an n-state AFA, then L^R is accepted by a 2^n-state DFA of which 2^{n-1} are final.* □

Our next aim is to get the converses of the above corollaries.

Proposition 5 ([7, Lemma 2])**.** *Let L be accepted by a 2^n-state MNFA whose reverse is a DFA. Then L is accepted by an n-state BFA.*

Proof Idea. Let $A=(Q,\Sigma,\cdot,I,F)$ be a MNFA with $Q=\{0,1\}^n$. Since A^R is a DFA, the MNFA A has a unique final state $f\in Q$, and moreover, for each $u\in Q$ and each $a\in\Sigma$ there is a unique state u' with $u'\cdot a=u$; denote this state by au. Construct a BFA $A'=(Q',\Sigma,\circ,g_s,F')$ where

- $Q'=\{q_1,q_2,\dots,q_n\}$,
- $g_s(u)=1$ iff $u\in I$,
- $F'=\{q_i\mid f_i=1\}$,
- $(q_1\circ a,q_2\circ a,\dots,q_n\circ a)(u)=au$.

Then $L(A)=L(A')$. □

Corollary 6. *If L is accepted by a 2^n-state DFA, then L^R is accepted by an n-state BFA.* □

Corollary 7. *If L is accepted by a 2^n-state DFA which has 2^{n-1} final states, then L^R is accepted by an n-state AFA.* □

We continue with the simulation of BFAs by DFAs.

Proposition 8 ([10, Theorem 7], [1, Theorem 2], [2, Theorem 5.2], [12, Corollary 3]). *Let L be a language over an alphabet Σ accepted by an n-state BFA. Then L is accepted by a DFA of at most 2^{2^n} states, and this upper bound is tight if $|\Sigma| \geq 2$.*

Proof Idea. If L is accepted by an n-state BFA, then by Proposition 2, it is accepted by a 2^n-state MNFA, and, consequently, by a 2^{2^n}-state DFA. For tightness, let K be the binary 2^n-state DFA from [12, Proposition 2] whose reversal K^R requires 2^{2^n} deterministic states. By Corollary 6, the language K^R is accepted by an n-state BFA. □

Finally, we consider the simulation of BFAs by NFAs, and provide an answer to an open problem from [3].

Theorem 9 ([7, Theorem 1]). *Let L be accepted by an n-state BFA. Then L is accepted by an NFA of at most $2^n + 1$ states. This upper bound is tight, and it can be met by a binary n-state AFA.*

Proof Idea. By Proposition 2, the language L is accepted by a 2^n-state MNFA, and, consequently, by a $(2^n + 1)$-state NFA. For tightness, let $n \geq 2$. Let L be the language accepted by the 2^n-state MNFA $A = (Q, \{a, b\}, \cdot, I, F)$ where

- $Q = \{0, 1, \ldots, 2^n - 1\}$,
- $I = \{0, 1, \ldots, 2^{n-1} - 1\}$,
- $F = \{2^n - 1\}$,
- $i \cdot a = \{(i + 1) \bmod 2^n\}$ for each $i \in Q$,
- $0 \cdot b = \{0\}$, $(2^n - 1) \cdot b = Q \setminus \{0\}$, and $i \cdot b = \emptyset$ is $i \in Q \setminus \{0, 2^n - 1\}$;

see Figure 1 for an illustration 1. The reverse A^R is a 2^n-state DFA which has 2^{n-1} final states. By Corollary 7, the language L is accepted by an n-state AFA. On the other hand, each singleton set is reachable and co-reachable in the MNFA A which means that every NFA accepting L has at least $2^n + 1$ states by [4, Lemma 9]. □

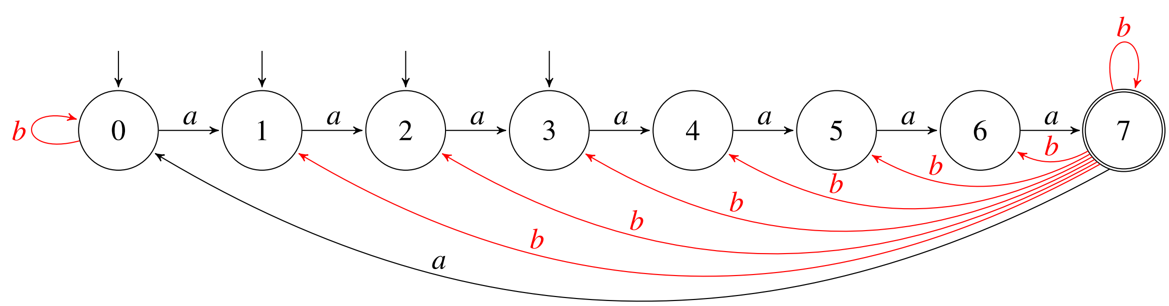

Figure 1: The MNFA A; $n = 3$.

4 Operational Complexity on Boolean and Alternating Finite Automata

In this section we use the four corollaries from the previous section to get the complexity of basic regular operations on languages represented by Boolean and alternating finite automata. The idea is as follows. Consider a binary operation and take languages K and L recognized by a 2^m-state and 2^n-state DFA, respectively, that are witnesses for the considered operation on DFAs. Then the languages K^R and L^R are accepted by an m-state and n-state BFA, respectively. Now it is enough to show that the language resulting from the operation applied to the languages K^R and L^R requires large enough BFA. In the case of AFAs, we start with DFAs with half of their states final that are hard for the considered operation on DFAs. We illustrate this idea for the concatenation operation.

Theorem 10 (Concatenation on BFAs). *Let K and L be languages over an alphabet Σ accepted by an m-state and n-state BFA, respectively. Then the language KL is accepted by a BFA of at most 2^m+n states, and this upper bound is tight if $|\Sigma| \geq 2$.*

Proof. To get an upper bound, let $A = (Q_A, \Sigma, \cdot_A, g_A, F_A)$ and $B = (Q_B, \Sigma, \cdot_B, g_B, F_B)$ be BFAs accepting the languages K and L, respectively. We first convert the BFA A to the 2^m-state MNFA $M = (Q_M, \Sigma, \cdot_M, g_M, F_M)$. Now we construct a BFA $C = (Q_M \cup Q_B, \Sigma, \cdot, g_M, F_B)$ with

$$q \cdot a = \begin{cases} q \cdot_M a, & \text{if } q \in Q_M \setminus F_M; \\ q \cdot_M a \vee g_B \cdot_B a, & \text{if } q \in F_M; \\ q \cdot_B a, & \text{if } q \in Q_B; \end{cases}$$

cf. [3, Theorem 9.2]. Then the BFA C has $2^m + n$ states and recognizes the language KL.

To get tightness, let K and L be Maslov's binary witnesses for concatenation on DFAs from [13], see Figure 2, accepted by a 2^n-state and 2^m-state DFA, respectively. Then every DFA accepting the language KL has at least $2^n 2^{2^m} - 2^{2^m-1}$ states. By Corollary 6, the languages L^R and K^R are accepted by m-state and n-state BFA, respectively. Next, we have $(L^R K^R)^R = KL$, so every DFA accepting the reverse of the concatenation $L^R K^R$ has at least $2^n 2^{2^m} - 2^{2^m-1}$ states. By Corollary 3, it follows that every BFA accepting $K^R L^R$ has at least $\lceil \log(2^n 2^{2^m} - 2^{2^m-1}) \rceil = 2^m + n$ states. □

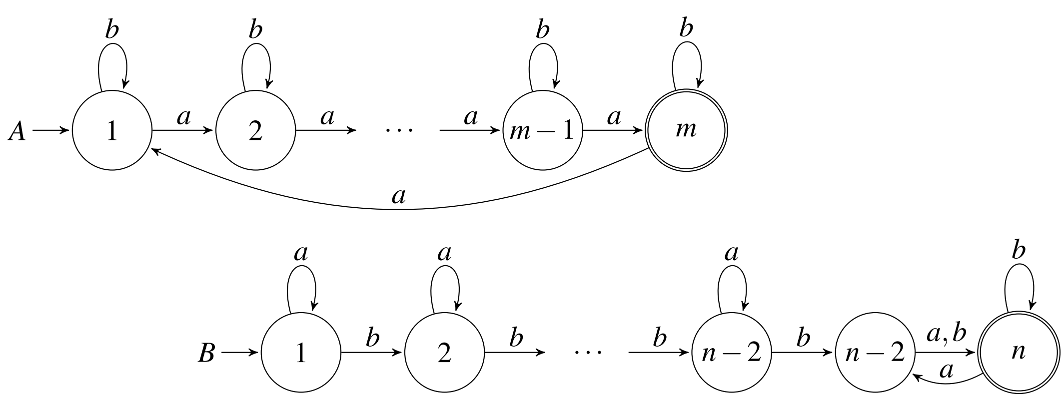

Figure 2: Maslov's witness DFAs for concatenation meeting the upper bound $m2^n - 2^{n-1}$.

Theorem 11 (Concatenation on AFAs). *Let K and L be languages over an alphabet Σ accepted by an m-state and n-state AFA, respectively. Then the language KL is accepted by a AFA of at most 2^m+n+1 states, and this upper bound is tight if $|\Sigma| \geq 2$.*

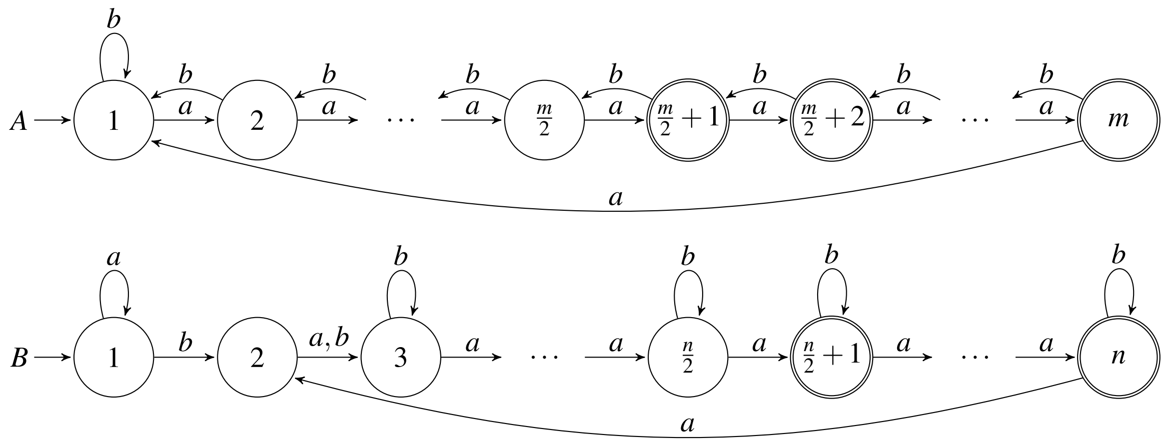

Figure 3: Witness DFAs for concatenation with half of states final meeting the upper bound $m2^n - \frac{m}{2}2^{n-1}$.

Proof. The upper bound follows from the previous theorem since one more state is enough to get an AFA equivalent to a given BFA. To get tightness, we use languages K and L accepted by 2^n-state and 2^m-state witness DFAs for concatenation with half of their states final from [5, Theorem 4.7], see Figure 3. Then the minimal DFA for KL has $2^n2^{2^m} - 2^{n-1}2^{2^m-1}$ states, of which more that 2^{2^m+n-1} states are final [5, Lemma 6.4]. Then the languages L^R and K^R are accepted by an m-state and n-state AFA, respectively. Next, we have $(L^RK^R)^R = KL$, so every AFA for L^RK^R has at least $\lceil\log(2^n2^{2^m} - 2^{n-1}2^{2^m-1})\rceil = 2^m+n$ states. If an AFA of 2^m+n states would accept L^RK^R, then the reverse of this language, that is, the language KL would be accepted by a DFA of 2^{2^m+n} states with 2^{2^m+n-1} final states. However, the minimal DFA for KL has more than 2^{2^m+n-1} final states, a contradiction. □

Hence, the upper bound 2^m+n+1 for concatenation on AFAs from [3, Theorem 9.3] is tight. This provides an answer to the second open problem from [3]. A similar idea as for concatenation also works for square, and left and right quotients. Our results for the star operation are covered by the next theorem.

Theorem 12 (Star on BFAs and AFAs). *If L is accepted by an n-state BFA, then L^* is accepted by a 2^n-state AFA. Moreover, there exists a binary language L accepted by an n-state AFA such that every BFA for L^* has at least 2^n states.*

Proof. If L is accepted by an n-state BFA, then L^R is accepted by a 2^n-state DFA by Corollary 3. Then $(L^R)^*$ is accepted by a 2^{2^n}-state DFA with half of its state final [6, Proposition 8]. Next, we have $(L^*)^R = (L^R)^*$. Hence L^* is accepted by an n-state AFA by Corollary 7.

To get tightness, let L be the Palmovský's witness DFA for star with 2^n states half of which are final [14, Theorem 4.4], see Figure 4. Then L^R is accepted by an n-state AFA by Corollary 7. Next, we have $((L^R)^*)^R = ((L^*)^R)^R = L^*$, and every DFA for L^* has at least $2^{2^n-1} + 2^{2^n-1+2^{n-1}}$ states. It follows that every BFA for $(L^R)^*$ has at least $\lceil\log(2^{2^n-1} + 2^{2^n-1+2^{n-1}})\rceil = 2^n$ states by Corollary 3. □

Similar arguments work for reversal. If L is accepted by an n-state BFA, that L^R is accepted by a 2^n-state DFA, a special case of AFA. For tightness, we take the language L accepted by a 2^n-state Šebej's DFA from [9, Fig. 6] with half of its states final. Then L^R ia accepted by an n-state AFA, while every DFA for L^R has at least 2^{2^n} states. Hence, every BFA for $L = (L^R)^R$ has at least 2^n states by Corollary 6.

We conclude this section with Boolean operations. Denote by $\mathrm{bsc}(L)$ the number of states in a minimal, with respect to the number of states, BFA accepting L. Define $\mathrm{asc}(L)$ in an analogous way.

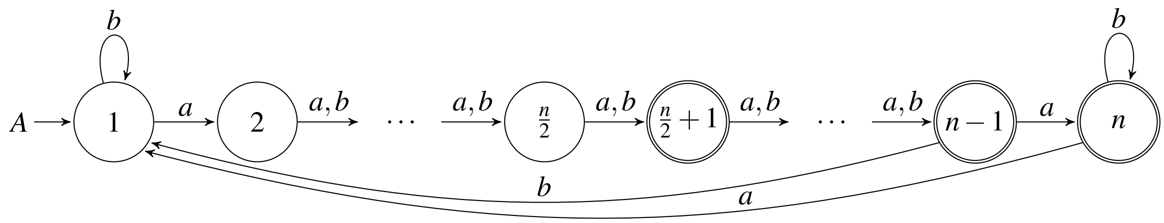

Figure 4: Witness DFA for star with half of states final meeting the upper bound $2^{n-1}+2^{n-1-\frac{n}{2}}$.

Proposition 13. *Let L be a regular language. Then $bsc(L) = bsc(L^c)$ and $asc(L) = asc(L^c)$.*

Proof. If L is accepted by a minimal n-state BFA, then L^R is accepted by a 2^n-state DFA by Corollary 3. It follows that $(L^R)^c = (L^c)^R$ is accepted by a 2^n state DFA, and therefore L^c is accepted by an n-state BFA by Corollary 6. Moreover, the language L^c cannot be accepted by a smaller BFA because otherwise the language $L = (L^c)^c$ would be accepted by a smaller BFA as well. In the case of AFAs, the DFAs for L^R and $(L^R)^c$ have 2^n states and 2^{n-1} final states, and we use Corollaries 4 and 7 to get the result. $\square$

Theorem 14. *Let K and L be languages over Σ accepted by an m-state and n-state AFA, respectively. Then $K \cup L$ is accepted by an AFA of at most $m+n+1$ states, and this upper bound is tight if $|\Sigma| \geq 1$.*

Proof. The language $K \cup L$ can be accepted by a $(m+n)$-state BFA constructed from the two AFAs by setting the initial function to the disjunction of the corresponding initial states. The upper bound for AFAs follows. For tightness, let K be the language accepted by the unary 2^m-state DFA with state set $\{0,1,\ldots,2^m-1\}$, the initial state 0, the set of final states $\{2^{m-1}, 2^{m-1}+1,\ldots,2^m-1\}$, and transitions given by $i \cdot a = (i+1) \bmod 2^m$. Then $K^R = K$ is accepted by an m-state AFA. Next, let L be a language accepted by a (2^n-1)-state unary DFA with state set $\{0,1,\ldots,2^n-2\}$, the initial state 0, the set of final states $\{2^{n-1}, 2^{m-1}+1,\ldots,2^m-2\}$, and transitions given by $i \cdot a = (i+1) \bmod 2^m-1$. Then we can add an unreachable final state to this DFA to get an equivalent 2^n-state DFA with half of its states final. Hence $L^R = L$ is accepted by an n-state AFA. As shown in [11, Lemma 4.2, Theorem 4.4], the minimal DFA for $K \cup L$ has $2^m(2^n-1)$ states, of which more than 2^{m+n+1} are final. It follows that every AFA for $K \cup L$ has at least $m+n+1$ states. $\square$

By Proposition 13 and De Morgan's laws, the complement of the languages described in the previous proof are witnesses for intersection. The case of difference is analogous. The same languages give a lower bound $m+n$ for symmetric difference on AFAs [11, Lemma 4.2] which is also an upper bound; notice that the symmetric difference of two DFAs with half of their states final is accepted by a DFA with half of its states final. Finally, exactly the same languages serve as witnesses for Boolean operations on BFAs [11, Theorem 4.3].

In the unary case, the reverse of any language is the same language, and the right quotient is the same as the left quotient of the corresponding languages. Moreover, we can show that the complexity of star, concatenation, and square on unary BFAs is $2n, m+n$, and $n+1$, respectively. It follows that whenever we used a binary alphabet to describe witnesses for the corresponding operations on BFAs and AFAs, it was always optimal.

The exact complexity of star, concatenation, and square on unary AFAs remains open since the complexity of these operations on unary DFAs with half of their states final is not known. The complexity of less common regular operations like shuffle, cyclic shift, or square root, would be of interest as well.

References

[1] J.A. Brzozowski & E.L. Leiss (1980): *On equations for regular languages, finite automata, and sequential networks*. Theor. Comput. Sci. 10, pp. 19–35, doi:10.1016/0304-3975(80)90069-9.

[2] A.K. Chandra, D. Kozen & L.J. Stockmeyer (1981): *Alternation*. J. ACM 28(1), pp. 114–133, doi:10.1145/322234.322243.

[3] Abdelaziz Fellah, Helmut Jürgensen & Sheng Yu (1990): *Constructions for alternating finite automata*. Int. J. Comput. Math. 35(1-4), pp. 117–132, doi:10.1080/00207169008803893.

[4] M. Hospodár (2021): *Power, positive closure, and quotients on convex languages*. Theor. Comput. Sci. 870, pp. 53–74, doi:10.1016/j.tcs.2021.02.002.

[5] M. Hospodár & G. Jirásková (2018): *The complexity of concatenation on deterministic and alternating finite automata*. RAIRO Theor. Informatics Appl. 52(2-3-4), pp. 153–168, doi:10.1051/ita/2018011.

[6] M. Hospodár, G. Jirásková & I. Krajňáková (2018): *Operations on Boolean and alternating finite automata*. In F.V. Fomin & V.V. Podolskii, editors: CSR 2018, LNCS, vol. 10846, Springer, pp. 181–193, doi:10.1007/978-3-319-90530-3_16.

[7] G. Jirásková (2012): *Descriptional complexity of operations on alternating and Boolean automata*. In E.A. Hirsch, J. Karhumäki, A. Lepistö & M. Prilutskii, editors: CSR 2012, LNCS, vol. 7353, Springer, pp. 196–204, doi:10.1007/978-3-642-30642-6_19.

[8] G. Jirásková & I. Krajňáková (2019): *Square on deterministic, alternating, and Boolean finite automata*. Int. J. Found. Comput. Sci. 30(6-7), pp. 1117–1134, doi:10.1142/S0129054119400318.

[9] G. Jirásková & J. Šebej (2012): *Reversal of binary regular languages*. Theor. Comput. Sci. 449, pp. 85–92, doi:10.1016/j.tcs.2012.05.008.

[10] D. Kozen (1976): *On parallelism in Turing machines*. In: 17th Annual Symposium on Foundations of Computer Science, Houston, Texas, USA, 25-27 October 1976, IEEE Computer Society, pp. 89–97, doi:10.1109/SFCS.1976.20.

[11] I. Krajňáková (2020): *Finite Automata and Operational Complexity*. Ph.D. thesis, Comenius University in Bratislava, Faculty of Mathematics, Physics and Informatics. Available at `https://www.mat.savba.sk/musav/autoreferaty/Krajnakova-dizertacna_praca.pdf`.

[12] E.L. Leiss (1981): *Succint representation of regular languages by Boolean automata*. Theor. Comput. Sci. 13, pp. 323–330, doi:10.1016/S0304-3975(81)80005-9.

[13] A. N. Maslov (1970): *Estimates of the number of states of finite automata*. Soviet Math. Doklady 11(5), pp. 1373–1375. Available at `https://www.mathnet.ru/php/archive.phtml?wshow=paper&jrnid=dan&paperid=35742&option_lang=eng`.

[14] M. Palmovský (2016): *Kleene closure and state complexity*. RAIRO Theor. Informatics Appl. 50(3), pp. 251–261, doi:10.1051/ita/2016024.

[15] S. Yu (1997): *Regular Languages*. In G. Rozenberg & A. Salomaa, editors: Handbook of Formal Languages, Volume 1: Word, Language, Grammar, Springer, pp. 41–110, doi:10.1007/978-3-642-59136-5_2.