Descriptional Complexity of Operations on Alternating and Boolean Automata*

Galina Jirásková

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

Abstract. The paper shows that the tight bound for the conversion of alternating finite automata into nondeterministic finite automata with a single initial state is $2^n + 1$. This solves an open problem stated by Fellah *et al.* (Intern. J. Computer Math. 35, 1990, 117–132). Then we examine the complexity of basic operations on languages represented by boolean and alternating finite automata. We get tight bounds for intersection and union, and for concatenation and reversal of languages represented by boolean automata. In the case of star, and of concatenation and reversal of AFA languages, our upper and lower bounds differ by one.

1 Introduction

Boolean and alternating finite automata [1,2,8,9] are generalizations of nondeterministic finite automata. They recognize regular languages, however, they may be exponentially smaller, in terms of the number of states, than equivalent nondeterministic automata.

Fellah *et al.* [3] studied alternating finite automata (AFAs), that is, boolean automata, in which the initial boolean function is given by a projection. They proved that every AFA of n states can be simulated by a nondeterministic finite automaton *with a single initial state* of at most $2^n + 1$ states, and left as an open problem the tightness of this upper bound. Our first result provides an answer to this problem by describing an n-state binary AFA whose equivalent NFAs with a single initial state have at least $2^n + 1$ states.

Then we examine the complexity of basic regular operations on languages represented by boolean and alternating automata. In the case of union and intersection, we get the tight bounds $m + n$ and $m + n + 1$ on the boolean and alternating state complexity, respectively. Next we show that the boolean state complexity of concatenation is $2^m + n$, and of reversal 2^n. As for the alternating state complexity of concatenation and reversal, our upper and lower bounds differ by one. The same is true for star in both boolean and alternating case.

To get the results, we use known results on the state complexity of operations on regular languages [4,5,6,7,11,14], as well as the fact that if the reversal of a language is accepted by the minimal deterministic automaton of n states, than every boolean automaton for this language has at least $\log n$ states.

* Research supported by VEGA grant 2/0183/11.

E. Hirsch et al. (Eds.): CSR 2012, LNCS 7353, pp. 196–204, 2012.

2 Preliminaries

This section gives basic definitions and notations. For all unexplained notions, the reader may refer to [10,12,13].

If Σ is a non-empty finite alphabet, then Σ^* is the set of all strings over Σ, including the empty string ε. A *language* over alphabet Σ is any subset of Σ^*.

A *boolean finite automaton* (BFA) is a quintuple $A = (Q, \Sigma, \delta, g_s, F)$, where Q is a finite non-empty set of states, $Q = \{q_1, \ldots, q_n\}$, Σ is an input alphabet, δ is the transition function that maps $Q \times \Sigma$ into the set $\mathcal{B}_n$ of boolean functions of n boolean variables $q_1, \ldots, q_n$, $g_s \in \mathcal{B}_n$ is the initial boolean function, and $F \subseteq Q$ is the set of final states. For example, let $A_1 = (\{q_1, q_2\}, \{a, b\}, \delta, q_1 \wedge q_2, \{q_2\})$, where transition function δ is given in Table 1.

Table 1. The transition function of boolean automaton A_1

δ	a	b
q_1	$q_1 \vee q_2$	1
q_2	q_2	$q_1 \wedge \overline{q_2}$

The transition function δ is extended to the domain $\mathcal{B}_n \times \Sigma^*$ as follows: For all g in $\mathcal{B}_n$, a in Σ, and w in Σ^*,

$$\delta(g, \varepsilon) = g;$$
$$\text{if } g = g(q_1, \ldots, q_n), \text{ then } \delta(g, a) = g(\delta(q_1, a), \ldots, \delta(q_n, a));$$
$$\delta(g, wa) = \delta(\delta(g, w), a).$$

Next, let $f = (f_1, \ldots, f_n)$ be the boolean vector with $f_i = 1$ iff $q_i \in F$. The language accepted by BFA A is the set $L(A) = \{w \in \Sigma^* \mid \delta(g_s, w)(f) = 1\}$. In our example we have

$$\delta(g_s, ab) = \delta(q_1 \wedge q_2, ab) = \delta(\delta(q_1 \wedge q_2, a), b) = \delta((q_1 \vee q_2) \wedge q_2, b) =$$
$$(1 \vee (q_1 \wedge \overline{q_2})) \wedge (q_1 \wedge \overline{q_2}) = q_1 \wedge \overline{q_2}.$$

To determine whether $ab \in L(A_1)$, we evaluate $\delta(g_s, ab)$ at vector $f = (0, 1)$. We obtain 0, hence $ab \notin L(A_1)$. On the other hand, we have $abb \in L(A_1)$ since $\delta(g_s, abb) = \delta(q_1 \wedge \overline{q_2}, b) = 1 \wedge (\overline{q_1} \vee q_2) = \overline{q_1} \vee q_2$, which gives 1 at $(0, 1)$.

A boolean finite automaton A is alternating (AFA) if the initial function is given by a projection $g(q_1, \ldots, q_n) = q_i$. It is nondeterministic with multiple initial states (NNFA) if g_s and $\delta(q_k, a)$ are of the form $\bigvee_{i \in I} q_i$. If, moreover, $g_s = q_i$, then automaton A is nondeterministic with a single initial state (NFA). If, moreover, $\delta(q_k, a)$ are of the form q_i, automaton A is deterministic (DFA).

The reverse A^R of NNFA A is obtained from A by swapping the role of the initial and final states, and by reversing all the transitions. The reverse of NNFA A accepts language $L(A)^R = \{w^R \mid w \in L\}$; where w^R is the mirror image of string w defined by $\varepsilon^R = \varepsilon$ and $(wa)^R = aw^R$.

3 Optimal Simulation of AFAs by NFAs

This section provides an answer to an open question stated by Fellah *et al.* [3] whether or not the upper bound $2^n + 1$ on AFA-to-NFA coversion is tight. We start with conversion of boolean automata to NNFAs. Then we show that every NNFA of 2^n states whose reverse is deterministic may be represented by an n-state boolean automaton. This allow us to describe boolean automata by NNFAs in our worst-case example.

Lemma 1. *If a language L is accepted by an n-state boolean automaton, then L is accepted by an NNFA of at most 2^n states.*

Proof. Let $A = (Q, \Sigma, \delta, g_s, F)$ be a boolean automaton with $Q = \{q_1, \ldots, q_n\}$. Like in [3], construct the NNFA $A' = (\{0,1\}^n, \Sigma, \delta', S, \{f\})$, where for every $u = (u_1, \ldots, u_n)$ in $\{0,1\}^n$ and every a in Σ,

$$\begin{aligned} \delta'(u, a) &= \{u' \in \{0,1\}^n \mid \delta(q_i, a)(u') = u_i \text{ for } i = 1, \ldots, n\}, \\ S &= \{b \in \{0,1\}^n \mid g_s(b) = 1\}, \\ f &= (f_1, \ldots, f_n) \in \{0,1\}^n \text{ with } f_i = 1 \text{ iff } q_i \in F. \end{aligned}$$

The proof of $L(A') = L(A)$ is almost the same as in [3, Theorem 4.1]. □

Lemma 2. *Let $A = (Q, \Sigma, \delta, S, F)$ be an NNFA such that $|Q| = 2^n$. Let the reverse of A be deterministic (and complete). Then there exists an n-state boolean automaton A' such that $L(A) = L(A')$. Moreover, if A has 2^{n-1} initial states, then A' may be taken to be alternating.*

Proof. Assume $Q = \{0, 1, \ldots, 2^n - 1\}$. Since A^R is deterministic, NNFA A has exactly one final state, and assume $F = \{k\}$. Moreover, for every symbol a in Σ and every state i in Q, there is *exactly one* state j in Q such that j goes to i by a in NNFA A.

For a state i in Q, let $\mathrm{bin}(i) = (b_1, \ldots, b_n)$ be the binary n-tuple such that $b_1 b_2 \cdots b_n$ is the binary notation of i on n digit with leading zeros if necessary.

Define an n-state boolean automaton $A' = (Q', \Sigma, \delta', g_s, F')$, where $Q' = \{q_1, \ldots, q_n\}$, $F' = \{q_\ell \mid \mathrm{bin}(k)_\ell = 1\}$, and for each i in Q and a in Σ,

$$(\delta'(q_1, a), \ldots, \delta'(q_n, a))(\mathrm{bin}(i)) = \mathrm{bin}(j) \text{ where } i \in \delta(j, a), \text{ and}$$
$$g_s(\mathrm{bin}(i)) = 1 \text{ iff } i \in S.$$

Then $L(A') = L(A)$. If A has 2^{n-1} initial states, assume $S = \{2^{n-1}, \ldots, 2^n - 1\}$. Now to get an AFA, let $g_s = q_1$, that is $g_s(b_1, \ldots, b_n) = 1$ iff $b_1 = 1$. □

The next lemma shows that there exists an NNFA of 2^n states and 2^{n-1} initial states whose reverse is deterministic, and such that every equivalent NFA requires at least $2^n + 1$ states.

Lemma 3. *Let L be the language accepted by 2^n-state NNFA of Fig. 1. Then every NFA for L has at least $2^n + 1$ states.*

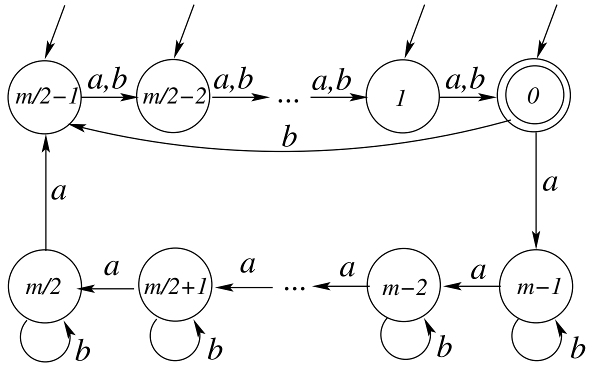

Fig. 1. The NNFA for Lemma 3; $m = 2^n$

Proof. Let $m = 2^n$ and L be the language accepted by the NNFA of Fig. 1. Let N be an NFA (that is, an NNFA with a single initial state) for L. Consider the set of m pairs of strings

$$\mathcal{A} = \{(a^i, a^{m-i}) \mid i = 1, 2, \ldots, m-1\} \cup \{(a^{m-1}b, \varepsilon)\}.$$

For every pair in $\mathcal{A}$, the concatenation of the first part of the pair and its second part results in string a^m or $a^{m-1}b$, both of which are in L. On the other hand, for two distinct pairs in $\mathcal{A}$, the concatenation of the first part of one of the two pairs and the second part of the other pair results in a string in

$$\{a^k \mid m/2 \le k \le m-1 \text{ or } 3m/2 \le k \le 2m-1\} \cup \{a^{m-1}ba^\ell \mid m/2+1 \le \ell \le m-1\}.$$

No such string is in L. It follows that $\mathcal{A}$ is a fooling set for L.

Now fix accepting computations of N on strings $a^i a^{m-i}$ $(1 \le i \le m-1)$ and $a^{m-1}b$, and let p_i $(1 \le i \le m-1)$ and p_m be the states on these computations that are reached after reading a^i resp. $a^{m-1}b$. Since $\mathcal{A}$ is a fooling set for L, states $p_1, \ldots, p_m$ must be pairwise distinct. Now let p_0 be (the sole) initial state of N. Consider the strings $a^{m/2-1}, a, ba^{m/2-1}$ that are in L, so are accepted from state p_0. Then

$$p_0 \notin \{p_n\} \cup \{p_1, \ldots, p_{m/2-1}\}$$

since otherwise one of strings $a^{m-1}b \cdot a^{m/2-1}$ or $a^i \cdot a^{m/2-1}$ with $1 \le i \le m/2-1$ would be accepted by N. However, none of these strings is in L. Next,

$$p_0 \notin \{p_{m/2}, \ldots, p_{m-2}\}$$

since otherwise one of strings $a^j \cdot a$ with $m/2 \le j \le m-2$ would be accepted by N. None of them is in L. Finally, $p_0 \ne p_{m-1}$ since otherwise the string $a^{m-1} \cdot ba^{m/2-1}$, that is not in L, would be accepted by N. Therefore, NFA N must have at least $m+1$ states, and the lemma follows. $\square$

The next result gives the optimal simulation of boolean automata by NFAs. The worst-case language is accepted by an AFA over a two-letter alphabet.

Theorem 1. *Let L be a language accepted by an n-state boolean automaton. Then $2^n + 1$ states are sufficient and necessary in the worst case for nondeterministic finite automata with a single initial state to accept language L. The upper bound is met by an n-state* alternating *automaton over a* binary *alphabet.*

Proof. By Lemma 1, language L is accepted by an 2^n-state NNFA. By adding a new initial state going by the empty string to the initial states of the NNFA, we get an NFA of $2^n + 1$ states for L.

For tightness, consider the language L accepted by the 2^n-state NNFA of Fig. 1. The NNFA has 2^{n-1} initial states, and its reverse is deterministic. By Lemma 2, language L is accepted by an n-state AFA. By Lemma 3, every NFA for L has at least $2^n + 1$ states. This proves the theorem. □

4 Boolean and Alternating State Complexity of Basic Regular Operations

This section examines complexity of basic regular operations on languages represented by boolean and alternating automata.

Recall that the state complexity of a regular language L, $\mathrm{sc}(L)$, is the smallest number of states in any DFA accepting L. Similarly define nondeterministic, alternating, and boolean state complexity of a regular language L, in short $\mathrm{nsc}(L), \mathrm{asc}(L)$, and $\mathrm{bsc}(L)$, respectively, as the smallest number of states in any NFA, AFA, and BFA for L, respectively. The following results are well known.

Lemma 4 ([1,9,3]). *If L is accepted by a boolean automaton of n-states, then L^R is accepted by a DFA of 2^n states. If L is accepted by an AFA of n-states, then L^R accepted by DFA of 2^n states of which 2^{n-1} are final. If* $\mathrm{sc}(L^R) = 2^n$ *then* $\mathrm{bsc}(L) \geq n$. *If* $\mathrm{sc}(L^R) = 2^n$ *and minimal dfa for L^R has more than 2^{n-1} or less than 2^{n-1} final states, then* $\mathrm{asc}(L) \geq n + 1$.

We start with union and intersection, and show that the boolean state complexity of both operations is $m+n$, while their alternating state complexity is $m+n+1$.

Theorem 2 (Union and Intersection on BFAs). *Let K and L be languages over an alphabet Σ with* $\mathrm{bsc}(K) = m$ *and* $\mathrm{bsc}(L) = n$. *Then*
1) $\mathrm{bsc}(K \cup L) \leq m + n$,
2) $\mathrm{bsc}(K \cap L) \leq m + n$,
and both bounds are tight if $|\Sigma| \geq 2$.

Proof. Let languages K and L be accepted by BFAs $(Q_A, \Sigma, \delta_A, g_A, F_A)$ and $(Q_B, \Sigma, \delta_B, g_B, F_B)$, respectively. Let $|Q_A| = m$, $|Q_B| = n$, and $Q_A \cap Q_B = \emptyset$. Then the languages $K \cup L$ and $K \cap L$ are accepted by boolean automata $(Q_A \cup Q_B, \Sigma, \delta, g_A \vee g_B, F_A \cup F_B)$ and $(Q_A \cup Q_B, \Sigma, \delta, g_A \wedge g_B, F_A \cup F_B)$, resp., where $\delta(p, a) = \delta_A(p, a)$ if $p \in Q_A$ and $\delta(p, a) = \delta_B(p, a)$ if $p \in Q_B$.

For tightness, consider languages

$$K^R = \{w \in \{a,b\}^* \mid |w|_a \equiv 0 \bmod 2^m\}, \text{ and}$$
$$L^R = \{w \in \{a,b\}^* \mid |w|_b \equiv 0 \bmod 2^n\}.$$

Both languages are accepted by DFAs of 2^m and 2^n states, respectively. Therefore, languages K and L are accepted by boolean automata of m and n states, respectively. Next we have $(K \cup L)^R = K^R \cup L^R$, and it is known [5,11,14] that the minimal DFA for $K^R \cup L^R$ has 2^{m+n} states. It follows that every boolean automaton for $K \cup L$ has at least $m+n$ states. The same argument holds for intersection. □

Theorem 3 (Union and Intersection on AFAs). *Let K and L be languages over an alphabet Σ with* $\mathrm{asc}(K) = m$ *and* $\mathrm{asc}(L) = n$. *Then*
1) $\mathrm{asc}(K \cup L) \leq m+n+1$,
2) $\mathrm{asc}(K \cap L) \leq m+n+1$,
and both bounds are tight if $|\Sigma| \geq 2$.

Proof. The upper bounds are from [3]. For tightness, consider languages $(K')^R$ and $(L')^R$ accepted by DFAs obtained from the DFAs for K^R and L^R in the previous lemma by making states $2^{m-1}, \ldots, 2^m - 1$ in the DFA for K, and states $2^{n-1}, \ldots, 2^n - 1$ in the DFA for L final. Since both DFAs have half of states final, languages K' and L' are accepted by alternating finite automata of m and n states, respectively. To accept language $(K' \cup L')^R$, we still need 2^{m+n} deterministic states, but this time, the number of final states in the minimal DFA for $(K' \cup L')^R$ is more than 2^{m+n-1}. It follows that the minimal AFA for $(K' \cup L')$ has at least $m+n+1$ states. In the case of intersection, the minimal 2^{m+n}-state DFA for $(K' \cap L')^R$ has less then 2^{m+n-1} final states, so $\mathrm{asc}(K' \cap L') \geq m+n+1$. □

Now we consider concatenation. First we show that its boolean state complexity is $2^m + n$. In the case of alternating state complexity, we get an upper bound $2^m + n + 1$, and a lower bound $2^m + n$.

Theorem 4 (Concatenation on BFAs). *Let K and L be languages over an alphabet Σ with* $\mathrm{bsc}(K) = m$ *and* $\mathrm{bsc}(L) = n$. *Then* $\mathrm{bsc}(KL) \leq 2^m + n$, *and the bound is tight if* $|\Sigma| \geq 2$.

Proof. To prove the upper bound, first transform the BFA for K into an NNFA with 2^m states which accepts language K. Then, using idea in [3, Theorem 9.2], we get a boolean automaton of $2^m + n$ states for language KL.

We now prove tightness. It is well known that the tight bound on the state complexity of concatenation is $(m-1)2^n + 2^{n-1}$ and the bound is met by binary DFA languages [11,14]. Now let L^R and K^R be the Maslov's [11] binary witnesses for concatenation with 2^n and 2^m states, respectively. Then $\mathrm{bsc}(K) \leq m$ and $\mathrm{bsc}(L) \leq n$. Moreover, $\mathrm{sc}(KL)^R = \mathrm{sc}(L^R K^R) = (2^n - 1) \cdot 2^{2^m} + 2^{2^m - 1} \geq 2^{n-1} \cdot 2^{2^m} + 2^{n-1} \cdot 2^{2^m - 1} \geq 2^{n-1} 2^{2^m}(1 + 1/2)$. It follows that $\mathrm{bsc}(KL) \geq \lceil \log(2^{n-1} 2^{2^m}(1 + 1/2)) \rceil = 2^m + n$, and the theorem is proved. □

Theorem 5 (Concatenation on AFAs). *Let K and L be languages over an alphabet Σ with* $\operatorname{asc}(K) = m$ *and* $\operatorname{asc}(L) = n$. *Then* $\operatorname{asc}(KL) \leq 2^m + n + 1$. *The bound* $2^m + n$ *is met if* $|\Sigma| \geq 2$.

Proof. The upper bound is from [3]. For tightness, we need L^R and K^R with half of states final. In such a case, the upper bound for concatenation is $m/2 \cdot 2^n + m/2 \cdot 2^{n-1}$ [14] and is met by binary languages [4]. Now let L^R and K^R be the binary witnesses from [4] with 2^n and 2^m states, respectively, half of which are final in both DFAs. Then $\operatorname{asc}(K) \leq m$ and $\operatorname{asc}(L) \leq n$. The minimal DFA for $(KL)^R = L^R K^R$ has $2^{n-1} \cdot 2^{2^m} + 2^{n-1} \cdot 2^{2^m - 1}$, so $\operatorname{asc}(KL) \geq 2^m + n$. This completes the proof. □

The next two results show that the boolean state complexity of reversal is 2^n, while its alternating state complexity is at least 2^n and at most $2^n + 1$.

Theorem 6 (Reversal on BFAs). *Let L be a language over an alphabet Σ with* $\operatorname{bsc}(L) = n$. *Then* $\operatorname{bsc}(L^R) \leq 2^n$, *and the bound is tight if* $|\Sigma| \geq 2$.

Proof. If L is accepted by an n-state boolean automaton, then L^R is accepted by an 2^n-state NNFA. Thus $\operatorname{bsc}(L^R) \leq 2^n$. The upper bound on the state complexity of reversal of regular languages is 2^n, and it is known to be tight in the binary case [7,9]. Let L^R be the binary witness from [7, Theorem 5] with 2^n states. Then $\operatorname{bsc}(L) \leq n$, and $\operatorname{sc}((L^R)^R) = 2^{2^n}$. Therefore, $\operatorname{bsc}(L^R) \geq 2^n$, and our proof is complete. □

Theorem 7 (Reversal on AFAs). *Let L be a language over an alphabet Σ with* $\operatorname{asc}(L) = n$. *Then* $\operatorname{asc}(L^R) \leq 2^n + 1$. *The bound* 2^n *is met if* $|\Sigma| \geq 2$.

Proof. If L is accepted by an n-state AFA, then L^R is accepted by an $(2^n + 1)$-state NFA. Thus $\operatorname{asc}(L^R) \leq 2^n + 1$. For the lower bound, we need a witness for reversal with half of states final. Such an example is given in [9, Proposition 2] with 2^n states. This proves lower bound and completes the proof. □

The last operation under consideration is star operation The following two theorems show that both boolean and alternating complexity of star are at least 2^n and at most $2^n + 1$.

Theorem 8 (Star on BFAs). *Let L be a language over an alphabet Σ with* $\operatorname{bsc}(L) = n$. *Then* $\operatorname{bsc}(L^*) \leq 2^n + 1$. *The bound* 2^n *is met if* $|\Sigma| \geq 2$.

Proof. If L is accepted by an n-state boolean automaton, then L^* is accepted by an $(2^n + 1)$-state NFA. Thus $\operatorname{bsc}(L^*) \leq 2^n + 1$. The upper bound on the state complexity of star of regular languages is $2^{n-1} + 2^{n-2}$, and is known to be tight in the binary case [14]. Let L^R be the binary witness from [14] with 2^n states. Then $\operatorname{bsc}(L) \leq n$, and

$$\operatorname{sc}((L^*)^R) = \operatorname{sc}((L^R)^*) = 2^{2^n - 1} + 2^{2^n - 2} = 2^{2^n - 1}(1 + 2^{-1}).$$

Hence $\operatorname{bsc}(L^*) \geq 2^n$, and the theorem follows. □

Theorem 9 (Star on AFAs). *Let L be a language over an alphabet Σ with* $\mathrm{asc}(L) = n$. *Then* $\mathrm{asc}(L^*) \leq 2^n + 1$. *The bound* 2^n *is met if* $|\Sigma| \geq 2$.

Proof. The upper bound follows like in the previous theorem. For the lower bound, we need L^R with half of states final. In such a case, the upper bound for star is $2^{n-1} + 2^{n-1-\ell}$, where ℓ is the number of final states different from the initial state. Such a bound is met for every ℓ ($1 \leq \ell \leq n-1$) in the binary case [6]. Let L^R be the binary witness from [6, Theorem 5] with 2^n states and $\ell = 2^{n-1}$ final states distinct from the initial state. Then $\mathrm{asc}(L) \leq n$, and

$$\mathrm{sc}((L^*)^R) = \mathrm{sc}((L^R)^*) = 2^{2^n-1} + 2^{2^n-1-n/2} = 2^{2^n-1}(1 + 2^{-n/2}).$$

Thus $\mathrm{asc}(L^*) \geq 2^n$, which proves the theorem. □

5 Conclusions

We proved the tight bound on the number of states of nondeterministic finite automaton with a single initial state that are sufficient and necessary in the worst case to simulate a boolean or alternating finite automaton of n states is $2^n + 1$. This solves an open problem from [3].

Then we examined the boolean and alternating state complexity of basic regular operations. Our results are summarised in the following table. All the worst-case examples are defined over a binary alphabet. The tightness of the upper bounds for star of BFAs and AFAs, as well as for concatenation and reversal of AFAs, remains open.

Table 2. Boolean and alternating state complexity of basic regular operations

	union	intersection	concatenation	reversal	star
BFAs	$m+n$	$m+n$	2^m+n	2^n	$\geq 2^n$ $\leq 2^n+1$
AFAs	$m+n+1$	$m+n+1$	$\geq 2^m+n$ $\leq 2^m+n+1$	$\geq 2^n$ $\leq 2^n+1$	$\geq 2^n$ $\leq 2^n+1$

Acknowledgement. I would like to thank Markus Holzer and Martin Kutrib for reading a preliminary version of the proof of Theorem 1, and for interesting discussions on the topic.

References

1. Brzozowski, J., Leiss, E.: On equations for regular languages, finite automata, and sequential networks. Theoret. Comput. Sci. 10, 19–35 (1980)
2. Chandra, A.K., Kozen, D.C., Stockmeyer, L.J.: Alternation. JACM 28, 114–133 (1981)

3. Fellah, A., Jürgensen, H., Yu, S.: Constructions for alternating finite automata. Intern. J. Computer Math. 35, 117–132 (1990)
4. Jirásek, J., Jirásková, G., Szabari, A.: State complexity of concatenation and complementation. Internat. J. Found. Comput. Sci. 16, 511–529 (2005)
5. Jirásková, G., Masopust, T.: Complexity in Union-Free Regular Languages. In: Gao, Y., Lu, H., Seki, S., Yu, S. (eds.) DLT 2010. LNCS, vol. 6224, pp. 255–266. Springer, Heidelberg (2010)
6. Jirásková, G., Masopust, T.: State Complexity of Projected Languages. In: Holzer, M., Kutrib, M., Pighizzini, G. (eds.) DCFS 2011. LNCS, vol. 6808, pp. 198–211. Springer, Heidelberg (2011)
7. Jirásková, G., Šebej, J.: Note on Reversal of Binary Regular Languages. In: Holzer, M., Kutrib, M., Pighizzini, G. (eds.) DCFS 2011. LNCS, vol. 6808, pp. 212–221. Springer, Heidelberg (2011)
8. Kozen, D.: On parallelism in turing machines. In: Proc. 17th FOCS, pp. 89–97 (1976)
9. Leiss, E.: Succinct representation of regular languages by boolean automata. Theoret. Comput. Sci. 13, 323–330 (1981)
10. Leiss, E.: On generalized language equations. Theoret. Comput. Sci. 14, 63–77 (1981)
11. Maslov, A.N.: Estimates of the number of states of finite automata. Soviet Math. Doklady 11, 1373–1375 (1970)
12. Sipser, M.: Introduction to the theory of computation. PWS Publishing Company, Boston (1997)
13. Yu, S.: Regular languages. In: Rozenberg, G., Salomaa, A. (eds.) Handbook of Formal Languages, ch. 2, vol. I, pp. 41–110. Springer, Heidelberg (1997)
14. Yu, S., Zhuang, Q., Salomaa, K.: The state complexity of some basic operations on regular languages. Theoret. Comput. Sci. 125, 315–328 (1994)