IFIG

RESEARCH

REPORT

INSTITUT FÜR INFORMATIK

UNARY LANGUAGE OPERATIONS AND THEIR NONDETERMINISTIC STATE COMPLEXITY

Markus Holzer   Martin Kutrib

IFIG RESEARCH REPORT 0107

NOVEMBER 2001

Institut für Informatik
JLU Gießen
Arndtstraße 2
D-35392 Giessen, Germany
Tel: +49-641-99-32141
Fax: +49-641-99-32149
mail@informatik.uni-giessen.de
www.informatik.uni-giessen.de

JUSTUS-LIEBIG-UNIVERSITÄT GIESSEN

IFIG Research Report
IFIG Research Report 0107, November 2001

Unary Language Operations and Their Nondeterministic State Complexity

Markus Holzer[1]

Institut für Informatik, Technische Universität München
Arcisstraße 21, D-80290 München, Germany

Martin Kutrib[2]

Institut für Informatik, Universität Giessen
Arndtstr. 2, D-35392 Giessen, Germany

Abstract. We investigate the costs, in terms of states, of operations on infinite and finite unary regular languages where the languages are represented by nondeterministic finite automata. In particular, we consider Boolean operations, concatenation, iteration, and λ-free iteration. Most of the bounds are tight in the exact number of states, i.e. the number is sufficient and necessary in the worst case. For the complementation of infinite languages a tight bound in the order of magnitude is shown.

CR Subject Classification (1998): F.1, F.4.3

[1]E-mail: holzer@informatik.tu-muenchen.de
[2]E-mail: kutrib@informatik.uni-giessen.de

1 Introduction

Finite automata are used in several applications and implementations in software engineering, programming languages and other practical areas in computer science. They are one of the first and most intensely investigated computational models. Nevertheless, some challenging problems of finite automata are still open. An important example is the question how many states are sufficient and necessary to simulate two-way nondeterministic finite automata with two-way deterministic finite automata. The problem has been raised in [14] and partially solved in [17]. A lower bound and an interesting connection with the open problem whether DLOGSPACE equals NLOGSPACE or not is given in [1].

Since regular languages have many representations in the world of finite automata it is natural to investigate the succinctness of their representation by different types of automata in order to optimize the space requirements. It is well known that nondeterministic finite automata (NFA) can offer exponential saving in space compared with deterministic finite automata (DFA), but the problem to convert a given DFA to an equivalent minimal NFA is PSPACE-complete [7]. Since minimization of NFAs is also PSPACE-complete, conversions from nondeterministic to deterministic variants are of particular interest. Concerning the number of states asymptotically tight bounds are $O(n^n)$ for the two-way DFA to one-way DFA conversion, $O(2^{n^2})$ for the two-way NFA to one-way DFA conversion, and 2^n for the one-way NFA to one-way DFA conversion, for example. For finite languages over a k-letter alphabet the NFA to DFA conversion has been solved in [15] with a tight bound of $O(k^{\frac{n}{\log_2 k+1}})$. A valuable source for further results and references is [2].

Related to these questions are the costs (in terms of states) of operations on regular languages with regard to their representing devices. For example, converting a given NFA to an equivalent DFA gives an upper bound for the NFA state complexity of complementation. In recent years results for many operations have been obtained. For DFAs state-of-the-art surveys can be found in [20, 21]. The general nondeterministic case has been studied in [4].

When certain problems are computationally hard in general, a natural question concerns simpler versions. To this regard promising research has been done for unary languages. It turned out that this particular case is essentially different from the general case. For example, the minimization of NFAs becomes NP-complete instead of PSPACE-complete [6, 18]. The problem of evaluating the costs of unary automata simulations has been raised in [17]. In [3] it has been shown that the unary NFA to DFA conversion takes $e^{\Theta(\sqrt{n\cdot\ln(n)})}$ states, the NFA to two-way DFA conversion has been solved with a bound of $O(n^2)$ states, and the costs of the unary two-way to one-way DFA conversion reduces to $e^{\Theta(\sqrt{n\cdot\ln(n)})}$. Several more results can be found in [10, 11].

State complexity results concerning operations on unary regular languages represented by DFAs are covered by the surveys [20, 21]. Estimations of the average state complexity are shown in [12].

Here we investigate the costs (in terms of states) of operations on infinite and finite unary regular languages represented by NFAs. In particular, we consider Boolean operations and catenation operations. The reversal of a unary language is trivial. Most of the bounds are tight in the exact number of states, i.e. the number is sufficient and necessary in the worst case. For the complementation of infinite languages a tight bound in the order of magnitude is shown. The technical depth of our results varies from immediate to more subtle extensions to previous work. Indeed the technique to prove minimality for DFAs is not directly applicable to the case of NFAs. Therefore, we mostly have to use counting arguments to prove our results on NFA minimality with respect to the number of states.

In the next section we define the basic notions and present preliminary results. Section 3 is devoted to the study of infinite languages. The NFAs for finite unary languages essentially have a simple structure. By this observation the results for finite languages are derived in Section 4.

2 Preliminaries

We denote the positive integers $\{1, 2, \dots\}$ by $\mathbb{N}$, the set $\mathbb{N} \cup \{0\}$ by $\mathbb{N}_0$, and the *powerset* of a set S by 2^S. The *empty word* is denoted by λ. For the *length of w* we write $|w|$. We use $\subseteq$ for inclusions and $\subset$ if the inclusion is strict. By $\gcd(x_1, \dots, x_k)$ we denote the *greatest common divisor* of the integers $x_1, \dots, x_k$, and by $\text{lcm}(x_1, \dots, x_n)$ their *least common multiple*. If two numbers x and y are relatively prime (i.e. $\gcd(x, y) = 1$) we write $x \perp y$.

Definition 1 *A* nondeterministic finite automaton (NFA) *is a system* $\mathcal{A} = \langle S, A, \delta, s_0, F \rangle$, *where*

1. *S is the finite set of* internal states,
2. *A is the finite set of* input symbols,
3. *$s_0 \in S$ is the* initial state,
4. *$F \subseteq S$ is the set of* accepting states, *and*
5. *$\delta : S \times A \to 2^S$ is the* transition function.

The set of rejecting states is implicitly given by the partitioning, i.e. $S \setminus F$.

An NFA is called *unary* if its set of input symbols is a singleton. Throughout the paper we use $A = \{a\}$. In some sense the transition function is complete. W.l.o.g. we may require δ to be a total function, since whenever the operation of an NFA is supposed not to be defined, then δ can map to the empty set which, trivially, belongs to 2^S. Thus, in some sense the NFA need not be complete. However, throughout the paper we assume that the NFAs are always *reduced*. This means that there are no unreachable states and that from any state an accepting state can be reached. An NFA is said to be *minimal* if its number of states is minimal with respect to the accepted language. Since every n-state NFA with λ-transitions can be transformed to an equivalent n-state NFA without λ-transitions [5] for state complexity issues there is no difference between the absence and presence of λ-transitions. For convenience, we consider NFAs without λ-transitions only.

As usual the transition function δ is extended to a function $\Delta : S \times A^* \to 2^S$ reflecting sequences of inputs as follows: $\Delta(s, wa) = \bigcup_{s' \in \Delta(s,w)} \delta(s', a)$, where $\Delta(s, \lambda) = \{s\}$ for $s \in S$, $a \in A$, and $w \in A^*$.

Let $\mathcal{A} = \langle S, A, \delta, s_0, F \rangle$ be an NFA, then a word $w \in A^*$ is *accepted* by $\mathcal{A}$ if $\Delta(s_0, w) \cap F \neq \emptyset$. The *language accepted* by $\mathcal{A}$ is:

$$L(\mathcal{A}) = \{w \in A^* \mid w \text{ is accepted by } \mathcal{A}\}$$

The next preliminary result involves NFAs directly. It is a key tool in the following sections, and can be proved by a simple pumping argument.

Lemma 2 *Let $n \geq 1$ be an arbitrary integer. Then $n+1$ resp. n states are sufficient and necessary in the worst case for an* NFA *to accept the language $\{a^n\}^+$ resp. $\{a^n\}^*$.*

3 Operations on Regular Languages

3.1 Boolean Operations

We start our investigations with Boolean operations on NFAs that accept unary regular languages. At first we consider the union. Due to the lack of suited tools and methods such as unique minimization etc. the proof of the lower bound refers specifically to the structures of the witness automata.

Theorem 3 *For any integers $m, n \geq 1$ let $\mathcal{A}$ be a unary m-state and $\mathcal{B}$ be a unary n-state* NFA*. Then $m+n+1$ states are sufficient for an* NFA *$\mathcal{C}$ to accept the language $L(\mathcal{A}) \cup L(\mathcal{B})$.*

Proof. To construct an $(m+n+1)$-state NFA for the language $L(\mathcal{A}) \cup L(\mathcal{B})$, we simply use a new initial state and connect it to the states of $\mathcal{A}$ and $\mathcal{B}$ that are reached after the first state transition. During the first transition, $\mathcal{C}$ nondeterministically guesses whether the input may belong to $L(\mathcal{A})$ or $L(\mathcal{B})$. Subsequently, $\mathcal{A}$ or $\mathcal{B}$ is simulated. Obviously, $L(\mathcal{C}) = L(\mathcal{A}) \cup L(\mathcal{B})$ and $|S| = |S_A| + |S_B| + 1 = m + n + 1$. □

Theorem 4 *Let $m, n > 1$ be two integers such that neither m is a multiple of n nor n is a multiple of m. Then there exist a unary m-state* NFA *$\mathcal{A}$ and a unary n-state* NFA *$\mathcal{B}$ such that any* NFA *$\mathcal{C}$ accepting $L(\mathcal{A}) \cup L(\mathcal{B})$ needs at least $m + n + 1$ states.*

Proof. W.l.o.g. we may assume $m > n$. Since m is not a multiple of n we obtain $n > \gcd(m, n)$.

Let $\mathcal{A}$ be an m-state NFA that accepts the language $\{a^m\}^*$ and $\mathcal{B}$ be an n-state NFA that accepts $\{a^n\}^*$. Let $\mathcal{C}$ be an NFA with initial state s_0 for the language

$L(\mathcal{A}) \cup L(\mathcal{B})$. In order to prove that $\mathcal{C}$ has at least $m+n+1$ states assume that it has at most $m+n$ states.

At first we consider the input a^{m+n} and show that it does not belong to $L(\mathcal{C})$. Since $2m > m+n > m$ the input does not belong to $L(\mathcal{A})$. If it would belong to $L(\mathcal{B})$, then there were a constant $c > 2$ such that $m+n = c \cdot n$. Therefore, $m = (c-1) \cdot n$ and, thus, m would be a multiple of n what contradicts the assumption of the theorem.

The next step is to show that each of the states $s_0 \vdash s_1 \vdash \cdots \vdash s_{m-1} \vdash s_m$ which are passed through when accepting the input a^m either is not in a cycle or is in a cycle of length m. To this end assume contrarily some state s_i is in a cycle of length $x \neq m$, i.e. s_i is reachable from s_i by processing x input symbols. Due to the number of states we may assume $x \leq m+n$.

By running several times through the cycle a^{m+x}, a^{m+2x} and a^{m+3x} are accepted.

We observe $m+x$ is not a multiple of m. If it is not a multiple of n we are done. Otherwise, x cannot be a multiple of n since m is not a multiple of n. Moreover, now we observe $m+2x$ is not a multiple of n since $m+x$ is and x is not. If $m+2x$ is not a multiple of of m we are done.

Otherwise we consider $m+3x$ which now cannot be a multiple of m since $m+2x$ is and x is not. If $m+3x$ is not a multiple of n we are done.

Otherwise we summarize the situation: $m+3x$ and $m+x$ are multiples of n which implies $2x$ is a multiple of n. Since $m+2x$ is a multiple of m we conclude that $2x$ is a multiple of m, too. Moreover, x is neither a multiple of m nor of n.

From $x \leq m+n < 2m \Longrightarrow 2x < 4m$ we derive $2x \in \{m, 3m\}$. If $2x = m$, then m is a multiple of n, a contradiction. So let $2x = 3m$. Since $2x$ is a multiple of n there exists a constant $c \in \mathbb{N}$ such that $3m = cn$. From $x \leq m+n$ follows $\frac{3}{2}m \leq m+n$. Therefore $\frac{1}{2}m \leq n$ which together with $n < m$ implies $c \in \{4, 5, 6\}$.

It holds $\frac{3}{c}m = n$. On the other hand, $m + x = m + \frac{3}{2}m = \frac{5}{2}m$ is a multiple of n. Therefore $\frac{5}{2}/\frac{3}{c} = \frac{5 \cdot c}{6}$ must belong to $\mathbb{N}$ for $c \in \{4, 5, 6\}$. Thus $c = 6$, but in this case $\frac{1}{2}m = n \Longrightarrow m = 2n$ and m is a multiple of n what contradicts the assumption of the theorem.

In order to complete the proof of the theorem now we come back to the sequence of states $s_0 \vdash s_1 \vdash \cdots \vdash s_{m-1} \vdash s_m$ passed through when accepting the input a^m. Correspondingly let $s_0 \vdash s'_1 \vdash \cdots \vdash s'_{n-1} \vdash s'_n$ be an accepting sequence for a^n.

Since $L(\mathcal{C})$ is an infinite language $\mathcal{C}$ must contain some cycle. If a s_i from $s_0, \ldots, s_m$ is in a cycle, then its length is m states. State s_i may not be in the sequence $s_0 \vdash s'_1 \vdash \cdots \vdash s'_n$ because in this case a^{m+n} would be accepted. In order to accept a^n but to reject $a^1, \ldots, a^{n-1}$ in any case, the states $s_0, s'_1, \ldots, s'_n$ must be pairwise different. Altogether this results in at least $n+1+m$ states.

If on the other hand a s_j from $s_0, s'_1, \ldots, s'_n$ is in a cycle, then it may not be in $s_0, s_1, \ldots, s_m$. Otherwise the cycle length must be m and a^{m+n} would be accepted. Concluding as before this case results in at least $m+n+1$ states, too.

Finally, if neither a state from $s_0, s_1, \dots, s_m$ nor a state from $s'_1, \dots, s'_n$ is in a cycle, then obviously $s_0, \dots, s_m$ must be pairwise different, but all states s'_j may or may not appear in $s_0, \dots, s_m$. This takes at least $m+1$ states. So there remain at most $n-1$ states for a cycle. Therefore, the cycle length x is at most $n-1$. Now we consider an accepting computation for the input a^{mn}. It must have run through the cycle. Running once more through the cycle leads to acceptance of a^{mn+x} which does not belong to $L(\mathcal{C})$.

So in any case we obtain a contradiction to the assumption that $\mathcal{C}$ has at most $m+n$ states. □

Next we are going to prove a tight bound for the intersection. The upper bound is obtained by the simple cross-product construction. The lower bound requires $m \perp n$. In [13] unary languages are studied whose *deterministic* state complexities are not relatively prime.

Theorem 5 *For any integers $m, n \geq 1$ let $\mathcal{A}$ be a unary m-state and $\mathcal{B}$ be a unary n-state* NFA. *Then $m \cdot n$ states are sufficient for an* NFA *$\mathcal{C}$ to accept the language $L(\mathcal{A}) \cap L(\mathcal{B})$. If $m \perp n$, then there exist a unary m-state* NFA *$\mathcal{A}$ and a unary n-state* NFA *$\mathcal{B}$ such that any* NFA *$\mathcal{C}$ accepting $L(\mathcal{A}) \cap L(\mathcal{B})$ needs at least $m \cdot n$ states.*

Proof. As witness languages consider $L(\mathcal{A}) = \{a^m\}^*$ and $L(\mathcal{B}) = \{a^n\}^*$. Since $m \perp n$ the intersection $L(\mathcal{A}) \cap L(\mathcal{B})$ is $\{a^{m \cdot n}\}^*$. Due to Lemma 2 any NFA accepting the intersection needs at least $m \cdot n$ states. □

For complementation of unary NFAs a crucial role is played by the function $F(n) = \max\{\mathrm{lcm}(x_1, \dots, x_k) \mid x_1, \dots, x_k \in \mathbb{N} \wedge x_1 + \cdots + x_k = n\}$ which gives the maximal order of the cyclic subgroups of the symmetric group of n symbols. For example, the first seven values of F are $F(1) = 1$, $F(2) = 2$, $F(3) = 3$, $F(4) = 4$, $F(5) = 6$, $F(6) = 6$, $F(7) = 12$ due to the sums $1 = 1$, $2 = 2$, $3 = 3$, $4 = 4$, $5 = 2 + 3$, $6 = 1 + 2 + 3$ (or $6 = 6$) and $7 = 3 + 4$.

Since F depends on the irregular distribution of the prime numbers we cannot expect to express $F(n)$ explicitly by n. The function itself has been investigated by Landau [8, 9] who has proved the asymptotic growth rate

$$\lim_{n \to \infty} \frac{\ln(F(n))}{\sqrt{n \cdot \ln(n)}} = 1$$

A bound immediately derived from Landau's result is:

$$\ln(F(n)) \in \Theta(\sqrt{n \cdot \ln(n)})$$

For our purposes the implied rough estimation $F(n) \in e^{\Theta(\sqrt{n \cdot \ln(n)})}$ suffices. Shallit and Ellul [16] pointed out that the bound $F(n) \in O(e^{\sqrt{n \cdot \ln(n)}})$ which is claimed in [3] is not correct. They deduced finer bounds from a result in [19] where the currently best known approximation for F has been proved. Nevertheless, in [3] it has been shown that for any unary n-state NFA there exists an equivalent $O(F(n))$-state deterministic finite automaton.

This upper bound for the transformation to a deterministic finite automaton is also an upper bound for the costs of the unary NFA complementation, since the unary deterministic complementation neither increases nor decreases the number of states (simply interchange accepting and rejecting states). The next theorem is an immediate consequence.

Theorem 6 *For any integer $n \geq 1$ the complement of a unary n-state* NFA *language is accepted by an $O(F(n))$-state* NFA.

This expensive upper bound is tight in the order of magnitude.

Theorem 7 *For any integer $n > 1$ there exists a unary n-state* NFA $\mathcal{A}$ *such that any* NFA *accepting the complement of $L(\mathcal{A})$ needs at least $\Omega(F(n))$ states.*

Proof. Let $x_1, \ldots, x_k \in \mathbb{N}$ be integers such that $x_1 + \cdots + x_k = n - 1$ and $\text{lcm}(x_1, \ldots, x_k) = F(n-1)$. If one of the integers x_i may be factorized as $x_i = y \cdot z$ such that $y, z > 1$ and $y \perp z$, then x_i can be replaced by $y, z, 1, 1, \ldots, 1$ where the number of ones is $y \cdot z - (y + z)$. Obviously neither the sum nor the least common multiple is affected. So we may assume that all x_i are powers of prime numbers. If two of the integers are powers of the same prime number, then the smaller one is replaced by a corresponding number of ones. Again neither the sum nor the least common multiple is affected. But now we may assume without loss of generality that all x_i are relatively prime, i.e. $x_i \perp x_j$ for $i \neq j$.

Now define for $1 \leq i \leq k$ the languages $L_i = \{a^{x_i}\}^*$ and consider the union of their complements: $L = \overline{L_1} \cup \overline{L_2} \cup \cdots \cup \overline{L_k}$.

Since $\overline{L_i}$ is acceptable by a x_i-state NFA $\mathcal{A}_i$ the language L is acceptable by an NFA $\mathcal{A}$ with at most $1 + x_1 + \cdots + x_k = n$ states. To this end we introduce a new initial state and connect it nondeterministically to the states of the $\mathcal{A}_i$ that are reached after their first state transition. Since all x_i are relatively prime the complement of L is $\overline{L} = \{a^{\text{lcm}(x_1,\ldots,x_k)}\}^*$. Therefore, the complement of the n-state language L needs $\text{lcm}(x_1, \ldots, x_k) = F(n-1)$ states. Since $F(n)$ is of order $e^{\Theta(\sqrt{n \cdot \ln(n)})}$ it follows $F(n-1)$ is of order $\Omega(F(n))$. $\square$

3.2 Catenation Operations

The lower bound of the concatenation misses the upper bound by one state. It is an open question how to close the gap by more sophisticated constructions or witness languages.

Theorem 8 *For any integers $m, n > 1$ let $\mathcal{A}$ be a unary m-state* NFA *and $\mathcal{B}$ be a unary n-state* NFA. *Then $m + n$ states are sufficient for an* NFA $\mathcal{C}$ *to accept the language $L(\mathcal{A})L(\mathcal{B})$. Moreover, there exist a unary m-state* NFA $\mathcal{A}$ *and a unary n-state* NFA $\mathcal{B}$ *such that any* NFA $\mathcal{C}$ *accepting $L(\mathcal{A})L(\mathcal{B})$ needs at least $m + n - 1$ states.*

Proof. The upper bound is due to the observation that in $\mathcal{C}$ one has simply to connect the accepting states in $\mathcal{A}$ with the states in $\mathcal{B}$ that follow the initial state.

Let $\mathcal{A} = \langle S_A, \{a\}, \delta_A, s_{0,A}, F_A\rangle$ and $\mathcal{B} = \langle S_B, \{a\}, \delta_B, s_{0,B}, F_B\rangle$ such that $S_A \cap S_B = \emptyset$, then $\mathcal{C} = \langle S, \{a\}, \delta, s_0, F\rangle$ is defined according to $S = S_A \cup S_B$, $s_0 = s_{0,A}$, $F = F_A \cup F_B$ if $\lambda \in L(\mathcal{B})$, $F = F_B$ otherwise, and for all $s \in S$:

$$\delta(s,a) = \begin{cases} \delta_A(s,a) & \text{if } s \in S_A \setminus F_A \\ \delta_A(s,a) \cup \delta_B(s_{0,B},a) & \text{if } s \in F_A \\ \delta_B(s,a) & \text{if } s \in S_B \end{cases}$$

Let $L(\mathcal{A})$ be the m-state language $\{a^k \mid k \equiv m-1 \pmod{m}\}$ and $L(\mathcal{B})$ be the n-state language $\{a^k \mid k \equiv n-1 \pmod{n}\}$.

The shortest word in $L(\mathcal{A})$ respectively $L(\mathcal{B})$ is a^{m-1} respectively a^{n-1}. Therefore the shortest word in $L(\mathcal{C})$ is a^{m+n-2}. Assume contrarily to the assertion the NFA $\mathcal{C}$ has at most $m+n-2$ states. Let $\mathcal{C}$ accept the input a^{m+n-2} by running through the state sequence $s_0 \vdash s_1 \vdash \cdots \vdash s_{m+n-2}$ where all states except s_{m+n-2} are non-accepting. Due to the assumption at least one of the non-accepting states s_i must appear at least twice in the sequence. This implies that there exists an accepting computation that does not run through the cycle $s_i \vdash \cdots \vdash s_i$. So an input whose length is at most $m+n-3$ would be accepted, a contradiction. □

The constructions yielding the upper bounds for the iteration and λ-free iteration are similar. The trivial difference between both operations concerns the empty word only. Moreover, the difference does not appear for languages containing the empty word. Nevertheless, in the worst case the difference costs one state.

Theorem 9 *For any integer $n > 2$ let $\mathcal{A}$ be a unary n-state* NFA. *Then $n+1$ resp. n states are sufficient and necessary in the worst case for an* NFA *to accept the language $L(\mathcal{A})^*$ resp. $L(\mathcal{A})^+$.*

Proof. Let $\mathcal{A} = \langle S_A, \{a\}, \delta_A, s_{0,A}, F_A\rangle$ be an n-state NFA. Then the transition function of an n-state NFA $\mathcal{C} = \langle S, \{a\}, \delta, s_0, F\rangle$ that accepts the language $L(\mathcal{A})^+$ is for $s \in S$ defined as follows:

$$\delta(s,a) = \begin{cases} \delta_A(s,a) & \text{if } s \notin F_A \\ \delta_A(s,a) \cup \delta_A(s_{0,A},a) & \text{if } s \in F_A \end{cases}$$

The other components remain unchanged, i.e., $S = S_A$, $s_0 = s_{0,A}$, and $F = F_A$. If the empty word belongs to $L(\mathcal{A})$ then the construction works fine for $L(\mathcal{A})^*$ also. Otherwise an additional state has to be added: Let $s_0' \notin S_A$ and define $S = S_A \cup \{s_0'\}$, $s_0 = s_0'$, $F = F_A \cup \{s_0'\}$ and for $s \in S$

$$\delta(s,a) = \begin{cases} \delta_A(s,a) & \text{if } s \notin F_A \cup \{s_0'\} \\ \delta_A(s,a) \cup \delta_A(s_{0,A},a) & \text{if } s \in F_A \\ \delta_A(s_{0,A},a) & \text{if } s = s_0' \end{cases}$$

In order to prove the tightness of the bounds for any $n > 2$ consider the n-state language $L = \{a^k \mid k \equiv n-1 \pmod{n}\}$.

At first we show that $n+1$ states are necessary for $\mathcal{C} = \langle S, \{a\}, \delta, s_0, F\rangle$ to accept $L(\mathcal{A})^*$.

Contrarily, assume $\mathcal{C}$ has at most n states. We consider words of the form a^i with $0 \leq i$. The shortest four words belonging to $L(\mathcal{A})^*$ are λ, a^{n-1}, a^{2n-2}, and a^{2n-1}. It follows $s_0 \in F$. Moreover, for a^{n-1} there must exist a path $s_0 \vdash s_1 \vdash \cdots \vdash s_{n-2} \vdash s_n$ where $s_n \in F$ and $s_1, \ldots, s_{n-2}$ are different non-accepting states. Thus, $\mathcal{C}$ has at least $n-2$ non-accepting states.

Assume for a moment F to be a singleton. Then $s_0 = s_n$ and for $1 \leq i \leq n-3$ the state s_0 must not belong to $\delta(s_i, a)$. Processing the input a^{2n-1} the NFA cannot enter s_0 after $2n-2$ time steps. Since $a^1 \notin L(\mathcal{A})^*$ the state s_0 must not belong to $\delta(s_0, a)$.

On the other hand, $\mathcal{C}$ cannot enter one of the states $s_1, \ldots, s_{n-3}$ since there is no transition to s_0. We conclude that $\mathcal{C}$ is either in state s_{n-2} or in an additional non-accepting state s_{n-1}. Since there is no transition such that $s_{n-2} \in \delta(s_{n-2}, a)$ in both cases there exists a path of length n from s_0 to s_0. But a^n does not belong to $L(\mathcal{A})^*$ and we have a contradiction to the assumption $|F| = 1$.

Due to our assumption $|S| \leq n$ we now have $|F| = 2$ and $|S| - |F| = n-2$. Let us recall the accepting sequence of states for the input a^{n-1}: $s_0 \vdash s_1 \vdash \cdots \vdash s_{n-2} \vdash s_n$. Both s_0 and s_n must be accepting states. Assume $s_n \neq s_0$. Since a^{2n-2} belongs to $L(\mathcal{A})^*$ there must be a possible transition $s_0 \vdash s_1$ or $s_n \vdash s_1$. Thus, a^{2n-2} is accepted by s_n. In order to accept a^{2n-1} there must be a corresponding transition from s_n to s_n or from s_n to s_0. In both cases the input a^n would be accepted. Therefore $s_n = s_0$.

By the same argumentation the necessity of a transition for the input symbol a from s_0 to s_0 or from s_0 to s_n follows. This implies that a^1 is accepted. From the contradiction follows $|S| > n$.

As an immediate consequence we obtain the tightness of the bound for $L(\mathcal{A})^+$. In this case $s_0 \in F$ is not required. Thus, just one accepting state is necessary.

□

4 Operations on Finite Languages

The situation for finite unary languages is easier since essentially the structure of the corresponding NFAs is simple.

In [15] it is stated that every finite unary n-state NFA language is acceptable by some complete deterministic finite automaton with at most $n+1$ states. A little bit more sophisticated, one observes that the minimum NFA for a finite unary language L has $n+1$ states if the longest word in L is of length n. Otherwise the NFA would run through a cycle when accepting a^n and, thus, L would be infinite. Now we can always construct a minimum NFA $\mathcal{A} = \langle S, \{a\}, \delta, s_0, F\rangle$

for L as follows: $S = \{s_0, s_1, \ldots, s_n\}$, $\delta(s_i, a) = \{s_{i+1}\}$ for $0 \leq i \leq n-1$, and $F = \{s_i \mid a^i \in L\}$.

From the construction it follows conversely that a minimum $(n+1)$-state NFA for a non-empty finite unary language accepts the input a^n. An immediate consequence is that we have only to consider the longest words in the languages in order to obtain the state complexity of operations that preserve the finiteness.

Theorem 10 *For any integers $m, n \geq 1$ let $\mathcal{A}$ be a unary m-state* NFA *and $\mathcal{B}$ be a unary n-state* NFA*. If $L(\mathcal{A})$ and $L(\mathcal{B})$ are finite, then* $\max(m,n)$, $\min(m,n)$ *respectively $m+n-1$ states are sufficient and necessary in the worst case for an* NFA $\mathcal{C}$ *to accept the language $L(\mathcal{A}) \cup L(\mathcal{B})$, $L(\mathcal{A}) \cap L(\mathcal{B})$ respectively $L(\mathcal{A})L(\mathcal{B})$.*

The complementation and iteration applied to finite languages yield to infinite languages. So in general for the lower bounds we cannot argue with the simple chain structure as before.

Theorem 11 *For any integer $n \geq 1$ let $\mathcal{A}$ be an n-state* NFA*. If $L(\mathcal{A})$ is finite, then $n+1$ states are sufficient and necessary in the worst case for an* NFA $\mathcal{C}$ *to accept the complement of $L(\mathcal{A})$.*

Proof. W.l.o.g. we may assume that $\mathcal{A}$ has the simple chain structure with states from s_0 to s_{n-1} as mentioned before. By interchanging accepting and non-accepting states we obtain an NFA that processes all inputs up to a length $n-1$ as required. But all longer words a^k, $k \geq n$, are belonging to the complement of $L(\mathcal{A})$. So it suffices to add a new accepting state s_n and two transitions from s_{n-1} to s_n and from s_n to s_n, in order to complete the construction of $\mathcal{C}$.

The tightness of the bound can be seen for the n-state NFA language $L = \{a^k \mid 0 \leq k \leq n-1\}$. Since a^n is the shortest word belonging to the complement of L from the proof of Theorem 8 follows that $\mathcal{C}$ has at least $n+1$ states. □

The state complexity for the iterations in the finite language case is as for infinite languages if the iteration is λ-free. If not the costs are reduced by two states.

Lemma 12 *For any integer $n > 1$ let $\mathcal{A}$ be a unary n-state* NFA*. If $L(\mathcal{A})$ is finite, then $n-1$ resp. n states are sufficient and necessary in the worst case for an* NFA *to accept the language $L(\mathcal{A})^*$ resp. $L(\mathcal{A})^+$.*

Proof. For the upper bounds we can adapt the construction of Theorem 9. The accepting states are connected to the states following the initial state. That is all for λ-free iterations.

For iterations we have to provide acceptance of the empty word. The following two observations let us save two states compared with infinite languages. First, the initial state is never reached again after initial time. Second, since the underlying language is finite and the accepting automaton is reduced there must exist an accepting state s_f for which the state transition is not defined. We can take s_f as new initial state and delete the old initial state what altogether leads to an $(n-1)$-state NFA for the iteration.

The bound for the λ-free iteration is reached for the language $L = \{a^{n-1}\}$ which requires n states. For the accepting $L^+ = \{a^{n-1}\}^+$ at least n states are necessary.

The bound for the iteration is reached for the language $L = \{a^n\}$ that requires $n+1$ states. Clearly, in order to accept $\{a^n\}^*$ at least n states are necessary. $\square$

For the sake of completeness the trivial bounds for the reversal are adduced in Table 1. The results concerning DFAs are covered by [20, 21].

	NFA		DFA	
	finite	infinite	finite	infinite
$\cup$	$\max\{m,n\}$	$m+n+1$	$\max\{m,n\}$	mn
$\sim$	$n+1$	$e^{\Theta(\sqrt{n\cdot\ln(n)})}$	n	n
$\cap$	$\min\{m,n\}$	mn	$\min\{m,n\}$	mn
R	n	n	n	n
$\cdot$	$m+n-1$	$O(m+n)$	$m+n-2$	mn
$*$	$n-1$	$n+1$	$n^2-7n+13$	$(n-1)^2+1$
$+$	n	n		

Table 1: Comparison of unary NFA and unary DFA state complexities.

References

[1] Berman, P. and Lingas, A. *On the complexity of regular languages in terms of finite automata*. Technical Report 304, Polish Academy of Sciences, 1977.

[2] Birget, J.-C. *State-complexity of finite-state devices, state compressibility and incompressibility*. Mathematical Systems Theory 26 (1993), 237–269.

[3] Chrobak, M. *Finite automata and unary languages*. Theoretical Computer Science 47 (1986), 149–158.

[4] Holzer, M. and Kutrib, M. *State complexity of basic operations on nondeterministic finite automata*. Implementation and Application of Automata (CIAA '02), 2002, pp. 151–160.

[5] Hopcroft, J. E. and Ullman, J. D. *Introduction to Automata Theory, Language, and Computation*. Addison-Wesley, Reading, Massachusetts, 1979.

[6] Hunt, H. B., Rosenkrantz, D. J., and Szymanski, T. G. *On the equivalence, containment, and covering problems for the regular and context-free languages*. Journal of Computer and System Sciences 12 (1976), 222–268.

[7] Jiang, T. and Ravikumar, B. *Minimal NFA problems are hard*. SIAM Journal on Computing 22 (1993), 1117–1141.

[8] Landau, E. *Über die Maximalordnung der Permutationen gegebenen Grades*. Archiv der Math. und Phys. 3 (1903), 92–103.

[9] Landau, E. *Handbuch der Lehre von der Verteilung der Primzahlen*. Teubner, Leipzig, 1909.

[10] Mereghetti, C. and Pighizzini, G. *Two-way automata simulations and unary languages*. Journal of Automata, Languages and Combinatorics 5 (2000), 287–300.

[11] Mereghetti, C. and Pighizzini, G. *Optimal simulations between unary automata*. SIAM Journal on Computing 30 (2001), 1976–1992.

[12] Nicaud, C. *Average state complexity of operations on unary automata*. Mathematical Foundations of Computer Science (MFCS '99) LNCS 1672, 1999, pp. 231–240.

[13] Pighizzini, G. and Shallit, J. O. *Unary language operations, state complexity and Jacobsthal's function*. International Journal of Foundations of Computer Science 13 (2002), 145–159.

[14] Sakoda, W. J. and Sipser, M. *Nondeterminism and the size of two way finite automata*. ACM Symposium on Theory of Computing (STOC '78), 1978, pp. 275–286.

[15] Salomaa, K. and Yu, S. *NFA to DFA transformation for finite languages over arbitrary alphabets*. Journal of Automata, Languages and Combinatorics 2 (1997), 177–186.

[16] Shallit, J. O. and Ellul, K. Personal communication, October 2002.

[17] Sipser, M. *Lower bounds on the size of sweeping automata*. Journal of Computer and System Sciences 21 (1980), 195–202.

[18] Stockmeyer, L. J. and Meyer, A. R. *Word problems requiring exponential time*. ACM Symposium on Theory of Computing (STOC '73), 1973, pp. 1–9.

[19] Szalay, M. *On the maximal order in* S_n *and* S_n^*. Acta Arithm. 37 (1980), 321–331.

[20] Yu, S. *State complexity of regular languages*. Journal of Automata, Languages and Combinatorics 6 (2001), 221–234.

[21] Yu, S. *State complexity of finite and infinite regular languages*. Bulletin of the EATCS 76 (2002), 142–152.