Simplifying Nondeterministic Finite Cover Automata

Cezar Câmpeanu
Department of Computer Science and Information Technology,
The University of Prince Edward Island, Canada
ccampeanu@upei.ca

The concept of Deterministic Finite Cover Automata (DFCA) was introduced at WIA '98, as a more compact representation than Deterministic Finite Automata (DFA) for finite languages. In some cases representing a finite language, Nondeterministic Finite Automata (NFA) may significantly reduce the number of states used. The combined power of the succinctness of the representation of finite languages using both cover languages and non-determinism has been suggested, but never systematically studied. In the present paper, for nondeterministic finite cover automata (NFCA) and l-nondeterministic finite cover automaton (l-NFCA), we show that minimization can be as hard as minimizing NFAs for regular languages, even in the case of NFCAs using unary alphabets. Moreover, we show how we can adapt the methods used to reduce, or minimize the size of NFAs/DFCAs/l-DFCAs, for simplifying NFCAs/l-NFCAs.

1 Introduction

The race to find more compact representation for finite languages was started in 1959, when Michael O. Rabin and Dana Scott introduced the notion of Nondeterministic Finite Automata, and showed that the equivalent Deterministic Finite Automaton can be, in terms of number of states, exponential larger than the NFA. Since, it was proved in [25] that we can obtain a polynomial algorithm for minimizing DFAs, and in [16] was proved that an $O(n \log n)$ algorithm exists. In the meantime, several heuristic approaches have been proposed to reduce the size of NFAs [2, 18], but it was proved by Jiang and Ravikumar [19] that NFA minimization problems are hard; even in case of regular languages over a one letter alphabet, the minimization is NP-complete [10, 19].

On the other hand, in case of finite languages, we can obtain minimizing algorithms [22, 26] that are in the order of $O(n)$, where n is the number of states of the original DFA. In [4, 6, 21] it has been shown that using Deterministic Finite Cover Automata to represent finite languages, we have minimization algorithms as efficient as the best known algorithm for minimizing DFAs for regular languages.

The study of the state complexity of operations on regular languages was initiated by Maslov in 1970 [22, 23], but has not become a subject of systematic study until 1992 [27]. The special case of state complexity of operations on finite languages was studied in [5].

Nondeterministic state complexity of regular languages was also subject of interest, for example in [12, 13, 14, 15]. To find lower bounds for the nondeterministic state complexity of regular languages, the fooling set technique, or the extended fooling set technique may be used [3, 9, 10].

In this paper we show that NFCA state complexity for a finite language L can be exponentially lower than NFA or DFCA state complexity of the same language. We modify the fooling set technique for cover automata, to help us prove lower bounds for NFCA state complexity in section 3. We also show that the (extended) fooling set technique is not optimal, as we have minimal NFCAs with arbitrary number of states, and the largest fooling set has constant size. In section 4 we show that minimizing NFCAs is hard, and in section 5 we show that heuristic approaches for minimizing DFAs or NFAs need a special

Z. Ésik and Z. Fülöp (Eds.): Automata and Formal Languages 2014 (AFL 2014)
EPTCS 151, 2014, pp. 162–173, doi:10.4204/EPTCS.151.11

treatment when applied to NFCAs, as many results valid for the DFCAs are no longer true for NFCAs. In section 6, we formulate a few open problems and future research directions.

2 Notations and definitions

The number of elements of a set T is denoted by $\#T$. In case Σ is an alphabet, i.e, finite non-empty set, the free monoid generated by Σ is Σ^*, and it is the set of all words over Σ. The length of a word $w = w_1w_2\ldots w_n$, $n \geq 0$, $w_i \in \Sigma$, $1 \leq i \leq n$, is $|w| = n$. The set of words of length equal to l is Σ^l, the set of words of length less than or equal to l is denoted by $\Sigma^{\leq l}$. In a similar fashion, we define $\Sigma^{\geq l}$, $\Sigma^{<l}$, or $\Sigma^{>l}$. A finite automaton is a structure $A = (Q, \Sigma, \delta, q_0, F)$, where Q is a finite non-empty set called the set of states, Σ is an alphabet, $q_0 \in Q$, $F \subseteq Q$ is the set of final states, and δ is the transition function. For delta, we distinguish the following cases:

- if $\delta : Q \times \Sigma \overset{\circ}{\longrightarrow} Q$, the automaton is deterministic; in case δ is always defined, the automaton is complete, otherwise it is incomplete;
- if $\delta : Q \times \Sigma \longrightarrow 2^Q$, the automaton is non-deterministic.

The language accepted by an automaton is defined by: $L(A) = \{w \in \Sigma^* \mid \delta(\{q_0\}, w) \cap F \neq \emptyset\}$, where $\delta(S, w)$ is defined as follows:

$$\delta(S, \varepsilon) = S,$$

$$\delta(S, wa) = \bigcup_{q \in \delta(S,w)} \delta(\{q\}, a).$$

Of course, $\delta(\{q\}, a) = \{\delta(q, a)\}$ in case the automaton is deterministic, and $\delta(\{q\}, a) = \delta(q, a)$, in case the automaton is non-deterministic.

Definition 1 *Let L be a finite language, and l be the length of the longest word w in L, i.e., $l = \max\{|w| \mid w \in L\}$[1]. If L is a finite language, L' is a cover language for L if $L' \cap \Sigma^{\leq l} = L$.*

A cover automaton for a finite language L is an automaton that recognizes a cover language, L', for L. An l-NFCA A is a cover automaton for the language $L(A) \cap \Sigma^{\leq l}$.

One could plainly see that any automaton that recognizes L is also a cover automaton.

The level of a state $s \in Q$ in a cover automaton $A = (Q, \Sigma, \delta, q_0, F)$ is the length of the shortest word that can reach the state s, i.e., $level_A(s) = min\{|w| \mid s \in \delta(q_0, w)\}$.

Let us denote by $x_A(s)$ the smallest word w, according to quasi-lexicographical order, such that $s \in \delta(q_0, w)$, see [6] for a similar definition in case of DFCA. Obviously, $level_A(s) = |x_A(s)|$.

For a regular language L, $\equiv_L$ denotes the Myhil-Nerode equivalence of words [17].

The similarity relation induced by a finite language L is defined as follows[6]: $x \sim_L y$, if for all $w \in \Sigma^{\leq l - \max\{|x|,|y|\}}$, $xw \in L$ iff $yw \in L$. A dissimilar sequence for a finite language L is a sequence $x_1, \ldots, x_n$ such that $x_i \not\sim_L x_j$, for all $1 \leq i, j \leq n$ and $i \neq j$.

Now, we need to define the similarity for states in an NFCA, since it was the main notion used for DFCA minimization.

Definition 2 *In an NFCA $A = (Q, \Sigma, \delta, q_0, F)$, two states $p, q \in Q$ are similar, written $s \sim_A q$, if $\delta(p, w) \cap F \neq \emptyset$ iff $\delta(q, w) \cap F \neq \emptyset$, for all $w \in \Sigma^{\leq l - \max\{level(p), level(q)\}}$.*

[1] We use the convention that $max\emptyset = 0$.

In case the NFCA A is understood, we may omit the subscript A, i.e., we write $p \sim q$ instead of $p \sim_A q$, also we can write $level(p)$ instead of $level_A(p)$.

We consider only non-trivial NFCAs for L, i.e., NFCAs such that $level(p) \leq l$ for all states p. In case $level(p) > l$, p can be eliminated, and the resulting NFA is still a NFCA for L. In this case, if $p \sim q$, then either $p, q \in F$, or $p, q \in Q \setminus F$, because $|\varepsilon| \leq l - \max\{level(p), level(q)\}$.

Deterministic state complexity of a regular language L is defined as the number of states of the minimal deterministic automaton recognizing L, and it is denoted by $sc(L)$:

$$sc(L) = \min\{\#Q \mid A = (Q, \Sigma, \delta, q_o, F), \text{ deterministic, complete, and } L = L(A)\}.$$

Non-deterministic state complexity of a regular language L is defined as the number of states of the minimal non-deterministic automaton recognizing L, and it is denoted by $nsc(L)$:

$$nsc(L) = \min\{\#Q \mid A = (Q, \Sigma, \delta, q_o, F), \text{ non-deterministic and } L = L(A)\}.$$

For finite languages L, we can also define deterministic cover state complexity $csc(L)$ and non-deterministic cover state complexity $ncsc(L)$:

$$\begin{aligned} csc(L) &= \min\{\#Q \mid A = (Q, \Sigma, \delta, q_o, F), \text{ deterministic, complete, and} \\ &\quad L = L(A) \cap \Sigma^{\leq l}\}, \\ ncsc(L) &= \min\{\#Q \mid A = (Q, \Sigma, \delta, q_o, F), \text{ non-deterministic, and} \\ &\quad L = L(A) \cap \Sigma^{\leq l}\}. \end{aligned}$$

Obviously, $ncsc(L) \leq nsc(L) \leq sc(L)$, but also $ncsc(L) \leq csc(L) \leq sc(L)$. Thus, non-deterministic finite cover automata can be considered to be one of the most compact representation of finite languages.

3 Lower-bounds and Compression Ratio for NFCAs

We start this section analyzing few examples where nondeterminism, or the use of cover language, reduce the state complexity. Let us first analyze the type of languages where non-determinism, combined with cover properties, reduce significantly the state complexity.

We choose the language $L_{F_{m,n}} = \{a,b\}^{\leq m} a \{a,b\}^{n-2}$, where $m, n \in \mathbb{N}$. In Figure 1 we can see an NFA recognizing L with $m+n$ states. We must note that the longest word in the language has $m+n-1$ letters. Let us analyze if the automaton in Figure 1 is minimal. The fooling set technique, introduced in [7] and [8], and used to prove the lower-bound for state complexity of NFAs, is stated in [3, 7] as follows:

Lemma 1 *Let $L \subseteq \Sigma^*$ be a regular language, and suppose there exists a set of pairs $S = \{(x_i, y_i) \mid 1 \leq i \leq n\}$, with the following properties:*

1. *If $x_i y_i \in L$, for $1 \leq i \leq n$ and $x_i y_j \notin L$, for all $1 \leq i, j \leq n$, $i \neq j$, then $nsc(L) \geq n$. The set S is called* a fooling set *for L.*
2. *If $x_i y_i \in L$, for $1 \leq i \leq n$ and for $1 \leq i, j \leq n$, if $i \neq j$, implies either $x_i y_j \notin L$ or $x_j y_i \notin L$, then $nsc(L) \geq n$. The set S is called* an extended fooling set *for L.*

Now consider the following set of pairs of words: $S = \{(b^m a b^j, b^{n-2-j}) \mid 0 \leq j \leq n-2\} \cup \{(a^i, b^{m-i} a b^{n-2}) \mid 0 \leq i \leq m\} = \{(x_k, y_k) \mid 1 \leq k \leq m+n\}$.

For $(x_k, y_k) \in S$, we have that

1. $x_k y_k = b^m ab^j b^{n-2-j} = b^m ab^{n-2} \in L$, or
2. $x_k y_k = a^i b^{m-i} ab^{n-2} \in L$.

Let us examine for each $1 \le k,h \le m+n$, $k \ne h$ if the words $x_k y_h$ and $x_h y_k$ are also in L. We have the following possibilities:

1. Case I
   (a) $x_k y_h = b^m ab^i b^{n-2-j} \notin L$, and
   (b) $x_h y_k = b^m ab^j b^{n-2-i} \notin L$.
2. Case II
   (a) $x_k y_h = a^i b^{m-j} ab^{n-2} \in L$, if $i < j$, but
   (b) $x_h y_k = a^j b^{m-i} ab^{n-2} \notin L$, if $i < j$ (because $|a^j b^{m-i} ab^{n-2}| = m+n-1+j-i > m+n-1$).
3. Case III
   (a) $x_k y_h = b^m ab^j b^{m-i} ab^{n-2} \notin L$ (because $|b^m ab^j b^{m-i} ab^{n-2}| = m+1+j+m+1+n-2 > m+n-1$), or
   (b) $x_h y_k = a^i b^{n-2-j} \in L$ if $n-2-j+1+i > n$, or $x_h y_k = a^i b^{n-2-j} \notin L$ $n-2-j+1+i < n$.

From the statement 2. of Lemma 1, it follows that the NFA is minimal. We must note the following:

1. we cannot use the weak form 1 to prove the lower-bound;
2. when proving the lower-bound, we concatenate words to obtain a word of length greater than the maximum length of the words in the language, and that's why $x_i y_j$ is rejected. Since in case of cover automata such words will be automatically rejected, there is no doubt that any fooling set type technique we may use to prove the lower-bound for NFCAs must consider the length, and we should ignore the cases when the length exceeds the maximal one.

Hence, the fooling set technique introduced in [7] and [8], and used to prove the lower-bound for state complexity of NFAs, can be modified to prove a lower-bound for minimal NFCAs, and it can be formulated for cover languages as an adaptation of Theorem 1 in [10].

Lemma 2 *Let $L \subseteq \Sigma^{\le l}$ be a finite language such that the longest word in L has the length l, and suppose there exists a set of pairs $S = \{x_i, y_i) \mid 1 \le i \le n\}$, with the following properties:*

1. *If $x_i y_i \in L$ for $1 \le i \le n$ and for $1 \le i,j \le n$, $i \ne j$, and $x_i y_j \in \Sigma^{\le l}$, we have that $x_i y_j \notin L$, then $ncsc(L) \ge n$.*

   The set S is called a fooling set *for L.*
2. *If $x_i y_i \in L$, for $1 \le i \le n$ and for $1 \le i,j \le n$, if $i \ne j$, implies either $x_i y_j \in \Sigma^{\le l}$ and $x_i y_j \notin L$, or $x_j y_i \in \Sigma^{\le l}$ and $x_j y_i \notin L$ for all, then $ncsc(L) \ge n$.*

   The set S is called an extended fooling set *for L.*

Proof Assume there exists an NFCA $A = (Q, \Sigma, \delta, q_0, F)$, with m states accepting L. For each i, $1 \le i \le n$, $x_i y_i \in L$, therefore we must have a state $s_i \in \delta(q_0, x_i)$ and $\delta(s_i, y_i) \cap F \ne \emptyset$. In other words, there exists a state $f_i \in F$ and $f_i \in \delta(s_i, y_i)$.

1. We claim $s_i \notin \delta(q_0, x_j)$ for all $j \ne i$. If $s_i \in \delta(q_0, x_j)$, then $f_i \in \delta(s_i, y_i) \subseteq \delta(q_0, x_j y_i)$, and because $|x_j y_i| \le l$, it follows that $x_j y_i \in L$, a contradiction.

2. We consider the function $f : \{1,\ldots,n\} \longrightarrow Q$ defined by $f(i) = s_i$, s_i as above. We claim that f is injective. If $f(i) = f(j)$, then $\delta(f(i),y_i) = \delta(f(j),y_i)$, also $\delta(f(j),y_j) = \delta(f(i),y_j)$. Because $\delta(f(i),y_i) \cap F \neq \emptyset$, we also have that $\delta(f(j),y_i) \cap F \neq \emptyset$, and because $|x_iy_j| \leq l$, it follows that $x_iy_j \in L$, a contradiction. If $|x_jy_i| \leq l$, using the same reasoning, will follow that $x_jy_i \in L$. In both cases we have a contradiction, thus Q must have at least n elements.□

For the example above, we discover that we cannot have more than one pair of the form $(a^i, b^{m-i}ab^{n-2})$, thus, applying the extended fooling set technique for NFCAs, the minimum number of states in a minimal NFCA is at least $n-2+1+1 = n$. This proves that the NFCA presented in Figure 2 is minimal.

It is easy to check that any two distinct words $w_1, w_2 \in \Sigma^{\leq n-1}$, $w_1 \neq w_2$, are not similar with respect to $\sim_L$. It follows that for the language presented in Figure 1, $csc(L) \geq 2^{n-1}$. One can also verify that for two distinct words uay and wax, if $|y| \neq |x|$, $|x|,|y| \leq n-2$, they are distinguishable; also, in case $|x| = |y| \leq n-2$, the word $a^{n-2-|x|}$ will distinguish between all the words for which $|u| < n-2-|x|$ or $|w| < n-2-|x|$, thus the number of states in the minimal DFA is even larger than $csc(L)$. In case $m = 2$ and $n = 4$, the minimal DFCA is presented in Figure 3. A simple computation shows us that the corresponding minimal DFA has 15 states.

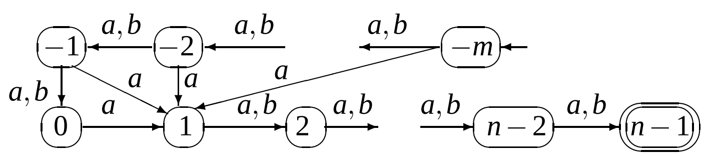

Figure 1: An NFA with $m+n$ states for the language $L_{F_{m,n}} = \{a,b\}^{\leq m}a\{a,b\}^{n-2}$.

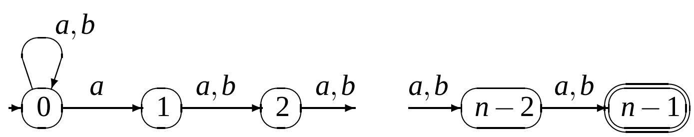

Figure 2: An NFCA with n states for the language $L_{F_{m,n}} = \{a,b\}^{\leq m}a\{a,b\}^{n-2}$, that is the same as the one in Figure 1. In case $m = 2$ and $n = 4$, the language is the same as the one described in Figure 3.
An equivalent minimal NFA has $m+n$ states.

This language example shows that NFCAs may be a much more compact representation for finite languages than NFAs, or even DFCAs, and motivates the study of such objects. In terms of compression, clearly the number of states in the NFCA is exponentially smaller than the number of states in the DFA, and in some cases, even exponentially smaller than in an NFA.

Let's set $\Sigma = \{a\}$, $l > k \geq 2$, and choose the following language:

$$L_{l,k} = a(\Sigma^{\leq l} - \{(a^k)^n \mid n \geq 0\}). \tag{1}$$

In Figure 4, the NFCA A_k accepts the language $L_{l,k}$, therefore $ncsc(L_{l,k}) \leq csc(L_{l,k}) \leq sc(L_{l,k}) \leq \min(l+1, k+1) = k+1$. It is known [7, 13, 24] that the automaton A_k is minimal NFA for $\bigcup_{l\in\mathbb{N}} L_{l,k}$, if k is

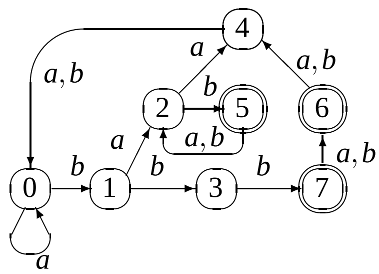

Figure 3: A minimal DFCA with 8 states for the language $L_{F_{2,4}} = \{a,b\}^{\leq 2}a\{a,b\}^2$, $l = 5$. The equivalent minimal DFA has 15 states.

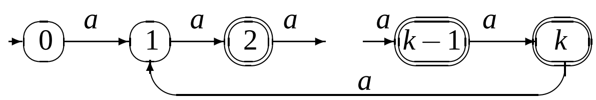

Figure 4: An NFA/NFCA A_k for $L_{l,k}$.

a prime number. However, this may not be a minimal NFCA, as illustrated by the example in Figure 5, where A_7 is not minimal for $L_{9,7}$, even if it is minimal NFA for the cover language.

We apply the extended fooling set technique for the language $L_{l,k}$. Because the alphabet is unary, all the words in an extended fooling set S are powers of a: $S \supseteq \{(a^{i_1}, a^{j_1}), (a^{i_2}, a^{j_2}), (a^{i_3}, a^{j_3}), \ldots, (a^{i_r}, a^{j_r})\}$, for some $r \in \mathbb{N}$. A simple computation shows that if $i_1, \ldots, i_r > 1$, and $i_1 + j_2 = z_{12}k + 1$ and $i_1 + j_3 = z_{13}k + 1$ for some $z_{12}, z_{13} \in \mathbb{N}$, then $i_2 + j_3 \neq z_{23}k + 1$ and $i_3 + j_2 \neq z_{32}k + 1$, for any $z_{23}, z_{32} \in \mathbb{N}$. It follows that $r \leq 3$.

Let A be an NFA accepting $L \supseteq L_k$, and we can consider that it is already in Chrobak normal form, as it is ultimately periodic. Thus, for each L, $nsc(L) \geq p_1 + \ldots p_s$, where p_i are primes, and each cycle has $p_i^{k_i}$ states, $1 \leq i \leq s$. Now, let us prove that A_k is minimal for some language $L_{l,k}$, $l \geq k$.

Assume there exists an automaton $B = (Q_B, \Sigma, \delta_B, q_{0,B}, F_B)$ with m states, $m \leq k+1$ such that $L(B) = L_{l,k}$. It follows that the language $L(B)$ will contain words with a length $x + hy$ for $x, y \leq k$, and all $h \in \mathbb{N}$. For h large enough, one of these words will be of length multiple of k plus 1, therefore, for large enough l, i.e., greater than some l_0, $L_{l,k} \neq L(B)$. Thus, the number of states in B is at least k. A_k is also a minimal NFCA for languages $L_{l,k}$, $l \geq l_0$, hence it follows that Theorem 7 in [10] is also valid for cover automata:

Theorem 1 *There is a sequence of languages $(L_{l,k})_{k \geq 2}$ such that the nondeterministic cover complexity*

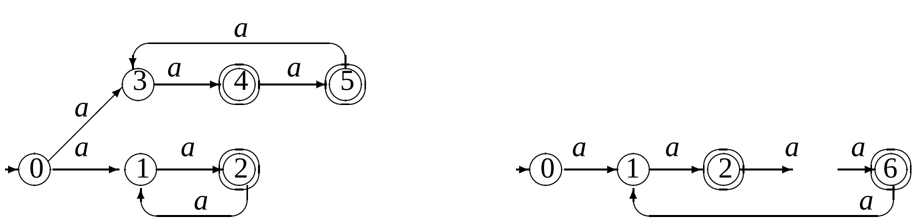

Figure 5: A minimal NFCA for $L_{9,7}$, left, and a minimal NFA for a cover language, right.

of $L_{l,k}$ is at least k, but the extended fooling set for $L_{l,k}$ is of size c, where c is a constant.

Now, we are ready to check how hard is to obtain this minimal representation of a finite language.

4 Minimization Complexity

In this section we show that minimizing NFCAs is hard, and we'll show it with the exact same arguments from [11], used to prove that minimizing NFAs is hard. We will describe the construction from [8, 11], showing that we can also use it with only a minor addition for cover NFAs. To keep the paper self contained, we include a complete description, and emphasize the changes required for the cover automata, rather than just presenting the differences.

Let us consider a logical formula $F \in 3SAT$, in the conjunctive normal form, i.e., $F = \bigwedge_{i=1}^{m} C_i$, where each clause C_i, $1 \le i \le m$, is defined using variables $x_1, \ldots, x_n$, $C_i = u_1 \vee u_2 \vee u_3$, and each u_j, $1 \le j \le 3$ are either x_i or $\neg x_i$. Let $p_1, p_2, \ldots, p_n$ be distinct prime numbers such that $p_1 < p_2 < \ldots < p_n$. We set $k = \prod_{i=1}^{n} p_i$, and using Chinese Remainder Theorem [20][2], it follows that the function $f : \mathbb{Z}_k \longrightarrow \prod_{i=1}^{n} \mathbb{Z}_{p_i}$ is bijective. We need to define a language L_F and a natural number l such that $L_F = \{a\}^*$, if and only if F is unsatisfiable, therefore, the finite language $L_F \cap \Sigma^{\le l}$ has $\{a\}^*$ as a cover language. We can construct an automaton B_i in $O(p_n)$ in a similar fashion as we build automata A_k that recognizes the language $L(B_i) = \{a^n \mid n \bmod p_i \notin \{0,1\}\}$. Let B be an automaton recognizing $\bigcup_{i=1}^{n} L(B_i)$. It is clear that it can be constructed in $O(n \cdot p_n)$ time. For each clause C_i such that a_1, a_2, a_3 is an assignment of its variables for which C_i is not satisfied, we define $L_{C_i} = \cap_{i=1}^{3} \{a^n \mid n \bmod p_i = a_i\}$. An automaton C_i accepting L_{C_i} can be constructed in $O(p_n^3)$ time[3]. Setting $L_F = \bigcup_{i=1}^{m} L_{C_i} \cup L(B)$, it follows that $L_F = \{a\}^*$ iff F is satisfiable. Moreover, L_F is a cyclic language with period at most k, thus setting $l = k$, we have that $L_F \cap \{a\}^{\le l}$ has $\{a\}^*$ as a cover language iff F is satisfiable. Since according to [1], primality test can be done in polynomial time, we can find the first n prime numbers in polynomial time, which means that our NFA construction can also be done in polynomial time. If F is unsatisfiable, then $ncsc(L) = 1$, if F is satisfiable, then the minimal period of L_F is $\frac{l}{2}$, according to [7, 8], and the minimal number of states in an NFA is at least equal to the largest prime number dividing its period, which is p_n. Using the same argument as in [11], it follows that the existence of a polynomial algorithm to decide if $ncsc(L) = o(n)$ implies that $nsc(L) = o(n)$, therefore we can solve $3SAT$ in polynomial time, i.e., $P = NP$. Consequently, we proved that

Theorem 2 *Minimizing either NFCAs or l-NFCAs is at least NP-hard.*

5 Reducing the Number of States of NFCAs

Assume the DFA $A = (Q, \Sigma, \delta, q_0, F)$ is minimal for L, and the minimal NFA is $A' = (Q', \Sigma, \delta', q_0, F)$, where $Q' = Q - \{d\}$, $\delta'(s,p) = \delta(s,p)$, if $\delta(s, p \in Q')$ and $\delta'(s,p) = \emptyset$ if $\delta(s,p) = d$. In other words, the minimal NFA is the same as the DFA, except that we delete the dead state. We may have a minimal DFCA as A, and A' as a minimal NFA, but not as a minimal NFCA, as illustrated by A_7 and $L_{9,7}$.

We need to investigate if classical methods to reduce the number of states in an NFA or DFA/DFCA can also be applied to NFCAs, thus, we first analyze the state merging technique. For NFAs, we distinguish between two main ways of merging states: (1) a weak method, where two states are merged

[2] Theorem I.3.3, page 21

[3] Using Cartesian product construction, for example.

by simply collapsing one into the other, and consolidate all their input and output transitions, and (2), a strong method, where one state is merged into another one by redirecting its input transitions toward the other state, and completely deleting it and all its output transitions. The same methods are considered for NFCAs.

Definition 3 *Let $A = (Q,\Sigma,\delta,q_0,F)$ be a NFCA for the finite language L.*

1. *We say that the state q is* weakly mergible *in state p if the automaton $A' = (Q',\Sigma,\delta',q_0,F')$, where $Q' = Q - \{q\}$, $F' = F \cap Q'$, and*

$$\delta(s,a) = \begin{cases} \delta(s,a), & \text{if } \delta(s,a) \subseteq Q' \text{ and } s \neq p, \\ (\delta(s,a) \setminus \{q\}) \cup \{p\}, & \text{if } q \in \delta(s,a) \text{ and } s \neq p, \\ (\delta(s,a) \cup \delta(q,a)) \setminus \{q\}, & \text{if } s = p \end{cases}$$

   is also a NFCA for L. In this case we write $p \precapprox q$.

2. *We say that the state q is* strongly mergible *in state p, if the automaton $A' = (Q',\Sigma,\delta',q_0,F')$, where $Q' = Q - \{q\}$, $F' = F \cap Q'$, and*

$$\delta(s,a) = \begin{cases} \delta(s,a), & \text{if } \delta(s,a) \subseteq Q' \\ (\delta(s,a) \setminus \{q\}) \cup \{p\}, & \text{if } q \in \delta(s,a), \end{cases}$$

   is also a NFCA for L. In this case we write $p \precsim q$.

In case $p \precapprox q$, $(L_p^L L_p^R \cup L_p^L L_q^R \cup L_q^L L_p^R \cup L_q^L L_q^R) \cap \Sigma^{\leq l} \subseteq L$ and in case $p \precsim q$, $L_q^L L_q^R \cap \Sigma^{\leq l} \subseteq (L_p^L L_p^R \cup L_q^L L_p^R) \cap \Sigma^{\leq l} \subseteq L$, where for $s \in Q$ $L_s^L = \{w \in \Sigma^* \mid s \in \delta(q_0,w)\}$ and $L_s^R = \{w \in \Sigma^* \mid \delta(s,w) \cap F \neq \emptyset\}$.

For the case of DFCAs, if A is a DFCA for L and two states are similar with respect to the similarity relation induced by A, then all the words reaching these states are similar. Moreover, if two words of minimal length reach two distinct states in a DFCA, and the words are similar with respect to L, then the states in the DFCA must be similar with respect to the similarity relation induced by A. These results are used for DFCA minimization, and we need to verify if they can be used in case of NFCAs. In the following lemmata we show that the corresponding results are no longer true.

Lemma 3 *Let $A = (Q,\Sigma,\delta,q_0,F)$ be a NFCA for the finite language L. It is possible that $x_A(s) \sim_L x_A(q)$, but s and q are not mergible.*

Proof For the automaton in Figure 5, left, $x_A(3) = x_A(1)$, but the states 1 and 3 are not mergible, as the resulting automaton would not reject a^7.

Lemma 4 *Let $A = (Q,\Sigma,\delta,q_0,F)$ be a NFCA for the finite language L, and $p,q \in Q$, $p \neq q$. It is possible to have $x,y \in \Sigma^*$, $p \in \delta(q_0,x)$, $q \in \delta(q_0,y)$, $p \sim q$, and $x \not\sim_L y$.*

Proof Consider the language $L = L(A) \cap \{a,b\}^{\leq 14}$, where A is depicted in Figure 5.

We have that:

- $aa \not\sim_L ba$, because $aaa \notin L$, but $baa \in L$;
- $2 \in \delta(0,ba)$, $7 \in \delta(0,aa)$, and
- $2 \sim_A 7$, because $\delta(2,a^{2k}) = \{2\} \subseteq F$, $\delta(2,a^{2k+1}) = \{1\} \cap F = \emptyset$, $\delta(7,a^{2k}) = \{7\} \subseteq F$, $\delta(7,a^{2k+1}) = \{6\} \cap F = \emptyset$, and $\delta(2,w) = \delta(7,w) = \emptyset$, for all $w \in \Sigma^* - \{a\}^*$.□

Let us verify the case when two states p,q are similar, or we can distinguish between them.

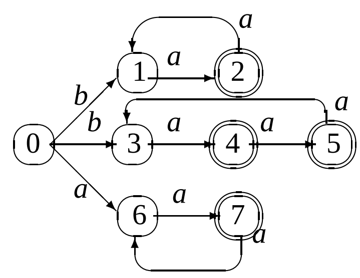

Figure 6: An example where $p \sim_A q$, $x \not\sim_L y$, but $p \in \delta(q_0,x)$ and $q \in \delta(q_0,y)$, $aa \not\sim_L ba$, $2 \in \delta(0,ba)$, $7 \in \delta(0,aa)$, and $2 \sim_A 7$.

Lemma 5 *Let $A = (Q,\Sigma,\delta,q_0,F)$ be a NFCA for the finite language L, $p,q \in Q$, $p \neq q$, and either $p,q \in F$, or $p,q \notin F$. Assume $r \in \delta(p,a)$ and $s \in \delta(q,a)$.*

1. *If $r \sim_A s$, for all possible choices of r and s, then $p \sim_A q$.*
2. *The converse is false, i.e., we may have $r \not\sim_A s$, for some r and s, and $p \sim_A q$.*

Proof Assume $p \not\sim_A q$, and let $w \in \Sigma^{\leq l-\max\{level(p),level(q)\}} \cap \Sigma^+$. Because either $p,q \in F$, or $p,q \notin F$, we have that $\delta(p,aw) \cap F \neq \emptyset$ and $\delta(q,aw) \cap F = \emptyset$, or $\delta(p,aw) \cap F = \emptyset$, and $\delta(q,aw) \cap F \neq \emptyset$. If $\delta(p,aw) \cap F \neq \emptyset$ and $\delta(q,aw) \cap F = \emptyset$, it follows that we have two states $r \in \delta(p,a)$ and $s \in \delta(q,a)$ such that $\delta(r,w) \cap F \neq \emptyset$, and $\delta(s,w) \cap F = \emptyset$. This proves that the first implication is true. For the second implication, consider the automaton depicted in Figure 5 with $l = 14$, and the following states p,q,r,s: $p = q = 0$, $r = 1$, $s = 3$, and the letter b. We have that $p \sim q$, $1,3 \in \delta(p,b) = \delta(q,b) = \delta(0,b)$, but $r \not\sim s$, because $\delta(1,a) \cap F = \emptyset$ and $\delta(3,a) \cap F = \{4\} \neq \emptyset$.$\square$

This result contrasts with the one for the deterministic case for cover automata, and the main reason is the nondeterminism, not the fact that we work with cover languages.

Next, we would like to verify if similar states can be merged in case of NFCAs, also to check which type of merge works. In case we have two similar states, we can strongly merge them as shown below. In the case of DFCAs, if two states are similar, these can be merged. We must ensure that the same result is also true for NFCAs, and the next theorem shows it.

Theorem 3 *Let $A = (Q,\Sigma,\delta,q_0,F)$ be an NFCA for L, and $p,q \in Q$ such that $p \neq q$, and $p \sim q$. Then we have*

1. *if $level_A(p) \leq level_A(q)$, then $p \precsim q$.*
2. *It is possible that $p \not\precsim q$.*

Proof For the first part, let A' be the automaton obtained from A by strongly merging q in p. We need to show that A' is a cover NFCA for L. Let $w = w_1 \dots w_n$ be a word in $\Sigma^{\leq l}$, $n \in \mathbb{N}$ and $w_i \in \Sigma$ for all i, $1 \leq i \leq n$. We now prove that $w \in L$ iff $\delta'(q_0,w) \cap F' \neq \emptyset$.

If we can find the states $\{q_0,q_1,\dots,q_n\}$ such that $q_1 \in \delta(q_0,w_1)$, $q_2 \in \delta(q_1,w_2)$, $\dots$, $q_n \in \delta(q_{n-1},w_n)$, $q_n \in F$ and $q \notin \{q_0,q_1,\dots,q_n\}$, then $q_1 \in \delta'(q_0,w_1)$, $q_2 \in \delta'(q_1,w_2)$, $\dots$, $q_n \in \delta'(q_{n-1},w_n)$, $q_n \in F'$, i.e., $\delta'(q_0,w) \cap F' \neq \emptyset$. Assume $q = q_j$, and j is the smallest with this property. If $j = n$, then $q \in F$, which implies $p \in F$, then $q_1 \in \delta'(q_0,w_1)$, $q_2 \in \delta'(q_1,w_2)$, $\dots$, $q_n \in \delta'(p,w_n)$, which means $\delta'(q_0,w) \cap F' \neq \emptyset$.

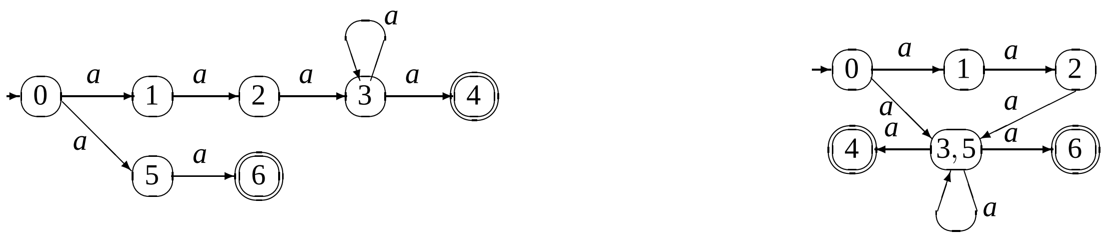

Figure 7: Example for weakly merging failure and similar states.

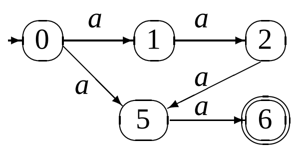

Figure 8: Example for strongly merging similar states for the example presented in Figure 7.

Assume the statements hold for $|w_j \dots w_n| < l'$ for $l' < l - |w|$ $(l - |w_1 ... w_j| \leq l - level(q))$, and consider the case when $|w_{j-1} w_j \dots w_n| = l'$. If for every non-empty prefix of $w_{j+1} \dots w_n$, $w_{j-1} \dots w_h$, $q \notin \delta(p, w_{j-1} \dots w_h)$, then $\delta(p, w_{j+1} \dots w_n) \in F - \{q\}$ iff $\delta(q, w_{j+1} \dots w_n) \in F - \{q\}$, i.e., $\delta'(p, w_{j+1} \dots w_n) \cap F' \neq \emptyset$ iff $\delta(q, w_{j+1} \dots w_n) \cap F \neq \emptyset$.

Otherwise, let h be the smallest number such that $q \in \delta(q, w_{j+1} \dots w_h$. Then $|w_{h+1} \dots w_n| < l'$ (and $p \in \delta'(p, w_j \dots w_h)$). By induction hypothesis, $\delta'(p, w_{h+1} \dots w_n) \cap F' \neq \emptyset$ iff $\delta(q, w_{h+1} \dots w_n) \cap F \neq \emptyset$. Therefore, $\delta(p, w_{j+1} \dots w_h w_{h+1} \dots w_n) \cap F' \neq \emptyset$ iff $\delta(q, w_{j+1} \dots w_h w_{h+1} \dots w_n) \cap F \neq \emptyset$, proving the first part. For the second part, consider the automaton in Figure 7 as a NFCA for $L = \{a^2, a^4\}$. We have that $l = 4$ and $3 \sim 5$, because $level(3) = 3$, and $\delta(3, \varepsilon) \cap F = \delta(5, \varepsilon) \cap F = \emptyset$ $\delta(3, a) \cap F = \{4\}$, $\delta(5, a) \cap F = \{6\}$. We cannot weakly merge state 3 with state 5, as we would recognize $a^3 \notin L$. In Figure 8 we have the result for strongly merging state 3 in state 5.

We can observe that strongly merging states does not add words in the language, while weakly merging may add words. Because any DFCA is also a NFCA, then some smaller automata can be obtained from larger ones without using state merging technique, and the following lemma presents such a case. Also, the automaton in Figure 2 is obtained from automaton in Figure 1 by strongly merging states $0, \dots -m+1$ into state $-m$.

Lemma 6 *Let $A = (Q, \Sigma, \delta, q_0, F)$ be an NFCA for L, and consider the reduced sub-automaton generated by state p, $A = (Q_R, \Sigma, \delta_R, p, F)$, i.e., Q_R contains only reachable and useful states, and δ_R is the induced transition function. If $\delta(s, a) \cap Q_R = \emptyset$, for all $s \in (Q \setminus Q_R)$, we can find two regular languages L_1,L_2 such that*

- $L_p = (L_1 \cup L_2) \cap \Sigma^{\leq l - level(p)}$, *and*
- $nsc(L_1) + nsc(L_2) < \# Q_R + 1$,

then A is not minimal.

Proof Let $A_i = (Q_i, \Sigma, \delta_i, q_{0,i}, F_i)$, $i = 1, 2$ be two NFAs for L_1 and L_2, and $L_p = (L_1 \cup L_2) \cap \Sigma^{\leq l - level(p)}$. We define the automaton $B = ((Q \setminus Q_R) \cup \{p\} \cup Q_1 \cup Q_2, \Sigma, \delta_B, q_0, F_B)$ as follows: $F = (F \setminus Q_R) \cup F_1 \cup F_2$, in case $p \notin F$, and $F = (F \setminus Q_R) \cup F_1 \cup F_2 \cup \{p\}$ in case $p \in F$. For the transition function, we have $\delta_B(s,a) = \delta(s,a)$ if $s \in (Q \setminus Q_R)$, $\delta_B(s,a) = \delta_i(s,a)$ if $s \in Q_i$, $i = 1,2$, and $\delta_B(p,a) = \delta_1(q_{0,1},a) \cup \delta_2(q_{0,2},a) \cup \delta(p,a) \setminus Q_R$, if $p \notin \delta(p,a)$, and $\delta_B(p,a) = \delta_1(q_{0,1},a) \cup \delta_2(q_{0,2},a) \cup \delta(p,a) \setminus Q_R \cup \{p\}$, if $p \in \delta(p,a)$. Obviously, the automaton B recognizes the cover language for L, and its state complexity is lower.

This technique was used to produce the minimal NFCA for $L_{9,7}$ in Figure 5.

6 Conclusion

In this paper we showed that NFCAs are a more compact representation of finite languages than both NFAs and DFCAs, therefore it is a subject worth investigating. We presented a lower-bound technique for state complexity of NFCAs, and proved its limitations. We showed that minimizing NFCAs has at least the same level of difficulty as minimizing general NFAs, and that extra information about the maximum length of the words in the language does not help reducing the time complexity. We checked if some of the results involving reducing the size of automata for NFAs and DFCAs are still valid for NFCAs, and showed that most of them are no longer valid. However, the method of strong merging states still works in case of NFCAs, and we showed that there are also other methods that could be investigated.

As future research, below is a list of problems we consider worth investigating:

1. check if the bipartite graph lower-bound technique can be applied for NFCAs;
2. find bounds for nondeterministic cover state complexity;
3. investigate the problem of magic numbers for NFCAs. In this case, we can relate either to DFCAs, or DFAs.

References

[1] M. Agrawal, N. Kayal & N. Saxena (2004): *PRIMES is in P. Annals of mathematics*, pp. 781–793. Available at `http://dx.doi.org/10.4007/annals.2004.160.781`.

[2] J. Amilhastre, P. Janssen & M-C. Vilarem (2001): *FA Minimisation Heuristics for a Class of Finite Languages*. *Lecture Notes in Computer Science* 2214, pp. 1 – 12. Available at `http://dx.doi.org/10.1007/3-540-45526-4_1`.

[3] J.-C. Birget (1992): *Intersection and union of regular languages and state complexity*. *Information Processing Letters* 43, pp. 185–190. Available at `http://dx.doi.org/10.1016/0020-0190(92)90198-5`.

[4] C. Câmpeanu, A. P˘aun & S. Yu (2002): *An Efficient Algorithm for Constructing Minimal Cover Automata for Finite Languages*. *International Journal of Foundations of Computer Science* 13(1), pp. 83 – 97. Available at `http://dx.doi.org/10.1142/S0129054102000960`.

[5] C. Câmpeanu, K. Culik II, K. Salomaa & S. Yu (2001): *State complexity of basic operations on finite languages*. *Lecture Notes in Computer Science* 2214, pp. 60–70. Available at `http://dx.doi.org/10.1007/3-540-45526-4_6`.

[6] C. Câmpeanu, N. Santean & S. Yu (1986): *Minimal cover-automata for finite languages*. *Theoretical Computer Science* 267(1-2), pp. 3–16. Available at `http://dx.doi.org/10.1016/S0304-3975(00)00292-9`.

[7] M. Chrobak (1986): *Finite Automata and Unary Languages*. *Theoretical Computer Science* 47(2), pp. 149–158. Available at `http://dx.doi.org/10.1016/0304-3975(86)90142-8`.

[8] G.Gramlich (2003): *Probabilistic and Nondeterministic Unary Automata*. *Lecture Notes in Computer Science* 2747, pp. 460 – 469. Available at `http://dx.doi.org/10.1007/978-3-540-45138-9_40`.

[9] I. Glaister & J. Shallit (1996): *A lower bound technique for the size of nondeterministic finite automata*. *Information Processing Letters* 59, pp. 75 – 77. Available at `http://dx.doi.org/10.1016/0020-0190(96)00095-6`.

[10] H. Gruber & M. Holzer (2006): *Finding lower bounds for nondeterministic state complexity is hard*. *Lecture Notes in Computer Science* 4036, pp. 363–374. Available at `http://dx.doi.org/10.1007/11779148_33`.

[11] H. Gruber & M. Holzer (2007): *Computational Complexity of NFA Minimization for Finite and Unary Languages*. *LATA* 8, pp. 261–272. Available at `http://www2.tcs.ifi.lmu.de/~gruberh/data/lata07-submission.pdf`.

[12] M. Holzer & M. Kutrib (2003): *State complexity of basic operations on nondeterministic finite automata*. *Lecture Notes in Computer Science* 2608, pp. 148–157. Available at `http://dx.doi.org/10.1007/3-540-44977-9_14`.

[13] M. Holzer & M. Kutrib (2003): *Unary language operations and their nondeterministic state complexity*. *Lecture Notes in Computer Science* 2450, pp. 162–172. Available at `http://dx.doi.org/10.1007/3-540-45005-X_14`.

[14] M. Holzer & M. Kutrib (2009): *Descriptional and computational complexity of finite automata*. *Lecture Notes in Computer Science* 5457, pp. 23–42. Available at `http://dx.doi.org/10.1007/978-3-642-00982-2_3`.

[15] M. Holzer & M. Kutrib (2009): *Nondeterministic finite automata - recent results on the descriptional and computational complexity*. *Int. J. Found. Comput. Sci* 20(4), pp. 563–580. Available at `http://dx.doi.org/10.1142/S0129054109006747`.

[16] John Hopcroft (1971): *An $n\log n$ Algorithm for Minimizing States in a Finite Automaton*. In Z. Kohavi & A. Paz, editors: *Theory of Machines and Computations*, Academic Press, New York, pp. 189–196.

[17] John E. Hopcroft & Jeffrey D. Ullman (1979): *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley.

[18] L. Ilie, G. Navarro & S. Yu (2004): *On NFA reductions*. *Lecture Notes in Computer Science Volume: Theory Is Forever Essays Dedicated to Arto Salomaa on the Occasion of His 70th Birthday* 3113, pp. 112–124. Available at `http://dx.doi.org/10.1007/978-3-540-27812-2_11`.

[19] T. Jiang & B. Ravikumar (1993): *NFA minimization problems are hard*. *SIAM Journal on Computing* 22(1), pp. 117–141.

[20] N. Koblitz (1994): *A Course in Number Theory and Criptography*. Springer. Available at `http://dx.doi.org/10.1007/978-1-4419-8592-7`.

[21] H. Körner (2003): *A Time and Space Efficient Algorithm for Minimizing Cover Automata for Finite Languages*. *International Journal of Foundations of Computer Science* 14(6), pp. 1071–1086. Available at `http://dx.doi.org/10.1142/S0129054103002187`.

[22] A. N. Maslov (1970): *Estimates of the number of states of finite automata*. *Soviet Mathematics Doklady* 11, pp. 1373–1374.

[23] A. N. Maslov (1973): *Cyclic shift operation for languages*. *Probl. Inf. Transm* 9, pp. 333–338.

[24] F. Mera & G. Pighizzini (2005): *Complementing unary nondeterministic automata*. *Theoretical Computer Science* 330, pp. 349–360. Available at `http://dx.doi.org/10.1016/j.tcs.2004.04.015`.

[25] E. F. Moore (1956): *Gedanken-experiments on sequential machines*. *Automata studies, Annals of mathematics studies* 34, pp. 129–153.

[26] D. Revuz (1992): *Minimisation of acyclic deterministic automata in linear time*. *Theoretical Computer Science* 92(1), pp. 181 – 189. Available at `http://dx.doi.org/10.1147/rd.32.0114`.

[27] S. Yu, K. Salomaa & Q. Zhuang (1994): *The state complexities of some basic operations on regular languages*. *Theoretical Computer Science* 125(2), pp. 315–328. Available at `http://dx.doi.org/10.1016/0304-3975(92)00011-F`.