Scientific Annals of Computer Science vol. 25 (1), 2015, pp. 3–28
doi: 10.7561/SACS.2015.1.3

Non-Deterministic Finite Cover Automata[1]

Cezar CÂMPEANU[2]

Abstract

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 using a Non-deterministic Finite Automata (NFA) may significantly reduce the number of required states. 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 non-deterministic finite cover automata (NFCA) and l-non-deterministic 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.

Keywords: Regular languages, finite languages, cover automata, l-cover automata, similarity relation

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 Non-deterministic Finite Automata, and showed that the equivalent Deterministic

[1]This is an extended journal version of [4].

[2]Department of Computer Science and Information Technology, Faculty of Science, University of Prince Edward Island, Charlottetown, Prince Edward Island, Canada, E-mail: ccampeanu@upei.ca

Finite Automaton can be, in terms of number of states, exponential larger than the NFA. Since, it was proved in [28] that we can obtain a polynomial algorithm for minimizing DFAs, and in [19] 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, 21], and it was proved by Jiang and Ravikumar [22] that NFA minimization problems are hard; even in case of regular languages over a one letter alphabet, the minimization is NP-complete [13, 22].

On the other hand, in case of finite languages, we can obtain minimizing algorithms [25, 29] that are in the order of $O(n)$, where n is the number of states of the original DFA. In [6, 8, 24] 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 [25, 26], but has not become a subject of systematic study until 1994 [31]. The special case of state complexity of operations on finite languages was studied in [7].

Non-deterministic state complexity of regular languages was also subject of interest, for example in [15, 16, 17, 18]. To find lower bounds for the non-deterministic state complexity of regular languages, the fooling set technique, or the extended fooling set technique may be used [3, 11, 13].

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, Theorem 4. 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 treatment when applied to NFCAs, as many results valid for the DFCAs are no longer true for NFCAs. We show a connection between fooling sets and NFA state reduction in Section 6. In section 7, 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 empty words, i.e., the word with no letters, is denoted by ε. The length of a word $w = w_1 w_2 \dots w_n$, $n \geq 0$, $w_i \in \Sigma$, $1 \leq i \leq n$, is $|w| = n$, in particular $|\varepsilon| = 0$ (for $n = 0$, $w = \varepsilon$). 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 the function δ, 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., $l = \max\{|w| \mid w \in L\}$[3]. 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 obviously see that any automaton that recognizes L is also a cover automaton.

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

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 [8] for a similar definition in case of DFCA. Obviously, $level_A(s) = |x_A(s)|$.

For a regular language L, $\equiv_L$ denotes the Myhill-Nerode equivalence of words [20, 30].

The similarity relation induced by a finite language L is defined as follows[8]: $x \sim_L y$, 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 $p \sim_A q$, if $\delta(p, w) \cap F \neq \emptyset$ iff $\delta(q, w) \cap F \neq \emptyset$, for all $w \in \Sigma^{\leq l - \max\{level(p), level(q)\}}$.*

In all cases when the automaton 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 an NFCA for L.

In case $level(p) \leq l$, $level(q) \leq l$, and $p \sim q$, 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 $sc(L)$:

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

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 $ncsc(L)$:

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

Obviously, $ncsc(L) \le nsc(L) \le sc(L)$, but also $ncsc(L) \le csc(L) \le sc(L)$. 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 non-determinism, or the use of a cover language, reduce the state complexity. Let us first analyze the type of languages where non-determinism, combined with cover properties, significantly reduce the state complexity.

We choose the language $L_{F_{m,n}} = \{a,b\}^{\le m}a\{a,b\}^{n-2}$, where $m,n \in \mathbb{N}$. In Figure 1, we can see an NFA recognizing $L_{F_{m,n}}$ with $m+n$ states. Please note that the longest word in the language has $m+n-1$ letters.

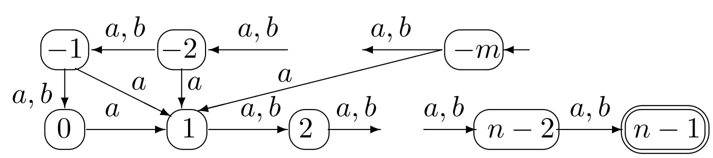

Figure 1: An NFA with $m+n$ states for the language $L_{F_{m,n}} = \{a,b\}^{\le m}a\{a,b\}^{n-2}$.

Let us analyze if the automaton in Figure 1 is minimal. The fooling set technique, introduced in [10] and [12] and used to prove the lower-bound for state complexity of NFAs, is stated in [3, 10] 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 language $L_{F_{m,n}}$ and following set of pairs of words, $S = S_1 \cup S_2 = \{(x_k, y_k) \mid 1 \leq k \leq m+n\}$, where $S_1 = \{(b^m a b^j, b^{n-2-j}) \mid 0 \leq j \leq n-2\}$ and $S_2 = \{(a^i, b^{m-i} a b^{n-2}) \mid 0 \leq i \leq m\}$.

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

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

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

1. Case I

   $(x_k, y_k) = (b^m a b^i, b^{n-2-i}) \in S_1$ and $(x_h, y_h) = (b^m a b^j, b^{n-2-j}) \in S_1$

   (a) $x_k y_h = b^m a b^i b^{n-2-j} \notin L_{F_{m,n}}$, and

   (b) $x_h y_k = b^m a b^j b^{n-2-i} \notin L_{F_{m,n}}$.

2. Case II

   $(x_k, y_k) = (a^i, b^{m-i} a b^{n-2}) \in S_2$ and $(x_h, y_h) = (a^j, b^{m-j} a b^{n-2}) \in S_2$

   (a) $x_k y_h = a^i b^{m-j} a b^{n-2} \in L_{F_{m,n}}$, if $i < j$, but

   (b) $x_h y_k = a^j b^{m-i} a b^{n-2} \notin L_{F_{m,n}}$, if $i < j$ (because $|a^j b^{m-i} a b^{n-2}| = m+n-1+j-i > m+n-1$).

3. Case III

   $(x_k, y_k) = (b^m a b^j, b^{n-2-j}) \in S_1$ and $(x_h, y_h) = (a^i, b^{m-i} a b^{n-2}) \in S_2$

   (a) $x_k y_h = b^m a b^j b^{m-i} a b^{n-2} \notin L_{F_{m,n}}$ (because $|b^m a b^j b^{m-i} a b^{n-2}| = m+1+j+m-i+1+n-2 > m+n-1$).

¿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, ignoring the cases when the length exceeds the maximal one.

Hence, the fooling set technique introduced in [10] and [12], 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 [13].

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

1. *For all i, j, $1 \leq i, j \leq n$, such that $x_i y_j \in \Sigma^{\leq l}$, if $i \neq j$, we have that $x_i y_j \notin L$. Then we have that $ncsc(L) \geq n$.*

   The set S is called a fooling set *for L.*

2. *For all i, j, $1 \leq i, j \leq n$, if $i \neq j$, we have either $x_i y_j \in \Sigma^{\leq l}$ and $x_i y_j \notin L$, or $x_j y_i \in \Sigma^{\leq l}$ and $x_j y_i \notin L$. Then we have that $ncsc(L) \geq n$.*

   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 and $m < n$. For each i, $1 \leq i \leq n$, $x_i y_i \in L$, therefore we must have a state $s_i \in \delta(q_0, x_i)$ and $\delta(s_i, y_i) \cap F \neq \emptyset$. 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 $j \neq i$, thus $m \geq n$. 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 $|x_j y_i| \leq l$, 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, thus $m \geq n$. 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 $|x_i y_j| \leq l$, it follows that $x_i y_j \in L$, a contradiction. If $|x_j y_i| \leq l$, using the same reasoning, will follow that $x_j y_i \in L$.

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.

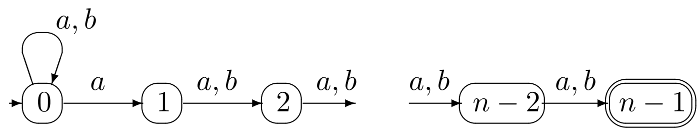

Figure 2: An NFCA with n states for the language $L_{F_{m,n}} = \{a,b\}^{\leq m}a\{a,b\}^{n-2}$, that is the same as the one in Figure 1. In case $m = 2$ and $n = 4$, the language is the same as the one described in Figure 3.
An equivalent minimal NFA has $m + n$ states.

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 for $L_{F_{2,4}}$ has 15 states.

For helping the reader to better understand the compression power of NFCA over NFAs, DFCAs, or DFAs, we present corresponding automata for a smaller example, i.e., for the language $L_{F_{2,3}}$. In this case, a minimal NFCA presented in Figure 4 has 3 states, a minimal NFA in Figure 5 has 4 states, a minimal DFCA in Figure 7 has also 4 states, and the minimal DFA in Figure 6 has 11 states.

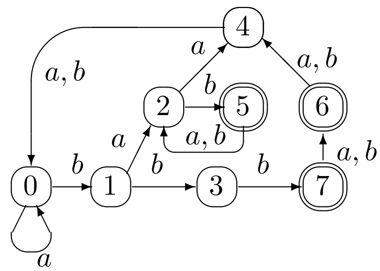

Figure 3: A minimal DFCA with 8 states for the language $L_{F_{2,4}} = \{a,b\}^{\leq 2}a\{a,b\}^2$, $l = 5$ and the equivalent minimal DFA has 15 states.

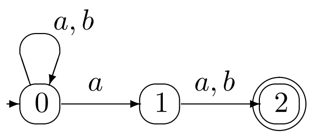

Figure 4: An NFCA with $n = 3$ states for the language $L_{F_{2,3}} = \{a,b\}^{\leq 2}a\{a,b\}$

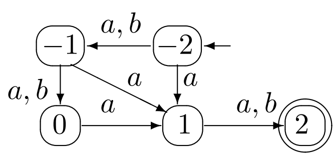

Figure 5: An NFA with $2 + 3 = 5$ states for the language $L_{F_{2,3}} = \{a,b\}^{\leq 2}a\{a,b\}$

We can observe that we do have the following similarities in Figure 6: $7 \sim 3$, $8 \sim 4$, $9 \sim 1$, $10 \sim 0$, thus we can obtain the corresponding minimal DFCA in Figure 7.

These language examples show 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

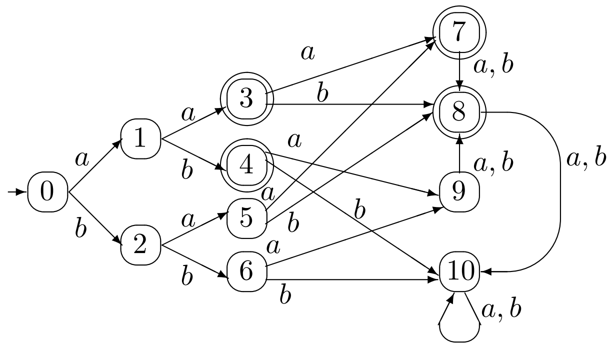

Figure 6: The minimal DFA with 11 states for the language $L_{F_{2,3}} = \{a,b\}^{\leq 2}a\{a,b\}$

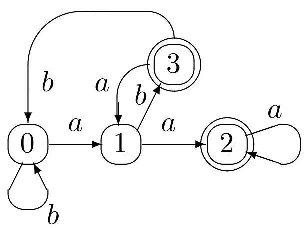

Figure 7: A minimal DFCA with 4 states for the language $L_{F_{2,3}} = \{a,b\}^{\leq 2}a\{a,b\}$, $l = 4$

NFA.

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

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

In Figure 8, the NFA A_{X_k} accepts the language $L_{X_k} = a(\Sigma^* - \{(a^k)^n \mid n \geq 0\}) = \{aaa^i \mid i \neq k-1 \bmod k\}$, which is a cover language for $L_{X_{l,k}}$. In other words, A_k is an NFCA for $L_{X_{l,k}}$, therefore $ncsc(L_{X_{l,k}}) \leq csc(L_{X_{l,k}}) \leq sc(L_{X_{l,k}}) \leq \min(l+1, k+1) = k+1$.

It is known [10, 16, 27] that the automaton A_{X_k} is minimal NFA for $L_{X_k} = \bigcup_{l \in \mathbb{N}, l > k} L_{X_{l,k}}$, if k is a prime number. However, this may not be a minimal NFCA, as illustrated by the example in Figure 9, where A_{X_7} is not a minimal NFCA for $L_{X_{9,7}}$, even if it is minimal NFA for the cover language

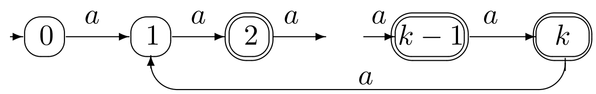

Figure 8: An NFA/NFCA A_k for $L_{l,k}$. In this particular case, A_k is also the minimal DFA.

L_{X_k} .[4]

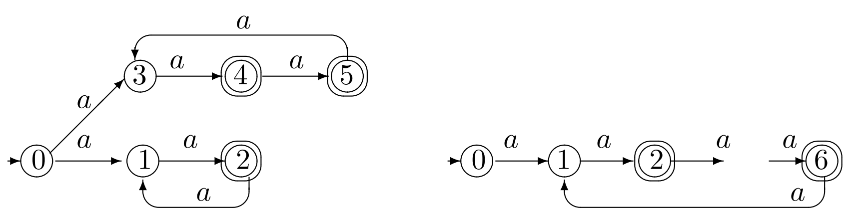

Figure 9: A minimal NFCA for $L_{X_{9,7}}$, left, and a minimal NFA, A_{X_7}, for a cover language, right.

We apply the extended fooling set technique for the language $L_{X_{l,k}}$. Because the alphabet is unary, all the words in an extended fooling set S are powers of a. Thus, considering only pairs in the fooling set S, such that the first word is not ε we have that for some $r \in \mathbb{N}$: $S \supseteq \{(a^{i_1}, a^{j_1}), (a^{i_2}, a^{j_2}), (a^{i_3}, a^{j_3}), \ldots, (a^{i_r}, a^{j_r})\}$, and $1 \leq i_1, \ldots, i_r \leq k$.

We show that r cannot be greater than 3, thus S has at most 4 elements. It is enough to take four pairs (a^{i_1}, a^{j_1}),(a^{i_2}, a^{j_2}),(a^{i_3}, a^{j_3}), (a^{i_4}, a^{j_4}), and show that they cannot have the extended fooling set property. We have $a^{i_1}a^{j_2} \notin L_{X_{l,k}}$, or $a^{i_2}a^{j_1} \notin L_{X_{l,k}}$, and $a^{i_1}a^{j_3} \notin L_{X_{l,k}}$, or $a^{i_3}a^{j_1} \notin L_{X_{l,k}}$, and $a^{i_1}a^{j_4} \notin L_{X_{l,k}}$, or $a^{i_4}a^{j_1} \notin L_{X_{l,k}}$. Without any loss of generality, we may assume that $i_1 + j_2 = z_{12}k + 1$ and $i_1 + j_3 = z_{13}k + 1$, all the other cases being similar, as they are just permutations of indexes, or replacing i's by j's. If $i_1, i_2, i_3 \geq 1$, 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 $z_{23}, z_{32} \in \mathbb{N}$, because $i_2 + j_3 = z_{23}k + 1$ would imply $i_1 + j_2 + i_2 + j_3 = xk + 2$, $x = z_{12} + z_{23}$, which means $j_2 + i_2 = xk + 2 - z_{13}k - 1 = yk + 1$, for $y = x - z_{13}$, a contradiction,

[4]Please note that for any finite language, there are infinitely many cover languages, as in Definition 1.

and $i_3 + j_2 = z_{32}k + 1$ would imply $i_3 + j_2 + i_1 + j_3 = xk + 2$, $x = z_{13} + z_{32}$, i.e., $i_3 + j_3 = xk + 2 - z_{12}k - 1 = yk + 1$, for $y = x - z_{12}$, which is also not possible.

It follows that $r \leq 3$, thus any extended fooling set for $L_{X_{l,k}}$ has at most 4 elements.

Let A be an NFA accepting $L \supseteq L_{X_k}$, 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 for k prime, A_{X_k} is minimal for some language $L_{X_{l,k}}$, $l > k$.

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 $L(B) = L_{X_{l,k}}$. 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, because k is prime, therefore, for large enough l, i.e., greater than some $l_{0,k}$, $L_{X_{l,k}} \neq L(B)$, because $a^{zk+1} \in L(B) \setminus L_{X_{l,k}}$, for some $z \in \mathbb{N}$. Thus, the number of states in B is at least $k+2$. The automaton A_{X_k} is also a minimal NFCA for languages $L_{X_{l,k}}$, if $l \geq l_{0,k}$, hence it follows that Theorem 7 in [13] is also valid for cover automata:

Theorem 1 *There is a sequence of languages $(L_{X_{l,k}})_{k \geq 2}$ such that the nondeterministic cover complexity of $L_{X_{l,k}}$ is at least k, but the extended fooling set for $L_{X_{l,k}}$ 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 [14], used to prove that minimizing NFAs is hard. We will describe the construction from [12, 14], showing that we can also use it with only a minor addition for NFCAs. 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.

The idea of proving that NFCA minimization is NP-hard is the following: we take an arbitrary logical formula in conjunctive normal form F, and we build two languages L_C and L_B such that their union $L_F = L_B \cup L_C$ should not include $\{a\}^*$ if F is satisfiable, in other words, $\{a\}^*$ is an l-cover language for $L_F \cap a^{\leq l}$, iff the formula is not satisfiable. 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, $1 \leq i \leq m$, is defined using variables $x_1, \ldots, x_n$, $C_i = u_1 \vee u_2 \vee u_3$, and each u_j, $1 \leq j \leq 3$ 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 [23][5], it follows that there exists a bijective function $f : \mathbb{Z}_k \longrightarrow \prod_{i=1}^{n} \mathbb{Z}_{p_i}$, such that $f(x) = (x \bmod p_1, \ldots, x \bmod p_n)$. 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 $L_F \cap \Sigma^{\leq l}$ has $\{a\}^*$ as a cover language.

In a similar fashion as we built the automata A_{X_k}, we can construct an automaton B_i that recognizes the language $L(B_i) = \{a^n \mid n \bmod p_i \notin \{0, 1\}\}$ in $O(p_n)$ time. Let B be an automaton recognizing $L_B = \bigcup_{i=1}^{n} L(B_i)$. It is straightforward 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[6]. For every sequence s of 0s and 1s of length n, there is an unique number $m \in \mathbb{Z}_k$, such that $f(m) = s$. In [14], the binary sequence s of length n is called *representation*, and its corresponding number m is called *assignment*. The $range(f)$ may contain other sequences in $\prod_{i=1}^{n} \mathbb{Z}_{p_i}$, and using the above observation, we deduct that for the language L_B, we have

$$L_B = \{a^i \mid i \text{ does not represent an assignment}\},$$

while for L_C, we have

$$L_C = \{a^i \mid f(i) \text{ does not satisfy } F\}.$$

We set $L_F = L_C \cup L_B$, where $L_C = \bigcup_{i=1}^{m} L_{C_i}$. If F is satisfiable, then L_F is a cyclic language with period at most k, and the minimal period of L_F is $\frac{l}{2}$, according to [10, 12]. Thus, setting $l = k$, we have that $L_F \cap \{a\}^{\leq l}$ has $\{a\}^*$ as a cover language, iff F is unsatisfiable. It follows that for some $l \in \mathbb{N}$, $\{a\}^*$ is an l-cover language for $L_F \cap \Sigma^{\leq l}$, iff F is unsatisfiable. Please note that the construction is similar to the one in [14], but in our case, we also prove that the constant l exists, and the language constructed is an l-cover language.

[5] Theorem I.3.3, page 21

[6] Using Cartesian product construction, for example.

Since according to [1] primality test can be done in polynomial time, we can find the first n prime numbers in polynomial time, meaning that our NFA construction can also be done in polynomial time. Hence, checking if $\{a\}^*$ is an l-cover language for $L_F \cap \{a\}^{\leq l}$, is in NP.

If F is unsatisfiable, then $ncsc(L) = 1$, otherwise the minimal number of states in an NFA is at least equal to the largest prime number dividing its period, p_n. To prove that finding the minimal NFCA for L_F is NP-hard, we use the same argument as in [14]: the existence of a polynomial algorithm to decide if $ncsc(L) = o(n)$ implies that $nsc(L) = o(n)$, which implies that we can solve $3SAT$ in polynomial time, i.e., $P = NP$. This means that minimizing NFCAs is at least NP-hard. Consequently, we proved that:

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

In the next section we analyze some methods to reduce the number of states of a NFCA, because any minimization algorithm would be at least exponential.

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_{X_7} and $L_{X_{9,7}}$.

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 by simply collapsing one into the other and consolidate all their input and output transitions, [5], 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 [9]. We must note that in case of NFCAs, the size of an NFA without mergeable states is bounded, as we cannot have a path from the initial state to the final ones that is longer than l. This observation contrasts with the result obtained in [9], for NFAs, where it is presented a class of arbitrary large

NFAs without any group of a fixed size k, of mergeable states. The same methods are considered for NFCAs.

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

1. *We say that the state q is* weakly mergeable *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 an NFCA for L. In this case we write $p \precapprox q$.

2. *We say that the state q is* strongly mergeable *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 an 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 an NFCA for the finite language L. It is possible that $x_A(s) \sim_L x_A(q)$, but s and q are not mergeable.*

Proof: For the automaton in Figure 9, left, $x_A(3) = x_A(1)$, but the states 1 and 3 are not mergeable, as the resulting automaton would not reject a^7. $\square$

Lemma 4 *Let $A = (Q, \Sigma, \delta, q_0, F)$ be an 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 10.

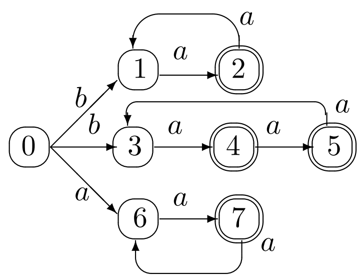

Figure 10: An example where $p \sim_A q$, $x \not\sim_L y$, but $p \in \delta(q_0, x)$ and $q \in \delta(q_0, y)$, namely, $aa \not\sim_L ba$, $2 \in \delta(0, ba)$, $7 \in \delta(0, aa)$, and $2 \sim_A 7$.

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 $w \in \Sigma^* - \{a\}^*$.

$\square$

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

Lemma 5 *Let $A = (Q, \Sigma, \delta, q_0, F)$ be an 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 10 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 $p \sim_A q$, $1, 3 \in \delta(p, b) = \delta(q, b) = \delta(0, b)$, but $r \not\sim_A s$, because $\delta(1, a) \cap F = \emptyset$ and $\delta(3, a) \cap F = \{4\} \neq \emptyset$. □

This result contrasts with the one for the deterministic case for cover automata, and the main reason is the non-determinism, 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 in Theorem 3. 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 $p \sim_A q$. 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 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 $\delta'(q_0, w) \cap F' \neq \emptyset$.

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 11 as an NFCA for $L = \{a^2, a^4\}$. We have that $l = 4$ and $3 \sim_A 5$, 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 12 we have the result for strongly merging state 3 in state 5.

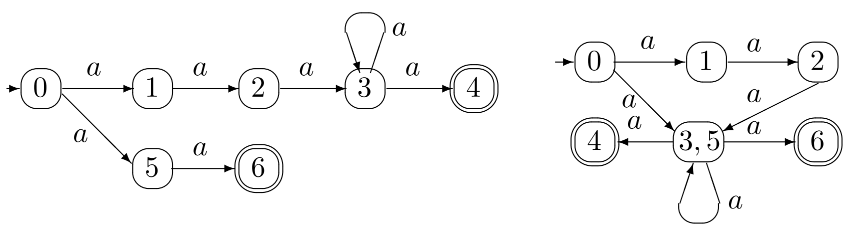

Figure 11: Example for weakly merging failure and similar states.

$\square$

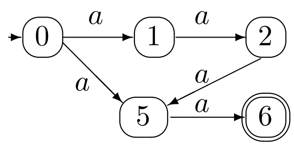

Figure 12: The result for strongly merging similar states for the example presented in Figure 11.

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 an 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: $F_B = (F \setminus Q_R) \cup F_1 \cup F_2$, in case $p \notin F$, and $F_B = (F \setminus Q_R) \cup F_1 \cup F_2 \cup \{p\}$ 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 $L_{X_{9,7}}$ in Figure 9.

6 State Merging and Fooling Sets

In this section we analyze the relation between mergeable states and fooling sets, more precisely, we would like to use a fooling set to identify states that are mergeable or not. We will consider both strong and weak mergeability. First, we start with a technical lemma.

Lemma 7 *Let $S = \{(x_i, y_i) \mid 1 \leq i \leq n\}$ be a(n) (extended) fooling set for the finite language L. If i, j are such that $1 \leq i, j \leq n$, $i \neq j$, then either $|x_i y_j| \leq l$, or $|x_j y_i| \leq l$.*

Proof: Assume $|x_j y_i| > l$. It follows that $|x_j| + |y_i| > l$, i.e., $|y_i| > l - |x_j|$. Because $x_i y_i \in L$, $|x_i y_i| \leq l$, i.e., $|x_i| + |y_i| \leq l$, hence $|y_i| \leq l - |x_i|$, thus $l - |x_j| < l - |x_i|$, which means that $|x_i| < |x_j|$. We have that $|x_i y_j| = |x_i| + |y_j| < |x_j| + |y_j| \leq l$. □

If a fooling set for a finite language L has exactly the number of states in a given NFCA, then the NFCA is minimal. In case the NFCA has more states, it could still be minimal. However, we would like to investigate if it is not minimal, and some states might be either weakly, or strongly mergeable.

The following result identifies, for a given fooling set, the states that are meargeable in an NFCA.

Theorem 4 *Let $A = (Q, \Sigma, \delta, q_0, F)$ be an NFCA for L, and $S = \{(x_i, y_i) \mid 1 \leq i \leq n\}$ a fooling set for L. Let $p, q \in Q$, $p \neq q$ be two states, and i, j, $1 \leq i, j \leq n$, $i \neq j$, be such that $p \in \delta(q_0, x_i)$ and $q \in \delta(q_0, x_j)$. Then the following statements hold true*

1. *If $\delta(q, y_j) \cap F \neq \emptyset$, and $\delta(p, y_i) \cap F \neq \emptyset$, then $p \not\precapprox q$.*
2. *If $\delta(p, y_j) \cap F \neq \emptyset$, and $|x_j y_i| > l$ then $p \not\precapprox q$.*
3. *If $|x_i y_j| > l$ and $\delta(p, y_i) \cap F \neq \emptyset$, then $p \not\precsim q$.*

Proof: Because S is a fooling set, it follows that $x_i y_j \notin L$ and $x_j y_i \notin L$. We now analyze each case of the theorem:

1. If $p \precapprox q$, it follows that by merging q with p, we obtain an equivalent NFCA $B = (Q - \{q\}, \Sigma, \delta_B, q_0, F - \{q\})$ such that $\emptyset \neq \delta(q, y_j) \cap F \subseteq \delta_B(p, y_j) \cap F_B$. Because $x_i y_j \notin L$, we must have $|x_i y_j| > l$, which implies $|x_j y_i| \leq l$. ¿From $q \in \delta(q_0, x_j)$, it follows $p \in \delta_B(q_0, x_j)$. But $\delta(p, y_i) \cap F \neq \emptyset$, therefore $\delta_B(q_0, x_j y_i) \cap F \neq \emptyset$, i.e., $x_j y_i \in L$, which is a contradiction.

2. If $p \precapprox q$, it follows that by merging q with p, we obtain an equivalent NFCA $B = (Q - \{q\}, \Sigma, \delta_B, q_0, F - \{q\})$ such that $\emptyset \neq \delta(p, y_j) \cap F \subseteq \delta_B(p, y_j) \cap F_B$. Because S is a fooling set and $|x_j y_i| > l$, using Lemma 7 we have that $|x_i y_j| \leq l$. But $\delta_B(q_0, x_i y_j) \cap F \neq \emptyset$, therefore $x_i y_j \in L$, which is a contradiction.

3. If $p \precsim q$, it follows that by merging q into p, we obtain an equivalent NFCA $B = (Q - \{q\}, \Sigma, \delta_B, q_0, F - \{q\})$ such that $\emptyset \neq \delta(q, y_i) \cap F \subseteq \delta_B(p, y_i) \cap F_B$.

   Because $|x_i y_j| > l$, using Lemma 7, we have that $|x_j y_i| \leq l$. ¿From $q \in \delta(q_0, x_j)$, it follows $p \in \delta_B(q_0, x_j)$. But $\delta(p, y_i) \cap F \neq \emptyset$, therefore $\delta_B(q_0, x_j y_i) \cap F \neq \emptyset$, i.e., $x_j y_i \in L$, which is a contradiction.

□

Because in the proof of 2 and 3 of Theorem 4 we use the condition on the length of the words in the fooling set, only one of the words x_iy_j or x_jy_i has to be tested if it is in L. Thus, if S is an extended fooling set, both 2 and 3 of Theorem 4 will hold.

For 1 of Theorem 4, and the case where S is an extended fooling set, we must consider two cases:

1. $x_iy_j \notin L$, and
2. $x_jy_i \notin L$.

If $x_iy_j \notin L$, then we have the same proof. If $x_jy_i \notin L$, from $\delta(p, y_i) \cap F \neq \emptyset$, it follows that $|x_jy_i| > l$, and using Lemma 7, we have that $|x_iy_j| \leq l$. Because $\delta_B(q_0, x_iy_j) \subseteq \delta(q, y_j) \setminus \{q\} \neq \emptyset$, it follows that $x_iy_j \in L$, which is a contradiction.

Hence, Theorem 4 holds for both fooling sets and extended fooling sets. The reverse is not true, as the fooling set technique does not provide a tight lower bound for the number of states. In the following examples, we show that if some initial conditions are not satisfied, then the states can be mergeable.

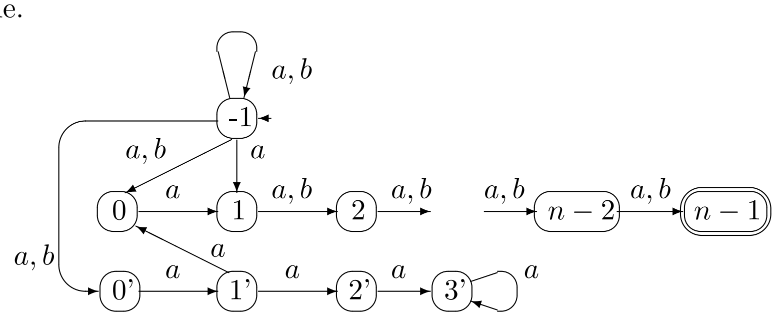

Figure 13: Example of fooling set and mergeable states not satisfying the conditions in Theorem 4.

Example 1 *In the following example, Figure 13, we have an NFCA with* $n+5$ *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. A fooling set or extended fooling set can be* $S = \{(a^i, ab^{n-1-i}) \mid 0 \leq i \leq n\}$*, which guarantees that the minimal NFCA has at least* n *states. An equivalent minimal NFA has* $m + n$ *states.*

We also have the following:

1. *If* $p = 0$ *and* $q = 1'$*, then* $p \precapprox q$*. However, for the following values,* $i = 1$ *and* $j = 2$*, we have that* $p \in \delta(-1, x_i)$ *and* $q \in \delta(-1, x_j)$*. In this case,* $x_i = a$*,* $x_j = aa$*,* $y_i = ab^{n-1-1}$*,* $y_j = ab^{n-1-2}$*, and* $\delta(q, y_j) \cap F = \emptyset$*.*

2. *For* $p = 0$ *and* $q = 2'$*,* $i = 1$*, and* $j = n - 1$*, we have* $x_i = a$*,* $x_j = a^{n-1}$*,* $y_i = ab^{n-1-1}$*,* $y_j = a$*,* $0 \precapprox 2'$*, and* $\delta(q, y_i) \cap F = \emptyset$*, but* $|x_i y_j| < l$*, and* $|x_j y_i| > l$*.*

3. *For* $p = 0$*,* $q = 3'$*,* $i = 1$*, and* $j = 4$*, we have* $x_i = a$*,* $x_j = aaaa$*,* $y_i = ab^{n-1-1}$*,* $y_j = ab^{n-1-4}$*.* $0 \precsim 3'$*, but* $0 \not\precapprox 3'$*, and* $\delta(q, y_j) \cap F = \emptyset$*.*

Remark 1 *For* $p = 0$ *and* $q = 2'$*,* $i = 1$*, and* $j = 3$*, we have* $x_i = a$*,* $x_j = aaa$*,* $y_i = ab^{n-1-1}$*,* $y_j = ab^{n-1-3}$*,* $0 \precsim 2'$*, and* $\delta(q, y_j) \cap F = \emptyset$*.*

The examples presented, together with the Theorem 4, show the difficulty of finding mergeable states in an NFCA, even if we already know a fooling set. This suggests that expanding the study, even for the case of NFAs, may produce some useful practical results.

7 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 proved 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 proved 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.

We also present an interesting connection between mergeability and fooling sets, that could be further extended.

As future research, we list below some problems that we consider worth investigating:

1. check if the bipartite graph lower-bound technique can be applied for NFCAs;

2. find bounds for non-deterministic 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, and N. Saxena. PRIMES is in P. *Annals of mathematics*, 160(2):781–793, 2004. doi:10.4007/annals.2004.160.781.

[2] J. Amilhastre, P. Janssen, and M-C. Vilarem. FA minimisation heuristics for a class of finite languages. In O. Boldt and H. Jürgensen, editors, *Proceedings of the 4th International Workshop on Implementing Automata (WIA' 99)*, volume 2214, pages 1–12, 2001. doi:10.1007/3-540-45526-4_1.

[3] J.-C. Birget. Intersection and union of regular languages and state complexity. *Information Processing Letters*, 43(4):185–190, 1992. doi:10.1016/0020-0190(92)90198-5.

[4] C. Câmpeanu. Simplifying nondeterministic finite cover automata. In Z. Ésik and Z. Fülöp, editors, *Proceedings of the 14th International Conference on Automata and Formal Languages (AFL 2014)*, volume 151 of *EPTCS*, pages 162–173, 2014. doi:10.4204/EPTCS.151.11.

[5] C. Câmpeanu, N. Sântean, and S. Yu. Mergible states in large NFA. *Theoretical Computer Science*, 330(1):23–34, 2004. doi:10.1016/j.tcs.2004.09.008.

[6] C. Câmpeanu, A. Păun, and S. Yu. An efficient algorithm for constructing minimal cover automata for finite languages. *International Journal of Foundations of Computer Science*, 13(1):83–97, 2002. doi:10.1142/S0129054102000960.

[7] C. Câmpeanu, K. Culik II, K. Salomaa, and S. Yu. State complexity of basic operations on finite languages. In O. Boldt and H. Jürgensen, editors, *Proceedings of the 4th International Workshop on Implementing Automata (WIA' 99)*, volume 2214 of *Lecture Notes in Computer Science*, pages 60–70, 2001. doi:10.1007/3-540-45526-4_6.

[8] C. Câmpeanu, N. Sântean, and S. Yu. Minimal cover-automata for finite languages. *Theoretical Computer Science*, 267(1-2):3–16, 2001. doi:10.1016/S0304-3975(00)00292-9.

[9] C. Câmpeanu, N. Sântean, and S. Yu. A family of NFAs free of state reductions. *Journal of Automata Languages and Combinatorics*, 12(1):69–78, 2007. Available from: http://dl.acm.org/citation.cfm?id=1463362.1463367.

[10] M. Chrobak. Finite automata and unary languages. *Theoretical Computer Science*, 47:149–158, 1986. doi:10.1016/0304-3975(86)90142-8.

[11] I. Glaister and J. Shallit. A lower bound technique for the size of nondeterministic finite automata. *Information Processing Letters*, 59(2):75–77, 1996. doi:10.1016/0020-0190(96)00095-6.

[12] G. Gramlich. Probabilistic and nondeterministic unary automata. In B. Rovan and P. Vojtás, editors, *Proceedings of the 28th International Symposium on Mathematical Foundations of Computer Science (MFCS 2003)*, volume 2747 of *Lecture Notes in Computer Science*, pages 460–469, 2003. doi:10.1007/978-3-540-45138-9_40.

[13] H. Gruber and M. Holzer. Finding lower bounds for nondeterministic state complexity is hard. In Oscar H. Ibarra and Zhe Dang, editors, *Proceedings of the 10th International Conference on Developments in Language Theory (DLT 2006)*, volume 4036 of *Lecture Notes in Computer Science*, pages 363–374, 2006. doi:10.1007/11779148_33.

[14] H. Gruber and M. Holzer. Computational complexity of NFA minimization for finite and unary languages. In R. Loos, S.Z. Fazekas, and C. Martín-Vide, editors, *Proceedings of the 1st International Conference on Language and Automata Theory and Applications (LATA 2007)*, pages 261–272, 2007. Available from: http://www.hermann-gruber.com/data/lata07-final.pdf.

[15] M. Holzer and M. Kutrib. State complexity of basic operations on nondeterministic finite automata. In J.-M. Champarnaud and D. Maurel, editors, *Implementation and Application of Automata*, volume 2608 of *Lecture Notes in Computer Science*, pages 148–157. 2003. doi:10.1007/3-540-44977-9_14.

[16] M. Holzer and M. Kutrib. Unary language operations and their nondeterministic state complexity. In M. Ito and M. Toyama, editors, *Proceedings of the 6th International Conference on Developments in Language Theory (DLT 2002)*, volume 2450 of *Lecture Notes in Computer Science*, pages 162–172, 2003. doi:10.1007/3-540-45005-X_14.

[17] M. Holzer and M. Kutrib. Descriptional and computational complexity of finite automata. In A.H. Dediu, A.M. Ionescu, and C. Martin-Vide, editors, *Proceedings of the Third International Conference on Language and Automata Theory and Applications (LATA 2009)*, volume 5457 of *Lecture Notes in Computer Science*, pages 23–42, 2009. doi:10.1007/978-3-642-00982-2_3.

[18] M. Holzer and M. Kutrib. Nondeterministic finite automata – recent results on the descriptional and computational complexity. *International Journal of Foundation of Computing Science*, 20(4):563–580, 2009. doi:10.1142/S0129054109006747.

[19] J. Hopcroft. An $n \log n$ algorithm for minimizing states in a finite automaton. In Z. Kohavi and A. Paz, editors, *Theory of Machines and Computations*, pages 189–196. Academic Press, New York, 1971.

[20] J.E. Hopcroft and J.D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, 1979.

[21] L. Ilie, G. Navarro, and S. Yu. On NFA reductions. In J. Karhumäki, H. Maurer, G. Păun, and G. Rozenberg, editors, *Theory Is Forever - Essays Dedicated to Arto Salomaa on the Occasion of His 70th Birthday*, volume 3113 of *Lecture Notes in Computer Science*, pages 112–124. 2004. doi:10.1007/978-3-540-27812-2_11.

[22] T. Jiang and B. Ravikumar. Minimal NFA problems are hard. In Javier Leach Albert, Burkhard Monien, and Mario Rodríguez Artalejo, editors, *Automata, Languages and Programming*, volume 510 of *Lecture Notes in Computer Science*, pages 629–640. Springer Berlin Heidelberg, 1991. doi:10.1007/3-540-54233-7_169.

[23] N. Koblitz. *A Course in Number Theory and Cryptography*. Springer, 1994. doi:10.1007/978-1-4419-8592-7.

[24] H. Körner. A time and space efficient algorithm for minimizing cover automata for finite languages. *International Journal of Foundations of Computer Science*, 14(6):1071–1086, 2003. doi:10.1142/S0129054103002187.

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

[26] A. N. Maslov. Cyclic shift operation for languages. *Problems of Information Transmission*, 9:333–338, 1973.

[27] F. Mera and G. Pighizzini. Complementing unary non-deterministic automata. *Theoretical Computer Science*, 330(2):349–360, 2005. doi:10.1016/j.tcs.2004.04.015.

[28] E. F. Moore. Gedanken-experiments on sequential machines. *Automata Studies, Annals of Mathematics Studies*, 34:129–153, 1956.

[29] D. Revuz. Minimisation of acyclic deterministic automata in linear time. *Theoretical Computer Science*, 92(1):181–189, 1992. doi:10.1016/0304-3975(92)90142-3.

[30] S. Yu. Regular languages. In Grzegorz Rozenberg and Arto Salomaa, editors, *Handbook of Formal Languages*, volume 1, pages 41–110. Springer, 1997. Available from: http://dl.acm.org/citation.cfm?id=267846.267848.

[31] S. Yu, Q. Zhuang, and K. Salomaa. The state complexities of some basic operations on regular languages. *Theoretical Computer Science*, 125(2):315–328, 1994. doi:10.1016/0304-3975(92)00011-F.