Symmetric Difference NFAs: the State of the Art

Lynette van Zijl Jaco Geldenhuys

Computer Science
Stellenbosch University
South Africa
{lynette,jaco}@cs.sun.ac.za

Abstract

Symmetric difference automata are a special case of automata with multiplicities, and have received attention since 1997[16]. This chapter provides a unified survey of the current state of the art for symmetric difference automata.

1 Introduction

This chapter is meant as an informal background and survey of symmetric difference nondeterministic finite automata ($\oplus$-NFAs). Examples are given to explain the various concepts, and most results are listed without proof. We concentrate on $\oplus$-NFAs over a one-letter alphabet (unary $\oplus$-NFAs). We refer the reader to another chapter in this book for a more mathematical background, which also covers non-unary alphabets in $\oplus$-NFAs.

1.1 Motivation

The usefulness of traditional NFAs ($\cup$-NFAs) for modelling many different problems in Computer Science is indisputable. Their main advantage lies in the fact that, for practical applications, they offer a natural modelling mechanism for sequentiality, and they typically do so with a smaller number of states than their equivalent DFAs. Therefore, $\cup$-NFAs are highly applicable to, for example, problems in pattern recognition in strings. Problems that are inherently dependent on cycles (such as random number generation, encryption/decryption, and perfect hashing) are less efficient to model with NFAs.

Symmetric difference NFAs ($\oplus$-NFAs), on the other hand, are ideally suited to model cyclic applications, as unary $\oplus$-NFAs are equivalent to linear feedback shift registers (LFSRs). Or, from an algebraic point of view, a unary $\oplus$-NFA is a linear machine over the Galois field GF(2). Also noteworthy is that $\oplus$-NFAs perform better than $\cup$-NFAs as far as their state complexity is concerned. Champarnaud [1] showed that, given an arbitrary n-state $\oplus$-NFA, there is a 50% probability that its equivalent minimal DFA has $O(2^n)$ states. In contrast, this probability is much lower for $\cup$-NFAs (see Section 3).

1.2 History

A $\oplus$-NFA is, theoretically speaking, simply a multiplicity automaton over the field with two elements [11]. The particular idiosyncracies of these automata were first investigated in detail in [16], as an example of generalized nondeterminism. Under generalized nondeterminism, when the subset construction is applied to an NFA to obtain its equivalent DFA, any binary associative commutative set operation is allowed in the subset construction. In the case of traditional NFAs, the operation is union and we therefore refer to traditional NFAs as $\cup$-NFAs. Similarly, one could consider intersection, symmetric difference, XNOR, and so on, as the set operation, to

obtain ∩-NFAs, ⊕-NFAs, and XNOR-NFAs, respectively. ⊕-NFAs and ∩-NFAs were investigated extensively in [16]. The main theme in that work was the descriptional complexity offered by generalized nondeterminism.

Since any ⊕-NFA has an equivalent DFA, it follows that the class of ⊕-NFAs accepts the set of regular languages. Or, they are equal in power to ∪-NFAs as far as language recognition is concerned. The properties of interest for ⊕-NFAs therefore are those that differentiate them from ∪-NFAs. For example, for ⊕-NFAs it is "easy" (in terms of the number of states of the resulting NFA) to obtain an NFA for the complement or symmetric difference of regular languages, where for ∪-NFAs it is again easy to obtain the NFA for the union of two regular languages – as the names would suggest. The descriptional complexity of ⊕-NFAs, however, first raised interest in these machines (for a detailed discussion, see Section 3). Noteworthy is that unary n-state ⊕-NFAs can have equivalent minimal DFAs with $O(2^n)$ states – this bound is not reachable for unary ∪-NFAs. Moreover, it is possible to specify *how many* unary languages are succinctly representable by ⊕-NFAs, and more surprisingly, exactly *which* unary languages have succinct ⊕-NFA representations. In the binary case, ⊕-NFAs are almost always succinct. Also of importance in practical applications, is that the minimization and equivalence testing of ∪-NFAs are known to be exponential, whereas Vuillemin and Gama [19] showed the existence of cubic equivalence and minimization algorithms for ⊕-NFAs.

Some related issues based on symmetric difference were investigated since 1997. For example, Okhotin [10] investigated the use of the symmetric difference operator in language equations, and showed that symmetric difference does not change the expressive power in language equations. Other work was done on magic numbers for ⊕-NFAs [6], minimization of ⊕-NFAs [12, 17, 19], and the ambiguity of ⊕-NFAs [12, 13, 14, 18].

We start the chapter with basic definitions and mathematical background on ⊕-NFAs.

2 Definitions and Examples

We consider the classic automata-theoretic definition of ⊕-NFAs in this section. For the linear algebra approach, the reader is referred to the relevant chapter in this book.

Let 2^Q represent the power set of any finite set Q (that is, 2^Q is the set of all subsets of Q).

Definition 1 *A ⊕-NFA is a 5-tuple $M = (Q, \Sigma, \delta, Q_0, F)$, where Q is a finite set of states, Σ is a finite alphabet, δ is a transition function such that $\delta : Q \times \Sigma \to 2^Q$, $Q_0 \subseteq Q$ is a set of start states, and $F \subseteq Q$ is a set of final states.*

The transition function δ can be extended to sets of states, as follows:

$$\delta(A, a) = \bigoplus_{q \in A} \delta(q, a) \tag{1}$$

for any $a \in \Sigma$ and $A \in 2^Q$.

We can also extend δ from single alphabet symbols to strings, in the usual manner:

$$\delta(A, \epsilon) = A$$

and

$$\delta(A, aw) = \delta(\delta(A, a), w)$$

for any $a \in \Sigma$, $w \in \Sigma^*$ and $A \in 2^Q$.

Symmetric difference is in essence a *parity* operation, like XOR in Boolean logic. Hence, a ⊕-NFA accepts a word $w \in \Sigma^*$ if the number of possible nondeterministic acceptance paths are odd. That is, w is accepted if $|F \cap \delta(q_0, w)| \mod 2 = 1$.

Equation (1) allows the construction of the DFA equivalent to a given ⊕-NFA. The classic proof of Hopcroft and Ullman to show that there is a DFA equivalent to (accepting the same language as) any given ∪-NFA, can be trivially adapted to the case of ⊕-NFAs (see [16] for more

detail). Hence, for any $\oplus$-NFA there is an equivalent DFA, and it follows that the $\oplus$-NFAs can only recognize the regular languages.

In Example 1 below, we highlight the difference between $\oplus$-NFAs and $\cup$-NFAs.

Example 1 *Let M be an NFA defined by*

$$M = (\{q_0, q_1, q_2, q_3\}, \{a, b\}, \delta, \{q_0\}, \{q_1, q_3\})$$

with δ given by

δ	a	b
q_0	$\{q_0, q_1, q_3\}$	$\{q_3\}$
q_1	$\{q_1, q_2\}$	$\{q_3\}$
q_2	$\{q_0, q_2\}$	$\{q_3\}$
q_3	$\emptyset$	$\{q_3\}$

.

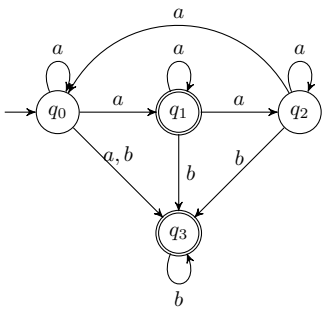

Figure 1: The NFA M for Example 1

Suppose first that M is to be considered as a $\cup$-NFA. Then the DFA M' equivalent to M can be constructed by the subset construction, to yield (see Figure 2):

δ	a	b
$[q_0]$	$[q_0, q_1, q_3]$	$[q_3]$
$[q_3]$	$\emptyset$	$[q_3]$
$[q_0, q_1, q_3]$	$[q_0, q_1, q_2, q_3]$	$[q_3]$
$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_2, q_3]$	$[q_3]$

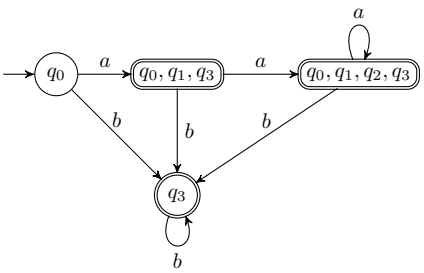

Figure 2: The DFA M' for M as a $\cup$-NFA

On the other hand, if M is considered as a $\oplus$-NFA, the DFA M'' equivalent to M would be given by:

δ	a	b
$[q_0]$	$[q_0, q_1, q_3]$	$[q_3]$
$[q_3]$	$\emptyset$	$[q_3]$
$[q_0, q_1, q_3]$	$[q_0, q_2, q_3]$	$[q_3]$
$[q_0, q_2, q_3]$	$[q_1, q_2, q_3]$	$[q_3]$
$[q_1, q_2, q_3]$	$[q_0, q_1]$	$[q_3]$
$[q_0, q_1]$	$[q_0, q_2, q_3]$	$\emptyset$

and graphically

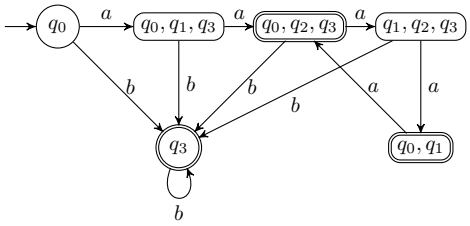

Figure 3: The DFA M'' for M as a $\oplus$-NFA

It is easy to see that M' and M'' accept different languages – for example, the word a is in $L(M')$, but not in $L(M'')$. Also, note the difference in final state calculation, where the DFA state $[q_0, q_1, q_3]$ is a final state in M', but not in M'' (because there is an even number of final states from M in $[q_0, q_1, q_3]$).

□

The reader may note that, whereas it is quite easy to analyse the behaviour of $\cup$-NFAs based on their graphical representation, this is not the case with $\oplus$-NFAs. Hence, other approaches are necessary. We first consider the case of unary $\oplus$-NFAs (that is, $\oplus$-NFAs with alphabet size one). The chapter on the linear algebra approach shows how to generalize the approach to more general alphabets.

2.1 The unary case

For any $\oplus$-NFA with a single alphabet symbol, one can consider the $\oplus$-NFA simply as a linear machine over the Galois field GF(2) [16] (in GF(2), the addition of an even number of 1's equals zero, similar to the XOR operation in Boolean logic). That is, such an $\oplus$-NFA is equivalent to a linear feedback shift register (LFSR), and the analysis of the behaviour of LFSRs are well understood [3]. If one encodes the transition matrix of a given unary $\oplus$-NFA as a matrix $\mathbf{A}$ such that

$$a_{ij} = \begin{cases} 1 & \text{if } q_i \in \delta(q_j, a) \\ 0 & \text{otherwise,} \end{cases}$$

the multiplication of these matrices over GF(2) mimics the behaviour of the $\oplus$-NFA as it reads a word. Note the similarity to the path through a graph by means of multiplying its adjency matrix.

If we encode the transition matrix of a unary $\oplus$-NFA M as above, the resulting matrix in GF(2) is known as the *characteristic* matrix $\mathbf{A}$ of M. It is then possible to find the so-called *characteristic polynomial* $c(x)$ of M, as $c(x) = \det(\mathbf{A} - \mathbf{I}x)$, where $\mathbf{I}$ is the identity matrix. The properties of $c(x)$ then determine the behaviour of M. We list a number of such properties below. For proof or more details, the reader is referred to either [3] or [16]:

- If $\mathbf{A}$ is non-singular and $c(x)$ is primitive and irreducible, then M' has $2^n - 1$ reachable states in a single cycle of length $2^n - 1$.

- If $\mathbf{A}$ is non-singular and $c(x)$ is irreducible but not primitive, then M' has a cycle of length b, where b is a factor of $2^n - 1$. No other cycle lengths are possible.
- If $\mathbf{A}$ is singular, then M' has a cycle length of b, or b transient states which ends in a cycle of length one. Here b is either less than 2^{n-1}, or can be constructed as $\text{lcm}(2^{n_1} - 1, 2^{n_2} - 1, \ldots, 2^{n_j} - 1)$, where $n = n_1 + n_2 + \ldots + n_j$.
- If $\mathbf{A}$ is non-singular and $c(x)$ is reducible, then M' has a number of states already occuring in one of the above cases.

Example 2 illustrates these concepts.

Example 2 *Consider the unary $\oplus$-NFA $M = (\{q_1, q_2, q_3\}, \{a\}, \delta, \{q_1\}, \{q_3\})$ with δ given by*

δ	a
q_1	$\{q_2\}$
q_2	$\{q_3\}$
q_3	$\{q_1, q_3\}$.

Then $\mathbf{A}$ is given by

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}.$$

Calculating $c(x)$ using standard algebra leads to $c(x) = x^3 - x^2 - 1$. Note that $c(x)$ is a primitive irreducible polynomial in $GF(2)$. If we encode the start state as a column vector $y(0)$, and compute $\mathbf{A}^k y(0)$, we end up with the k-th entry in the on-the-fly subset construction on M. For example, with the start state q_1 encoded as $y(0) = [1\ 0\ 0]^T$, we see that $\mathbf{A}^4$ is given by

$$\mathbf{A}^4 = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix},$$

and hence $\mathbf{A}^4 y(0)$ is given by $[1\ 1\ 1]$. This corresponds to the state $[q_1, q_2, q_3]$, which is reached after four applications of the subset construction on M. Similarly, $\mathbf{A}^6 y(0)$ is given by $[0\ 1\ 1]$, which corresponds to $[q_2, q_3]$.

□

3 Descriptional Complexity

3.1 Succinctness

The early work in $\oplus$-NFAs centered on the descriptional complexity advantages offered by these machines. The descriptional complexity of unary $\oplus$-NFAs were analysed in detail in [16], and we summarise those results below. The descriptional complexity of binary $\oplus$-NFAs were investigated only from an experimental point of view in [16]. Champarnaud [1] made some progress towards a theoretical analysis in the binary case, but a solid algebraic foundation for the analyis of the descriptional complexity of $\oplus$-NFAs with multiple alphabet symbols was only presented by Vuillemin [19] in 2009, and subsequently in [13].

Any n-state NFA is said to be succinct if its equivalent minimal DFA has $O(2^n)$ states. The upper and lower bounds on the number of states for $\cup$-NFAs are known to be $O(2^n)$ and $O(n)$ in the binary case, and $f(n)$ and $O(n)$ in the unary case. Here $f(n)$ is Landau's function, and is given by

$$f(n) = \max\{\text{lcm}(r_1, \ldots, r_k) |\ r_1 + \ldots + r_k = n\}.$$

Finding a good approximation for $f(n)$ is known as Landau's problem [8]; Chrobak [2] uses the approximation

$$f(n) = e^{\sqrt{n \log n}}$$

as the succinctness upper bound for unary $\cup$-nfas.

The lower and upper bound for $\oplus$-NFAs is $O(n)$ and $O(2^n)$ in both the unary and binary cases. The notable difference between $\cup$-NFAs and $\oplus$-NFAs therefore occurs in the upper bound for the unary case. To show that this bound is reachable, is easy. Since unary $\oplus$-NFAs correspond to linear machines over GF(2), one can consider the characteristic polynomial $c(x)$ of a given n-state unary $\oplus$-NFA M. If $c(x)$ is primitive and irreducible, it is known that $c(x)$ gives rise to a linear machine with a cycle length of $2^n - 1$ exactly. In other words, the DFA equivalent to M will be a cycle of length $2^n - 1$, containing all possible subsets of states of M excluding the empty state. Theorem 1 below then shows that the DFA must be minimal.

Theorem 1 *Any n-state unary $\oplus$-NFA M with primitive irreducible characteristic polynomial, has an equivalent minimal DFA M' with exactly $2^n - 1$ states.*

Proof: For any state q in M', let the *state size* of q denote the number of states of M used to label q in M'.

Label each state in M' with a 1 if it is a final state (that is, if it contains an odd number of final states of M), and label it with 0 if it is not a final state. Since M' forms a cycle (of length $2^n - 1$), this cycle of zeroes and ones forms a primitive word w iff M' is minimal. Now, M' has exactly $s = \binom{n}{k}$ states of size k, and each of these states may be final or not, depending on whether it contains an odd number of final states from M, or not. Therefore, M' always has 2^{n-1} final states (by simply considering the possible number of groupings of states of M in M' with an odd number of final states from M). Since there is always exactly one more final state than the total number of non-final states, it is not possible to write the word w as $w = (w_0 \ldots w_t)^m$ for any values of t and m, and hence M' is always minimal.

□

Given the result above, we can now quantify *the number of* unary regular languages that are succinctly representable by $\oplus$-NFAs.

Theorem 2 *There are at most $n\varphi(2^n-1)$ unary regular languages that are succinctly representable by n-state $\oplus$-NFAs, where $\varphi(m) = m(1 - 1/p_1)(1 - 1/p_2)\ldots(1 - 1/p_k)$ is the Euler function indicating the number of positive integers less than m which are relatively prime to n.*

Proof: The number of primitive polynomials of order n in GF(2) is given by $\frac{1}{n}\ \varphi(2^n - 1)$ [3]. If $m = p_1^{a_1} p_2^{a_2} \ldots p_k^{a_k}$ is the prime power factorization of m, where the a_i are positive integers and the p_i are distinct primes, then $\varphi(m)$ can be calculated as $\varphi(m) = m(1-1/p_1)(1-1/p_2)\ldots(1-1/p_k)$. Each such polynomial corresponds to exactly one n-state $\oplus$-NFA M with one start state and one final state. There are n possible start state sets, and n possible final state sets for each such $\oplus$-NFA, so that there are $n\varphi(2^n-1)$ possible $\oplus$-NFAs. From Theorem 1 above, all the corresponding DFAs are minimal. However, some of the languages represented by these DFAs may be equivalent, and hence there are at most $n\varphi(2^n - 1)$ unary regular languages that are succinctly representable by n-state $\oplus$-NFAs.

□

The reader may note that there are **no** infinite unary regular languages that can be succinctly represented by $\cup$-NFAs, while every unary $\oplus$-NFA with a primitive characteristic polynomial succinctly represents a unary regular language.

We can now give the exact form of the unary languages that are succinctly represented by n-state unary $\oplus$-NFAs.

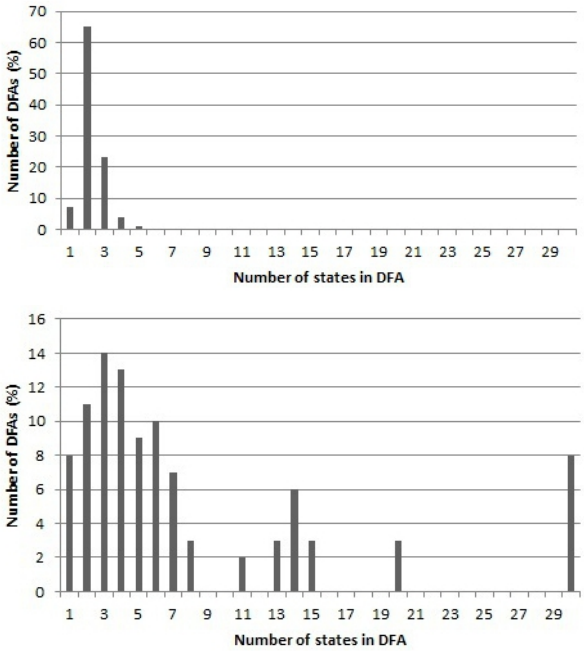

Figure 4: The percentage of minimal DFAs with k states from 1 million randomly generated 5-state unary $\cup$-NFAs (top) and 1 million randomly generated 5-state unary $\oplus$-NFAs (bottom).

Theorem 3 *Any succinct n-state $\oplus$-NFA accepts a unary regular language of the form*

$$\mathcal{L} = \bigcup_{k \in A} \{a^{i(2^n-1)+k} | i \geq 0\},$$

for some set $A \subseteq \{0, 1, 2, \ldots, 2^n - 2\}$.

Proof: Take M to be an n-state unary $\oplus$-NFA with a primitive characteristic polynomial. Then the DFA M' equivalent to M has a cycle of length $2^n - 1$, and accepts a language of the form $\mathcal{L}$ given above.

□

For a comparison between the behaviour of unary $\cup$-NFAs and unary $\oplus$-NFAs as far as succinctness is concerned, some experimental analysis was performed in [16]. For each experiment, a large number of randomly generated n-state $\cup$-NFAs and $\oplus$-NFAs were created, converted to DFAs, and minimized. Only one copy of homomorphic DFAs was retained. The number of states in each of the DFAs was determined, and a count was made of the number of DFAs with m states, where $1 \leq m \leq 2^n - 1$. These results are an indication of *how often* an n-state $\cup$-NFA or $\oplus$-NFA is succinct. Figure 4 and Figure 5 give a typical result for unary and binary NFAs, respectively. Of particular interest in the binary case, is that many more $\oplus$-NFAs tend to be succinct than is the case for $\cup$-NFAs.

From a practical point of view, it becomes interesting to know whether any DFA can have a succinct description as a $\oplus$-NFA. Iwama [6] considered *magic numbers* for which it will not be possible to find succinct n-state $\cup$-NFAs. In other words, is there a magic number α with $n < \alpha < 2^n$, such that no DFA with α states can be simulated by an NFA with n states? Jiraskova [7] showed that no such magic numbers exist for $\cup$-NFAs for a sufficiently large alphabet size. In the unary case, it follows from Chrobak's bound [2] that $e^{\sqrt{n \log n}}$ is indeed magic.

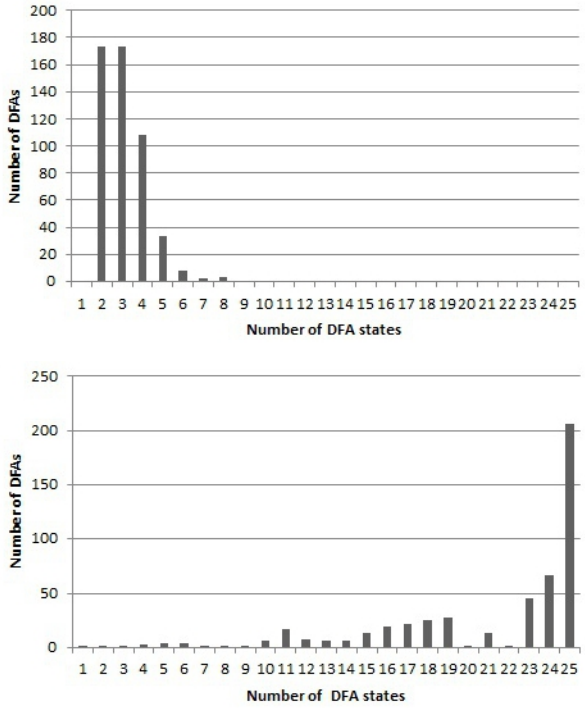

Figure 5: The number of minimal DFAs with k states from 500 randomly generated 5-state binary $\cup$-NFAs (top) and 500 randomly generated 5-state binary $\oplus$-NFAs (bottom).

However, the graph in Figure 4 seems to indicate that many magic numbers exist for unary $\oplus$-NFAs. In [15] we considered non-magic numbers for $\oplus$-NFAs, and in [16] we related magic numbers for unary $\oplus$-NFAs back to the order of the roots of the characteristic polynomial in the multiplicative group of GF(2^n).

In summary, there are two main known differences between the descriptional complexity of unary $\cup$-NFAs and unary $\oplus$-NFAs. The first is the fact that unary $\oplus$-NFAs has the tight upper bound of $O(2^n)$, while unary $\cup$-NFAs cannot reach this bound. Secondly, there are many known magic numbers for which no succinct $\oplus$-NFAs exist.

4 Ambiguity

In any nondeterministic automaton, a measure of its nondeterminism is of interest. For $\cup$-NFAs, various such measures have been defined [5]. One is the *ambiguity* of a given $\cup$-NFA [20]. That is, how many different acceptance paths exist for a word w in the language of the $\cup$-NFA?

A $\cup$-NFA is said to be k-ambiguous if there are at most k different paths accepting any word in its language. If $k = 1$, then the $\cup$-NFA is unambiguous. If k is a constant, then the $\cup$-NFA is finitely ambiguous. The $\cup$-NFA is polynomially ambiguous if the number of accepting paths is polynomially bound in the length of any input word, and exponentially ambiguous if the number of accepting paths is exponentially bound in the length of any input word.

Ambiguity results for $\oplus$-NFAs were investigated in [13, 18], and we discuss those results in this section.

Ambiguity for $\oplus$-NFAs is defined exactly as for $\cup$-NFAs; that is, each ambiguity class represents that at most $f(n)$ paths are needed to accept any word of length n in the language accepted by the $\oplus$-NFA. It is known [18] that it is possible to construct examples of $\oplus$-NFAs that exhibit ambiguity behaviour for each ambiguity class (see Example 3).

To visually illustrate acceptance paths, one can construct the execution tree of a given NFA [4]. The root of the tree is labelled with the start node of the NFA, and all the possible resulting states on reading the first letter of the input word is on level 1 of the tree. So, the states on level i represent all the states that can be reached after having read the first i letters of the input word. Note that, for $\cup$-NFAs, the union of the nodes on level i represent the corresponding node in the equivalent DFA. For $\oplus$-NFAs, the *symmetric difference* of the nodes on a given level represent the corresponding node in the equivalent DFA. Therefore, for a $\oplus$-NFA, all occurrences of a given node label on a specific level in the execution tree, are cancelled out if there is an even number of occurrences of that node label on that level. On the other hand, if there is an odd number of occurrences, then none cancel out on that level.

Example 3 *The family M_n of unary $\oplus$-NFAs given in Figure 6 is unambiguous. It is easy to see that the final state q_{n-1} can only occur either once or twice on every path in the execution tree (see Figure 7, which shows the execution tree of M_3 on input word aaaaaaa).*

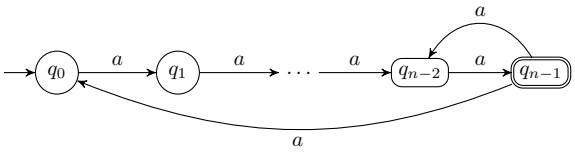

Figure 6: A family M_n of unambiguous unary $\oplus$-NFAs

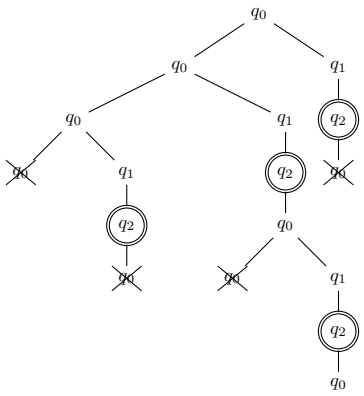

Figure 7: Execution tree of M_3 on string *aaaaaaa*.

Similarly, the same family M_n of unary $\oplus$-NFAs in Figure 6 is finitely ambiguous if M_n has an odd number $k > 1$ of final states.

The interested reader may note that M_n is succinct if its characteristic polynomial $c(x) = x^n + x^{n-1} + 1$ is primitive for a given n.

Figure 8 shows a family of polynomially ambiguous $\oplus$-NFAs [18].

Finally, let M_n^e be a $\oplus$-NFA with start state set $\{q_0\}$ and final state set $\{q_0\}$, and transition function δ given by

$$\delta(q_i, a) = \begin{cases} \{q_0, q_1, q_2, \ldots, q_{n-1}\}, & \text{if } i = 0, \\ \{\overline{q_i}\}, & \text{otherwise.} \end{cases}$$

It was shown in [18] that the family $\{M_n^e\}_{n\geq 3}$ of $\oplus$-NFAs is exponentially ambiguous. Figure 9 illustrates M_3^e.

□

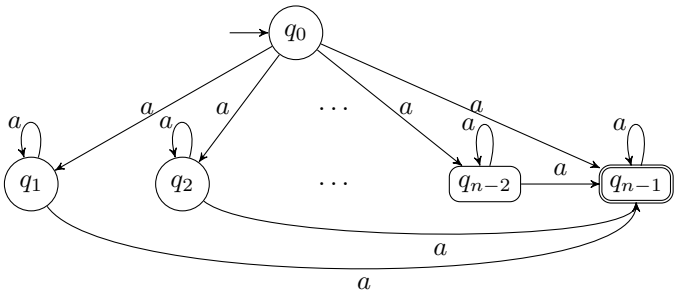

Figure 8: $\oplus$-NFA M_n^p.

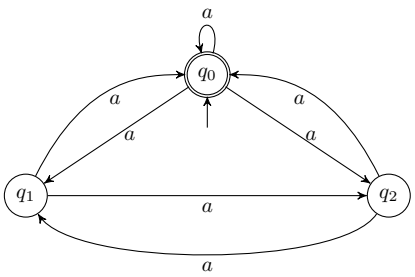

Figure 9: $\oplus$-NFA M_3^e.

The aim in [18] was to consider the descriptional complexity of ambiguous $\oplus$-NFAs, and hence some questions remain open. For example, regular languages that can be represented by $\oplus$-NFAs that *strictly* belong to a given ambiguity class have not been demonstrated yet.

Another question regarding ambiguity, is whether it would be expensive in terms of the number of states, to remove the ambiguity. For $\cup$-NFAs, Okhotin showed that in the unary case, there is an exponential trade-off in converting unary $\cup$-NFAs to unambiguous unary $\cup$-NFAs [9]. That is, there are unary languages that can be represented by n-state $\cup$-NFAs, but the smallest unambiguous unary $\cup$-NFA that accepts the language has $g(n) + n^2$ states, where $g(n)$ is Landau's function [2].

The same question for $\oplus$-NFAs has a far more general result. In [13], it was shown that for every unary n-state $\oplus$-NFA, there is an equivalent finitely ambiguous $\oplus$-NFA with exactly n states. Therefore, every unary $\oplus$-NFA can be rewritten in an equivalent finitely ambiguous form, without a state explosion as in the case of unary $\cup$-NFAs. This linearity result follows from the theorem that, for any unary $\oplus$-NFA, it is possible to give an equivalent unary $\oplus$-NFA with only one start state (resp. final state) by changing the basis of the transition matrix:

Theorem 4 *Let $M = (Q, \{a\}, \delta, Q_0, F)$ be a unary n-state $\oplus$-NFA that accepts a non-empty language L. Then for any non-empty subset of states $X \subseteq Q$ there exists unary $\oplus$-NFAs M' and M'', both accepting L, such that:*

1. *$M' = (Q, \{a\}, \delta', X, F')$, and*
2. *$M'' = (Q, \{a\}, \delta'', Q_0'', X)$.*

Corollary 1 *There are no unary languages which require polynomially or exponentially ambiguous $\oplus$-NFAs.*

It is interesting to note that, even for complementation, the unary $\oplus$-NFAs show better conversion results than the unary $\cup$-NFA. For unambiguous unary n-state $\cup$-NFAs [9], the state complexity of complementation lies between the bounds $\frac{1}{42}n\sqrt{n}$ (for $n > 867$) and $e^{\Theta}(\sqrt[3]{n \ln^2 n})$. For unambiguous unary n-state $\oplus$-NFAs, we have that (see [14]):

Theorem 5 *Let M be a unary $\oplus$-NFA accepting the language L. Then there is an $n+1$-state unary $\oplus$-NFA M' such that M' accepts $\overline{L}$.*

and hence

Corollary 2 *Assume the language L is accepted by an n-state unary finitely ambiguous $\oplus$-NFA M. Then there is an $n+1$-state finitely ambiguous unary $\oplus$-NFA M' that accepts $\overline{L}$.*

Even when severely restricting the amount of nondeterminism, there are interesting differences between $\cup$-NFAs and $\oplus$-NFAs. Suppose that the only nondeterminism allowed is that there may be multiple initial states – such machines are known as k-DFAs (in the union case) or k-$\oplus$-DFAs. It is easy to see that both k-DFAs and k-$\oplus$-DFAs are at most finitely ambiguous [13]. However, it has been shown [14] that there exists a family $\{M_n\}_{n\geq 2}$ of unary k-DFAs that are finitely ambiguous when considered as k-DFAs, but are unambiguous when considered as k-$\oplus$-DFAs (see Figure 10).

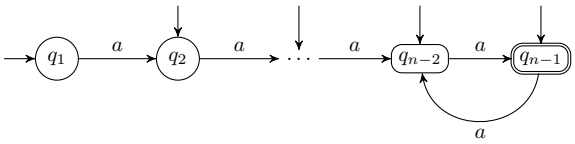

Figure 10: A family of unambiguous k-$\oplus$-DFAs

Moreover, there is a polynomial time algorithm to determine whether a given unary k-$\oplus$-DFA is unambiguous [14].

5 Future Work

Unary $\oplus$-NFAs have been investigated in some detail, but many issues remain outstanding. A mathematical framework for $\oplus$-NFAs with larger alphabet size has recently been established, but not much work has been done on these machines.

We list some of the topics that could be investigated with regards to $\oplus$-NFAs: two-way $\oplus$-NFAs, a generalization of $\oplus$-NFAs to accept infinite words (a Büchi-$\oplus$-NFA), the application of $\oplus$-NFAs in Angluin learning, and many more.

References

[1] J.-M. Champarnaud, G. Hansel, T. Paranthon, and D. Ziadi. Random generation models for NFAs. *Journal of Automata, Languages and Combinatorics*, 9(2):203216, 2004.

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

[3] L. Dornhoff and F. Hohn. *Applied Modern Algebra*. Macmillan Publishing Company, 1978.

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

[5] J. Hromkovic, J. Karhumäki, H. Klauck, G. Schnitger, and S. Seibert. Measures of nondeterminism in finite automata. In *Proceedings of the 27th International Colloquium on Automata, Languages and Programming*, pages 199–210. Springer, 2000.

[6] K. Iwama, A. Matsuura, and M. Paterson. A family of NFAs which need $2^n - \alpha$ deterministic states. *Theoretical Computer Science*, 301:451–462, 2003.

[7] G. Jiraskova. Note on minimal finite automata. In J. Sgall, A. Pultr, and P. Kolman, editors, *Proceedings of the 26th International Symposium on the Mathematical Foundations of Computer Science*, volume 2136 of *Lecture Notes in Computer Science*, pages 421–431. Springer, 2001.

[8] E. Landau. Ueber die Maximalordung der Permutationen gegebenen Grades. *Archiv der Mathematic und Physik*, 3:92–103, 1903.

[9] A. Okhotin. Unambiguous finite automata over a unary alphabet. In P. Hlinený and A. Kucera, editors, *Proceedings of the 35th International Symposium on the Mathematical Foundations of Computer Science*, volume 6281 of *Lecture Notes in Computer Science*, pages 556–567. Springer, August 2010.

[10] A. Okhotin. Language equations with symmetric difference. *Fundamenta Informatika*, 116(1-4):205–222, 2012.

[11] M.P. Schützenberger. On the definition of a family of automata. *Information and Control*, 4(2):245–270, September 1961.

[12] A.B. van der Merwe, H. Tamm, and L. van Zijl. Minimal DFA for symmetric difference NFA. In *Proceedings of 14th International Workshop on Descriptional Complexity of Formal Systems (DCFS)*, volume 7386 of *Lecture Notes in Computer Science*, pages 307–318. Springer, July 2012.

[13] A.B. van der Merwe, L. van Zijl, and J. Geldenhuys. Ambiguity of unary symmetric difference NFAs. In *Proceedings of the International Colloquium on Theoretical Aspects of Computing*, volume 6916 of *Lecture Notes in Computer Science*, pages 256–266. Springer, September 2011.

[14] A.B. van der Merwe, L. van Zijl, and J. Geldenhuys. Ambiguity and structural ambiguity of symmetric difference NFAs. *Theoretical Computer Science*, 2013. Accepted, to appear.

[15] L. van Zijl. Magic numbers for symmetric difference NFAs. *Theoretical Computer Science*, 88:325–349, 1991.

[16] L. van Zijl. *Generalized Nondeterminism and the Succinct Representation of Regular Languages*. PhD thesis, Stellenbosch University, November 1997.

[17] L. van Zijl, J. Daciuk, and G. Müller. Minimization of unary symmetric difference NFAs. *South African Computer Journal*, pages 64–74, July 2005.

[18] L. van Zijl and J. Geldenhuys. Descriptional complexity of ambiguity in symmetric difference NFAs. *Journal of Universal Computer Science*, 17(6):874–890, 2011.

[19] J. Vuillemin and N. Gama. Compact normal form for regular languages as Xor automata. In *Proceedings of the 14th International Conference on Implementation and Application of Automata*, CIAA '09, pages 24–33, Berlin, Heidelberg, 2009. Springer.

[20] A. Weber and H. Seidl. On the degree of ambiguity of finite automata. *Theoretical Computer Science*, 88:325–349, 1991.