Boolean Automata and Atoms of Regular Languages

Hellis Tamm ✉
Department of Software Science, Tallinn University of Technology, Estonia

Abstract

We examine the role that atoms of regular languages play in boolean automata. We observe that the size of a minimal boolean automaton of a regular language is directly related to the number of atoms of the language. We present a method to construct minimal boolean automata, using the atoms of a given regular language. The "illegal" cover problem of the Kameda-Weiner method for NFA minimization implies that using the union operation only to construct an automaton from a cover – as is the case with NFAs –, is not sufficient. We show that by using the union and the intersection operations (without the complementation operation), it is possible to construct boolean automata accepting a given language, for a given maximal cover.

2012 ACM Subject Classification Theory of computation → Formal languages and automata theory

Keywords and phrases Boolean automaton, Regular language, Atoms

Digital Object Identifier 10.4230/LIPIcs.MFCS.2021.86

Funding This work was supported by the Estonian Ministry of Education and Research grant IUT33-13 and by the Estonian Research Council grant PRG1210.

Acknowledgements The author is indebted to late Janusz Brzozowski for suggesting the topic and for collaboration during the early stages of this work.

1 Introduction

Nondeterministic finite automata (NFAs) were introduced by Rabin and Scott [19] in 1959, and since then this model of computation has been extended in many ways. Notably, in 1976, Kozen [14] introduced a model of parallel computation based on a generalization of nondeterminism which was later developed into the notion of alternating finite automaton (AFA) by Chandra, Kozen, and Stockmeyer [7]. Independently from this work, Brzozowski and Leiss [5] introduced an equivalent concept of a boolean finite automaton (BFA), using a different notation. Often, the notions of AFA and BFA are used interchangeably. We use the terminology and notation of [5].

The transition function of a BFA uses boolean combinations of its states, generalizing the notion of NFAs which can be interpreted as using unions of states as transition targets. However, the expressive powers of BFAs and NFAs are the same, that is, BFAs only accept regular languages [5, 7]. Importantly, BFAs can succinctly represent regular languages: for any $n \geqslant 1$ there exists a BFA with n states such that the minimal deterministic finite automaton (DFA) of the same language has $2^{(2^n)}$ states [7, 16]. Also, it is known that any regular language is accepted by an n-state boolean automaton if and only if its reverse language is accepted by a DFA with at most 2^n states [15, 16]. However, there are languages such that their boolean, nondeterministic, and deterministic complexities coincide [17].

Atoms [6] of a regular language L can be considered as its building blocks, since any quotient of L, including L itself, is a disjoint union of atoms. Recently, several old results of automata theory have been revisited using atoms of regular languages: Brzozowski's double-reversal method for minimizing a DFA [4] was generalized in [6], the Kameda-Weiner

method of finding a minimal NFA [13] was reinterpreted in terms of atoms and generalized in [21], and lower bound methods for the size of an NFA [3, 10, 11] were presented in terms of quotients and atoms of regular languages in [22].

We examine the role that atoms play in boolean automata. We show that the size of a minimal BFA of a regular language is directly related to the number of atoms of that language. More exactly, we observe that if a language L has m atoms, then a minimal BFA of L has $\lceil \log_2 m \rceil$ states. Furthermore, we show how to construct a minimal BFA using the atoms of a given language. We note that constructions of a minimal BFA presented in the literature [16, 15, 9] have been using a DFA of the reverse language. We think that using atoms to build BFAs is more natural and goes well with the narrative of considering atoms as building blocks of a language. Our method of constructing a minimal BFA using the atoms of a language ensures that the resulting BFA is atomic; that is, the languages associated with the states of a BFA are unions of atoms. Consequently, every regular language has an atomic minimal BFA; however, we also show that a minimal BFA is not necessarily atomic. For comparison, not every language has an atomic minimal NFA [6]. Symmetric difference NFAs – a subclass of boolean automata that use only the symmetric difference operation in the transition function – are known to be able to succinctly represent some regular languages [24]. Interestingly, every minimal symmetric difference NFA is atomic [23].

We revisit the Kameda-Weiner method of NFA minimization [13] which constructs NFAs from grid covers of a special matrix. However, not every cover yields an NFA that would accept a given language. The problem of "illegal" covers of the Kameda-Weiner matrix has been of interest for decades [13, 11, 21, 22]. We show that one can construct a BFA for a given language, using any cover of the Kameda-Weiner matrix. One can see this result as a solution to the problem of interpreting grid covers of the Kameda-Weiner matrix in terms of finite automata accepting a given language. The "illegal" cover problem implies that using the union operation only to construct such an automaton – as is the case with NFAs –, is not sufficient. We show that by using the union and the intersection operations (without the complementation operation), it is possible to construct boolean automata accepting a given language, for a given maximal cover. We note that by a result in [9], for any BFA of n states, there is an equivalent BFA with $2n$ states that uses the union and the intersection operations only. However, in certain cases, our method can produce such a BFA with less states.

We mention that learning regular languages via AFAs has been studied in [1, 2].

We also note that recently, symbolic versions of AFAs and BFAs have been introduced [8, 20], and it has been claimed that in the symbolic setting, these two automata models become importantly different [20].

2 Automata, Languages, and Equations

A *boolean finite automaton (BFA)* is a quintuple $\mathcal{B} = (Q, \Sigma, \delta, f_0, F)$ where $Q = \{q_0, \ldots, q_{n-1}\}$ is a finite, non-empty set of *states*, Σ is a finite non-empty *alphabet*, $\delta : Q \times \Sigma \to B_Q$ is the *transition function*, where B_Q is the free boolean algebra generated by Q, $f_0 \in B_Q$ is the *initial function* in B_Q, and $F \subseteq Q$ is the set of *final states*. We denote the empty word by ε. The transition function is extended to the function $\delta : B_Q \times \Sigma^* \to B_Q$ as follows. For every $q_i \in Q$, $a \in \Sigma$, $w \in \Sigma^*$, and $f \in B_Q$, $\delta(q_i, \varepsilon) = q_i$, and $\delta(q_i, aw) = f_{i,a}(\delta(q_0, w), \ldots, \delta(q_{n-1}, w))$, where $f_{i,a}(q_0, \ldots, q_{n-1}) = \delta(q_i, a)$, and $\delta(f, w) = f(\delta(q_0, w), \ldots, \delta(q_{n-1}, w))$. Let $\varphi : Q \to \{0, 1\}$ be defined by setting $\varphi(q_i) = 1$ if $q_i \in F$, and $\varphi(q_i) = 0$ otherwise, for $q_i \in Q$. The *language accepted* by a BFA $\mathcal{B}$ is $L(\mathcal{B}) = \{w \in \Sigma^* \mid \delta(f_0, w)(\varphi(q_0), \ldots, \varphi(q_{n-1})) = 1\}$. Two boolean automata are *equivalent* if they accept the same language. The *right language* of a state q of $\mathcal{B}$ is $L(\mathcal{B}_q)$, where $\mathcal{B}_q = (Q, \Sigma, \delta, q, F)$.

If the functions f_0 and $\delta(q,a)$ are unions of states for every $q \in Q$ and $a \in \Sigma$, then $\mathcal{B}$ is a *nondeterministic finite automaton (NFA)*. Traditionally, for NFAs these functions have been presented as sets of states. We prefer to use the traditional notation of an NFA $\mathcal{N} = (Q, \Sigma, \delta, I, F)$, where Q, Σ, and F are as in BFA, $I \subseteq Q$ is the set of *initial states*, and $\delta : Q \times \Sigma \to 2^Q$ is the *transition function*. The *reverse* of an NFA $\mathcal{N} = (Q, \Sigma, \delta, I, F)$ is the NFA $\mathcal{N}^R = (Q, \Sigma, \delta^R, F, I)$, where $q \in \delta^R(p,a)$ if and only if $p \in \delta(q,a)$ for $p, q \in Q$ and $a \in \Sigma$.

If $f_0 \in Q$ and $\delta(q,a) \in Q$ for every $q \in Q$ and $a \in \Sigma$, then $\mathcal{B}$ is a *deterministic finite automaton (DFA)*. The *boolean* (*nondeterministic*, *deterministic*, respectively) *complexity* of a regular language L, denoted by $bc(L)$ ($nc(L)$, $dc(L)$, respectively) is the minimal number of states of a boolean (nondeterministic, deterministic, respectively) automaton of L.

A *boolean system of equations (BSE)* B with variables $L_0, \ldots, L_{n-1}$ is a set of language equations

$$L_i = \bigcup_{a \in \Sigma} aF_{i,a}(L_0, \ldots, L_{n-1}) \cup L_i^{\varepsilon}, \quad i = 0, \ldots, n-1, \tag{1}$$

where $F_{i,a}$ is a boolean function of the variables $L_0, \ldots, L_{n-1}$, $L_i^{\varepsilon} = \{\varepsilon\}$ if $\varepsilon \in L_i$, and $L_i^{\varepsilon} = \emptyset$ otherwise, together with the initial function $F_0(L_0, \ldots, L_{n-1})$. The language defined by a BSE B is $L(B) = F_0(L_0, \ldots, L_{n-1})$.

Any BSE defines a BFA and *vice versa*. There is a one-one correspondence between the state set $Q = \{q_0, \ldots, q_{n-1}\}$ of a BFA $\mathcal{B}$ and the set of language variables $\{L_0, \ldots, L_{n-1}\}$ of the corresponding BSE B; there is a transition from $q_i \in Q$ with $a \in \Sigma$ to a boolean function $f_{i,a}$ in B_Q if and only if $F_{i,a}$ is the corresponding function of variables $\{L_0, \ldots, L_{n-1}\}$ where the *disjunction*($\vee$), *conjunction*($\wedge$), and *negation*($\neg$) operations are replaced by the set operations *union*($\cup$), *intersection*($\cap$), and *complement*($^-$), respectively, and the constants 0 and 1 are replaced by $\emptyset$ and Σ^*, respectively, and a similar correspondence is between the initial functions f_0 and F_0. Also, any state q_i of $\mathcal{B}$ is final if and only if $L_i^{\varepsilon} = \{\varepsilon\}$. In the rest of the paper, we treat boolean automata and their corresponding systems of equations interchangeably.

The *left quotient*, or simply *quotient*, of a language L by a word $w \in \Sigma^*$ is the language $w^{-1}L = \{x \in \Sigma^* \mid wx \in L\}$. It is well known that the left quotients of L are the right languages of the states of the minimal DFA of L.

An *atom* of a regular language L with quotients $K_0, \ldots, K_{n-1}$ is any non-empty language of the form $\widetilde{K_0} \cap \cdots \cap \widetilde{K_{n-1}}$, where $\widetilde{K_i}$ is either K_i or $\overline{K_i}$, and $\overline{K_i}$ is the complement of K_i with respect to Σ^* [6]. An atom is *initial* if it has L (rather than $\overline{L}$) as a term; it is *final* if it contains ε. There is exactly one final atom, the atom $\widehat{K_0} \cap \cdots \cap \widehat{K_{n-1}}$, where $\widehat{K_i} = K_i$ if $\varepsilon \in K_i$, and $\widehat{K_i} = \overline{K_i}$ otherwise. If $\overline{K_0} \cap \cdots \cap \overline{K_{n-1}}$ is an atom, then it is called the *negative* atom, all the other atoms are *positive*. Thus atoms of L are pairwise disjoint languages uniquely determined by L; they define a partition of Σ^*. Every quotient K_i (including L) is a (possibly empty) union of atoms. Hence, atoms can be considered as building blocks of regular languages. We also note that atoms of L are the classes of the *left congruence* ${}_L{\equiv}$ of L defined as follows: for $x, y \in \Sigma^*$, $x\,{}_L{\equiv}\,y$ if for every $u \in \Sigma^*$, $ux \in L$ if and only if $uy \in L$ [12].

Let $A = \{A_0, \ldots, A_{m-1}\}$ be the set of atoms of L, let I_A be the set of initial atoms, and let A_{m-1} be the final atom. The *átomaton* of L is the NFA $\mathcal{A} = (A, \Sigma, \alpha, I_A, \{A_{m-1}\})$ where $A_j \in \alpha(A_i, a)$ if and only if $A_j \subseteq a^{-1}A_i$, for all $A_i, A_j \in A$ and $a \in \Sigma$. It was shown in [6] that the atoms of L are the right languages of the states of the átomaton, and that the reverse NFA of the átomaton is the minimal DFA of the reverse language L^R of L.

The BSE corresponding to the átomaton $\mathcal{A}$, also called the *atom equations*, is as follows:

$$A_i = \bigcup_{a\in\Sigma} a\Big(\bigcup_{A_j\subseteq a^{-1}A_i} A_j\Big) \cup A_i^{\varepsilon}, \quad i = 0,\ldots,m-1, \tag{2}$$

where $A_i^{\varepsilon} = \emptyset$ for $i = 0,\ldots,m-2$, and $A_{m-1}^{\varepsilon} = \{\varepsilon\}$, with the initial function $L = \bigcup_{A_i\in I_A} A_i$.

A BFA $\mathcal{B}$ is *atomic* if the right languages of its states are unions of atoms of $L(\mathcal{B})$.

3 Boolean Atoms

Let L be a regular language over Σ and let $\mathcal{B}$ be a BFA of L with variables $L_0,\ldots,L_{n-1}$.

A *boolean atom* of $\mathcal{B}$ is any non-empty language $\widetilde{L_0}\cap\cdots\cap\widetilde{L_{n-1}}$, where $\widetilde{L_i}$ is either L_i or its complement $\overline{L_i}$ with respect to Σ^*. Similarly to the atoms of L, boolean atoms of $\mathcal{B}$ are pairwise disjoint, defining a partition of Σ^*.

We study the relationship between boolean atoms of $\mathcal{B}$ and atoms of L. To avoid confusion between these two notions, we also call the atoms of L the *language atoms*.

▶ **Proposition 1.** *Every atom of L is a union of boolean atoms of $\mathcal{B}$.*

Proof. Every quotient of L (including L itself) is obtained as a boolean combination of some L_i's. Hence, every quotient of L can be expressed as a union of intersections involving uncomplemented and complemented variables from $\{L_0,\ldots,L_{n-1}\}$. By adding in the missing variables, every quotient can be expressed as a union of boolean atoms. Also, the complement of any quotient is a union of boolean atoms. Since an intersection of unions of boolean atoms is a union of boolean atoms, the proposition holds. ◀

▶ **Corollary 2.** *Every boolean atom of $\mathcal{B}$ is a subset of some atom of L.*

Proof. Since both the boolean atoms of $\mathcal{B}$ and the atoms of L define a partition of Σ^*, the corollary follows from Proposition 1. ◀

▶ **Proposition 3.** *A BFA $\mathcal{B}$ is atomic if and only if its boolean atoms are equal to the atoms of L.*

Proof. First assume that $\mathcal{B}$ is atomic. Then every language L_i is a union of some atoms of L, and so is its complement $\overline{L_i}$. Therefore, any boolean atom $B_i = \widetilde{L_0}\cap\cdots\cap\widetilde{L_{n-1}}$ is an intersection of unions of language atoms, which is a union of language atoms. On the other hand, by Corollary 2, B_i is a subset of some language atom. We conclude that B_i is equal to some atom of L.

Conversely, if the boolean atoms of $\mathcal{B}$ are equal to the atoms of L, then since every L_i is a union of boolean atoms, it is as well a union of language atoms. Hence, $\mathcal{B}$ is atomic. ◀

We note that boolean atoms are a generalization of *partial atoms* of NFAs, introduced in [6]. More exactly, given an NFA with its language equations, its partial atoms are the boolean atoms of the corresponding BFA.

4 Constructing Minimal Boolean Automata Using Atoms

It is known that if a regular language L is accepted by an n-state BFA, then the reverse language L^R is accepted by a DFA with at most 2^n states [7, 16]. We also recall the following theorem by Leiss [16]:

▶ **Theorem 4.** *Let $\mathcal{D}$ be a DFA with m states. There exists a BFA with $\lceil \log_2 m \rceil$ states which accepts the reverse language of $\mathcal{D}$.*

Consequently, a minimal BFA of a language L has $\lceil \log_2 m \rceil$ states, where m is the number of states of the minimal DFA of L^R. Since the átomaton of L is isomorphic to the reverse automaton of the minimal DFA of L^R [6], we make the following observation:

▶ **Theorem 5.** *A minimal BFA of a regular language L has $\lceil \log_2 m \rceil$ states, where m is the number of atoms of L.*

We note that Leiss [16] also describes a method to construct a BFA of L^R with $\lceil \log_2 m \rceil$ states by using a DFA of L with m states. Also, Kozen [15] (p. 327) discusses how to construct an AFA of L with k states by using a DFA of L^R with 2^k states, and vice versa.

We present a method to construct a minimal BFA of a regular language, by using its atoms.

Let L be a regular language over an alphabet Σ, and let $A = \{A_0, \ldots, A_{m-1}\}$ be the set of atoms of L, with a subset $I_A \subseteq A$ of initial atoms and the final atom A_{m-1}. Let $k = \lceil \log_2 m \rceil$. We show how to construct an atomic BFA with k variables $L_0, \ldots, L_{k-1}$ denoting some (not yet identified) languages over Σ. Let us consider the set S of all intersections in the form $\widetilde{L_0} \cap \ldots \cap \widetilde{L_{k-1}}$ where $\widetilde{L_i}$ is either L_i or $\overline{L_i}$. Clearly, there are 2^k such intersections, and the union of all these intersections is Σ^*. Also, we note that since $k = \lceil \log_2 m \rceil$, the inequality $m \leqslant 2^k$ holds.

Let us denote any intersection in S by $X_P = \bigcap_{i \in P} L_i \cap \bigcap_{i \in \overline{P}} \overline{L_i}$, where $P \subseteq \{0, \ldots, k-1\}$ and $\overline{P} = \{0, \ldots, k-1\} \setminus P$. Now, let us choose any subset $S_m = \{X_{P_0}, \ldots, X_{P_{m-1}}\}$ of S consisting of m intersections, and set any $X_{P_j} \in S_m$ to be equal to some atom A_j of L. We note that S_m is the set of boolean atoms of the BFA we will be constructing, and by Proposition 3, this BFA will be atomic. For instance, we may choose $P_0 = \{0, \ldots, k-1\}$, $P_1 = \{0, \ldots, k-2\}$, $P_2 = \{0, \ldots, k-3, k-1\}$, $P_3 = \{0, \ldots, k-3\}$, etc., and form the following equations between the boolean atoms and the atoms of L:

$$\begin{aligned}
L_0 \cap L_1 \cap \ldots \cap L_{k-2} \cap L_{k-1} &= A_0, \\
L_0 \cap L_1 \cap \ldots \cap L_{k-2} \cap \overline{L_{k-1}} &= A_1, \\
L_0 \cap L_1 \cap \ldots \cap \overline{L_{k-2}} \cap L_{k-1} &= A_2, \\
L_0 \cap L_1 \cap \ldots \cap \overline{L_{k-2}} \cap \overline{L_{k-1}} &= A_3, \\
&\ldots \\
\widetilde{L_0} \cap \widetilde{L_1} \cap \ldots \cap \widetilde{L_{k-2}} \cap \widetilde{L_{k-1}} &= A_{m-1},
\end{aligned}$$

where $\widetilde{L_i}$ is L_i if $n_i = 0$, and $\widetilde{L_i}$ is $\overline{L_i}$ if $n_i = 1$, where $n_0 n_1 \ldots n_{k-1}$ is the binary representation of the number $m-1$ using k bits. For every $X_{P_j} \notin S_m$, that is, for $j = m, \ldots, 2^k - 1$, we let $X_{P_j} = \emptyset$.

Clearly, every language L_i, where $i = 0, \ldots, k-1$, is the union of those boolean atoms X_{P_j}, where L_i is uncomplemented, that is, $L_i = \bigcup_{i \in P_j} X_{P_j}$.

We derive boolean equations for $L_0, \ldots, L_{k-1}$, using the equations above together with the atom equations (2):

$$L_i = \bigcup_{i \in P_j} X_{P_j} = \bigcup_{i \in P_j} A_j = \bigcup_{i \in P_j} \Big(\bigcup_{a \in \Sigma} a\Big(\bigcup_{A_h \subseteq a^{-1} A_j} A_h \Big) \cup A_j^{\varepsilon} \Big) =$$
$$\bigcup_{a \in \Sigma} a\Big(\bigcup_{i \in P_j} \bigcup_{A_h \subseteq a^{-1} A_j} X_{P_h} \Big) \cup \bigcup_{i \in P_j} A_j^{\varepsilon},$$

that is, we obtain the equations

$$L_i = \bigcup_{a\in\Sigma} a\Big(\bigcup_{i\in P_j}\ \bigcup_{A_h\subseteq a^{-1}A_j} X_{P_h}\Big) \cup L_i^{\varepsilon}, \quad i = 0,\ldots,k-1, \tag{3}$$

where $L_i^{\varepsilon} = \{\varepsilon\}$ if $i \in P_{m-1}$, and $L_i^{\varepsilon} = \emptyset$ otherwise.

We also notice that $L = \bigcup_{A_h\subseteq L} A_h = \bigcup_{A_h\subseteq L} X_{P_h}$. That is, since L is a union of some of its atoms A_h, it is the union of the corresponding boolean atoms X_{P_h}. Hence, the equations (3) together with the initial function $L = \bigcup_{A_h\subseteq L} X_{P_h}$ form an atomic minimal BFA of L.

In case our goal would be to obtain an atomic minimal BFA with the initial function $L = L_0$, we would have to be able to assign all the initial atoms of L to boolean atoms which have L_0 (rather than $\overline{L_0}$) as a term, and all the other language atoms to boolean atoms with $\overline{L_0}$. It is not difficult to see that this is possible if and only if the condition

$$m - 2^{\lceil \log_2 m\rceil - 1} \leqslant |I_A| \leqslant 2^{\lceil \log_2 m\rceil - 1} \tag{4}$$

holds.

The method described above constructs an atomic minimal BFA of a language. However, there may exist non-atomic minimal BFAs as well. The following example illustrates the constructions of minimal BFAs and presents a case of a non-atomic minimal BFA.

▶ **Example 6.** Let us consider a regular language L from [6, 18], defined by the following nondeterministic system of equations, with $L = L_0$:

$$\begin{aligned}
L_0 &= aL_1 \cup b(L_1 \cup L_2),\\
L_1 &= aL_3 \cup b(L_0 \cup L_3),\\
L_2 &= a(L_0 \cup L_2 \cup L_3) \cup \varepsilon,\\
L_3 &= aL_3 \cup bL_1.
\end{aligned}$$

It was shown in [6] that the corresponding NFA is a minimal NFA of L, however, it is not atomic. The language L has 6 atoms, denoted by A, B, C, D, E, and F, from which B, D, and F are the initial atoms, and A is the final atom. The atom equations are as follows:

$$\begin{aligned}
A &= a(A \cup B) \cup \varepsilon,\\
B &= aC \cup bA,\\
C &= b(B \cup D),\\
D &= bC,\\
E &= aD,\\
F &= a(E \cup F) \cup b(E \cup F),
\end{aligned}$$

with the initial function $L = B \cup D \cup F$. One can verify that $L_0 = B \cup D \cup F$, $L_1 = C \cup E \cup F$, and $L_3 = D \cup E \cup F$, whereas $L_2 = A \cup E \cup F'$, where $F' = a(E \cup F)$ is a subset of F.

The boolean atoms of the original system of equations are found as the following non-empty intersections:

$$\begin{aligned}
L_0 \cap L_1 \cap L_2 \cap L_3 &= F',\\
L_0 \cap L_1 \cap \overline{L_2} \cap L_3 &= F \setminus F',\\
L_0 \cap \overline{L_1} \cap \overline{L_2} \cap L_3 &= D,\\
L_0 \cap \overline{L_1} \cap \overline{L_2} \cap \overline{L_3} &= B,\\
\overline{L_0} \cap L_1 \cap L_2 \cap L_3 &= E,\\
\overline{L_0} \cap L_1 \cap \overline{L_2} \cap \overline{L_3} &= C,\\
\overline{L_0} \cap \overline{L_1} \cap L_2 \cap \overline{L_3} &= A.
\end{aligned}$$

Hence, the boolean atoms are A, B, C, D, E, F', and $F \setminus F'$. We note that F is a union of two boolean atoms, while the other language atoms coincide with boolean atoms.

In the following, we drop the union symbols when representing unions of atoms; for example, $B \cup D \cup F$ is denoted by BDF. A minimal BFA of this language has $\lceil \log_2 6 \rceil = 3$ states. We construct an atomic minimal BFA by the approach described above. Let us denote the states of a minimal BFA by variables L_0, L_1, L_2, and form the following equations:

$$\begin{aligned}
L_0 \cap L_1 \cap L_2 &= A,\\
L_0 \cap L_1 \cap \overline{L_2} &= B,\\
L_0 \cap \overline{L_1} \cap L_2 &= C,\\
L_0 \cap \overline{L_1} \cap \overline{L_2} &= D,\\
\overline{L_0} \cap L_1 \cap L_2 &= E,\\
\overline{L_0} \cap L_1 \cap \overline{L_2} &= F,\\
\overline{L_0} \cap \overline{L_1} \cap L_2 &= \emptyset,\\
\overline{L_0} \cap \overline{L_1} \cap \overline{L_2} &= \emptyset.
\end{aligned}$$

From these equations we obtain $L_0 = ABCD$, $L_1 = ABEF$, and $L_2 = ACE$. All the variables correspond to final states because the corresponding languages contain the final atom A. Using the atom equations, we get the following equations:

$$\begin{aligned}
ABCD &= aABC \cup bABCD \cup \varepsilon,\\
ABEF &= aABCDEF \cup bAEF \cup \varepsilon,\\
ACE &= aABD \cup bBD \cup \varepsilon,
\end{aligned}$$

with $L = BDF$. By replacing atoms with the corresponding boolean expressions over the variables L_0, L_1, and L_2, and after a few straightforward simplifications, we obtain the following system of equations, with $L = \overline{L_2}$:

$$\begin{aligned}
L_0 &= a(L_0 \cap (L_1 \cup L_2)) \cup bL_0 \cup \varepsilon,\\
L_1 &= a\Sigma^* \cup b((L_1 \cap L_2) \cup \overline{L_0}) \cup \varepsilon,\\
L_2 &= a((L_0 \cap L_1) \cup (\overline{L_1} \cap \overline{L_2})) \cup b(L_0 \cap \overline{L_2}) \cup \varepsilon.
\end{aligned}$$

These equations form an atomic minimal BFA for L.

We may also obtain a minimal BFA for L with $L = L_0$, since the condition (4) holds for L. For instance, we could have the atom assignments as follows (with the rest of the intersections set to be empty):

$$\begin{aligned}
L_0 \cap L_1 \cap L_2 &= B,\\
L_0 \cap L_1 \cap \overline{L_2} &= D,\\
L_0 \cap \overline{L_1} \cap L_2 &= F,\\
\overline{L_0} \cap L_1 \cap L_2 &= A,\\
\overline{L_0} \cap L_1 \cap \overline{L_2} &= C,\\
\overline{L_0} \cap \overline{L_1} \cap L_2 &= E.
\end{aligned}$$

Now we obtain the languages $L_0 = BDF$, $L_1 = ABCD$, and $L_2 = ABEF$, where L_1 and L_2 correspond to the final states. Using the atom equations, we get the following equations:

$$\begin{aligned}
BDF &= aCEF \cup bACEF,\\
ABCD &= aABC \cup bABCD \cup \varepsilon,\\
ABEF &= aABCDEF \cup bAEF \cup \varepsilon,
\end{aligned}$$

with $L = BDF$. Similarly as above, we obtain the following language equations, with $L = L_0$:

$$\begin{aligned}
L_0 &= a((\overline{L_0} \cap \overline{L_2}) \cup (\overline{L_1} \cap L_2)) \cup b(\overline{L_0} \cup (L_0 \cap \overline{L_1})),\\
L_1 &= a((L_1 \cap L_2) \cup (\overline{L_0} \cap \overline{L_2})) \cup bL_1 \cup \varepsilon,\\
L_2 &= a\Sigma^* \cup b((\overline{L_0} \cap L_2) \cup (L_0 \cap \overline{L_1})) \cup \varepsilon.
\end{aligned}$$

These equations form another atomic minimal BFA of L.

We can also construct a non-atomic minimal BFA for L. Let us take $L_0 = BDF$, $L_1 = CEF$, and $L_2 = ADEF'$, where L_2 is a final state. We note that L_2 is not atomic. Using the atom equations and the equation for F', we get the following equations:

$$\begin{aligned} BDF &= aCEF \cup bACEF, \\ CEF &= aDEF \cup bBDEF, \\ ADEF' &= aABDEF \cup bC \cup \varepsilon, \end{aligned}$$

with $L = BDF$. We obtain the following language equations, with $L = L_0$:

$$\begin{aligned} L_0 &= aL_1 \cup b((\overline{L_0} \cap \overline{L_1} \cap L_2) \cup L_1), \\ L_1 &= a((L_0 \cap L_1) \cup (L_0 \cap L_2) \cup (L_1 \cap L_2)) \cup b(L_0 \cup (L_1 \cap L_2)), \\ L_2 &= a(L_0 \cup L_2) \cup b(\overline{L_0} \cap L_1 \cap \overline{L_2}) \cup \varepsilon. \end{aligned}$$

These equations form a non-atomic minimal BFA of L. ⌟

5 Boolean Automata and Quotient-Atom Matrix

Kameda and Weiner [13] studied the problem of finding a minimal NFA of a language L, using a matrix based on the states of the minimal DFAs of L and L^R. They suggested a method of constructing NFAs, utilizing grid covers of this matrix. However, an NFA formed this way does not necessarily accept L. Therefore, the method of finding a minimal NFA tests grid covers of the matrix in the order of increasing size to see if they are "legal". The first legal NFA, that is, an NFA found to accept L, is a minimal one.

In this section, we revisit the Kameda-Weiner method and its recent reinterpretation in terms of atoms of the language [21]. We will show that if one aims to form a BFA rather than an NFA, then the problem of "illegal" covers disappears. That is, one can form a BFA for L, using any cover of the quotient-atom matrix.

First, we describe the Kameda-Weiner method in terms of quotients and atoms as presented in [21]. It was shown in [21] that the matrix used by Kameda and Weiner can be viewed as the *quotient-atom matrix* of the language, that is, the matrix with its rows corresponding to the non-empty quotients, and the columns, to the positive atoms of the language. Any (i, j)-entry of this matrix is 1 if the quotient K_i has the atom A_j as its subset, and 0 otherwise. A *grid* of the matrix is the direct product $g = P \times R$ of a set P of quotients with a set R of atoms, such that every atom in R is a subset of every quotient in P. A grid $g = P \times R$ is *maximal* if there is no other grid $g' = P' \times R'$ such that $P \subseteq P'$ and $R \subseteq R'$. A *grid cover* of the matrix is a set $G = \{g_0, \ldots, g_{k-1}\}$ of grids, such that every 1-entry (K_i, A_j) belongs to some grid g_h of G. A grid cover is maximal if it consists of maximal grids. Clearly, any grid cover can be made maximal by replacing every non-maximal grid $g = P \times R$ in it by the maximal grid $g' = P' \times R'$ such that $P \subseteq P'$ and $R \subseteq R'$.

Let f_G be the function that assigns to every non-empty quotient K_i the subset of G, consisting of grids $g = P \times R$ such that $K_i \in P$. The Kameda-Weiner method constructs the NFA $\mathcal{N}_G = (G, \Sigma, \eta_G, I_G, F_G)$, where G is a maximal grid cover, $I_G = f_G(K_0)$, $g \in F_G$ if and only if $g \in f_G(K_i)$ implies that $\varepsilon \in K_i$, and transition function is computed by the so-called *intersection rule* $\eta_G(g, a) = \bigcap_{K_i \in P} f_G(a^{-1}K_i)$, for a grid $g = P \times R$ in G and $a \in \Sigma$. The NFA $\mathcal{N}_G$ may or may not accept L. A cover G is called "legal" if $L(\mathcal{N}_G) = L$. To find a minimal NFA of a language L, covers of the matrix are tested in the order of increasing size to see if they are legal; the first legal NFA is a minimal one.

In [21], the Kameda-Weiner method was interpreted in terms of atoms, with the key observation being that any maximal grid can be seen as the set of atoms it involves. We call a set $\{L_0, \ldots, L_{k-1}\}$ of languages a *language cover* for L if for every L_i there is a quotient

K_j such that $L_i \subseteq K_j$ and if every quotient of L is a union of some L_i's. A language cover $\{L_0, \ldots, L_{k-1}\}$ is *atomic* if every L_i, where $i = 0, \ldots, k-1$, is a union of atoms of L. Now, for any grid cover $G = \{g_0, \ldots, g_{k-1}\}$ there is a corresponding atomic language cover $C = \{U(R_0), \ldots, U(R_{k-1})\}$, where $g_i = P_i \times R_i$ and $U(R_i) = \bigcup_{A_j \in R_i} A_j$ for $i = 0, \ldots, k-1$. The following theorem was proved in [21]:

▶ **Theorem 7.** *Let $G = \{g_0, \ldots, g_{k-1}\}$ be a cover consisting of maximal grids $g_i = P_i \times R_i$, $i = 0, \ldots, k-1$, and let $\mathcal{N}_G = (G, \Sigma, \eta_G, I_G, F_G)$ be the corresponding NFA, obtained by the intersection method. It holds that $g_i \in I_G$ if and only if $U(R_i) \subseteq L$, and $g_i \in F_G$ if and only if $\varepsilon \in U(R_i)$. For any $g_i, g_j \in G$ and $a \in \Sigma$, $g_j \in \eta_G(g_i, a)$ if and only if the inclusion $U(R_j) \subseteq a^{-1}U(R_i)$ holds.*

By Theorem 7, it is easy to see that forming the NFA $\mathcal{N}_G$ corresponds to defining this NFA with variables $L_0, \ldots, L_{k-1}$ using the language equations

$$L_i = \bigcup_{a \in \Sigma} a\Big(\bigcup_{U(R_j) \subseteq a^{-1}U(R_i)} L_j \Big) \cup L_i^{\varepsilon}, \quad i = 0, \ldots, k-1, \tag{5}$$

where $L_i^{\varepsilon} = \{\varepsilon\}$ if $\varepsilon \in U(R_i)$, and $L_i^{\varepsilon} = \emptyset$ otherwise, and the initial function $\bigcup_{U(R_i) \subseteq L} L_i$. We recall that this NFA not necessarily accepts L.

Gruber and Holzer [11] defined a canonical bipartite graph G_L of a language, which is a graph representation of the Kameda-Weiner matrix. By denoting the bipartite dimension of G_L, that is, the minimum number of bicliques to cover the edges of G_L, by $d(G_L)$, it is clear that the size of the minimal grid cover of the quotient-atom matrix is equal to $d(G_L)$. It has been a long-standing problem to identify languages with a legal minimal cover [11, 13, 21, 22].

It is known that $dc(L) \leqslant 2^{d(G_L)}$ [11]. Since $dc(L^R) = m$, where m is the number of atoms of L [6], and $d(G_{L^R}) = d(G_L)$ [11, 13], we get that $m \leqslant 2^{d(G_L)}$. By Theorem 5, a minimal BFA of L has $bc(L) = \lceil \log_2 m \rceil$ states, implying that $bc(L) \leqslant d(G_L)$. Hence, the following statement holds:

▶ **Theorem 8.** *A minimal BFA of a regular language L has at most $d(G_L)$ states.*

Theorem 8 provides an upper bound to the size of a minimal BFA of a language in terms of the size of a minimal cover of the quotient-atom matrix/graph of the language. Since a minimal NFA has at least $d(G_L)$ states [11, 13, 21], the inequalities

$$bc(L) \leqslant d(G_L) \leqslant nc(L)$$

hold. Also, we can state the following corollary:

▶ **Corollary 9.** *A minimal NFA is a minimal BFA of L if and only if the equalities $bc(L) = d(G_L) = nc(L)$ hold.*

In the following sections, we present two methods to construct a BFA, using a cover of the quotient-atom matrix/graph.

5.1 Constructing a General BFA

Given a language cover $C = \{L_0, \ldots, L_{k-1}\}$ of L, we call any non-empty intersection $\widetilde{L_0} \cap \ldots \cap \widetilde{L_{k-1}}$ where $\widetilde{L_i}$ is either L_i or $\overline{L_i}$, a *cover atom* of C. Similarly to the atoms of L, cover atoms are pairwise disjoint and their union is Σ^*.

The following proposition is a slightly modified version of the result in [6][1]:

[1] The result in [6] concerns the case where the language cover consists of the right languages of an NFA accepting L; however, its proof holds for any cover.

▶ **Proposition 10.** *Given a language cover C of L, every cover atom of C is a subset of some atom of L.*

We note that cover atoms of the cover $K = \{K_0, \ldots, K_{n-1}\}$ consisting of the quotients of L, are clearly the atoms of L. Moreover, the following proposition holds:

▶ **Proposition 11.** *The cover atoms of a cover C of L are equal to the atoms of L if and only if C is atomic.*

Proof. Let $C = \{L_0, \ldots, L_{k-1}\}$ be an atomic language cover of L, meaning that every $L_i \in C$ is a union of atoms of L. In this case every complement language $\overline{L_i}$, for $i = 0, \ldots, k-1$, is also a union of atoms. Therefore, any cover atom as an intersection of unions of atoms, is a union of atoms. However, since by Proposition 10, any cover atom is a subset of some atom of L, we conclude that cover atoms are equal to the atoms of L.

If the cover C is not atomic, then there is some $L_i \in C$ that is not a union of atoms of L. However, since L_i is the union of those cover atoms that have L_i uncomplemented in their intersection, there exists at least one non-atomic cover atom of C. ◀

Let us now consider any grid cover $G = \{g_0, \ldots, g_{k-1}\}$ of the quotient-atom matrix and the corresponding language cover $C = \{U(R_0), \ldots, U(R_{k-1})\}$ of L, with $g_i = P_i \times R_i$ and $U(R_i) = \bigcup_{A_j \in R_i} A_j$ for $i = 0, \ldots, k-1$. Let the set of cover atoms of C be $X = \{X_0, \ldots, X_{\ell-1}\}$. Since the cover C is atomic, by Proposition 11, the cover atoms of C are the atoms of L. Hence, any atom A_h of L can be expressed as $A_h = \bigcap_{A_h \in R_i} U(R_i) \cap \bigcap_{A_h \notin R_i} \overline{U(R_i)}$.

Now, we can form a BFA with the set $\{L_0, \ldots, L_{k-1}\}$ of variables, using the correspondence between the variables L_i and the languages $U(R_i)$, where $i = 0, \ldots, k-1$. We obtain the following equations for a BFA, using the atom expressions above and the atom equations (2):

$$L_i = \bigcup_{a \in \Sigma} a\Big(\bigcup_{A_j \in R_i} \bigcup_{A_h \subseteq a^{-1}A_j} \Big(\bigcap_{A_h \in R_i} L_i \cap \bigcap_{A_h \notin R_i} \overline{L_i} \Big) \Big) \cup L_i^{\varepsilon}, \quad i = 0, \ldots, k-1, \tag{6}$$

where $L_i^{\varepsilon} = \{\varepsilon\}$ if $\varepsilon \in U(R_i)$, and $L_i^{\varepsilon} = \emptyset$ otherwise. The initial function of the BFA is $L = \bigcup_{U(R_i) \subseteq L} L_i$. Clearly, this BFA is atomic, because every L_i is a union of atoms. We also note that boolean atoms of this BFA are equal to the cover atoms.

5.2 Constructing a BFA Without Complementation

We show that any maximal grid cover can be used to construct a BFA that uses the union and the intersection operations only, without a need for the complementation operation. We use the following definition from [21]:

▶ **Definition 12.** *A set R of atoms is* maximal *if $R = \{A_j \mid A_j \subseteq \bigcap_{U(R) \subseteq K_i} K_i\}$, where $U(R) = \bigcup_{A_j \in R} A_j$ is the union of atoms in R.*

We observe that the intersection of any quotients of L corresponds to a maximal set of atoms:

▶ **Proposition 13.** *Given any set $P \subseteq K$ of quotients of L, the set $R = \{A_j \mid A_j \subseteq \bigcap_{K_h \in P} K_h\}$ of atoms in their intersection, is maximal.*

Proof. Let $P \subseteq K$ be any set of quotients of L and let $R = \{A_j \mid A_j \subseteq \bigcap_{K_h \in P} K_h\}$ be the set of atoms in their intersection. Since $U(R) = \bigcap_{K_h \in P} K_h = \bigcap_{U(R) \subseteq K_i} K_i$, we get that $R = \{A_j \mid A_j \subseteq \bigcap_{U(R) \subseteq K_i} K_i\}$. Thus, R is maximal. ◀

▶ **Proposition 14.** *If a set R of atoms is maximal, then the set $\{A_j \mid A_j \subseteq a^{-1}U(R)\}$, where $a \in \Sigma$, is also maximal.*

Proof. Let R be a maximal set of atoms and $a \in \Sigma$. Then $U(R) = \bigcap_{U(R)\subseteq K_i} K_i$ and we get $a^{-1}U(R) = a^{-1}\bigcap_{U(R)\subseteq K_i} K_i = \bigcap_{U(R)\subseteq K_i} a^{-1}K_i$. Since for any quotient K_i of L, $a^{-1}K_i$ is also a quotient of L, the language $a^{-1}U(R)$ is equal to an intersection of some quotients. By Proposition 13, we conclude that the set $\{A_j \mid A_j \subseteq a^{-1}U(R)\}$ is maximal. ◀

Let us consider a maximal grid cover $G = \{g_0, \ldots, g_{k-1}\}$ of the quotient-atom matrix and the corresponding language cover $C = \{U(R_0), \ldots, U(R_{k-1})\}$ of L, with $g_i = P_i \times R_i$ and $U(R_i) = \bigcup_{A_j \in R_i} A_j$ for $i = 0, \ldots, k-1$. It is known that any maximal grid involves a maximal set of atoms [21]. Hence, every R_i, where $i = 0, \ldots, k-1$, is a maximal set of atoms. We denote by $P_{i,a}$, where $a \in \Sigma$, the set $P_{i,a} = \{K_h \mid a^{-1}U(R_i) \subseteq K_h\}$ of those quotients of L that include $a^{-1}U(R_i)$ as a subset. By Proposition 14, the equality $a^{-1}U(R_i) = \bigcap_{K_h \in P_{i,a}} K_h$ holds. Also, we recall that every quotient K_h is a union of some elements of the cover C, that is, $K_h = \bigcup_{U(R_i)\subseteq K_h} U(R_i)$.

We construct a BFA with the set $\{L_0, \ldots, L_{k-1}\}$ of variables, using the correspondence between the variables L_i and the languages $U(R_i)$, where $i = 0, \ldots, k-1$. The language equations are as follows:

$$L_i = \bigcup_{a\in\Sigma} a\Big(\bigcap_{K_h \in P_{i,a}} \bigcup_{U(R_j)\subseteq K_h} L_j\Big) \cup L_i^{\varepsilon}, \quad i = 0, \ldots, k-1, \tag{7}$$

where $L_i^{\varepsilon} = \{\varepsilon\}$ if $\varepsilon \in U(R_i)$, and $L_i^{\varepsilon} = \emptyset$ otherwise. The initial function is $L = \bigcup_{U(R_i)\subseteq L} L_i$. This BFA is atomic, since every L_i is a union of atoms of L.

We note that the BFA corresponding to the equations (7) uses only the union and the intersection operations. That is, the complementation operation is not needed when we use a maximal cover as a basis for the states of a BFA.

One can see this result as a solution to the problem of interpreting grid covers of the Kameda-Weiner matrix, or equivalently, biclique edge covers of the quotient-atom graph, in terms of finite automata accepting a given language. The "illegal" cover problem mentioned above implies that using the union operation only to construct such an automaton – as is the case with NFAs –, is not sufficient. However, we showed that with the union and the intersection operations it is possible to construct boolean automata accepting a given language, for a given maximal cover.

We note that by a result in [9], for any BFA of n states, there is an equivalent BFA with $2n$ states that uses the union and the intersection operations only. However, since the inequality $bc(L) \leqslant d(G_L)$ holds for any language L, and because we can construct a BFA with $d(G_L)$ states, using the union and the intersection operations only, our method can produce such a BFA for any language L for which the inequality $d(G_L) < 2bc(L)$ holds, with less states.

6 Conclusions

We have started a study of the role that atoms of a regular language have in the context of boolean automata, their relationship with boolean atoms, and how they can be used to construct BFAs.

The problem of "illegal" covers of the Kameda-Weiner matrix used for NFA minimization has been of interest for a long time. We presented a new interpretation of the covers in terms of BFAs, so that every cover becomes "legal". We showed that it is sufficient to use the union and the intersection operations to construct a BFA for a given language, corresponding to a maximal cover of the matrix. Moreover, the resulting BFAs are atomic.

These are part of the evidence of the significant role that atoms play in the theory of automata.

References

1 Dana Angluin, Sarah Eisenstat, and Dana Fisman. Learning regular languages via alternating automata. In Qiang Yang and Michael J. Wooldridge, editors, *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25–31, 2015*, pages 3308–3314. AAAI Press, 2015.

2 Sebastian Berndt, Maciej Liskiewicz, Matthias Lutter, and Rüdiger Reischuk. Learning residual alternating automata. In Satinder P. Singh and Shaul Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4–9, 2017, San Francisco, California, USA*, pages 1749–1755. AAAI Press, 2017.

3 Jean-Camille Birget. Intersection and union of regular languages and state complexity. *Inf. Process. Lett.*, 43(4):185–190, 1992.

4 Janusz A. Brzozowski. Canonical regular expressions and minimal state graphs for definite events. In *Proc. Symp. on Mathematical Theory of Automata*, volume 12 of *MRI Symposia Series*, pages 529–561. Polytechnic Press, Polytechnic Institute of Brooklyn, N.Y., 1963.

5 Janusz A. Brzozowski and Ernst L. Leiss. On equations for regular languages, finite automata, and sequential networks. *Theor. Comput. Sci.*, 10:19–35, 1980.

6 Janusz A. Brzozowski and Hellis Tamm. Theory of átomata. *Theor. Comput. Sci.*, 539:13–27, 2014.

7 Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. Alternation. *J. ACM*, 28(1):114–133, 1981.

8 Loris D'Antoni, Zachary Kincaid, and Fang Wang. A symbolic decision procedure for symbolic alternating finite automata. In *Proceedings of the Thirty-Fourth Conference on the Mathematical Foundations of Programming Semantics, MFPS 2018, Dalhousie University, Halifax, Canada, June 6–9, 2018*, volume 341 of *Electronic Notes in Theoretical Computer Science*, pages 79–99. Elsevier, 2018.

9 Abdelaziz Fellah, Helmut Jürgensen, and Sheng Yu. Constructions for alternating finite automata. *Int. J. of Comput. Math.*, 35(1-4):117–132, 1990.

10 Ian Glaister and Jeffrey O. Shallit. A lower bound technique for the size of nondeterministic finite automata. *Inf. Process. Lett.*, 59(2):75–77, 1996.

11 Hermann Gruber and Markus Holzer. Finding lower bounds for nondeterministic state complexity is hard. In *Proc. of DLT 2006*, volume 4036 of *Lecture Notes in Computer Science*, pages 363–374. Springer, 2006.

12 Szabolcs Iván. Complexity of atoms, combinatorially. *Inf. Process. Lett.*, 116(5):356–360, 2016.

13 Tsunehiko Kameda and Peter Weiner. On the state minimization of nondeterministic finite automata. *IEEE Trans. Computers*, 19(7):617–627, 1970.

14 Dexter Kozen. On parallelism in Turing machines. In *17th Annual Symposium on Foundations of Computer Science, Houston, Texas, USA, 25–27 October 1976*, pages 89–97, 1976.

15 Dexter Kozen. *Theory of Computation*. Texts in Computer Science. Springer, 2006.

16 Ernst L. Leiss. Succinct representation of regular languages by boolean automata. *Theor. Comput. Sci.*, 13:323–330, 1981.

17 Ernst L. Leiss. Succinct representation of regular languages by boolean automata II. *Theor. Comput. Sci.*, 38:133–136, 1985.

18 Oliver Matz and Andreas Potthoff. Computing small finite nondeterministic automata. In U. H. Engberg, K. G. Larsen, and A. Skou, editors, *Proceedings of the Workshop on Tools and Algorithms for Construction and Analysis of Systems*, BRICS Note Series, pages 74–88. BRICS, 1995.

19 Michael O. Rabin and Dana S. Scott. Finite automata and their decision problems. *IBM J. Res. Dev.*, 3(2):114–125, 1959.

20 Caleb Stanford, Margus Veanes, and Nikolaj Bjørner. Symbolic boolean derivatives for efficiently solving extended regular expression constraints. In Stephen N. Freund and Eran Yahav, editors, *PLDI '21: 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation, Virtual Event, Canada, June 20–25, 20211*, pages 620–635. ACM, 2021.

21 Hellis Tamm. New interpretation and generalization of the Kameda-Weiner method. In *43rd Int. Colloq. on Automata, Languages, and Programming (ICALP 2016)*, volume 55 of *Leibniz Int. Proc. in Informatics (LIPIcs)*, pages 116:1–116:12, Dagstuhl, Germany, 2016. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

22 Hellis Tamm and Brink van der Merwe. Lower bound methods for the size of nondeterministic finite automata revisited. In *Language and Automata Theory and Applications – 11th International Conference, LATA 2017, Umeå, Sweden, March 6-9, 2017, Proceedings*, volume 10168 of *Lecture Notes in Computer Science*, pages 261–272, 2017.

23 Brink van der Merwe, Hellis Tamm, and Lynette van Zijl. Minimal DFA for symmetric difference NFA. In *Descriptional Complexity of Formal Systems – 14th International Workshop, DCFS 2012, Braga, Portugal, July 23-25, 2012. Proceedings*, volume 7386 of *Lecture Notes in Computer Science*, pages 307–318. Springer, 2012.

24 Lynette van Zijl. On binary $\oplus$-nfas and succinct descriptions of regular languages. *Theor. Comput. Sci.*, 328(1-2):161–170, 2004.