Languages of Words of Low Automatic Complexity Are Hard to Compute

Joey Chen
Department of Mathematics, National University of Singapore, Singapore

Bjørn Kjos-Hanssen
Department of Mathematics, University of Hawai'i at Mānoa, Honolulu, HI, USA

Ivan Koswara
School of Computing, National University of Singapore, Singapore

Linus Richter[1]
Department of Mathematics, National University of Singapore, Singapore

Frank Stephan
Department of Mathematics, National University of Singapore, Singapore
School of Computing, National University of Singapore, Singapore

Abstract

The automatic complexity of a finite word (string) is an analogue for finite automata of Sipser's distinguishing complexity (1983) and was introduced by Shallit and Wang (2001). For a finite alphabet Σ of at least two elements, we consider the non-deterministic automatic complexity given by *exactly* – yet not necessarily *uniquely* – accepting automata: a word $x \in \Sigma^*$ has exact non-deterministic automatic complexity $k \in \mathbb{N}$ if there exists a non-deterministic automaton of k states which accepts x while rejecting every other word of the same length as x, and no automaton of fewer states has this property. Importantly, and in contrast to the classical notion, the witnessing automaton may have multiple paths of computation accepting x. We denote this measure of complexity by A_{Ne}, and study a class of languages of low A_{Ne}-complexity defined as $L_q = \{ x \in \Sigma^* \colon A_{Ne}(x) < q|x| \}$, which is parameterised by rationals $q \in (0, 1/2)$ (generalising a class of sets first studied by Kjos-Hanssen). We show that for every $q \in (0, 1/2)$, this class is neither context-free nor recognisable by certain Boolean circuits. In the process, we answer an open question of Kjos-Hanssen quantifying the complexity of $L_{1/3}$ in terms of Boolean circuits, and also prove the Shannon effect for A_{Ne}.

2012 ACM Subject Classification Theory of computation → Grammars and context-free languages; Theory of computation → Regular languages

Keywords and phrases Automatic complexity, automata theory, formal languages, Boolean circuits, Shannon effect

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2025.24

Related Version *Full Version*: https://doi.org/10.48550/arXiv.2510.07696 [4]

Funding *Bjørn Kjos-Hanssen*: This work was partially supported by a grant from the Simons Foundation (#704836 to Bjørn Kjos-Hanssen).
Linus Richter: This work was fully supported by Singapore Ministry of Education grant MOE-000538-01.
Frank Stephan: This work was partially supported by Singapore Ministry of Education grant MOE-000538-01.

Acknowledgements Parts of this work have appeared in the first author's Bachelor's thesis submitted to the National University of Singapore.

[1] Corresponding author

1 Introduction

Automatic complexity is a notion of complexity of finite words (strings) determined by witnessing automata, first introduced by Shallit and Wang in [35] as a Turing computable alternative to Kolmogorov complexity. It is an analogue for finite automata of Sipser's distinguishing complexity [37]. Classically, the automatic complexity of a word x over a finite alphabet Σ refers to the cardinality – counted in number of states[2] – of the smallest deterministic finite automaton which accepts x and rejects every other word of the same length as x [35]. The notion as well as variations of it have proven interesting for multiple reasons. For instance, since automatic complexity is Turing computable, it can be used in the study of computational complexity: the computational complexity of sets of binary words of low automatic complexity has helped prove missing relationships in the Complexity Zoo [1] (see [20, Theorem 39] for an example). Further, the detailed investigation of words in terms of their automatic complexity [17, 16] has shed light on *computable* notions of randomness, which are unavailable from the viewpoint of Kolmogorov complexity [21, 40, 28].

In this paper, we study a weakening of a variation of automatic complexity due to Hyde [12], and show that it generates classes of words too complicated to be captured by pushdown automata, nor by certain classes of constant-depth Boolean circuits – both of which are notably computationally more powerful than finite automata. This provides further evidence towards the conjecture that automatic complexity is hard to compute (see e.g. [15]).

1.1 Technical Background

Fix a finite alphabet Σ of at least two elements. In usual Kleene notation, we denote by Σ^* the set of all finite words of elements from Σ. We denote the empty string by ε, and the set of non-empty words by $\Sigma^+ = \Sigma^* \setminus \{\varepsilon\}$. By an **automaton** we always mean a non-deterministic finite automaton, unless otherwise stated. We do not allow ε-transitions.

▶ **Definition 1.** *Let $x \in \Sigma^*$. An automaton M* **exactly accepts** *x if M accepts x, and whenever both $y \neq x$ and $|y| = |x|$ then M rejects y.*

The pumping lemma shows that this definition is maximally restrictive on the number of words accepted by the witnessing automaton; trying to strengthen the definition by asking for outright uniqueness of the accepted word only leads to trivialities.

▶ **Definition 2.** *The* **automatic complexity** *of $x \in \Sigma^*$ is given by*

$$A_D(x) = \min\{\, k \in \mathbb{N} \colon \text{there exists a DFA of } k \text{ states which exactly accepts } x \,\}.$$

For a reference on contemporary automatic complexity, see e.g. the recent [19]. The subscript D stands for "deterministic", indicating that $A_D(x)$ is determined by the smallest DFA. By definition, it is clear that A_D is well-defined, and even computable (for every $n \in \mathbb{N}$, there are only finitely many DFAs, and each can be simulated in finite time). However – similar to the unnatural properties of *plain* compared to *prefix-free* Kolmogorov complexity – the measure A_D has the following properties, which may render it undesirable as a natural measure of complexity of words. These were first described in [13]:

[2] A version of automatic complexity counting the number of *transitions* has been studied by Serna [32]; see also Shallit and Breitbart [34].

1. A_D is not invariant under natural transformations on strings, such as reversals. For instance, Hyde and Kjos-Hanssen have verified computationally that $A_D(011100) = 4 < 5 = A_D(001110)$.
2. The DFA witnessing $A_D(x)$ often appears unnatural, in the sense that determinism requires $A_D(x)$ to be total: in many cases, an automaton non-"deterministically" witnessing $A_D(x)$ needs to be augmented by an extra state to which every non-accepting path leads.

To overcome these obstacles, Hyde introduced automatic complexity witnessed by the smallest *non-deterministic* finite automaton (NFA) [12].

▶ **Definition 3.** *Let $x \in \Sigma^*$. An automaton M* **uniquely accepts** *x if M exactly accepts x and there is only one path in M which accepts x.*

Clearly, every DFA which exactly accepts x also uniquely accepts x. For NFAs, however, this is not the case. An NFA uniquely accepts x if and only if the NFA exactly accepts x and the NFA is unambiguous on $\Sigma^{|x|}$. Though Hyde [12] required the NFA to be unambiguous on $\Sigma^{|x|}$, she noted that the complexity based on NFAs is much more flexible and many words have a smaller complexity in her version than if only DFAs are considered. This led her to:

▶ **Definition 4.** *Let $x \in \Sigma^*$. The* **unique non-deterministic automatic complexity** *of x is given by*

$$A_N(x) = \min\{\, k \in \mathbb{N} : \text{there exists an NFA of } k \text{ states which uniquely accepts } x \,\}.$$

▶ Remark 5. We note that this notion is usually called "non-deterministic automatic complexity". As we study an ostensibly weaker notion below, we emphasise the additional strength of the notion defined in Definition 4 by adding the attribute "unique".

While it is well-known that NFAs and DFAs recognise exactly the same class of languages – the regular languages (see e.g. [33, 14] for a comprehensive background on automata theory) – the respective notions of automatic complexity differ. The following properties of A_N have been derived by Hyde and Kjos-Hanssen alongside co-authors, and others. Let $M_N(x)$ denote both the **minimal automaton witnessing** $A_N(x)$ and the directed graph representing it.

▶ **Lemma 6.** *Let $x \in \Sigma^*$.*
1. *$A_N(x) \leq (|x|/2) + 1$ follows from exhibiting suitable NFAs* [12].
2. *$M_N(x)$ is planar* [2].

Building upon Hyde's work from [12], in the present paper we study more closely the notion of automatic complexity induced by a weaker class of machines: the class of exactly but not necessarily uniquely accepting automata.

▶ **Definition 7.** *Let $x \in \Sigma^*$. The* **non-deterministic automatic complexity** *of x is*

$$A_{Ne}(x) = \min\{\, k \in \mathbb{N} : \text{there exists an NFA of } k \text{ states which exactly accepts } x \,\}.$$

Since every NFA which uniquely accepts x also exactly accepts x, we have $A_{Ne}(x) \leq A_N(x)$. Whether equality holds is still open (Question 48). In [20], Kjos-Hanssen investigated the complexity of certain languages induced by A_N in terms of more complicated models of computation, e.g. pushdown automata. In particular, he showed:

▶ **Theorem 8.**
1. *$\{\, x \in \{0,1,2\}^* : A_N(x) \leq |x|/2 \,\}$ is not context-free.*
2. *$\{\, x \in \{0,1\}^* : A_N(x) \leq |x|/3 \,\}$ cannot be recognised by constant-depth circuits with semi-unbounded fan-in, using Boolean $\wedge$- and $\vee$-gates.*

Results of this type motivate this paper: we investigate the impact of exactness on the behaviour of automatic complexity, which we describe via theorems akin to Theorem 8.

1.2 Our Theorems and the Structure of This Paper

We investigate the complexity of A_{Ne} as a function in terms of the complexity of the language of A_{Ne}-complicated words. Explicitly, we investigate the following class of languages first defined[3] by Kjos-Hanssen [20], and prove results on their complexities.

▶ **Definition 9.** *For $q \in (0, 1/2)$, define $L_q = \{ x \in \Sigma^* : A_{Ne}(x) < q|x| \}$.*

In Section 2, we isolate complexity results on the L_q-sets which follow from a fine-grained investigation of its elements. For instance, in Proposition 17 we isolate an upper bound of the Kolmogorov complexity of words in L_q. This gives a small-to-large result – a theorem about elements which provides information about sets – in the form of Corollary 19, which shows that the cardinality of $L_q \cap \Sigma^n$ is in $o(|\Sigma|^n)$. This observation also yields a proof of the **Shannon effect** for A_{Ne}:

▶ **Theorem 22.** *Let $A_{Ne}(\Sigma^n) = \max_{x \in \Sigma^n} A_{Ne}(x)$. For almost every $x \in \Sigma^*$ we have*

$$A_{Ne}(x) \geq A_{Ne}\left(\Sigma^{|x|}\right) - o\left(A_{Ne}\left(\Sigma^{|x|}\right)\right).$$

In Section 3, we show that pushdown automata are not powerful enough to characterise A_{Ne}-complicated words, which the following theorems show.

▶ **Theorem 34.** *For every $q \in (0, 1/2)$, the language L_q is not context-free.*

▶ **Theorem 35.** *For every $q \in (0, 1/2)$, the language $\Sigma^* \setminus L_q$ is not context-free.*

In Section 4, we consider the complexity of L_q in terms of Boolean circuits. To do so, we use two classical types of Boolean circuits – $\mathbf{SAC}^0$, defined in Section 4.1, and $\bigoplus\mathbf{SAC}^0$, defined in Section 4.2 – and apply a counting argument to prove:

▶ **Theorem 39.** *Let $q \in (0, 1/2)$ and $|\Sigma| = 2$. Then $L_q \notin \mathbf{SAC}^0$ and $\Sigma^* \setminus L_q \notin \mathbf{SAC}^0$.*

▶ **Theorem 46.** *Let $q \in (0, 1/2)$ and $|\Sigma| = p$ for some prime p. Then $L_q \notin \bigoplus\mathbf{SAC}^0$ and $\Sigma^* \setminus L_q \notin \bigoplus\mathbf{SAC}^0$.*

As a special case, we show that $L_{1/3}$ is not $\bigoplus\mathbf{SAC}^0$-recognisable, answering a question of Kjos-Hanssen [20, p. 351]. Omitted proofs as well as a partial generalisation to alphabets of non-prime cardinality can be found in the related full version of this paper [4].

In Section 5, we conclude this paper by giving a few open questions.

2 Combinatorial Properties of L_q

In this section, we derive combinatorial properties of L_q which are needed in the sequel, particularly to prove Theorem 34. Fix $q \in (0, 1/2)$. Firstly, we show that L_q satisfies a strong closure property: any word $x \in \Sigma^*$ can be extended to some word $y \in \Sigma^*$ for which $y \in L_q$.

▶ **Proposition 10.** *Suppose $x \in \Sigma^*$. If $m > |x|/q$ then $x^m \in L_q$.*

Proof. Let $n = |x|$ and suppose $x = x_0 \cdots x_{n-1} \in \Sigma^*$. Now build an NFA as follows: there are n states $\{s_0, \ldots, s_{n-1}\}$, with s_0 being both the start and unique accepting state. Transitions are given by $s_i \xrightarrow{x_i} s_{i+1}$ for $i < n-1$ and $s_{n-1} \xrightarrow{x_{n-1}} s_0$. It is readily seen that this automaton witnesses $A_{Ne}(x^m) \leq |x| < qm < qm|x| = q|x^m|$, as needed. ◀

[3] In [20, Def. 17], Kjos-Hanssen has considered the complementary decision problem, given by $q|x| < A_{Ne}(x)$. We note that our class $\{ L_q : q \in (0, 1/2) \}$ is more general.

While the previous proposition employs repetition of words to push the non-deterministic automatic complexity down, in the following lemma we show that spacing out bits of information achieves the same effect. W.l.o.g., assume $0 \in \Sigma$. For notation, if $x = x_0 \cdots x_{n-1} \in \Sigma^*$ then define the **(Hamming) weight of** x by $\text{weight}(x) = |\{\, k < n \colon x_k \neq 0 \,\}|$.

▶ **Lemma 11** (Gap Lemma). *For every $c \in \mathbb{N}$ there exists $n \in \mathbb{N}$ such that if $x \in \Sigma^n$ and $\text{weight}(x) \leq c$ then $x \in L_q$.*

Note that, in the statement above, n depends on q, which we fixed at the beginning of this section. Before we give the proof, we need the following number-theoretical lemma, called Bertrand's postulate (for a proof see e.g. [29]). Let $\mathbb{P}$ denote the set of prime numbers.

▶ **Lemma 12** (Bertrand's postulate). *If $h > 1$ then $\mathbb{P} \cap (h, 2h)$ is non-empty.*

Proof of Proposition 11. Fix $c \in \mathbb{N}$. For each $n \in \mathbb{N} \setminus \{0, 1\}$, we define a finite sequence of primes by $(p_1(n), \ldots, p_c(n))$ as follows: put $p_1(n) = \min\left(\mathbb{P} \cap (\sqrt[c]{n}, 2\sqrt[c]{n})\right)$ and

$$p_{i+1}(n) = \min(\mathbb{P} \cap (p_i, 2p_i)) \qquad \text{for } i = 1, 2, \ldots, c-1.$$

Since $n > 1$, Bertrand's Postulate shows that this is well-defined. Now, let

$$Q_i(n) = \left(\frac{1}{p_i(n)}\right) \prod_{j=1}^{c} p_j(n)$$

Bertrand's postulate alongside a short calculation implies $p_c(n) < 2^{c-1} p_1(n) < 2^c \sqrt[c]{n}$, and so

$$Q_i(n) \leq (p_c(n))^{c-1} \leq 2^{c(c-1)} n^{\frac{c-1}{c}}$$

which proves that $Q_i(n) \in O\left(n^{\frac{c-1}{c}}\right)$. This also shows that, in the limit, $Q_i(n) < n$. Similarly, $Q_i(n)p_i(n) > (p_1(n))^c > (\sqrt[c]{n})^c = n$ and hence, again in the limit, $Q_i(n) < n < Q_i(n)p_i(n)$. For $x \in \Sigma^n$ with $\text{weight}(x) \leq c$, write $x = w_0 0^{\ell_1} w_1 \cdots 0^{\ell_k} w_k$ for some $k \leq c$. Given our fixed $q \in (0, 1/2)$, we now choose $n \in \mathbb{N}$ sufficiently large so that we may assume the following (write $Q_i = Q_i(n)$):

- $|w_i| \leq cQ_1$ for $i = 0, 1, \ldots, k$.
- $\ell_i \geq Q_1$ for $i = 1, \ldots, k$.

Now, write $\ell_i = a_i Q_i + r_i$ where $0 \leq r_i < Q_i$. Since $Q_i p_i > n$ we must have $a_i < p_i$; otherwise, $|\ell_i| > n = |x|$, a contradiction. Hence, consider the automaton M in Figure 1.

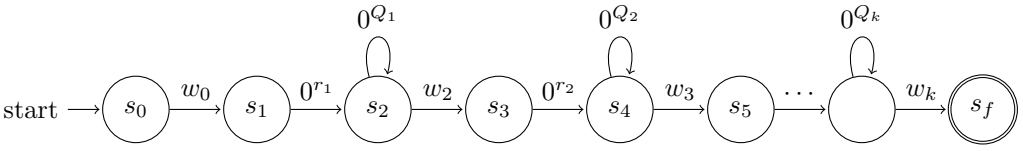

Figure 1 The non-deterministic automaton witnessing the "gap lemma".

We show that M is as required. First, by definition, M accepts x. To show exactness, suppose $y \in \Sigma^n$ and that M accepts y. If $x \neq y$, assume w.l.o.g. that $M(y)$ traverses the 0^{Q_1}-loop fewer than a_i-many times. Since $|y| = n$, $M(y)$ must go through the remaining loops more often to make up for the Q_1-deficit. However, the equation $Q_1 = d_2 Q_2 + \ldots + d_k Q_k$ has no integer solution, since p_1 divides the right-hand side yet not Q_1. Thus, M cannot accept y, as needed. Finally, recall that $r_i, |w_i| \leq cQ_1 \in O\left(n^{\frac{c-1}{c}}\right)$. Thus, for n large enough, inspection of the automaton M shows that $A_{Ne}(x) \leq 3kcQ_1 \in O\left(n^{\frac{c-1}{c}}\right)$ for every $x \in \Sigma^n$, and so $A_{Ne}(x) < q|x|$. ◀

As a consequence of our proof, we also obtain the following (we thank the anonymous reviewer for pointing this out).

▶ **Corollary 13.** *For every $c \in \mathbb{N}$ and every $q \in (0, 1/2)$ there exists $n \in \mathbb{N}$ such that if $x \in \Sigma^{>n}$ and* $\mathrm{weight}(x) \leq c$ *then $x \in L_q$.*

Our next result studies the small-scale structure of words in L_q. We say w is a **factor of** x if there exist $u, v \in \Sigma^*$ for which $x = uwv$; we write $w \preceq x$. If $u \in \Sigma^+$ or $v \in \Sigma^+$ then w is a **proper factor of** x; we write $w \prec x$. Call a non-empty word w a **square** if there exists $v \prec w$ for which $w = vv$; we write $w = v^2$.

▶ **Proposition 14.** *Let $n \geq 4$ and $x \in L_q \cap \Sigma^n$. There exists a proper factor $w \prec x$ of length $|w| \geq \left(\frac{1-2q}{2}\right)\sqrt{n}$ for which there are $u, v \in \Sigma^+$ with $|u| = |v| \leq |w|$ and $uw = wv \prec x$. Further, $uwv \prec x$.*

Note that if $|u| = |v| = |w|$ then the conclusion of Proposition 14 yields a square. To prove the general case of Proposition 14, we again need a classical auxiliary result, in this case due to Lyndon and Schützenberger [27].

▶ **Theorem 15** (The First Lyndon-Schützenberger-Theorem)**.** *Suppose $x, y \in \Sigma^*$. Then $xy = yx$ if and only if there exists $z \in \Sigma^*$ and $k, \ell \in \mathbb{N}$ for which $x = z^k$ and $y = z^\ell$.*

Note that the First Lyndon-Schützenberger-Theorem characterises *bordered words*[4] – those which have a non-trivial decomposition of the form $uw = wv$ – as those generated by powers of a common word z. This will be important in the proof of Proposition 14. We also require the following combinatorial lemma.

▶ **Lemma 16.** *Suppose $x \in \Sigma^n$ for some $n \geq 4$. Assume $x \in L_q$, and let $M_{Ne}(x)$ be the witnessing automaton with accepting run $(q_0, \ldots, q_n)$. Then*

$$|\{\, k \in \mathbb{N} \colon (\exists\, i, j)(i < j < k \wedge q_i = q_j = q_k)\,\}| \geq (1 - 2q)n.$$

Proof. Consider the list of states $(q_0, \ldots, q_n)$. Since $q < 1/2$, we have $2qn < n$. In particular, $n = 2qn + (1-2q)n$. Hence, by the pigeonhole principle, there exist at least $(1-2q)n$ indices at which some state is visited a third time. ◀

We now prove Proposition 14. Call triples (i, j, k) as provided by Lemma 16 **loop triples (for x)**. Before we give the proof of Proposition 14, we introduce the following notation: write $x_{[i,j]} = x_i \cdots x_j$. For instance, if $n \geq 4$, then $x_0 x_1 \cdots x_{n-1} = x_{[0,n-1]} = x_{[0,2]} x_{[3,n-1]}$.

Proof of Proposition 14. Let $x \in \Sigma^n$ be as assumed, and suppose $(q_0, \ldots, q_n)$ is the run of M_{Ne} which accepts x. Observe that if (i, j, k) is a loop triple for x (by Lemma 16 there are at least $(1 - 2q)n$ many), then the witnessing NFA $M_{Ne}(x)$ has completed at least two loops by the time it has read the word $x_{[0,k-1]}$. There are two cases.

1. There exists a loop triple (i, j, k) for which $\max\left(\left|x_{[i,j-1]}\right|, \left|x_{[j,k-1]}\right|\right) > (1 - 2q)\sqrt{n}$. Assume w.l.o.g. that $\left|x_{[j,k-1]}\right| \geq \left|x_{[i,j-1]}\right|$ and write $x = x_{[0,i-1]} x_{[i,j-1]} x_{[j,k-1]} x_{[k,n-1]}$. Since (i, j, k) is a loop triple, $q_i = q_j = q_k$, and thus $M_{Ne}(x)$ also accepts the word $x_{[0,i-1]} x_{[j,k-1]} x_{[i,j-1]} x_{[k,n-1]}$. Since $M_{Ne}(x)$ exactly accepts x, we have $x_{[j,k-1]} x_{[i,j-1]} =$

[4] For more on bordered words, see e.g. [30]. A more general characterisation is given by the Second Lyndon-Schützenberger-Theorem 18.

$x_{[i,j-1]}x_{[j,k-1]}$, and so Theorem 15 implies $x_{[i,j-1]} = z^k$ and $x_{[j,k-1]} = z^\ell$ for some $z \in \Sigma^+$ and $k, \ell \in \mathbb{N}$. Since $k, \ell \geq 1$, the decomposition of $x_{[i,k-1]}$ trivialises into a product of copies of z:

$$x_{[i,k-1]} = x_{[i,j-1]}x_{[j,k-1]} = zz^{k+\ell-1} = z^{k+\ell-1}z$$

As $\left|z^{k+\ell-1}\right| \geq \left|x_{[j,k-1]}\right| \geq (1-2q)n > (1-2q)\sqrt{n}$, we are done.

2. For all loop triples (i,j,k) we have $\max\left(\left|x_{[i,j-1]}\right|, \left|x_{[j,k-1]}\right|\right) \leq (1-2q)\sqrt{n}$.
By Lemma 16, there exist $(1-2q)n$ indices k for which there exist (i,j) such that (i,j,k) is a loop triple. Since every loop in a loop triple has length at most $(1-2q)\sqrt{n}$, the pigeonhole principle gives an $\ell \leq (1-2q)\sqrt{n}$ such that there exist at least $m \geq \sqrt{n}$ such indices k at which a loop of length ℓ was just completed (hence, we only focus on the *second* loops in each loop triple). Let this set of indices be given in ascending order, denoted by $\mathcal{K} = \{k_1, \ldots, k_m\}$, with associated loops $\rho_1, \ldots, \rho_m \prec x$, each of length ℓ.
We show that ρ_1 and ρ_m must be disjoint, i.e. share no states along their traversals in $M_{Ne}(x)$. Let q_{k_1} be the origin state of the loop ρ_1. By definition, ρ_1 is the second loop in the loop triple (i_1, j_1, k_1). Suppose τ is the first loop at q_{k_1} so that $\tau\rho_1$ is a loop triple at q_{k_1}. Then, if we read $b > (1-2q)\sqrt{n}$ letters along the loops at state q_{k_1}, then we could concatenate those loops with τ to obtain a loop triple, one of whose lengths exceeds $(1-2q)\sqrt{n}$, which contradicts the assumption of this case. Therefore, at state q_{k_1}, we can only read at most $(1-2q)\sqrt{n}$ letters of the factors contained in $\rho_1, \ldots, \rho_m$, before moving on to a different state, never to return. However, by construction, for every $i \leq m$ we know that x_{k_i} appears in ρ_i, and thus we must read at least $m \geq \sqrt{n}$ letters throughout all loops $\rho_1, \ldots, \rho_m$. Since $q < 1/2$, we have $(1-2q)\sqrt{n} < \sqrt{n} \leq m$; hence, the first and last loops ρ_1 and ρ_m must be disjoint. Thus, $x = u\ \rho_1\ y\ \rho_m\ u'$ where $u, y, u' \prec x$ and $|\rho_1| = |\rho_m| = \ell$. By exact acceptance of $M_{Ne}(x)$, we have

$$x = u\ (\rho_1)^2\ y\ u'$$

since $|\rho_1| = |\rho_m|$. Therefore, $\rho_1 y = y\rho_m$, and thus, with $y' = \rho_1 y$, we have $y'\rho_m = \rho_1 y'$. To show that y' has the desired length, note that $y\rho_m$ must contain the set $\{x_{k_2}, \ldots, x_{k_m}\}$; the loop ρ_1, since it is the first loop in $\mathcal{K}$, can only contain x_{k_1}. Since $n \geq 4$, we have

$$|y'| = |y\rho_m| \geq |\mathcal{K}| - 1 = m - 1 \geq \sqrt{n} - 2 \geq \frac{\sqrt{n}}{2}.$$ ◀

We now apply Proposition 14 to go even finer: instead of studying the complexity of L_q, we classify the complexity of *words* in L_q, using plain Kolmogorov complexity. Fix an alphabet Σ of cardinality k, and let C_k denote the **plain Kolmogorov complexity** on words in Σ:

$$C_k(x) = \min\{\, \ell(p) \colon U_k(p) = x \,\}.$$

where U_k is a universal Turing machine on the k-element alphabet. For details on Kolmogorov complexity, see e.g. [5].

▶ **Proposition 17.** *If $x \in \Sigma^n \cap L_q$, then*

$$C_k(x) \leq n - \frac{(1-2q)}{2}\sqrt{n} + 5\log_k(n) + O(1).$$

Its proof – which is included in the full version [4] – requires an extension of Theorem 15, which gives a sufficient and necessary criterion for the decomposition of words with same prefix and suffix. As it will be useful to us in the sequel outside of the proof of Proposition 17, we state it right here in the version of [33].

▶ **Theorem 18** (The Second Lyndon-Schützenberger-Theorem). *Let $x, y, z \in \Sigma^*$. Then $xy = yz$ iff there exist $e \in \mathbb{N} \setminus \{0\}$, $u \in \Sigma^+$ and $v \in \Sigma^*$ such that $x = uv$, $z = vu$, and $y = x^e u = u z^e$.*

With $|\Sigma| = k$ as before, note that the function which maps $x \in \Sigma^*$ to its C_k-witness is an injection. Hence, Proposition 17 immediately yields the following bound on $|L_q|$.

▶ **Corollary 19.** *The set $L_q \cap \Sigma^n$ has cardinality in $o(|\Sigma|^n)$.*

From Corollary 19, we now deduce the Shannon effect for A_{Ne}. Originally conjectured by Shannon [36] and proven (and named) by Lupanov for Boolean functions [25, 26], the Shannon effect says that most strings are of almost maximal complexity. We give a definition due to Wegener [45].

▶ **Definition 20.** *Let $P \subset \Sigma^*$. We say that* **almost all** x **have property** P *if*

$$\lim_{n\to\infty} \frac{|P \cap \Sigma^n|}{|\Sigma|^n} = 1.$$

▶ **Definition 21.** *Let Γ be a complexity measure defined on Σ^*. For $n \in \mathbb{N}$, let $\Gamma(\Sigma^n) = \max_{x\in\Sigma^n}(\Gamma(x))$. Then Γ has the* **Shannon effect** *if for almost all $x \in \Sigma^*$ we have*

$$\Gamma(x) \geq \Gamma\left(\Sigma^{|x|}\right) - o\left(\Gamma\left(\Sigma^{|x|}\right)\right).$$

By exhibiting upper and lower bounds of complexity for *all* words, it is readily seen that (plain and prefix-free) Kolmogorov complexity satisfy the Shannon effect [21, 41, 42, 23, 22], as do A_D [35] and A_n [12, 18]. The cardinality argument of Corollary 19 shows:

▶ **Theorem 22.** *A_{Ne} satisfies the Shannon effect.*

Proof. Fix $q = 1/(2+\epsilon)$ for some small $\epsilon > 0$. Since $A_{Ne}(x) \leq A_N(x) \leq (|x|/2) + 1$ (Lemma 6), identifying a suitable lower bound suffices. By Corollary 19, for $o(|\Sigma|^n)$-many words $x \in \Sigma^n$ we have $x \in L_q$. Hence, for almost all (as per Definition 20) $x \in \Sigma^n$,

$$\frac{n}{2+\epsilon} \leq A_{Ne}(x) \leq \frac{n}{2} + 1$$

and so $A_{Ne}(x)/|x|$ is sufficiently close to $1/2$ for typical large x. ◀

3 L_q Is Not Context-Free

Fix $q \in (0, 1/2)$ and suppose w.l.o.g. that $0, 1 \in \Sigma$. In this section, we demonstrate that L_q cannot be generated by a context-free grammar (CFG); hence, L_q is not context-free. To this end, we first define the concept of a *rich* CFG. We then prove that if a CFG generates L_q, it must be rich. Finally, we show that any rich CFG generates words of arbitrarily high complexity, which contradicts the fact that the CFG generates L_q.

We provide the required definitions. (For more details, see e.g. [33].) A **context-free grammar** (CFG) is a tuple $G = (V_T, V_N, S, P)$ where:

- V_T is the set of **terminal symbols**.
- V_N is the set of **non-terminal symbols**.
- $S \in V_N$ is the **start symbol**.
- P is a finite set of **productions**.

We also insist that $V_T \cap V_N = \emptyset$, and we define the set of **symbols** by $V = V_T \cup V_N$. Elements of V^* are called **sentential forms**. Productions in P are pairs (A, γ) where $A \in V_N$ and $\alpha \in V^*$. We denote such a production by $A \to \gamma$.

The **derivation relation** $\Longrightarrow$ is defined as follows: if $\alpha, \beta \in V^*$ then $\alpha \Longrightarrow \beta$ if $\alpha = \alpha' A \alpha''$ and $\beta = \alpha' \gamma \alpha''$ for some $\alpha', \alpha'' \in V^*$, and there exists a production $A \to \gamma$ in P. The transitive and reflexive closure of $\Longrightarrow$ is denoted by $\Longrightarrow^*$.

A language that is recognisable by a CFG is called a **context-free language** (CFL).

▶ **Definition 23.** *A CFG has* **no useless nonterminals** *if:*
1. *each nonterminal is reachable from the starting symbol; and*
2. *a terminal string can be derived from each nonterminal.*

▶ **Definition 24.** *Let G be a CFG. A nonterminal $A \in G$ is a* **rich nonterminal** *if for some words $v, w, x, y \in \Sigma^*$ we have $vwxy \neq \varepsilon$ and $A \Longrightarrow^* vAx \mid wAy$ as well as:*
1. *if $vw \neq \varepsilon$ then $vw \neq wv$; and*
2. *if $xy \neq \varepsilon$ then $xy \neq yx$.*

A **rich CFG** *has a rich nonterminal but no useless nonterminals. A* **rich CFL** *is generated by a rich CFG.*

Our motivation for rich CFGs follows from Theorem 15, however, we note here that, in style, our richness characterisation is similar to classical results by Ginsburg [9, Theorem 5.5.1], who characterised boundedness of CFLs via syntactical properties of grammars. Our syntactical notion of richness, similarly, characterises the complexity of generated languages, in our case L_q. The equivalence in Theorem 15 implies that a rich non-terminal can construct words which do not collapse to repeating copies of a common factor z. This is needed in Section 3.2, where we construct high-complexity words.

3.1 Only Rich CFGs Can Generate L_q

We require the following normal form theorem due to Greibach [10] (see [11, p. 277] for a modern exposition).

▶ **Theorem 25** (Greibach Normal Form Theorem)**.** *Every CFG with no ε-productions can be expressed in* **Greibach Normal Form***: all its production rules are of the form $A \to x\overline{A}$ where $x \in \Sigma$ and $\overline{A}$ is a finite word of nonterminals.*

The following corollary is immediate.

▶ **Corollary 26.** *Every CFL omitting ε is generated by a CFG in Greibach Normal Form.*

Our main result in this subsection is the following.

▶ **Theorem 27.** *If L_q is generated by a context-free grammar G, then G is rich.*

Proof. By our results in the previous section, L_q is non-empty; further, by definition, $\varepsilon \notin L_q$. So, by Corollary 26, there exists a CFG G in Greibach Normal Form which generates L_q. We show that G must be rich by a counting argument on the number of nonterminals of G. Let $k \in \mathbb{N}$ denote the number of nonterminals in G. Define

$$x_i = 0^i 1^{4k-i} \qquad \text{for } i = 1, 2, \ldots, 4k-1.$$

By Proposition 10, for every i there exists $m_i \in \mathbb{N}$ for which $x_i^{m_i} \in L_q$. Similarly, for each i there exists $m_i' \in \mathbb{N}$ for which the derivation tree of $x_i^{m_i'}$ has a branch which contains some nonterminal A at least $(4k)^2 + 1$ times. Let $M \in \mathbb{N}$ be sufficiently large to satisfy these

requirements for all x_i simultaneously. By the pigeonhole principle, there exist $i, j, \ell \leq 4k-1$ such that some nonterminal A appears at least $(4k)^2 + 1$ times in some branch of the derivation tree of each of x_i^M, x_j^M and x_ℓ^M.

Consider such a sufficiently long branch of the derivation tree of x_i^M, in which we choose to expand A at the end. Since G is in Greibach Normal Form, the derivation is of the form

$$S \Longrightarrow^* y_i^1 y_i^2 y_i^3 \dots y_i^s A z_i^s \dots z_i^3 z_i^2 z_i^1$$

from which x_i^M can be derived in at least $(4k)^2+1$ expansions of A. Observe that each $y_i^j \neq \varepsilon$, since G is in Greibach Normal Form. Consider the number of expansions of A in terms of blocks $B_1, B_2, \dots, B_n$ such that each block has cardinality $4k$. By assumption, $n \geq 4k+1$. Let A_m be the derivation of A from the expansions in block B_m. There are two cases:

1. For some $m \leq n$, $A_m = yA$ with $y \in \Sigma^*$ and[5] $|y| \geq 4k$.
2. For all $m \leq n$, $A_m = y_m A z_m$ with $y_m, z_m \in \Sigma^+$. Then, $A_{4k} = yAz$ with $|y|, |z| \geq 4k$.

Since these two cases apply to all x_i^M, x_j^M and x_ℓ^M, two of them must share the same case above. W.l.o.g. assume both x_i^M and x_j^M fall into case 2 (the argument for case 1 is similar). Hence, $T \Longrightarrow^* yAz$ (from the derivation of x_i^M) and $T \Longrightarrow^* vAw$ (from the derivation of x_j^M) where $|y|, |z|, |v|, |w| \geq 4k$. By definition, y, z contain i zeroes, while v, w contain j zeroes among the first $4k$ letters. It is now seen from the First Lyndon-Schützenberger-Theorem that $yv \neq vy$ and $zw \neq wz$. Hence, A is a rich nonterminal. ◀

3.2 Every Rich CFG Generates High-Complexity Words

In this section, we prove that every rich CFG generates words of arbitrarily high complexity relative to its length. In particular, there exists a word x for which $A_{Ne}(x) > q|x|$ for every $q \in (0, 1/2)$. This contradicts the fact that any rich CFG can generate L_q for any q, since any $x \in L_q$ satisfies $A_{Ne}(x) < q|x|$. We also isolate the following technical proposition.

▶ **Proposition 28.** *Suppose $u, v \in \Sigma^n$ with $uv \neq vu$. Then the following set is infinite:*

$$\mathcal{I}_{(u,v)} = \{\, x \in \{u, v\}^* \colon \textit{if } y \prec x \textit{ satisfies } |y| > 2\log(|x|) \textit{ then } y \textit{ occurs exactly once in } x \,\}$$

Proving Proposition 28 takes a few technical lemmas on the behaviour of non-commuting strings in formal languages. Denote the **set of factors** of $w \in \Sigma^*$ by $[w] = \{\, x \in \Sigma^* \colon x \prec w \,\}$. For convenience, we now fix some $n \in \mathbb{N}$ and a pair $u, v \in \Sigma^n$ for which $uv \neq vu$.

▶ **Lemma 29.** $uv, vu \notin [u^3] \cup [v^3]$

Proof. We give the argument for $uv \notin [u^3]$; the other parts are similar. Assume that $uv \in [u^3]$; thus write $u^3 = xuvy$ for some $x, y \in \Sigma^*$. Note that $|xy| = |u| = |v|$. Since $uv \neq vu$ we cannot have $x, y \in \{u, v\}$, and thus $|x|, |y| < |u| = |v|$. But now, by periodicity of u^3, we must have $xy = u$. Thus $u^3 = xyxyxy = xuvy$. Therefore, $uv = yxyx$, from which it follows that $u = xy = yx = v$, contradicting the fact that $uv \neq vu$. ◀

To motivate the next lemma, we provide the definition of string homomorphisms.

▶ **Definition 30.** *A function $h \colon \{0, 1, \dots, n-1\}^* \to \Sigma^*$ is a* **string homomorphism** *if for all $n_i \in \{0, 1, \dots, n-1\}$ we have $h(n_0 \cdots n_k) = h(n_0) \cdots h(n_k)$.*

[5] This is a consequence of the observation immediately following Theorem 25.

Observe that every such string homomorphism is uniquely defined by its action on the alphabet. Define a string homomorphism $h\colon \{0,1,2\} \to \{u,v\}^*$ given by

$$h(0) = uv \qquad h(1) = vu \qquad h(2) = u^3v^4$$

With this string homomorphism fixed, the following lemma is immediate from Lemma 29.

▶ **Lemma 31.** $u^4, v^4 \notin \bigcup\{\, [x] \colon x \in h(\{0,1\}^*)\,\}$

To give a proof of Proposition 28, we first code words as follows. For every $k \in \mathbb{N}$, let σ_k be the lexicographical concatenation of all positive integers which, coded in binary, have length k; each is then followed by a 2. For instance, $\sigma_2 = 002012102112$. We consider the images of these words under h, and collect some immediate properties of the σ_k and the $h(\sigma_k)$ below, whose proofs are readily deduced, hence omitted.

▶ **Lemma 32.** *Let $k \in \mathbb{N}$.*

1. $|\sigma_k| = 2^k(k+1)$
2. $|h(\sigma_k)| = 2^k|v|(2k+7)$
3. $2\log(|h(\sigma_k)|) \geq 2k + 14|v|$ *for sufficiently large k.*

To prove Proposition 28, we show that for large enough k, every substring of $h(\sigma_k)$ of length at least $2\log|h(\sigma_k)|$ must contain two copies of $h(2)$; since the word between any two copies of $h(2)$ is unique within $h(\sigma_k)$, the proposition is proven.

Proof of Proposition 28. Fix $k \in \mathbb{N}$ sufficiently large so as to satisfy Item 3, and consider the word $h(\sigma_k)$. By construction and the choice of k, if $y \prec h(\sigma_k)$ and $|y| \geq 2\log(|h(\sigma_k)|)$ then y contains two copies of $h(2)$. By definition, $v^4 \prec h(2)$; on the other hand, Lemma 31 shows that v^4 cannot be a factor of any $h(w)$ with $w \in \{0,1\}^*$. Hence, $v^4 \prec y$ must be a factor of some $h(2)$ occurring in $h(\sigma_k)$. We show that the copies of $h(2)$ in y and in $h(\sigma_k)$ overlap perfectly. Consider the word $h(2)z = xh(2)$ contained in $h(\sigma_k)$. This bordered word is in fact a proper square, which can seen by a case analysis on x. Write

$$\begin{aligned} x &= a|u| + \ell && \text{for some } a \in \mathbb{N}, 0 \leq \ell < |u| \\ u &= \alpha\beta && \text{where } |\alpha| = \ell \\ v &= \gamma\delta && \text{where } |\gamma| = \ell \end{aligned}$$

and note that this renaming implies $|\beta| = |\delta|$, and that

$$h(2)z = xh(2) = x_1x_2(\alpha\beta)^3(\gamma\delta)^4 = (\alpha\beta)^3(\gamma\delta)^4z.$$

There are four cases describing x_1; using Theorems 15 and 18, each will lead to a contradiction.

1. $a = 0$: Since $|\alpha| = \ell$, this case implies $\alpha\beta = \beta\alpha$. Further, $|\beta\gamma| = |\alpha\beta| = |\beta\alpha|$, and so $\alpha = \gamma$. By comparing lengths, it is easily seen that $\beta = \delta$, and so $u = \alpha\beta = \gamma\delta = v$, a contradiction.
2. $a = 1$: Since $u = \alpha\beta$, by comparing initial segments it is readily seen that in this case $uv \prec v^4$, contradicting Lemma 29.
3. $a = 2$ or $a = 3$: Since $xh(2) = h(2)z$, we have $|x| = |z|$. So, if $a = 2, 3$, then again by comparing initial segments it is readily seen that $uv \prec v^4$, contradicting Lemma 29.
4. $a > 3$: Here, $v^4 \prec z$. Since $z = h(w)$ for some $w \in \{0,1\}^*$, Lemma 29 gives a contradiction.

Hence, the copies of $h(2)$ appearing in y are exactly those appearing in $h(\sigma_k)$. But now, if $y' \prec h(\sigma_k)$ is of length at least $2\log(|h(\sigma_k|)$, then it contains a factor of the form $h(2)\rho h(2)$ where $\rho \in h(\{u,v\}^*)$. Each such ρ appears only once in $h(\sigma_k)$, by construction. Thus, for large enough k, the word $h(\sigma_k)$ is as required, and thus the set $\mathcal{I}_{(u,v)}$ is infinite. ◀

▶ **Theorem 33.** *If G is a rich CFG, then G generates a word $x \in \Sigma^*$ such that $A_{Ne}(x) > q|x|$ for every $q \in (0, 1/2)$.*

For notation, if $\sigma \in \{0,1\}^*$, let $\overline{\sigma}$ denote the reverse of σ. Further, if $x, y \in \Sigma^*$ satisfy $xz = zy$ and both $xz, zy \prec w$ such that xz and zy overlap at z, then call xzy its **union**, written as $xz \cup zy$. We use Proposition 28.

Proof. Let G be a rich CFG with rich nonterminal A and witnesses $x, y, x', y' \in \Sigma^*$ for which $A \Longrightarrow^* xAy \mid x'Ay'$ and $xx' \neq x'x$ and $yy' \neq y'y$. Define $u_1 = xx', v_1 = x'x$ and $u_2 = yy', v_2 = y'y$. Now define string homomorphisms g, h by:

$$g(0) = u_1v_1 \qquad g(1) = v_1u_1 \qquad g(2) = u_1^3v_1^4$$
$$h(0) = u_2v_2 \qquad h(1) = v_2u_2 \qquad h(2) = u_2^3v_2^4$$

Now, fix any $w_1, w_2, w_3 \in \Sigma^*$ for which

$$S \Longrightarrow^* w_1Aw_3 \qquad \text{and} \qquad A \Longrightarrow^* w_2. \tag{$*$}$$

By repeated application of the generation rules in $(*)$, it is readily seen that for any $m, k \in \mathbb{N}$, the word $y_{m,k}$ of the following form is generated by G:

$$y_{m,k} = (w_1\ g(\sigma_k))\ (x^m\ w_2\ y^m)\ (\overline{h(\sigma_k)}\ w_3).$$

We show that, for sufficiently large m, k, the word $y_{m,k}$ has large non-deterministic automatic complexity. Choose m, k large enough so that $|y_{m,k}| \gg |w_1w_2w_3|$, and let $n = |y_{m,k}|$. Since we may choose k, m freely, we may also impose that

$$2\log(n) \leq m|x| \leq 3\log(n) \in o(\sqrt{n}). \tag{$\dagger$}$$

Now, let $z \prec y_{m,k}$ whose length is in $O(\sqrt{n})$ be the first occurrence of a word in $y_{m,k}$ of the form $zb = cz$ for words $b, c \prec y_{m,k}$. Below, we show that this is only possible if $b = c = \varepsilon$.

By choosing m, k wisely, we may assume that $|z|$ is even. Further, it will be convenient to distinguish the words which make up the left-hand and right-hand squares of z; hence, write $z = z_1z_2 = z_1'z_2'$ so that $|z_1| = |z_2|$ and $z_1 = z_1', z_2 = z_2'$, and $z_1z_2b = cz_1'z_2'$.

We show that $z_1 \prec w_1g(\sigma_k)$; the case that $z_2' \prec \overline{h(\sigma_k)}w_3$ is similar. Note that, otherwise, we may choose k large enough so that z_1 intersects $\overline{h(\sigma_k)}w_3$, and in particular, we may enforce that this intersection $s \in \Sigma^*$ has length at least $2\log(n)$. By construction and the fact that $z_1z_2b = cz_1'z_2'$, the word s must appear twice in $\overline{h(\sigma_k)}$, which contradicts Proposition 28.[6]

Thus, $z_1 \prec w_1g(\sigma_k)$ and $z_2' \prec \overline{h(\sigma_k)}w_3$ imply that $x^mw_2y^m$ is a factor of the union $z_2b \cup cz_1'$. By a counting argument, it is seen that either $x^m \prec z_2$ or $y^m \prec z_1'$. If $x^m \prec z_2$ – the other case is similar – then also $x^m \prec z_2' \prec \overline{h(\sigma_k)}$. But this is impossible by Proposition 28 and since $|z_2'| \geq m|x| \geq 2\log(n)$ by $(\dagger)$. Therefore, we have arrived at a contradiction: we can only have $z_1z_2b = cz_1'z_2'$ if $b = c = \varepsilon$. But now, the contrapositive of Proposition 14 shows that $y_{m,k} \notin L_q$ for every $q \in (0, 1/2)$. Since $y_{m,k}$ is generated by G, the result is proven. ◀

Theorem 33 and Theorem 27 combined imply our main result of this section:

▶ **Theorem 34.** *For every $q \in (0, 1/2)$, the language L_q is not context-free.*

[6] See in particular the proof of Proposition 28 to note that $\mathcal{I}_{(u_2,v_2)}$ can be generated by sets of the form $h(\sigma_k)$, and by a similar argument, by those of the form $\overline{h(\sigma_k)}$.

A language L is **CFL-immune** if it contains no infinite context-free language as a subset. We note here that L_q cannot be CFL-immune, since for every letter $x \in \Sigma$, the regular language $\{x\}^+$ is contained in L_q (modulo finitely many words, depending on q), and each of its words has constant complexity. However, the following holds:

▶ **Theorem 35.** *For every $q \in (0, 1/2)$, the language $\Sigma^* \setminus L_q$ is CFL-immune.*

For the proof, we direct the reader to [4]. Since $A_{Ne}(x) \leq A_D(x)$ for all words x, Theorem 35 also implies:

▶ **Corollary 36.** *For every $q \in (0, 1/2)$, $\{\, x \in \Sigma^* \colon A_D(x) \geq q|x| \,\}$ is CFL-immune.*

We conjecture that recognising L_q can be done via a linearly bounded automaton.

4 L_q Cannot Be Recognised by Certain Constant-Depth Circuits

In this section, we expand on our work in Section 3 by investigating the complexity of L_q further. Instead of considering pushdown automata, in this section we consider constant-depth circuits. We show that two types of circuits cannot recognise L_q either, which is analogous to Theorem 34 for pushdown automata.

Fix $q \in (0, 1/2)$ and fix $\Sigma = \{0, 1\}$. We first provide the definitions of two types of constant depth circuits explicitly – the class $\mathbf{SAC}^0$ in Section 4.1, and $\bigoplus\mathbf{SAC}^0$ in Section 4.2 – and then show that neither can recognise L_q, nor its complement.

4.1 The Circuit Class SAC^0

Suppose $k \geq 1$. A language L is **$\mathbf{SAC}^k$-recognisable** if it is recognised by a polynomial-size, $O(\log^k n)$-depth, uniform[7] semi-unbounded fan-in circuit (a circuit is **semi-unbounded** if it has unbounded fan-in-$\vee$, bounded fan-in-$\wedge$, and admits negative literals but no other negations [3]; cf. [44, 20]). Of these classes of particular interest is $\mathbf{SAC}^1$, since it is the class **logCFL** of languages which are log-space reducible to context-free languages [43, 44]. More generally, $\mathbf{SAC}^k$ enjoys the following relationship with the classical classes $\mathbf{AC}^k$ and $\mathbf{NC}^k$:

$$\mathbf{NC}^k \subseteq \mathbf{SAC}^k \subseteq \mathbf{AC}^k \subseteq \mathbf{NC}^{k+1} \qquad \text{for all } k \geq 1.$$

Just like $\mathbf{NC}^k$ and $\mathbf{AC}^k$, the class $\mathbf{SAC}^k$ is also closed under complements [3, Corollary 15].

Here, we consider the class $\mathbf{SAC}^0$. Contrary to the classes above, $\mathbf{SAC}^0$ is *not* closed under complementation [3]. Note that $\mathbf{SAC}^0$-circuits have *constant* depth; hence, the $\mathbf{SAC}^0$-recognisable languages can be characterised by formulas in a simple propositional language, as expressed in Lemma 38. We give a formal definition of $\mathbf{SAC}^0$ due to Kjos-Hanssen [20].

▶ **Definition 37.** *A language $L \subset \{0, 1\}^*$ is **$\mathbf{SAC}^0$-recognisable** if there exists a family $(C_i)_{i\in\mathbb{N}}$ of Boolean circuits which recognises L and which satisfies the following:*

1. *Each C_i is defined over the basic set $\{\wedge, \vee\}$ and accepts negative literals.*
2. *The family $(C_i)_{i\in\mathbb{N}}$ has constant depth.*
3. *Each C_i has unbounded fan-in-$\vee$ and bounded fan-in-$\wedge$.*
4. *Each C_i accepts words of length i.*

[7] Requiring uniformity is debatable; see e.g. [20, Remark 29].

Note that, for the classes $\mathbf{SAC}^k$ with $k > 0$, the size of the circuit must be polynomial in n. However, this requirement is redundant for $\mathbf{SAC}^0$ (cf. [20, Remark 30]). An important characterisation of $\mathbf{SAC}^0$-recognisable languages follows (cf. [20] and [3, p. 560]).

▶ **Lemma 38.** *A language $L \subset \Sigma^*$ is $\mathbf{SAC}^0$-recognisable iff there exists $c \in \mathbb{N}$ such that: for every $n \in \mathbb{N}$ and every $x \in \Sigma^n$ there exist $k_n \in \mathbb{N}$ and a Boolean formula $\psi_n = \bigvee_{i=1}^{k_n} \varphi_{i,n}$ for which $\varphi_{i,n}$ is a conjunction of at most c literals, and*

$$x \in L \iff \psi_n(x) \text{ holds.}$$

Using this lemma, which can be deduced from the distributive properties of propositional languages, we conclude:

▶ **Theorem 39.** *Let $q \in (0, 1/2)$ and $|\Sigma| = 2$. Then $L_q \notin \mathbf{SAC}^0$ and $\Sigma^* \setminus L_q \notin \mathbf{SAC}^0$.*

Proof. The proof uses a counting argument using Lemma 38. First, suppose $L_q \in \mathbf{SAC}^0$, witnessed by a sequence of formulas $(\psi_n)_{n\in\mathbb{N}}$ and a constant c. Consider ψ_n. Since $\varphi_{1,n}$ mentions at most c variables, the circuit accepts every word which agrees on these c variables. Hence, ψ_n accepts at least 2^{n-c} words. Yet the order of L_q is $o(2^n)$, by Corollary 19, which contradicts the fact that $(\psi_n)_{n\in\mathbb{N}}$ recognises L_q.

Now, suppose $\Sigma^* \setminus L_q \in \mathbf{SAC}^0$, again accepted by $(\psi_n)_{n\in\mathbb{N}}$ with constant c. Separate the positive from the negative literals in $\varphi_{1,n}$; there are at most $c' \leq c$ such positive literals. Thus, for any word $x = x_1 \cdots x_n \in \Sigma^*$, if $x_i = 1$ for all such positive literals, and $x_i = 0$ everywhere else, then ψ_n accepts x. But for large enough n, such x is in L_q by Proposition 11, which contradicts the fact that $(\psi_n)_{n\in\mathbb{N}}$ recognises $\Sigma^* \setminus L_q$. ◀

4.2 The Circuit Class $\bigoplus\mathbf{SAC}^0$

In this section, we consider the class $\bigoplus\mathbf{SAC}^0$, whose definition differs from that of $\mathbf{SAC}^0$ only in the choice of base set. Let $\oplus$ denote the XOR operation. As before, fix $q \in (0, 1/2)$.

▶ **Definition 40.** *A language $L \subset \{0,1\}^*$ is $\bigoplus\mathbf{SAC}^0$-recognisable if there exists a family $(C_i)_{i\in\mathbb{N}}$ of Boolean circuits which recognises L and which satisfies the following:*

1. *Each C_i is defined over the basic set $\{\wedge, \oplus\}$ and accepts negative literals.*
2. *The family $(C_i)_{i\in\mathbb{N}}$ has constant depth.*
3. *Each C_i has unbounded fan-in-$\oplus$ and bounded fan-in-$\wedge$.*
4. *Each C_i accepts words of length i.*

From this definition and the following observation, we can investigate languages larger than binary. Recall that in the previous subsection, we focussed solely on the two-element alphabet $\{0, 1\}$. This was forced by the fact that Boolean expressions have trouble expressing Boolean operations on non-binary languages (e.g. what does $0 \wedge 2$ evaluate to?). This can be remedied in the class $\bigoplus\mathbf{SAC}^0$ for some languages, courtesy of the operator $\oplus$.

It is readily seen that $(\{0, 1\}, \oplus, \wedge)$ is isomorphic to the field of two elements $\mathbb{F}_2 = (\mathbb{Z}/2\mathbb{Z}, +, \times)$. (Studying Boolean circuits in terms of the arithmetic of $\mathbb{F}_2$ goes back to Gál and Wigderson [8]. We also mention here similarities to the work of Razborov-Smolensky [31, 39, 38].) To extend this equivalence beyond binary alphabets, take the field $\mathbb{F}_p$ for some prime $p > 2$. By interpreting $(\oplus, \wedge)$ as $(+, \times)$ mod p, we extend $\mathbf{SAC}^0$-recognisability to alphabets of prime cardinality. Below, we give a natural characterisation of $\bigoplus\mathbf{SAC}^0$-recognisability in terms of propositional formulas, similar to that in Lemma 38.[8]

[8] For a classical definition of $\bigoplus\mathbf{SAC}^0$ in terms of the complexity of Boolean circuits see e.g. [20, Section 4].

▶ **Definition 41.** *Let $|\Sigma| = p$ for some $p \in \mathbb{P}$. Then L is $\bigoplus\mathbf{SAC}^0$-recognisable if there exists $c \in \mathbb{N}$ such that: for every $n \in \mathbb{N}$ and every $x \in \Sigma^n$ there exists $k_n \in \mathbb{N}$ and a formula $\psi_n = \bigoplus_{i=1}^{k_n} \varphi_{i,n}$ for which $\varphi_{i,n}$ is a conjunction of at most c literals and*

$$x \in L \iff \psi_n(x) \neq 0.$$

▶ Remark 42. Observe that there is a subtle difference between $\mathbf{SAC}^0$ and $\bigoplus\mathbf{SAC}^0$ in the case $p = 2$. An $\mathbf{SAC}^0$ circuit accepts a word $x \in \Sigma^n$ if *any* term in the disjunction of $\psi_n(x)$ holds. On the contrary, in $\bigoplus\mathbf{SAC}^0$, the disjunction is interpreted as addition modulo 2, and hence, x is accepted only if the number of terms in the disjunction of ψ_n is odd. Also, note that Definition 41 requires a real-world formalism in which gates are able to carry out addition and multiplication modulo p as a primitive. This assumption is not needed when $p = 2$, as such Boolean circuits can be modelled using $\oplus$ and $\wedge$, as mentioned.

For completeness, we mention here that $\mathbf{SAC}^0 \neq \mathbf{coSAC}^0$ (see [3]), while $\mathbf{co}\bigoplus\mathbf{SAC}^0 = \bigoplus\mathbf{SAC}^0$ (inverting a polynomial in a finite field requires only a constant number of layers; we use this fact in the proof of Theorem 46). Further, $\mathbf{SAC}^0 \not\subseteq \bigoplus\mathbf{SAC}^0$ [20, Theorem 39].

Below, we prove the following complexity characterisation of alphabets of prime cardinality.

▶ **Theorem 46.** *Let $|\Sigma| = p$ for some $p \in \mathbb{P}$. Then $L_q \notin \bigoplus\mathbf{SAC}^0$ and $\Sigma^* \setminus L_q \notin \bigoplus\mathbf{SAC}^0$.*

By translating prime-cardinality-alphabets into finite fields, we may use the tools of field theory. In this section, we collect facts about finite fields which we require to prove Theorem 46.

▶ **Lemma 43.** *Let $\mathbb{F}$ be a finite field.*

1. *By prime decomposition, $\mathbb{F}$ has prime characteristic.*
2. *$\mathbb{F}$ has order p^n for some $p \in \mathbb{P}$. [7, 33.2, 33.10]*
3. *If $\mathbb{F}$ has order p^n then $\mathbb{F}$ has characteristic p. [6, Sec. 14.3]*
4. *For every $p \in \mathbb{P}$ and $n \in \mathbb{N}$, there is one field up to isomorphism of order p^n [7, 33.12]. This field has a subfield of order p, the* prime subfield.
5. *All functions from $\mathbb{F}$ to itself are polynomial functions. [7, Exercises 22: 31.c.]*
6. *If $\mathbb{F}$ has order p^n and $x \in \mathbb{F}$ then $x^{p^m} = x^{p^{m+n}}$ for all $m \in \mathbb{N}$. In particular, $x = x^{p^n}$, since the multiplicative subgroup of $\mathbb{F}$ has order $p^n - 1$. [6, p. 550]*

If $p \in \mathbb{P}$ and $n \in \mathbb{N}$, let $\mathbb{F}_{p^n}$ denote the (unique up to isomorphism) field of order p^n.

▶ **Lemma 44.** *Suppose $\varphi\colon \mathbb{F}_{p^n} \to \mathbb{F}_p$ is linear, i.e. $\varphi(x+y) = \varphi(x)+\varphi(y)$ and $\varphi(ax) = a\varphi(x)$ for all $x, y \in \mathbb{F}_{p^n}$ and $a \in \mathbb{F}_p$. Then there exist $a_1, \ldots, a_n \in \mathbb{F}_{p^n}$ for which $\varphi(x) = \sum_{i=1}^{n} a_i x^{p^i}$. In fact, every linear function from $\mathbb{F}_{p^n}$ to $\mathbb{F}_p$ arises in this way.*

For a proof and related details on *field traces*, see for instance [24, Theorem 2.24] and [24, Chapter 2.3]. In fact, their proof shows that there exists *one* $z \in \mathbb{F}_{p^n}$ for which $a_i = z^{p^i}$. We now give a characterisation of $\bigoplus\mathbf{SAC}^0$ in terms of finite fields and their operations. This characterisation is akin to that of $\mathbf{SAC}^0$ in Lemma 38 in terms of propositional formulas.

▶ **Proposition 45.** *Let $\phi_n\colon \mathbb{F}_p^n \to \mathbb{F}_{p^n}$ be a linear isomorphism of vector spaces over $\mathbb{F}_p$, and suppose $L \subset \Sigma^*$ is $\bigoplus\mathbf{SAC}^0$-recognisable. Then there exists a family of polynomials $(f_n)_{n\in\mathbb{N}}$ with $f_n\colon \mathbb{F}_{p^n} \to \mathbb{F}_p$ for which*

$$x \in L \cap \Sigma^n \iff (f_n \circ \phi_n)(x) \neq 0$$

and for which there exists $\ell \in \mathbb{N}$ such that for all $n \in \mathbb{N}$ we have $\deg(f_n) \leq p^n - p^{n-\ell}$.

For the proof, we direct the reader to [4]. We now combine the field-theoretic tools above to prove the main theorem of this section. Recall that $\mathbb{P}$ denotes the set of primes.

▶ **Theorem 46.** *Let $q \in (0, 1/2)$ and $|\Sigma| \in \mathbb{P}$. Then $L_q \notin \bigoplus\mathbf{SAC}^0$ and $\Sigma^* \setminus L_q \notin \bigoplus\mathbf{SAC}^0$.*

Proof. Suppose some circuit recognises L_q. By Proposition 45, there exists a family of polynomials (f_n) and $\ell \in \mathbb{N}$ for which $x \in L_q$ if and only if $(f_n \circ \phi_n)(x) \neq 0$ and $\deg(f_n) \leq p^n - p^{n-\ell}$. So, the number of roots of f_n – and, hence, the number of words not in L_q – is bounded above by $p^n - p^{n-\ell}$, so the cardinality of L_q is in $\Omega(p^n)$, contradicting Corollary 19.

For $\Sigma^* \setminus L_q$, note that the circuit can be augmented by a constant number of layers to flip the output of $f_n \circ \phi_n$ for any n (note that $\operatorname{ran}(f_n \circ \phi_n) \subseteq \mathbb{F}_p$). If $a_x = (f_n \circ \phi_n)(x) \neq 0$ then use Lemma 43 Item 6 to see that $a_x^p = a_x$; thus $a_x^{p-1} = 1$, and so the polynomial $\theta(x) = 1 - x^{p-1}$ satisfies $\theta(x) = 0 \iff a_x \neq 0$. As p is fixed, θ can be computed by a constant-depth circuit, which we may append to any $\bigoplus\mathbf{SAC}^0$-circuit recognising L_q to recognise $\Sigma^* \setminus L_q$. Since the former does not exist, neither does the latter. ◀

5 Open Questions

In this paper, we proved multiple results on the complexity of the measure A_{Ne} via the proxy family of sets $\{ L_q : q \in (0, 1/2) \}$. In particular, we showed that L_q is complicated from the viewpoint of pushdown automata (Theorems 34 and 35 and Corollary 36), and even certain Boolean circuits cannot recognise L_q, nor its complement (Theorems 39 and 46). We also proved the Shannon effect for A_{Ne} (Theorem 22). Pressing open questions pertain to refining these results on L_q – and, ultimately, to understanding the measure A_{Ne} even better.

In Section 4.2, and in Theorem 46 in particular, we considered alphabets of prime cardinality; however, our proof uses a non-standard definition of $\bigoplus\mathbf{SAC}^0$. We wonder:

▶ **Question 47.** Do the results from Theorem 46 apply to arbitrary alphabets using the definition of $\bigoplus\mathbf{SAC}^0$ given in Definition 40?

Finally, it is clear by definition that $A_{Ne}(x) \leq A_N(x)$ for all $x \in \Sigma^*$, for any finite alphabet Σ. This allows the extension of certain results from A_{Ne} to A_N. For instance, we note here that Theorems 33, 35, and 39 also apply to A_N since $A_{Ne}(x) \leq A_N(x)$. However, whether the equality $A_{Ne}(x) = A_N(x)$ holds in general remains the cardinal open question to fully understand the impact of exactness in Definition 7 compared to Definition 4.

▶ **Question 48.** Let $\Sigma = \{0, 1\}$. Does there exist $x \in \Sigma^*$ for which $A_{Ne}(x) < A_N(x)$?

Broadly, it is our hope that a better understanding of A_{Ne}, via proxies such as the L_q sets or otherwise, will lead to a better understanding of A_N, and (non-deterministic) automatic complexity in general. Similar to the comparison between regular languages being characterised by both NFAs and DFAs, a proof of $A_N = A_{Ne}$ would simplify computing the non-deterministic automatic complexity while preserving its naturalness as a measure of complexity. Conversely, $A_N \neq A_{Ne}$ would show that non-deterministic automatic complexity is dependent on the actual computation path, which would also be of philosophical interest.

References

1 Scott Aaronson. Complexity Zoo, 2025. URL: `https://complexityzoo.net/Complexity_Zoo` [cited 18 Apr 2025].

2 Achilles A. Beros, Bjørn Kjos-Hanssen, and Daylan Kaui Yogi. Planar digraphs for automatic complexity. In *Theory and applications of models of computation*, volume 11436 of *Lecture Notes in Comput. Sci.*, pages 59–73. Springer, Cham, 2019. `doi:10.1007/978-3-030-14812-6_5`.

3 Allan Borodin, Stephen A. Cook, Patrick W. Dymond, Walter L. Ruzzo, and Martin Tompa. Two applications of inductive counting for complementation problems. *SIAM J. Comput.*, 18(3):559–578, 1989. doi:10.1137/0218038.

4 Joey Chen, Bjørn Kjos-Hanssen, Ivan Koswara, Linus Richter, and Frank Stephan. Languages of Words of Low Automatic Complexity Are Hard to Compute, 2025. arXiv:2510.07696.

5 Rodney G. Downey and Denis R. Hirschfeldt. *Algorithmic randomness and complexity*. Theory and Applications of Computability. Springer, New York, 2010. doi:10.1007/978-0-387-68441-3.

6 David S. Dummit and Richard M. Foote. *Abstract algebra*. John Wiley & Sons, Inc., Hoboken, NJ, third edition, 2004.

7 John B Fraleigh. *A first course in abstract algebra*. Pearson Education, Philadelphia, PA, 7th edition, 2003.

8 Anna Gál and Avi Wigderson. Boolean complexity classes vs. their arithmetic analogs. *Random Struct. Algorithms*, 9(1-2):99–111, 1996. doi:10.1002/(sici)1098-2418(199608/09)9:1/2<99::aid-rsa7>3.3.co;2-o.

9 Seymour Ginsburg. *The mathematical theory of context-free languages*. McGraw-Hill Book Co., New York-London-Sydney, 1966.

10 Sheila A. Greibach. A New Normal-Form Theorem for Context-Free Phrase Structure Grammars. *J. ACM*, 12(1):42–52, January 1965. doi:10.1145/321250.321254.

11 John E Hopcroft, Rajeev Motwani, and Jeffrey D Ullman. *Introduction to automata theory, languages, and computation*. Pearson, Upper Saddle River, NJ, 3 edition, June 2006.

12 Kayleigh Hyde. Nondeterminstic Finite State Complexity. Master's thesis, University of Hawai'i at Mānoa, 2013. URL: http://hdl.handle.net/10125/29507.

13 Kayleigh K. Hyde and Bjørn Kjos-Hanssen. Nondeterministic automatic complexity of overlap-free and almost square-free words. *Electron. J. Combin.*, 22(3):Paper 3.22, 18, 2015. doi:10.37236/4851.

14 Bakhadyr Khoussainov and Anil Nerode. *Automata theory and its applications*, volume 21 of *Progress in Computer Science and Applied Logic*. Birkhäuser Boston, Inc., Boston, MA, 2001. doi:10.1007/978-1-4612-0171-7.

15 Bjørn Kjos-Hanssen. On the complexity of automatic complexity. *Theory Comput. Syst.*, 61(4):1427–1439, 2017. doi:10.1007/s00224-017-9795-4.

16 Bjørn Kjos-Hanssen. Automatic complexity of shift register sequences. *Discrete Mathematics*, 341(9):2409–2417, 2018. doi:10.1016/j.disc.2018.05.015.

17 Bjørn Kjos-Hanssen. Automatic complexity of Fibonacci and Tribonacci words. *Discrete Applied Mathematics*, 289:446–454, 2021. doi:10.1016/j.dam.2020.10.014.

18 Bjørn Kjos-Hanssen. An incompressibility theorem for automatic complexity. *Forum Math. Sigma*, 9:e62, 7, 2021. doi:10.1017/fms.2021.58.

19 Bjørn Kjos-Hanssen. *Automatic complexity—a computable measure of irregularity*, volume 12 of *De Gruyter Series in Logic and its Applications*. De Gruyter, Berlin, [2024] ©2024. doi:10.1515/9783110774870.

20 Bjørn Kjos-Hanssen. Maximal automatic complexity and context-free languages. In *Aspects of computation and automata theory with applications*, volume 42 of *Lect. Notes Ser. Inst. Math. Sci. Natl. Univ. Singap.*, pages 335–352. World Sci. Publ., Hackensack, NJ, [2024] ©2024. doi:10.1142/9789811278631_0013.

21 A. N. Kolmogorov. Three approaches to the definition of the concept "quantity of information". *Problemy Peredači Informacii*, 1(vyp. 1):3–11, 1965. URL: https://www.mathnet.ru/eng/ppi68.

22 L. A. Levin. Laws of Information Conservation (Nongrowth) and Aspects of the Foundation of Probability Theory. *Problems Inform. Transmission*, 10(3):206–210, 1974. URL: https://www.mathnet.ru/eng/ppi1039.

23 Leonid A. Levin. Some theorems on the algorithmic approach to probability theory and information theory: (1971 dissertation directed by a.n. kolmogorov). *Annals of Pure and*

Applied Logic, 162(3):224–235, 2010. Special Issue: Dedicated to Nikolai Alexandrovich Shanin on the occasion of his 90th birthday. doi:10.1016/j.apal.2010.09.007.

24 Rudolf Lidl and Harald Niederreiter. *Finite fields*, volume 20 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, second edition, 1997. With a foreword by P. M. Cohn. doi:10.1017/CBO9780511525926.

25 O. B. Lupanov. The synthesis of contact circuits. *Dokl. Akad. Nauk SSSR (N.S.)*, 119:23–26, 1958.

26 O. B. Lupanov. The schemes of functional elements with delays. *Problemy Kibernet.*, (23):43–81, 303, 1970.

27 R. C. Lyndon and M. P. Schützenberger. The equation $a^M = b^N c^P$ in a free group. *Michigan Math. J.*, 9:289–298, 1962. URL: http://projecteuclid.org/euclid.mmj/1028998766.

28 Per Martin-Löf. The definition of random sequences. *Information and Control*, 9:602–619, 1966. doi:10.1016/S0019-9958(66)80018-9.

29 Jaban Meher and M. Ram Murty. Ramanujan's proof of Bertrand's postulate. *Amer. Math. Monthly*, 120(7):650–653, 2013. doi:10.4169/amer.math.monthly.120.07.650.

30 Jakub Radoszewski, Wojciech Rytter, and Tomasz Waleń. Faster Algorithms for Ranking/Unranking Bordered and Unbordered Words. In Zsuzsanna Lipták, Edleno Moura, Karina Figueroa, and Ricardo Baeza-Yates, editors, *String Processing and Information Retrieval*, pages 257–271, Cham, 2025. Springer Nature Switzerland. doi:10.1007/978-3-031-72200-4_20.

31 A. A. Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical notes of the Academy of Sciences of the USSR*, 41(4):333–338, April 1987. doi:10.1007/BF01137685.

32 Maria José Serna. Asymptotical behaviour of some non-uniform measures. *RAIRO - Theoretical Informatics and Applications - Informatique Théorique et Applications*, 23(3):281–293, 1989. doi:10.1051/ita/1989230302811.

33 Jeffrey Shallit. *A Second Course in Formal Languages and Automata Theory*. Cambridge University Press, USA, first edition, 2008. doi:10.1017/CBO9780511808876.

34 Jeffrey Shallit and Yuri Breitbart. Automaticity I: Properties of a Measure of Descriptional Complexity. *Journal of Computer and System Sciences*, 53(1):10–25, 1996. doi:10.1006/jcss.1996.0046.

35 Jeffrey Shallit and Ming-Wei Wang. Automatic complexity of strings. *J. Autom. Lang. Comb.*, 6(4):537–554, 2001. 2nd Workshop on Descriptional Complexity of Automata, Grammars and Related Structures (London, ON, 2000). doi:10.25596/jalc-2001-537.

36 Claude. E. Shannon. The synthesis of two-terminal switching circuits. *The Bell System Technical Journal*, 28(1):59–98, 1949. doi:10.1002/j.1538-7305.1949.tb03624.x.

37 Michael Sipser. A complexity theoretic approach to randomness. In *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing*, STOC '83, pages 330–335, New York, NY, USA, 1983. Association for Computing Machinery. doi:10.1145/800061.808762.

38 R. Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, STOC '87, pages 77–82, New York, NY, USA, 1987. Association for Computing Machinery. doi:10.1145/28395.28404.

39 R. Smolensky. On representations by low-degree polynomials. In *Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science*, pages 130–138, 1993. doi:10.1109/SFCS.1993.366874.

40 Ray J Solomonoff. A preliminary report on a general theory of inductive inference, February 1960. URL: https://raysolomonoff.com/publications/z138.pdf.

41 Ray J Solomonoff. A formal theory of inductive inference. Part I. *Information and control*, 7(1):1–22, 1964. doi:10.1016/S0019-9958(64)90223-2.

42 Ray J Solomonoff. A formal theory of inductive inference. Part II. *Information and control*, 7(2):224–254, 1964. doi:10.1016/S0019-9958(64)90131-7.

43 I. H. Sudborough. On the tape complexity of deterministic context-free languages. *J. Assoc. Comput. Mach.*, 25(3):405–414, 1978. doi:10.1145/322077.322083.

44 H. Venkateswaran. Properties that characterize LOGCFL. *J. Comput. System Sci.*, 43(2):380–404, 1991. doi:10.1016/0022-0000(91)90020-6.

45 I. Wegener. *The Complexity of Boolean Functions*. Wiley Teubner on Applicable Theory in Computer Science. Wiley, 1987.