On the Degree of Extension of Some Models Defining Non-Regular Languages*

Victor Mitrana
Department of Information Systems,
Universidad Politécnica de Madrid
Calle Alan Turing s/n (Carretera de Valencia Km 7),
28031 Madrid, Spain
and
National Institute of R&D for Biological Sciences,
296 Independenței Bd., 060031, Bucharest. Romania
victor.mitrana@upm.es

Mihaela Păun
National Institute for R&D for Biological Sciences
296 Independenței Bd., 060031, Bucharest, Romania
mihaela.paun@incdsb.ro

This work is a survey of the main results reported for the degree of extension of two models defining non-regular languages, namely the context-free grammar and the extended automaton over groups. More precisely, we recall the main results regarding the degree on non-regularity of a context-free grammar as well as the degree of extension of finite automata over groups. Finally, we consider a similar measure for the finite automata with translucent letters and present some preliminary results. This measure could be considered for many mechanisms that extend a less expressive one.

1 Introduction

Language defining models play a central role in formal language theory, and in theoretical computer science. There have been defined very many such models with various motivations depending on the specific problems to be solved. In this work, we restrict ourselves to the most well-known devices: Chomsky generative grammars and finite automata. Regular languages are classically represented by: regular or right-linear grammars, many variants of finite automata, regular expressions, logical or algebraic formalisms. Due to their limited expressiveness, some of these models have been extended to more complex models such that the old model is just a particular case of the extended one. For instance, context-free grammars are natural extensions of regular or right-linear grammars, finite automata with valences [15], jumping automata [29], automata with translucent letters [31] are extensions of finite automata able to accept non-regular languages, etc. In their turn, context-free languages are classically represented by: context-free grammars, pushdown automata, logical and algebraic formalisms. Mainly, by the same reason as above, there have been proposed various extensions like context-sensitive grammars, grammars with regulated rewriting [11], etc.

In this work, we recall a measure for evaluating the degree of extension of a two such models, namely the context-free grammar as an extension of regular grammar, and the extended finite automaton over groups as an extension of the finite automaton. A similar measure is also considered for finite automata with translucent letters. Roughly speaking, this measure is defined as follows:
(i) by counting the maximal number of non-regular rules used in a derivation [6],
(ii) by evaluating the group memory used by the extended automata over groups [1].
(iii) by counting the number of jumping moves used by a finite automata with translucent letters.

*This work was performed through the Core Program within the National Research, Development and Innovation Plan 2022-2027, carried out with the support of MRID, project no. 23020101(SIA-PRO), contract no 7N/2022, and project no. 23020301(SAFE-MAPS), contract no 7N/2022.

Zs. Gazdag, Sz. Iván, G. Kovásznai (Eds.): 16th International Conference on Automata and Formal Languages (AFL 2023)
EPTCS 386, 2023, pp. 12–24, doi:10.4204/EPTCS.386.3

As far as the first measure is concerned, it is worth noting that similar investigations have been reported from the time of introducing the classes of regular and context-free languages. Here are several results giving sufficient conditions for a context-free grammar and a context-sensitive grammar to generate a regular and context-free language, respectively:

- Each context-free grammar that is not self-embedding generates a regular language [8].
- An arbitrary grammar in which no terminal is used as context and every rule generates at least one terminal, generates a context-free language [19].
- An arbitrary grammar generates a context-free language if the left side of every rule contains only one nonterminal, with terminal words as the only context [5].
- If every rule of an arbitrary grammar has as left context a word of terminal symbols at least as long as the right context, then the language generated is context-free [5].
- A grammar which has a partial ordering on its symbols, such that in every rule of the grammar every symbol on the left side is "smaller" than some symbol on the right generates a context-free language [21].
- In a grammar, the sets of terminal words generated by "one-way" and "two-ways" derivations are context-free [26, 27, 28] and [13].
- An arbitrary grammar such that in each of its non-context-free rules, the right side contains a word of terminals longer than any terminal word appearing between two nonterminals in the left side, generates a context-free language [2].

As one can see, some of the above conditions can be immediately checked, namely by examining the rules. In many cases, the complexity of a device generating a language is a function with nonnegative integer values: rational index [4], initial index [16], index of a context-free grammar [7], height of derivational trees [10], etc. Similar approaches have been reported in [6] for context-free and context-sensitive grammars, and [1] for extended automata over groups. In the sequel, we survey the most important results of these papers.

2 Preliminaries

We assume the reader is familiar with the basic definitions and concepts in formal language and automata theory and combinatorial algebra such as monoids and groups, presentations and generating sets, etc. For further details, we refer to [33] (for formal languages and automata theory), and [24, 32] (for combinatorial algebra).

We denote by $\mathbb{N}$ the set of nonnegative integers. An alphabet is a finite set of letters or symbols. For a set A we denote by $card(A)$ the cardinality of A. For a finite set V, called alphabet, we denote by $(V^*, \cdot, \varepsilon)$ the free monoid generated by V under the operation of concatenation with the neutral element ε. The elements of V^* are called words and ε is the empty word. For a word $x \in V^*$ we denote by $alph(x)$ the smallest subset of V such that $x \in alph^*(x)$. Given a set A, we denote by $\mathscr{P}_f(A)$ the family of all finite subsets of A. The free semigroup generated by V with concatenation is denoted by V^+. The length of $x \in V^*$ is denoted by $|x|$, $|x|_a$ is the number of occurrences of a in x, whereas $|x|_B$ is the number of occurrences of symbols $B \subseteq V$ in x. For a word $w = a_1 a_2 \ldots a_n$, $n \geq 1$, $a_i \in V$ for all $1 \leq i \leq n$, we write $\widetilde{w} = a_n \ldots a_2 a_1$.

By regular grammar we mean a grammar that is right-linear, hence a regular rule should be understood as a right-linear rule of one of the forms $A \to wB$, and $A \to w$, with A, B being nonterminals and w

being a word of terminals, possibly the empty word. In what follows we also use the regular expressions for defining regular languages. A context-free grammar $G = (N,T,S,P)$ is a *reduced* grammar if for any $X \in N$ we have the derivations $S \Longrightarrow^* \alpha X \beta$, for some $\alpha, \beta \in (N \cup T)^*$ (X is said to be *accessible*), and $X \Longrightarrow^* u$, with $u \in T^*$ (X is said to be *co-accessible*). A context-free grammar is *proper* if it has no λ-production (i.e. $X \to \lambda$, $X \in N$) and no chain-production (i.e., $X \to Y$, $X,Y \in N$). It is known that for every context-free grammar (which does not generate λ) there exists an equivalent proper and reduced context-free grammar.

For an arbitrary grammar $G = (N,T,S,P)$ we denote by

- $G(A) = (N,T,A,P)$, $A \in N$ the grammar in which the axiom S was replaced by another nonterminal, A.
- $G_{reg} = (N,T,S,P_{reg})$ the grammar obtained from G by considering the set $P_{reg} \subseteq P$ of regular productions of P only.
- $G_{cf} = (N,T,S,P_{cf})$ the grammar obtained from G by considering the set $P_{cf} \subseteq P$ of context-free productions of P only.

We denote by REG and CF the class of regular and context-free languages, respectively.

A finite *multiset* over a finite set A is a mapping $\sigma : A \longrightarrow \mathbb{N}$; $\sigma(a)$ expresses the number of copies of $a \in A$ in the multiset σ. In what follows, a multiset containing the elements $b_1, b_2, \ldots, b_r$, any element possibly being repeated one or more times in the sequence, will be denoted by $\langle b_1, b_2, \ldots, b_r \rangle$. Each multiset σ over a set A of cardinality n may also be viewed as an array of size n with non-negative entries.

For two functions $f,g : \mathbb{N} \longrightarrow \mathbb{N}$ we say that $f(n) \in \mathscr{O}(g(n))$ iff there is a constant $c > 0$ and $n_0 \geq 1$ such that $f(n) \leq cg(n)$ for all $n \geq n_0$. Equivalently, $f(n) \in \mathscr{O}(g(n))$ iff $\limsup\limits_{n\to\infty} \dfrac{f(n)}{g(n)} < \infty$. Following [3], we say that $f(n) \in \Omega(g(n))$ iff $\limsup\limits_{n\to\infty} \dfrac{f(n)}{g(n)} > 0$. Furthermore, we say that $f(n) \in o(g(n))$ iff $\lim\limits_{n\to\infty} \dfrac{f(n)}{g(n)} = 0$.

3 The degree of non-regularity

Given a context-free grammar $G = (N,T,S,P)$ a derivation step in G by using the rule $r \in P$ is denoted by $\Rightarrow_r$. For a derivation in G

$$D = (S \Rightarrow_{r_1} w_1 \Rightarrow_{r_2} w_2 \cdots \Rightarrow_{r_m} w_m = w),$$

where $w \in T^*$ and $r_i \in P$ for $1 \leq i \leq m$, we define the degree of non-regularity of w with respect to D by

$$dnreg_G(w,D) = \text{card}\{i \mid r_i \notin P_{reg}, 1 \leq i \leq m\}.$$

Less formally, $dnreg_G(w,D)$ is the number of non-regular rules applied in the derivation D of w in the grammar G. The degree of non-regularity of a terminal word w with respect to the grammar G is

$$dnreg_G(w) = \begin{cases} min\{dnreg_G(w,D) \mid D \text{ is a derivation of } w \text{ in } G\}, \\ 0, \text{ if } w \notin L(G). \end{cases}$$

In other words, the degree of non-regularity of a word with respect to a grammar is computed by taking into consideration the "least non-regular derivation" if there is one.

The degree of non-regularity of a context-free grammar G as above is a mapping from $\mathbb{N}$ to $\mathbb{N}$ defined by

$$dnreg_G(n) = max\{dnreg_G(w) \mid |w| = n, w \in T^+\}.$$

As one can see, the most "non-regular" word of each length is considered.

For a function $f : \mathbb{N} \to \mathbb{N}$ we now define the complexity class

$$\begin{aligned} DNREG(f(n)) &= \{L \mid L = L(G) \text{ for some context-free grammar } G \text{ and} \\ &\quad dnreg_G(n) \in \mathscr{O}(f(n))\}. \end{aligned}$$

Otherwise stated, a language has the degree of non-regularity $f(n)$ if and only if it belongs to $DNREG(f(n))$.

In the sequel we recall the main results about the degree on non-regularity. A simple remark turns out to be useful. If G is an arbitrary context-free grammar and G_1 is the reduced grammar obtained from G, $dnreg_G(n) = dnreg_{G_1}(n)$ holds for all n, because none of the removed productions contributes in any derivation of a terminal word in G. By several considerations, a similar situation holds if G is not proper and G_1 is the proper grammar obtained from G. Therefore, the context-free grammars considered in the sequel are reduced and proper.

A context-free grammar is said to be in *quasi normal form* if all its rules of are of the following forms:

(i) $A \to a$, where a is a terminal,

(ii) $A \to aB$, where a is a terminal and B is a nonterminal,

(iii) $A \to \alpha$, where α is a word of nonterminals of length at least 2.

Proposition 1 *For every context-free grammar G there exists an equivalent context-free grammar G' in quasi normal form such that $dnreg_G(n) = dnreg_{G'}(n)$ for all n.*

If the length of α is exactly 2 in every rule $A \to \alpha$ of a grammar in quasi normal form, we say that the grammar is in *quasi Chomsky normal form*. If we have a grammar in quasi normal form, each rule $A \to \alpha$, with $|\alpha| \geq 3$ can be replaced by a sequence of rules with the right-hand side of length 2. Hence, each grammar in quasi normal can be replaced by an equivalent grammar in quasi Chomsky normal form at a price of a constant number of times higher degree of non-regularity.

Let G be a context-free grammar and c be a positive integer; we define the language

$$L(G, \leq c) = \{w \in L(G) \mid dnreg_G(w) \leq c\}.$$

Clearly, if $dnreg_G(n) \leq c$ for any $n \geq 1$, then $L(G, \leq c) = L(G)$ holds.

Theorem 1 *$DNREG(1) = REG$. A language generated by a context-free grammar is finitely-non-regular if and only if it is regular.*

Theorem 2 *For any given context-free grammar G and a positive integer c, one can algorithmically decide whether $dnreg_G(n) \leq c$.*

It is worth mentioning that $dnreg_G(n) \leq c$, for a context-free grammar G and a positive integer c, implies that $L(G)$ is regular. However, if $dnreg_G(n) > c$, then nothing can be said about the regularity of $L(G)$. Even more, if $L(G)$ is regular, it does not generally follow that $dnreg_G(n) \in \mathscr{O}(1)$.

Theorem 3 *Given an unambiguous context-free grammar G, one can algorithmically decide whether* $dnreg_G(n) \in \mathscr{O}(1)$.

The problem turns out to be undecidable even for arbitrary linear context-free grammars. It is worth mentioning that this problem is not equivalent to the problem of whether or not a given context-free grammar generates a regular language, which is known to be undecidable.

Theorem 4 *Given a linear context-free grammar G, it is undecidable whether* $dnreg_G(n) \in \mathscr{O}(1)$.

If L is a language generated by a context-free grammar such that every derivation of each word $w \in L$ of length n needs a number of non-regular productions at most linear in n, then the language is said to be "at most linearly non-regular".

Theorem 5 $CF \subseteq DNREG(n)$. *Every context-free language is at most linearly-non-regular.*

The next result gives an evaluation of the degree of non-regularity of unambiguous context-free grammar generating a non-regular language.

Theorem 6 *Let G be an unambiguous context-free grammar generating a non-regular language. Then* $dnreg_G(n) \in \Omega(n)$.

In [6] one defines a complexity measure on pushdown automata which is related, to some extent, to the pushdown space complexity of languages introduced in [17].

Let $\Gamma = (Q, V, U, \delta, q_0, Z_0, F)$ be a pushdown automaton with the set of states Q, the input alphabet V, the stack alphabet U, the transition mapping δ, the initial state q_0, the initial stack symbol Z_0 and the set of accepting states F. We say that a transition $(s, \alpha) \in \delta(q, a, A)$, with $q, s \in Q$, $a \in V \cup \{\lambda\}$, $A \in U$, $\alpha \in U^*$, is a *push move*, if $|\alpha| \geq 2$, it is a *pop move* if $\alpha = \lambda$, and it is a *neutral move* if $\alpha \in U$. In [6] one defines the *push complexity* of a language as the number of push moves needed by a pushdown automaton to accept that language. Let $w \in V^*$ and

$$C : (q_o, w, Z_0) \vdash^* (a, \lambda, \lambda)$$

be a computation in Γ accepting the input word w with empty stack, see, e.g., Chapter 6 in [25]. Then, the number of push moves in the computation C, is denoted by $push_\Gamma(w, C)$. Furthermore, for every word $w \in V^*$ we define

$$push_\Gamma(w) = \begin{cases} min\{push_\Gamma(w, C) \mid \text{C is a computation accepting } w\}, \\ \text{if } w \text{ is accepted by } \Gamma, \\ 0, \text{ if } w \text{ is not accepted by } \Gamma. \end{cases}$$

We now define the function $push_\Gamma : \mathbb{N} \longrightarrow \mathbb{N}$ by

$$push_\Gamma(n) = max\{push_\Gamma(w) \mid |w| = n\}.$$

This function is called the push complexity of Γ. Note that if a pushdown automaton has stack space complexity $f(n)$, its push complexity is a function $g(n)$ such that $f(n) \in \mathscr{O}(g(n))$. As for the degree of non-regularity we set

$$PUSH_\lambda(f(n)) = \{L \mid L = L(\Gamma) \text{ for some pushdown automaton } \Gamma \text{ accepting with empty stack and } push_\Gamma(n) \in \mathscr{O}(f(n))\}.$$

Analogously, we define

$$PUSH_f(f(n)) = \{L \mid L = L(\Gamma) \text{ for some pushdown automaton } \Gamma \text{ accepting with final states and } push_\Gamma(n) \in \mathscr{O}(f(n))\}.$$

It is known how a pushdown automaton accepting with final states can be transformed into an equivalent one accepting with empty stack (Theorem 5.1 in [22]), and conversely (Theorem 5.2 in [22]). By these constructions the equality $PUSH_\lambda(f(n)) = PUSH_f(f(n))$.

The *push* measure will turn out to be very useful for our further investigation. Indeed, we claim that the two classes of languages $PUSH_\lambda(f(n))$ and $DNREG(f(n))$ are identical.

Theorem 7 *Let L be a deterministic context-free language that is not regular. If $L \in DNREG(f(n))$, then $f(n) \in \Omega(n)$.*

Theorem 8 *Both families $DNREG(\sqrt{n})$ and $DNREG(\log n)$ contain non-regular languages.*

A very natural problem arises: Are there other sublinear functions f such that $DNREG(f)$ does contain non-regular languages? The problem of finding other sublinear functions f such that $DNREG(f)$ contains non-regular languages is of interest from a computational point of view as well. By the next theorem, functions like $\log_p(n)$, for some $p \geq 2$, are of a special interest.

Theorem 9 *Let G be a context-free grammar in quasi Chomsky normal form generating a non-regular language such that $dnreg_G(n) \leq f(n)$. Then $L(G)$ is recognizable in $\mathscr{O}(n \cdot p^{f(n)})$ time, where p is the number of nonterminals of G.*

4 The degree of extension of finite automata over groups

Let $(M, \cdot, 1)$ be a group under an operation denoted by $\cdot$ with the neutral element denoted by 1. An extended finite automaton (EFA shortly) A over the group $(M, \cdot, 1)$ is defined formally as follows. $A = (Q, V, M, f, q_0, F)$, where Q, V, q_0, F have the same meaning as in a usual finite automaton, namely the set of states, the input alphabet, the initial state and the set of final states, respectively, and $f : Q \times V \longrightarrow \mathscr{P}_f(Q \times M)$. This is actually the extension of finite automata with additive or multiplicative valences to an arbitrary group, see [15] and the references therein.

This type of automaton can be viewed as a finite automaton having a register in which any element of M can be stored, let us call it "group memory". The relation $(q, m) \in f(s, a),\ q, s \in Q,\ a \in V,\ m \in M$ means that the automaton A changes its current state s into q, by reading the symbol a on the input tape, and stores $x \cdot m$ in the register, where x is the former content of the register. The initial value stored in the register is 1.

We shall use the notation

$$(q, aw, m) \models_A (s, w, mr) \text{ iff } (s, r) \in f(q, a)$$

for all $s, q \in Q\ a \in V,\ m, r \in M$. The reflexive and transitive closure of the relation $\models_A$ is denoted by $\models_A^*$. Sometimes, the subscript identifying the automaton will be omitted when it is self-understood.

The word $x \in V^*$ is accepted by the automaton A if, and only if, there is a final state q such that $(q_0, x, 1) \models^* (q, \varepsilon, 1)$. In other words, a string is accepted if the automaton completely reads the string and reaches a final state with the content of the register being the neutral element of M. The language accepted by an EFA A over a group as above is denoted by $L(A)$.

The following simple observation will be useful in what follows. If L is a language accepted by an EFA over some group M, there exists a finitely generated subgroup N of M such that L is accepted by an EFA over N. Indeed, since the EFA over some group has finitely many transitions, only finitely many elements of the group can be associated with these transitions. Consequently, the register can only ever hold values in the subgroup of the initial group generated by these elements, so it suffices to view the automaton as an EFA over this subgroup.

It is clear that some words in the language accepted by an EFA over a group can be accepted by computations containing "non-regular transitions", that is transitions that change the contents of the group memory. The use of these transitions can make EFA more powerful than

nite automata. A very simple example is a finite automaton that accepts the language $\{a^n b^m \mid n,m \geq 1\}$. If we extend this automaton such that each transition reading an a add the value 1 to its register and each transition reading a b subtracts 1 from the register, the new automaton is an EFA over the additive group of integers that accepts the non-regular language $\{a^n b^n \mid n \geq 1\}$. Consequently, EFA over groups are able to accept non-regular languages or even not context-free languages, see, e.g.,[12]. In the remainder of the present work we study "how much" group memory, defined as the number of non-regular transitions, needs an EFA for accepting a non-regular language.

Given an EFA $A = (Q,V,M,f,q_0,F)$ over a group $(M,\cdot,1)$, $w \in L(A)$, and a computation

$$C_A(w) : (q_0,w_0,m_0) \models_A (q_1,w_1,m_1) \models_A (q_2,w_2,m_2) \models_A \ldots \models_A (q_s,\varepsilon,m_s),$$

for some $s \geq 1$, where $w_0 = w$, $m_0 = m_s = 1$, we define the multiset $E(C_A(w)) = \langle m_i^{-1} m_{i+1} \mid 0 \leq i \leq s-1 \rangle$. In words, $E(C_A(w))$ contains all the elements of M used in the computation $C_A(w)$, each element appearing in exactly the same number of copies as that of times that element was used during the computation. Further on, let $N(C_A(w))$ be the integer defined by

$$N(C_A(w)) = \sum_{x \in M, x \neq 1} E(C_A(w))(x).$$

We now define the *group memory complexity* of the computation of A on the word w by

$$gmc_A(w) = \begin{cases} \min\{N(C_A(w)) \mid C_A(w) \text{ is a computation of } A \text{ on } w\} \\ 0, \text{ if } w \notin L(A). \end{cases}$$

In other words, the group memory complexity of a word with respect to an EFA over M is computed by taking into consideration the "least non-regular computation" if there is one. The group memory complexity of an EFA as above is a mapping from $\mathbb{N}$ to $\mathbb{N}$ defined by

$$gmc_A(n) = \max\{gmc_A(w) \mid |w| = n, w \in V^*\}.$$

As one can see, the most "non-regular" word of each length is considered.

Let A be an arbitrary EFA over some group and c be a positive integer; we define the language $L(A,\leq c) = \{w \in L(A) \mid gmc_A(w) \leq c\}$. Clearly, if $gmc_A(n) \leq c$ for any $n \geq 1$, then $L(A,\leq c) = L(A)$ holds. A natural question arises: Are there EFA accepting non-regular languages with a constant group memory complexity? We give a negative answer to the question through the following result:

Theorem 10 *Given an EFA A and a positive integer c, the language $L(A,\leq c)$ is regular.*

Theorem 11 *Let M be a group such that all its finitely generated subgroups are finite. Then the language accepted by any EFA over M is regular.*

It is worth mentioning that the proof of Theorem 10 is effective, that is a finite automaton recognizing $L(A, \leq c)$ can effectively be constructed. On the other hand, it is known that a pushdown automaton may be seen as an EFA over a free group [12, 9] or an EFA over a polycyclic monoid [8, 18]. Starting from these results we prove the next result.

Theorem 12 *For every EFA A over a free group or a polycyclic monoid and a positive integer c, the problem of whether or not $gmc_A(n) \leq c$ is decidable.*

Are there other classes of groups for which the question in the previous statement is decidable? Yes, actually this happens for every finitely generated abelian group. The reason is a fundamental result in the group theory.

Theorem 13 *Every finitely generated abelian group is the direct product of a finite number of cyclic groups.*

Consequently, the language accepted by an EFA over a finitely generated abelian group is either regular or is a language accepted by an EFA over a group $\mathbb{Z}^k \times H$, where k is a positive integer and H is a finite abelian group. Moreover, $\mathbb{Z}^k$ is the additive group of vectors of size k with integer entries. We now make use of the next result (Theorem 7 in [30]):

Theorem 14 *The language accepted by an EFA over an abelian group can be: (1) regular, (2) accepted by an EFA over by $\mathbb{Z}^k$, (3) accepted by an EFA over the multiplicative group of rationals.*

It follows that if a language L accepted by an EFA over a finitely abelian group is not regular, then there exists a positive integer k such that L is accepted by an EFA over $\mathbb{Z}^k$. We can now state

Theorem 15 *For every EFA A over a finitely generated abelian group and a positive integer c, the problem of whether or not $gmc_A(n) \leq c$ is algorithmically decidable.*

We now provide an EFA over an abelian group that accept non-regular languages and has a sublinear group memory complexity, namely a function in $\mathcal{O}(\sqrt{n})$.

Lemma 1 *There exists an EFA A over $\mathbb{Z} \times \mathbb{Z}_2$ such that $L(A)$ is not regular and $gmc_A(n) \in \mathcal{O}(\sqrt{n})$.*

Inspired by the Goldstine language:

$$G = \{a^{n_1} b a^{n_2} b \ldots a^{n_p} b \mid p \geq 1, n_i \geq 0, \text{ and } n_j \neq j \text{ for some } j, 1 \leq j \leq p\},$$

we define the non-regular language

$$\begin{aligned} L \quad = \quad & \{ba^{i_1} b a^{i_2} b \ldots a^{i_k} b c^m \mid k \geq 1, i_1, i_2, \ldots, i_k > 0, \text{ and} \\ & \text{there exists } 1 \leq j \leq k \text{ such that } i_j \neq j \text{ and } m = \left\{ \begin{array}{l} j - i_j, \text{ if } j > i_j, \\ 1, \text{ if } j < i_j \end{array} \right. \}. \end{aligned}$$

It can be routinely proved that L is not regular.

As it suffices to use Theorem 7 from [30] to simply replace the group $\mathbb{Z} \times \mathbb{Z}_2$ by $\mathbb{Z}$ in the statement of previous lemma, we can state:

Theorem 16 *Let M be a group having at least one infinite cyclic subgroup. There exists an EFA A over M such that $L(A)$ is not regular and $gmc_A(n) \in \mathcal{O}(\sqrt{n})$.*

By using a similar idea to that used in the proof of Lemma 1 we prove the next result, where $\mathbb{F}_2$ is the free group of rank 2.

Lemma 2 *There exists an EFA A over the group $\mathbb{F}_2 \times \mathbb{Z}_2$ such that $L(A)$ is not regular and $gmc_A(n) \in \mathcal{O}(\log n)$.*

As $\mathbb{Z}_2$ is a finite group, we state:

Theorem 17 *There exists an EFA A over the group $\mathbb{F}_2$ such that $L(A)$ is not regular and $gmc_A(n) \in \mathcal{O}(\log n)$.*

We now give an example of a non-regular language such that any EFA over some group that accepts this language has a group memory complexity in $\Omega(n)$.

Theorem 18 *If $L(A) = \{a^n b^n \mid n \geq 1\}$, where A is an EFA over some group, then $gmc_A(n) \in \Omega(n)$.*

Along these lines, two problems remain open here:

1. Are there other abelian or non-abelian groups for which the aforementioned problem is decidable?
2. Give a class of groups $\mathscr{M}$ such that for any group $M \in \mathscr{M}$ and an EFA A over M the problem of whether or not the group memory complexity of A is finite is decidable/undecidable.

We have provided examples of EFA over some groups that accept non-regular languages and have a sublinear group memory complexity, namely a function in $\mathcal{O}(\sqrt{n})$ or $\mathcal{O}(\log n)$. Is it true that for any sublinear integer-valued function f, there is an EFA A over some group M such that $L(A)$ is not regular and $gmc_A(n) \in \mathcal{O}(f(n))$?

Theorem 18 provides a non-regular language such that any EFA over some group that accepts it has a linear group memory complexity.

It is worth mentioning that we have not considered here the deterministic variants of EFA over groups which will be investigated in another work.

5 Jumping complexity of finite automata with translucent letters

A *noneterministic finite automaton with translucent letters* (FATL) is a NFA M as above, such that the transition relation is defined in the following way. First, we define the partial relation $\circlearrowleft$ on the set of all configurations of M: $(s,xay) \circlearrowleft (p,xy)$ iff $p \in \delta(s,a)$, and $\delta(s,b)$ is not defined for any $b \in alph(x)$, $s,p \in Q$, $a,b \in V$, $x \in V^+$, $y \in V^*$. We now write

$$(p,x) \models_M (q,y), \text{ if either } (p,x) \to (q,y) \text{ or } (p,x) \circlearrowleft (q,y).$$

The subscript M is omitted when it is understood from the context.

The language accepted by M is defined by

$$L(M) = \{x \in V^* \mid (q_0,x) \models^* (f,\varepsilon), f \in F\}.$$

We want to stress that the automaton has been introduced in [31], with a slightly different definition. Actually, our definition is an FATL in the normal form in [31] without a marker for the end of the input word. This automaton is also related to the *one way jumping automaton* introduced in [29] with the difference that after each jump it returns to its previous position and does not make shift of the jumped part to the end of the word.

Let M be an FATL; we consider $w \in L(M)$, and the accepting computation in M on the input w:

$$C_M(w) : (q_0,w) \models (q_1,w_1) \models (q_2,w_2) \models \ldots \models (q_m,\varepsilon),$$

with $q_i \in Q$, $1 \leq i \leq m$, and $q_m \in F$. We define

$$jc(C_M(w)) = \{i \geq 1 \mid (q_{i-1}, w_{i-1}) \circlearrowleft (q_i, w_i)\}.$$

In words, $jc(C_M(w))$ contains all the jumping steps in the computation $C_M(w)$.

We now define the *jumping complexity* of the computation of M on the word w by

$$jc_M(w) = \begin{cases} \min\{card(jc(C_M(w))) \mid C_M(w) \text{ is a computation of } M \text{ on } w\} \\ 0, \text{ if } w \notin L(M). \end{cases}$$

In other words, the jumping complexity of a word with respect to M is computed by taking into consideration the "least non-regular computation" if there is one. Equivalently, the jumping complexity of a word with respect to M is the number of jumping steps of a computation with the minimal number of jumping steps.

The jumping complexity of an automaton M as above is a mapping from $\mathbb{N}$ to $\mathbb{N}$ defined by

$$jc_M(n) = \max\{jc_M(w) \mid |w| = n, w \in V^*\}.$$

As one can see, the most "non-regular" word of each length is considered. It is clear that $jc_M(n) \leq n$ for every jumping automaton M as one letter is consumed in every step of a computation.

Let f be a function from $\mathbb{N}$ to $\mathbb{N}$; we define the family of languages

$$LJC((f(n)) = \{L \mid \exists \text{ FATL } M \text{ such that } L = L(M) \text{ and } jc_M(n) \in \mathcal{O}(f(n))\}.$$

Let M be an arbitrary jumping automaton and c be a positive integer; we define the language $L(M, \leq c) = \{w \in L(M) \mid jc_M(w) \leq c\}$. Clearly, if $jc_M(n) \leq c$ for any $n \geq 1$, then $L(M, \leq c) = L(M)$ holds. A natural question arises: Are there FATL accepting non-regular languages with a constant jumping complexity? We give a negative answer to the question through the following result:

Lemma 3 *Given an FATL M and a positive integer c, the language $L(M, \leq c)$ is regular.*

Consequently, we have

Theorem 19 *$JCL(1)$ equals the class of regular languages.*

By Lemma 3, a sufficient condition for an FATL to accept a regular language is to be of constant jumping complexity. Is this condition necessary as well? Were this the case, the problem of deciding whether or not the jumping complexity of a given FATL is constant would be undecidable. Indeed, this decidability problem would be equivalent to decide whether or not the language accepted by an FATL is a regular language, which is not decidable, see Proposition 8 in [31]. As it was expected, the condition is not necessary. It suffices to consider the FATL M defined by the transition mapping:

$$\delta(q_0, b) = q_1, \delta(q_1, b) = q_1, \delta(q_1, c) = q_2, \delta(q_2, a) = q_3,$$

with the final state q_3. The language accepted by this automaton is $L = \{b^n a b^m c \mid n + m \geq 1\} \cup \{b^n c a \mid n \geq 1\}$, which is regular. On the other hand, $jc_M(ab^n c) = n$, for any $n \geq 1$.

However, the decidability status of the following related problem can be partially settled: *Given an FATL M and a positive integer c, is it decidable whether or not $jc_M(n) \leq c$ for all $n \geq 1$?* By modifying the construction in the proof of Lemma 3 we may infer:

Lemma 4 *Given a deterministic FATL M, there exists a deterministic FATL M′ such that* $L(M') = \{w \in L(M) \mid jc_M(w) \geq 1\}$.

This lemma is crucial for the next result.

Theorem 20 *Given a deterministic FATL M and a positive integer c, it is algorithmically decidable whether or not the jumping complexity of M is bounded by c.*

It is worth mentioning that even with this result, the decidability status of the problem "Is the jumping complexity of a deterministic FATL finite?" is still open.

Obviously, each FATL has a jumping complexity which is situated between the constant function and the identity function. In other words, $JCL(n)$ equals the class of all languages accepted by FATL. It remains to investigate what happens between these two extremes. First, we show that there are languages which require a jumping complexity in $\Omega(n)$.

Theorem 21 *If* $L(M) = \{w \mid |w|_a = |w|_b = n \geq 1\}$, *where M is an FATL, then* $jc_M(n) \in \Omega(n)$.

Corollary 1 $JCL(n) \setminus JCL(1) \neq \emptyset$.

6 Final remarks

There are still some attractive problems, in our opinion, that remained unsolved here. One of the most intriguing is the existence of a lower bound for the degree of non-regularity for context-free languages which are not regular. As we have seen, there are context-free languages which are not regular having a sublinear degree of non-regularity. We strongly suspect a more general result: $REG \subset DNREG(f)$, *strict inclusion, for any function* f *that is not a constant*. We mention a few other problems excepting the problem discussed before Theorem 9. Given a context-free grammar G, is it decidable whether or not G has the least degree of non-regularity among all grammars generating $L(G)$?

As far as the degree of extension of finite automata over groups is concerned, we have proved that given an EFA A over a free group, a polycyclic monoid, or a finitely generated abelian group and a constant c, one can algorithmically decide whether or not the group memory complexity of A is bounded by c. Along these lines, two problems remain open here:

1. Are there other abelian or non-abelian groups for which the aforementioned problem is decidable?
2. Give a class of groups $\mathscr{M}$ such that for any group $M \in \mathscr{M}$ and an EFA A over M the problem of whether or not the group memory complexity of A is finite is decidable/undecidable.

We have provided examples of EFA over some groups that accept non-regular languages and have a sublinear group memory complexity, namely a function in $\mathscr{O}(\sqrt{n})$ or $\mathscr{O}(\log n)$. Is it true that for any sublinear integer-valued function f, there is an EFA A over some group M such that $L(A)$ is not regular and $gmc_A(n) \in \mathscr{O}(f(n))$?

Theorem 18 provides a non-regular language such that any EFA over some group that accepts it has a linear group memory complexity. It is worth mentioning that [1] has not considered the deterministic variants of EFA over groups which is to be further investigated.

Some of the above problems remained open as well as regards the jumping complexity of finite automata with translucent letters.

Generally, this could be a measure for investigating the degree of extension of many mechanisms that extend a less expressive one like context-free grammars with regulated rewriting, extended various types of finite automata and tree automata over groups, jumping automata, automata with translucent letters, etc. A first step in this direction has already been done in [14].

References

[1] F. Arroyo, V. Mitrana, A. Păun, M. Păun & J.R. Sánchez Couso (2020): *On the group memory complexity of extended finite automata over groups*. *J. Log. Algebraic Methods Program.* 117, p. 100605, doi:10.1016/j.jlamp.2020.100605.

[2] B.S. Baker (1974): *Non-context-free grammars generating context-free languages*. *Information and Control* 24, pp. 231 – 246, doi:10.1016/S0019-9958(74)80038-0.

[3] J.L. Balcazar, J. Diaz & J. Gabarró (1995): *Structural Complexity*. Springer-Verlag, Berlin, doi:10.1007/978-3-642-79235-9.

[4] L. Boasson, B. Courcelle & M. Nivat (1981): *The rational index, a complexity measure for languages*. *SIAM J. Computing* 10, pp. 284 – 296, doi:10.1137/0210020.

[5] R.V. Book (1972): *Terminal context in context-sensitive grammars*. *SIAM J. Computing* 1, pp. 20 – 30, doi:10.1137/0201003.

[6] H. Bordihn & V. Mitrana (2020): *On the degrees of non-regularity and non-context-freeness*. *J. Comput. Syst. Sci.* 108, pp. 104 – 117, doi:10.1016/j.jcss.2019.09.003.

[7] B. Brainerd (1968): *An analog of a theorem about context-free languages*. *Information and Control* 11, pp. 561 – 567, doi:10.1016/S0019-9958(67)90771-1.

[8] N. Chomsky & M. P. Schützenberger (1963): *The algebraic theory of context-free languages*. In: *Studies in Logic and the Foundations of Mathematics*, North-Holland, Amsterdam, pp. 118–161, doi:10.1016/S0049-237X(08)72023-8.

[9] J. M. Corson (2005): *Extended finite automata and word problems*. *J. Algebra Comput.* 15, pp. 455 – 466, doi:10.1142/S0218196705002360.

[10] K. Culik & H. Maurer (1978): *On the Derivation of Trees*. TR Vol, 18, Institut für Informationsverarbeitung (Graz).

[11] J. Dassow & G. Păun (1989): *Regulated Rewriting in Formal Language Theory*. Springer Berlin, Heidelberg, doi:10.1007/978-3-642-74932-2.

[12] J. Dassow & V. Mitrana (2000): *Finite automata over free groups*. *J. Algebra Comput.* 10, pp. 725 – 737, doi:10.1142/S0218196700000315.

[13] R. Evey (1963): *The Theory and Application of Pushdown Store Machines*. Doctoral Dissertation, Harvard University.

[14] S. Z. Fazekas, R. Mercaş & O. Wu (2022): *Complexities for jumps and sweeps*. *Journal of Automata, Languages and Combinatorics* 27, pp. 131–149, doi:10.25596/jalc-2022-131.

[15] H. Fernau & R. Stiebe (2002): *Sequential grammars and automata with valences*. *Theoret. Comput. Sci.* 276, pp. 377 – 405, doi:10.1016/S0304-3975(01)00282-1.

[16] J. Gabarro (1983): *Initial index:a new complexity function for languages*. In: *International Colloquium on Automata, Languages, and Programming ICALP*, LNCS 154 Springer Verlag, pp. 226–236, doi:10.1007/BFb0036911.

[17] J. Gabarro (1984): *Pushdown space complexity and related full-A.F.L.s*. In: *Annual Symposium on Theoretical Aspects of Computer Science STACS*, LNCS 166 Springer Verlag, pp. 250–259, doi:10.1007/3-540-12920-0_23.

[18] R. H. Gilman & M. Shapiro (1998): *On groups whose word problem is solved by a nested stack automaton*. *arXiv:math.GR/9812028*, doi:10.48550/arXiv.math/9812028.

[19] S. Ginsburg & S. Greibach (1966): *Mappings which preserve context-sensitive languages*. *Information and Control* 9, pp. 563 – 582, doi:10.1016/S0019-9958(66)80016-5.

[20] J. Goldstine (1972): *Substitution and bounded languages*. *J. Comput. Syst. Sci.* 6, pp. 9 – 29, doi:10.1016/S0022-0000(72)80038-2.

[21] T. Hibbard (1966): *Scan Limited Automata and Context Limited Grammars*. Doctoral Dissertation, University of California at Los Angeles.

[22] J.E. Hopcroft & J.D. Ullman (1979): *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, Reading, Mass.

[23] M. Kambites (2006): *Word problems recognisable by deterministic blind monoid automata*. Theoret. Comput. Sci. 362, pp. 232 – 237, doi:10.1016/j.tcs.2006.06.026.

[24] R. C. Lyndon & P. E. Schupp (1977): *Combinatorial Group Theory*. Springer-Verlag, Berlin, doi:10.1007/978-3-642-61896-3.

[25] C. Martín-Vide, V. Mitrana & Gh. Păun (2003): *Formal Languages and Applications*. Springer Verlag, doi:10.1007/978-3-540-39886-8.

[26] G. Matthews (1963): *Discontinuity and asymmetry in phrase structure grammars*. Information and Control 6, pp. 137–146, doi:10.1016/S0019-9958(63)90179-7.

[27] G. Matthews (1964): *A note on asymmetry in phrase structure grammars*. Information and Control 7, pp. 360–365, doi:10.1016/S0019-9958(64)90406-1.

[28] G. Matthews (1967): *Two-way languages*. Information and Control 10, pp. 111–119, doi:10.1016/S0019-9958(67)80001-9.

[29] A. Meduna & P. Zemek (2012): *Jumping finite automata*. Int. J. Found. Comput. Sci. 23, pp. 1555–1578, doi:10.1142/S0129054112500244.

[30] V. Mitrana & R. Stiebe (2001): *Extended finite automata over groups*. Discrete Appl. Math. 108, pp. 287 – 300, doi:10.1016/S0166-218X(00)00200-6.

[31] B. Nagy & L. Kovács (2014): *Finite automata with translucent letters applied in natural and formal language theory*. In: Transactions on Computational Collective Intelligence, LNCS 8790 Springer Verlag, pp. 107–127, doi:10.1007/978-3-662-44994-3_6.

[32] J. J. Rotman (1995): *An Introduction to the Theory of Groups*. Springer-Verlag, Berlin, doi:10.1007/978-1-4612-4176-8.

[33] G. Rozenberg & A. Salomaa (1997): *Handbook of Formal Languages*. Springer Verlag, doi:10.1007/978-3-642-59136-5.