Theoretical Computer Science 658 (2017) 74–84

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

On a class of languages with holonomic generating functions

Giusi Castiglione [a,*], Paolo Massazza [b]

[a] *Dipartimento di Matematica e Informatica, Università di Palermo, Italy*
[b] *Dipartimento di Scienze Teoriche e Applicate, Università dell'Insubria, Italy*

ARTICLE INFO

Article history:
Received 2 November 2015
Received in revised form 13 July 2016
Accepted 15 July 2016
Available online 26 July 2016

Keywords:
Holonomic functions
Parikh vectors
Context free languages
k-counter machines
Parikh automata

ABSTRACT

We define a class of languages (RCM) obtained by considering Regular languages, linear Constraints on the number of occurrences of symbols and Morphisms. The class RCM presents some interesting closure properties, and contains languages with holonomic generating functions. As a matter of fact, RCM is related to one-way 1-reversal bounded k-counter machines and also to Parikh automata on letters. Indeed, RCM is contained in $\mathcal{L}_{\mathsf{NFCM}}$ but not in $\mathcal{L}_{\mathsf{DFCM}}$, and strictly includes $\mathcal{L}_{\mathsf{LPA}}$. We conjecture that $\mathcal{L}_{\mathsf{DFCM}} \subset \mathsf{RCM}$.

1. Introduction

A well-known result of Chomsky–Schützenberger [6] states that the generating functions of regular languages are rational whereas the generating functions of unambiguous context-free languages are algebraic.

Thanks to this result, many enumeration problems have been solved by bijections with languages, i.e. by using the so called DVS methodology, for Delest–Viennot–Schützenberger. Indeed, once a bijection between a combinatorial structure C and a language L is found, the problem of determining the number of objects in C having size n is reduced to computing the coefficient a_n of the generating function of L, $\phi_L(x) = \sum a_n x^n$. Then, if $\phi_L(x)$ is rational or algebraic, the value a_n can be computed by standard techniques (see, for instance, [12]). For example, in [8] the generalized Dyck paths are described by generalized Dyck words, which belong to a context free language that lets us obtain a more precise asymptotic enumeration and statistics on such paths.

Furthermore, the use of generating functions allows us to use analytic methods as a powerful tool to determine properties of languages. For instance, a method to show that a context-free language is inherently ambiguous, employed by Flajolet in [9] and [10], consists of proving that it has a transcendental generating function. On the other hand, an open problem is that of determining the class of functions that contains the generating functions of all context-free languages.

Hence, one can pose the question of finding suitable classes of languages having generating functions that are interesting from the computational point of view (for instance, with respect to the complexity of the coefficient computation) and belong to classes of functions with closure properties that can be exploited for solving decision problems on languages.

In this context, the holonomic functions have been widely investigated since the end of 1980s. The class of the holonomic functions in one variable is an extension of the class of the algebraic functions and contains all the functions satisfying a linear differential equation with polynomial coefficients (see [21,22]).

[*] Corresponding author.
E-mail addresses: giuseppa.castiglione@unipa.it (G. Castiglione), paolo.massazza@uninsubria.it (P. Massazza).

http://dx.doi.org/10.1016/j.tcs.2016.07.022
0304-3975/© 2016 Elsevier B.V. All rights reserved.

Bertoni, Massazza, Sabadini 1992

A first use of holonomic functions in the context of formal languages was in [4], where the authors proved that the problem of deciding the holonomicity of the generating function of a context-free language is equivalent to the problem of deciding whether a context-free language is inherently ambiguous. Furthermore, a class of languages with holonomic 1993 generating functions, called LCL, has been introduced by one of the authors [17]. Such a class contains the languages that can be expressed as the intersection of an unambiguous context-free language and a language of words satisfying a set of linear constraints on the occurrences of symbols of the alphabet. The class LCL lacks closure properties with respect to most of the usual operations on languages. Thus, a first attempt of defining a subclass of LCL with interesting closure properties was done in [4], where it was shown that the equivalence problem is decidable for the class LCL_R (closed under intersection) consisting of languages that are the intersection of a regular language and a language of words satisfying a set of linear constraints on the occurrences of symbols of the alphabet. More recently, in [19], in order to 2008 extend the correspondence between languages and generating functions, the authors studied the shuffle product as operator on context-free languages and as operation on grammar rules.

The idea of using constraints and finite state automata in order to define "visible" languages is also at the basis of a family of automata called Parikh Automata (PA) and defined in [14,15]. In [5], the subclass LPA of Parikh Automata on letters has been defined. The aim of [5] CADILHAC et al. 2012 does not concern the study of the generating functions of the recognized languages but, mainly, the study of the closure and decidability properties of these models. Probably, the authors were not aware of the results which appeared in [4,17], and prove a proposition from which one can infer that the class $\mathcal{L}_{\mathsf{LPA}}$ of the languages recognized by automata in LPA is just the class LCL_R. As a consequence, the generating function of a language in $\mathcal{L}_{\mathsf{LPA}}$ is holonomic.

In this paper, we define a class of languages, called RCM, which strictly includes LCL_R while borrowing the main aspect from LCL, i.e. any language in RCM has a holonomic generating function. Moreover, RCM turns out to be closed under union and intersection. A language in RCM is specified by a triple $\langle R, C, \mu\rangle$ where R is a regular language (described by a finite state automaton or a regular expression), C is a set of linear constraints on the number of occurrences of symbols and μ is a morphism that is length preserving (i.e. $|w| = |\mu(w)|$ for all w) and injective on the language $R \cap [C]$ ($[C]$ is the set of words satisfying C). Therefore, $\langle R, C, \mu\rangle = \mu(R \cap [C])$.

In addition to the expressiveness, the interest in this class is manifold. The first is related to the properties of the class of the holonomic functions. More precisely, since the extraction of the nth coefficient $[x^n]\phi(x)$ of a holonomic function $\phi(x)$ can be done by means of efficient algorithms [16,18], it follows that the counting problem for RCM can be solved efficiently, once the differential equation satisfied by the generating function $\phi_L(x)$ of a language $L \in \mathsf{RCM}$ has been computed.

Furthermore, the closure properties of RCM (w.r.t. union and intersection) and of the class of the holonomic functions, let us show that the following problems are decidable for RCM: equivalence, inclusion, universe, disjointness, emptiness. Indeed, all of them can be reduced to the equality problem for the holonomic functions, which is well-known to be decidable [22].

Finally, it is interesting to study the relation between RCM and some well-known models such as reversal-bounded counter machines and Parikh automata. More precisely, we prove that RCM is strictly included in the class $\mathcal{L}_{\mathsf{NFCM}}$ of the languages recognized by one-way nondeterministic counter machines, whereas it is not included in $\mathcal{L}_{\mathsf{DFCM}}$ (deterministic), and strictly includes $\mathcal{L}_{\mathsf{LPA}}$, the class of languages recognized by Parikh automata on letters. Finally, we conjecture that $\mathcal{L}_{\mathsf{DFCM}}$ is included in RCM.

2. Preliminaries

In this section we give some notations and basics about languages, generating functions, and classes of languages and automata of our interest in the paper. Furthermore, we recall some of the well-known results about such classes.

Let $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_h\}$ be a finite alphabet and $w \in \Sigma^\star$. For all i, with $1 \le i \le h$, we denote by $|w|_{\sigma_i}$ the number of occurrences of the symbol σ_i in w. The *length* of w is $|w| = \sum_i |w|_{\sigma_i}$ and the *Parikh vector* of $w \in \Sigma^\star$ is the integer vector $[|w|_{\sigma_1}, |w|_{\sigma_2}, \ldots, |w|_{\sigma_h}]$.

Given a positive integer d, a *semilinear set* in $\mathbb{N}^d$ is a finite union of linear sets of the form $\{\mathbf{v}_0 + k_1\mathbf{v}_1 + \ldots + k_p\mathbf{v}_p | k_i \in \mathbb{N}\}$, with $\mathbf{v}_i \in \mathbb{N}^d$.

Let Γ, Σ be two finite alphabets. A morphism $\mu : \Gamma^\star \mapsto \Sigma^\star$ is said *length preserving* if for all $w \in \Gamma^\star$ one has $|\mu(w)| = |w|$.

In the following, we indicate by $\uplus$ the disjoint union of languages, that is, $L = A \uplus B$ means $L = A \cup B$ and $A \cap B = \emptyset$. We also denote by $\bar{L}$ the complement $\Sigma^\star \setminus L$ of a language $L \subseteq \Sigma^\star$.

2.1. Generating functions of languages

The *generating function* of a language L is the function $\phi_L(x) = \sum_{n \ge 0} a_n x^n$ where $a_n = |\{w \in L|\ |w| = n\}|$.

If μ is a length preserving morphism that is injective on L, then the languages L and $\mu(L)$ have the same generating function, that is, $\phi_L(x) = \phi_{\mu(L)}(x)$.

It is well-known that regular languages and unambiguous context-free languages have rational and algebraic generating functions, respectively. In the sequel, we deal with languages whose generating functions belong to the class of the holonomic functions, introduced by Bernstein in 1970s [2,3] and deeply investigated by Stanley, Lipshitz, Zeilberger and others [21,16,22,7]. Here we consider holonomic functions in one variable, which are defined as follows.

Definition 1. A function $f(x) : \mathbb{C} \mapsto \mathbb{C}$ is holonomic if and only if there exist an integer d and $d+1$ polynomials $p_i(x) \in \mathbb{C}[x]$, $0 \le i \le d$, with $p_d(x) \neq 0$, such that

$$\sum_{i=0}^{d} p_i(x) \frac{d^i}{dx^i} f(x) = 0.$$

The class of the holonomic functions contains all rational functions as well as all algebraic functions. Moreover, it admits interesting closure properties that are summarized in the following theorem.

Theorem 1 *([22]). The class of the holonomic functions is closed under the operations of sum, product, indefinite integration, differentiation, right composition with algebraic functions.*

Furthermore, given two differential equations (of order d_1 and d_2, respectively) satisfied by two holonomic functions $f_1(x)$ and $f_2(x)$, one can compute the differential equation satisfied by the function $f_1(x) - f_2(x)$ (of order $d = O(d_1 + d_2)$); then, the problem of deciding whether $f_1(x) = f_2(x)$ is reduced to computing the first $d+1$ coefficients in the Taylor's expansions at $x = 0$ of $f_1(x)$ and $f_2(x)$. We recall that a useful package for dealing with holonomic functions (Gfun, see [20]) is available for the computer algebra system Maple. Actually, one can use Gfun to find the annihilator of $f_1(x) \mathrm{op} f_2(x)$, for op $\in \{+, -, *\}$.

2.2. Classes of languages with constraints

We are interested in defining a class of languages whose words satisfy linear constraints on the number of occurrences of symbols in the alphabet.

Definition 2 *(linear constraint)*. A linear constraint on the occurrences of symbols of $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_h\}$ in $w \in \Sigma^\star$ is an expression of the form

$$\sum_{i=1}^{h} c_i |w|_{\sigma_i} \vartriangle c_{h+1},$$

where $c_i \in \mathbb{Z}$ and $\vartriangle \in \{<, \le, =, \neq, \ge, >\}$. Boolean combinations of linear constraints

Definition 3 *(system of linear constraints)*. A system of linear constraints C is either a linear constraint, or $C_1 \vee C_2$ or $C_1 \wedge C_2$ or $\neg C_1$, where C_1, C_2 are systems of linear constraints. Thus, no modulo constraints } They can be encoded in the automaton

We denote by $[C]$ the language of the words in $\Sigma^\star$ satisfying a system of linear constraints C.

A well-known combinatorial result states that a set of integral solutions of a system of linear inequalities C can be represented as the set of nonnegative linear combinations of a finite number of vectors. More precisely, the set of nonnegative integral solutions of C can be represented by the Hilbert basis as defined in [11]. On the other hand, given a finite set of vectors $I \subset \mathbb{N}^d$ one can easily compute a system C of linear constraints (on an alphabet with d variables) such that:

- the Parikh vector of a word $w \in [C]$ lies in the semilinear set generated by I;
- if $w \notin [C]$ then its Parikh vector does not belong to the semilinear set generated by I.

Notice that the number of constraints in C may grow exponentially in the number of vectors of I. The following example illustrates how C can be obtained from I.

Example 1. Let $\Sigma = \{a, b\}$ and consider the semilinear set

$$S = \{\mathbf{v} \in \mathbb{N}^2 | \exists i, j \ge 0, \mathbf{v} = (2, 9) + i(3, 5) + j(4, 7)\}.$$

If $i = j = 0$ one has the vector $(2, 9)$, which corresponds to the system of constraints $|w|_a = 2 \wedge |w|_b = 9$. For $i > 0$ and $j = 0$ one has the vectors $(3i+2, 5i+9)$. By setting $|w|_a = 3i+2$ and $|w|_b = 5i+9$, one gets the system $3|w|_b - 5|w|_a = 17$. Similarly, if $i = 0$ and $j > 0$ one obtains $4|w|_b - 7|w|_a = 22$. Lastly, for $i, j > 0$ one has $|w|_a = 3i+4j+2$, $|w|_b = 5i+7j+9$ and $|w|_b - |w|_a = 2i+3j-7$. Hence, $21|w|_b - 35|w|_a = 119$. Thus, the system associated with S is

$$(|w|_a = 2) \wedge (|w|_b = 9) \vee (3|w|_b - 5|w|_a = 17) \vee (4|w|_b - 7|w|_a = 22) \vee (21|w|_b - 35|w|_a = 119).$$

In [17], a class of languages called LCL was defined as follows.

Definition 4 *(*LCL*).* The class LCL is the class of all languages L such that $L = M \cap [C]$, where M is an unambiguous context free language and C a system of linear constraints.

For this class of languages the following result was proved.

Theorem 2 *([17, Thm. 2]). Let $L \in$ LCL. Then $\phi_L(x)$ is holonomic.*

Furthermore, in [17] it is implicitly shown that there exists an algorithm for computing an annihilator for the generating function of a language in LCL. In [4] a subclass of LCL was defined as follows.

Definition 5 *(*LCL_R*).* The class LCL_R is the class of all languages L such that $L = R \cap [C]$, where R is a regular language and C a system of linear constraints.

For example, the non-context-free language $\{a^n b^n c^n \,|\, n > 0\}$ belongs to LCL_R since it is the language of words $w \in a^\star b^\star c^\star$ such that $|w|_a = |w|_b \wedge |w|_b = |w|_c$.

The class LCL_R is closed under intersection but not closed under union and is composed by languages with holonomic generating function. Furthermore, the equivalence problem is decidable in LCL_R since it can be reduced to the equality problem for holonomic functions, which is decidable (see [22]).

2.3. Parikh automata

Let Σ be a finite alphabet, $d \in \mathbb{N}^+$, and D a finite subset of $\mathbb{N}^d$. The monoid morphism from $(\Sigma \times D)^\star$ to $\Sigma^\star$ defined by $(a, v) \longmapsto a$ is called the *projection* on Σ and the monoid morphism from $(\Sigma \times D)^\star$ to $\mathbb{N}^d$ defined by $(a, v) \longmapsto v$ is called the *extended Parikh image*.

A *Parikh automaton* of dimension d over Σ is a pair (A, C) where $A = (Q, \Sigma \times D, \delta, q_0, F)$ is a finite state automaton over $\Sigma \times D$, and $C \subseteq \mathbb{N}^d$ is a semilinear set. The language recognized by (A, C), denoted by $L(A, C)$, is the projection on Σ of the words of $L(A)$ whose extended Parikh image is in C. A Parikh automaton is said to be deterministic if for each $\cancel{q} \in Q$ p and $a \in \Sigma$, there exists at most one pair $(q, v) \in Q \times D$ such that $(p, (a, v), q) \in \delta$.

We denote by PA (resp. DetPA) the class of Parikh automata (resp. deterministic Parikh automata), and by $\mathcal{L}_{\mathsf{PA}}$ (resp. $\mathcal{L}_{\mathsf{DetPA}}$) the class of languages recognized by PA (resp. DetPA). A *Parikh automaton on letters* [15] is a Parikh automaton where globally for any two labels of transitions (a, v) and (a, u), one has that $u = v$. The class of languages recognized by letter Parikh automata is denoted by $\mathcal{L}_{\mathsf{LPA}}$.

We recall that $\mathcal{L}_{\mathsf{LPA}} \subsetneq \mathcal{L}_{\mathsf{DetPA}} \subsetneq \mathcal{L}_{\mathsf{PA}}$. On the other hand, from [5, Prop. 5.5] the following proposition can be inferred.

Proposition 1. $\mathcal{L}_{\mathsf{LPA}} = \mathrm{LCL}_R$. LPA are determinisable

As a consequence, the generating functions of the languages in $\mathcal{L}_{\mathsf{LPA}}$ are holonomic.

2.4. Reversal-bounded counter machines

A one-way k-counter machine is a finite automaton equipped with k counters. The operations admitted on each counter are the increment or the decrement by 1, as well as the comparison with 0. The machine is called l-reversal bounded if the count in each counter alternately increases and decreases at most l times. We refer to [13] for all the definitions and main results concerning the class DFCM(k, m, n) (resp. NFCM(k, m, n)) of *deterministic* (resp. *non-deterministic*) *(m, n)-reversal bounded k-counter machines*, that is, n-reversal bounded k-counter machines with a two-way input tape where the input head reverses direction at most m times. In particular, we are interested in the classes DFCM$(k, 0, 1)$ and NFCM$(k, 0, 1)$, where the input tape is one-way and the counters can change from increasing to decreasing mode at most once. Formally, $M \in$ DFCM$(k, 0, 1)$ is a 7-tuple $M = (k, Q, \Sigma, \$, \delta, q_0, F)$, where k indicates the number of counters, Q is a finite set of states, Σ is the input alphabet, \$ is the right end-marker, δ is the transition function, $q_0 \in Q$ is the initial state and $F \subseteq Q$ is the set of final states. The transition function is a mapping from $Q \times (\Sigma \cup \{\$\}) \times \{0, 1\}^k$ into $Q \times \{S, R\} \times \{-1, 0, +1\}^k$ such that if $\delta(q, a, c_1, \dots, c_k) = (p, d, d_1, \dots, d_k)$ and $c_i = 0$ for some i, then d_i has to be nonnegative to prevent negative values in a counter. The symbols S and R are used to indicate the movement of the input tape head ($S =$ stay, $R =$ right).

A word $w \in \Sigma^\star$ is accepted by M if and only if $\delta^\star(q_0, w\$, 0, \dots, 0) = (q, \$, c_1, \dots, c_k)$, for some $q \in F$, with $c_i \geq 0$ for $1 \leq i \leq k$. The language recognized by M, denoted by $L(M)$, is the set of all the words accepted by M.

Here we recall some results related to Parikh automata and counter machines (cf. [14,5]). The two classes DetPA and PA do not have the same expressive power. Example 4 shows a language that is in $\mathcal{L}_{\mathsf{PA}} \setminus \mathcal{L}_{\mathsf{DetPA}}$ (and also in RCM).

Proposition 2. $\mathcal{L}_{\mathsf{DetPA}} \subsetneq \mathcal{L}_{\mathsf{PA}}$.

Parikh automata and nondeterministic one-way reversal-bounded counter machines recognize the same languages.

Proposition 3. $\mathcal{L}_{\mathsf{LPA}} = \mathcal{L}_{\mathsf{NFCM}}$.

The two classes DetPA and DFCM have not the same expressive power. Example 5 shows a language that is in $\mathcal{L}_{\mathsf{DFCM}} \setminus \mathcal{L}_{\mathsf{DetPA}}$ (and also in RCM).

Proposition 4. $\mathcal{L}_{\mathsf{DetPA}} \subsetneq \mathcal{L}_{\mathsf{DFCM}}$.

3. The class RCM

Let Γ, Σ be two finite alphabets. Consider a language R on Γ, a system C of linear constraints on the number of occurrences of symbols in Γ and a morphism $\mu : \Gamma^\star \mapsto \Sigma^\star$. We denote by $\langle R, C, \mu \rangle$ the language $\mu(R \cap [C]) \subseteq \Sigma^\star$, and we define the class of languages RCM as follows.

Definition 6 *(*RCM*).* The class RCM is the class of the languages $\langle R, C, \mu \rangle$ where R is a regular language, C a system of linear constraints and μ a length preserving morphism that is injective on $R \cap [C]$.

Note that Definition 6 does not specify how the regular language R has to be represented. Most of the examples we provide use regular expressions, whereas finite automata are used when proving closure properties. Thus, we are free to choose the representation that looks more convenient depending on the context.

The class RCM contains languages that are context-free as well as languages that are not context-free. Here we give some examples to get familiarity with RCM.

Example 2. The (non-context-free) language $\{a^{2i}b^{3j}a^ib^j \mid i, j > 0\}$ is in RCM. Indeed, let $\Gamma = \{a_i, a_2, b_1, b_2\}$, $\Sigma = \{a, b\}$, $R = a_1^+b_1^+a_2^+b_2^+$ and $C = (|w|_{a_1} = 2|w|_{a_2}) \wedge (|w|_{b_1} = 3|w|_{b_2})$, with $\mu : \Gamma^\star \mapsto \Sigma^\star$ defined by $\mu(a_1) = \mu(a_2) = a$ and $\mu(b_1) = \mu(b_2) = b$. It is immediate to see that μ is length preserving and injective on R (hence also on $R \cap [C]$). Indeed, it is sufficient to note that $\mu(w) = a^ib^ja^kb^l$ if and only if $w = a_1^ib_1^ja_2^kb_2^l$. Hence, $\langle R, C, \mu \rangle = \{a^{2i}b^{3j}a^ib^j \mid i, j > 0\}$.

Example 3. The (non-context-free) language $\{va^{|v|}b^{|v|} \mid v \in \{a, b\}^\star\}$ is in RCM. Indeed, let $\Gamma = \{a_1, a_2, b_1, b_2\}$, $\Sigma = \{a, b\}$, $R = (a_1 + b_1)^\star a_2^\star b_2^\star$ and $C = (|w|_{a_2} = |w|_{a_1} + |w|_{b_1}) \wedge (|w|_{b_2} = |w|_{a_1} + |w|_{b_1})$, with $\mu : \Gamma^\star \mapsto \Sigma^\star$ defined by $\mu(a_1) = \mu(a_2) = a$ and $\mu(b_1) = \mu(b_2) = b$. In this case, the length preserving morphism μ is injective on $R \cap [C]$ (but not on R). Indeed, suppose that there exist $u, t \in R \cap [C]$ such that $\mu(u) = \mu(t) = va^{|v|}b^{|v|}$. Let x be the longest common prefix of u and t, that is, $u = x\sigma y$ $t = x\tau z$, with $\sigma \neq \tau$. Without loss of generality, let $\sigma = a_1$. Then, as $\mu(u) = \mu(t)$, one has $\tau = a_2$. This implies that $|u|_{a_1} + |u|_{b_1} > |t|_{a_1} + |t|_{b_1}$ and then, as both u and t satisfy C, one has $|u| > |t|$. This contradicts $\mu(u) = \mu(t)$ (recall that μ is length preserving). Hence, $\langle R, C, \mu \rangle = \{va^{|v|}b^{|v|} \mid v \in \{a, b\}^\star\} \in$ RCM.

$\cancel{\# a_1} + \# b_1 + 1 = \cancel{\# a_1} + \# a_2$

Example 4. By $v[i]$ we denote the ith letter of the word v. The (non-context-free) language $L = \{w \in \{a, b\}^\star \mid w[|w|_a] = b\}$ is in RCM. Indeed, let $\Gamma = \{a_1, a_2, b_1, b_2, b_3\}$, $\Sigma = \{a, b\}$, $R = (a_1 + b_1)^\star b_3(a_2 + b_2)^\star$ and $C = (|w|_{a_2} = |w|_{b_1} + 1)$, with $\mu : \Gamma^\star \mapsto \Sigma^\star$ defined by $\mu(a_1) = \mu(a_2) = a$ and $\mu(b_1) = \mu(b_2) = \mu(b_3) = b$. Trivially, $L = \langle R, C, \mu \rangle$. Furthermore, the morphism μ is length preserving and we have only to prove that it is injective on $R \cap [C]$ (note that μ is not injective on R). Indeed, suppose that there exist $u, v \in R \cap [C]$ such that $\mu(u) = \mu(v)$ and let x be their longest common prefix, that is, $u = x\sigma y$ and $v = x\tau z$, with $\sigma \neq \tau$. Suppose that $\mu(\sigma) = a$. Without loss of generality, one has $u = xa_1y$ and $v = xa_2z$. One has $|v|_{b_1} = |x|_{b_1}$ and then (by the constraints in C) also $|z|_{a_2} = |x|_{b_1} - 1$. Since $|y|_{a_2} \geq |x|_{b_1} + 1 > |z|_{a_2}$, it follows that $\mu(z) \neq \mu(y)$, that is, $\mu(u) \neq \mu(v)$.

Similarly, let $\mu(\sigma) = b$. Without loss of generality, one has $u = xb_1y$ and $v = xb_2z$. Therefore, $|z|_{a_2} = |x|_{b_1} + 1$, whereas $|y|_{a_2} \geq |x|_{b_1} + 2$. This means that $\mu(z) \neq \mu(y)$. Note that the case $u = xb_3y$ and $v = xb_iz$, with $1 \leq i \leq 2$, can not occur since it implies that b_3 is in x and so it would occur twice in u.

Example 5. Here, we consider the (non-context-free) language

$$\mathrm{NSUM} = \{a^n \spadesuit b^{i_1} \sharp b^{i_2} \sharp \cdots \sharp b^{i_m} \clubsuit c^{i_1 + \ldots + i_n} \mid k \geq n \geq 0 \wedge i_j \in \mathbb{N}\}$$

that is used in [5] to show that $\mathcal{L}_{\mathsf{DetPA}} \subsetneq \mathcal{L}_{\mathsf{DFCM}}$. Thus, let $L = \langle R, C, \mu \rangle$, where $R = a_1^\star s_1(b_1^\star s_2)^\star(b_2^\star s_3)^\star b_2^\star s_4 c_1^\star$ and $C = (|w|_{a_1} = |w|_{s_2}) \wedge (|w|_{b_1} = |w_{c_1}|)$, with $\mu(a_1) = a$, $\mu(b_1) = \mu(b_2) = b$, $\mu(c_1) = c$, $\mu(s_1) = \spadesuit$, $\mu(s_4) = \clubsuit$ and $\mu(s_2) = \mu(s_3) = \sharp$. It is immediate to see that NSUM $= L$. So, we have only to prove that the length preserving morphism μ is injective on $R \cap [C]$. We argue by contradiction and suppose that there exist $u, v \in R \cap [C]$ such that $\mu(u) = \mu(v) = a^n \spadesuit b^{i_1} \sharp \cdots b^{i_n} \sharp \gamma c^{i_1 + \ldots + i_n}$, for suitable integers $n, i_1, \ldots, i_n$ and a certain $\gamma \in (b_1^\star s_2)^\star(b_2^\star s_3)^\star b_2^\star s_4$.

Since $|u|_{a_1} = |u|_{s_2}$ and $|v|_{a_1} = |v|_{s_2}$, the word $x = a_1^n s_1 b_1^{i_1} s_2 \cdots b_1^{i_n} s_2$ is a common prefix of u and v. Furthermore, $y = s_4 c_1^{i_1 + \ldots + i_n}$ is a common suffix of u and v. Thus, one has $u = x\gamma y$ and $v = x\gamma' y$, with $|\gamma|_{s_2} = |\gamma'|_{s_2} = 0$; this means that γ and γ' are in $(b_2^\star s_3)^\star b_2^\star$. Hence, by definition of μ, $\gamma \neq \gamma'$ implies $\mu(\gamma) \neq \mu(\gamma')$, that is, $\mu(u) \neq \mu(v)$.

Note that RCM has been defined by using injective morphisms. The main motivation behind this choice is that it leads to the following statement about generating functions of languages in RCM. Furthermore, in Section 4 injective morphisms are used to study the relationship between RCM and other known classes of languages.

Theorem 3. *Every language in* RCM *has a holonomic generating function.*

In fact, they have the same g.f. as LCLR

Proof. Let $L \in$ RCM. By definition, one has $\phi_L(x) = \phi_{\mu(R\cap[C])}(x)$ and then, since μ is injective on $R \cap [C]$ and length preserving, $\phi_L(x) = \phi_{R\cap[C]}(x)$. Since $R \cap [C]$ is in LCL_R (a subclass of LCL), the generating function $\phi_{R\cap[C]}(x)$ is holonomic due to [17, Thm. 2]. □

Now we prove some closure properties of the class RCM. Notice that LCL_R is closed under union only if disjoint alphabets are considered.

Theorem 4. *The class* RCM *is closed under union.*

FLAWED

Close under union is open

Proof. Let $L = \langle R, C, \mu\rangle$ and $L' = \langle R', C', \mu'\rangle$ be two languages in RCM. Without loss of generality, we suppose that the alphabets of R and R' are disjoint, $\Gamma \cap \Gamma' = \emptyset$. Otherwise, we can rename the symbols of Γ' that are in $\Gamma \cap \Gamma'$ and then modify R', C' and μ' accordingly, obtaining $\tilde{L} = \langle \tilde{R}, \tilde{C}, \tilde{\mu}\rangle$ with $\mu'(R' \cap [C']) = \tilde{\mu}(\tilde{R} \cap [\tilde{C}])$ and $\tilde{R} \subseteq \tilde{\Gamma}^\star$, $\tilde{\Gamma} \cap \Gamma = \emptyset$. Moreover, we suppose that the empty word does not belong to $[C]$ or to $[C']$. We show that there exist suitable R'', C'' and μ'' such that $L'' = \langle R'', C'', \mu''\rangle = L \cup L'$.

Recall that regular languages are closed with respect to morphisms and inverse morphisms. Thus, we first compute the regular language $M = \mu(R) \cap \mu'(R')$, and then the two (regular) languages $R_1 = R \cap \mu^{-1}(M)$ and $R'_1 = R' \cap \mu'^{-1}(M)$ (one has $\mu(R_1) = \mu'(R'_1) = M$). Furthermore, let $R_2 = R \setminus R_1$ and $R'_2 = R' \setminus R'_1$.

From the automata recognizing R_1 and R'_1, say $A_1 = (Q_1, \Gamma, \delta_1, q_0, F_1)$ and $A'_1 = (Q'_1, \Gamma', \delta'_1, q'_0, F'_1)$, we construct an automaton $A' = (Q', \mathcal{T}, \delta', p_0, F')$ which resembles the usual product of automata (note that $\Gamma \cap \Gamma' = \emptyset$). More precisely, the alphabet of A' is $\mathcal{T} = \{\tau_{ij} | 1 \leq i \leq |\Gamma|, 1, \leq j \leq |\Gamma'|\}$, the set of states of A' is $Q' \subseteq Q_1 \times Q'_1$, the set of finals states is $F' = F_1 \times F'_1$ the initial state of A' is $p_0 = (q_0, q'_0)$ and the transition function δ' is defined as follows: $\delta'((q_i, q'_j), \tau_{rs}) = (q_h, q'_k)$ if and only if $\delta_1(q_i, \gamma_r) = q_h$, $\delta'_1(q'_j, \gamma'_s) = q'_k$ and $\mu(\gamma_r) = \mu'(\gamma'_s)$. Thus, A' accepts a word $\tau_{i_1 j_1}\tau_{i_2 j_2}\cdots\tau_{i_n j_n}$ if and only if $w = \gamma_{i_1}\gamma_{i_2}\cdots\gamma_{i_n} \in R_1$, $w' = \gamma'_{j_1}\gamma'_{j_2}\cdots\gamma'_{j_n} \in R'_1$ and $\mu(w) = \mu'(w')$.

Let $M' \subseteq \mathcal{T}^\star$ be the language recognized by A' and consider the system of constraints $C_{M'} = \tilde{C} \vee \tilde{C}'$ where the systems $\tilde{C}$ and $\tilde{C}'$ are obtained by replacing $|w|_{\gamma_i}$ with $\sum_j |w|_{\tau_{ij}}$ (in all the linear constraints of C) and $|w|_{\gamma'_j}$ with $\sum_i |w|_{\tau_{ij}}$ (in all the linear constraints of C'). Then, we define a morphism $\mu_{M'}$ that associates with a word $\tau_{i_1 j_1}\cdots\tau_{i_n j_n} \in M'$ the word $\mu(\gamma_{i_1}\cdots\gamma_{i_n}) = \mu'(\gamma'_{j_1}\cdots\gamma'_{j_n})$.

So, the three components needed to show that $L \cup L'$ is in RCM are:

$$R'' = R_2 \uplus R'_2 \uplus M',$$

$$C'' = C \vee C' \vee C_{M'},$$

$$\mu''(w) = \begin{cases} \mu(w) & \text{if} \quad w \in R_2, \\ \mu'(w) & \text{if} \quad w \in R'_2, \\ \mu_{M'}(w) & \text{if} \quad w \in M'. \end{cases}$$

It still remains to prove that $\mu''(R'' \cap [C'']) = L \cup L'$ and that μ'' is injective on $R'' \cap [C'']$. Notice that the alphabets of R_2, R'_2 and M' are disjoint and that a word w in R_2 (R'_2, M', resp.) satisfies C'' if and only if w satisfies C ($C', C_{M'}$, resp.). Therefore, one has

$$L = \mu(R \cap [C]) = \mu((R_2 \uplus R_1) \cap [C]) = \mu(R_2 \cap [C]) \uplus \mu(R_1 \cap [C])$$

and

$$L' = \mu'(R' \cap [C']) = \mu'((R'_2 \uplus R'_1) \cap [C']) = \mu'(R'_2 \cap [C']) \uplus \mu'(R'_1 \cap [C']).$$

Thus, one has

$$L \cup L' = \mu(R_2 \cap [C]) \uplus \mu'(R'_2 \cap [C']) \uplus (\mu(R_1 \cap [C]) \cup \mu'(R'_1 \cap [C']))$$

and it is sufficient to show that $\mu(R_1 \cap [C]) \cup \mu'(R'_1 \cap [C']) = \mu_{M'}(M' \cap [C_{M'}])$.

If w is in $\mu_{M'}(M' \cap [C_{M'}])$ then there exists $\tau_{i_1 j_1}\cdots\tau_{i_n j_n} \in M' \cap [C_{M'}]$ such that $v = \gamma_{i_1}\cdots\gamma_{i_n} \in R_1$, $v' = \gamma'_{j_1}\cdots\gamma'_{j_n} \in R'_1$ and $v \in [C]$ or $v' \in [C']$, with $\mu(v) = \mu'(v') = w$. Therefore w is also in $\mu(R_1 \cap [C]) \cup \mu'(R'_1 \cap [C'])$.

Now, consider a word w in $\mu(R_1 \cap [C]) \cup \mu(R'_1 \cap [C'])$. Without loss of generality, we suppose that there exists a word $v = \gamma_{i_1}\cdots\gamma_{i_n}$ in $R_1 \cap [C]$ such that $\mu(v) = w$. Since $w \in M$, there exists also a word $v' = \gamma'_{j_1}\cdots\gamma'_{j_n} \in R'_1$ such that $\mu'(v') = w$. So, the word $w' = \tau_{i_1 j_1}\cdots\tau_{i_n j_n}$ is in M' and also in $[C_{M'}]$, with $\mu_{M'}(w') = w$, that is to say, $w \in \mu_{M'}(M' \cap [C_{M'}])$.

this is not true

By definition, the morphism μ'' is injective on $R_2 \cap [C]$, $R'_2 \cap [C']$ and $M' \cap [C]_{M'}$. So, it is injective on $R'' \cap [C]''$ since the languages $\mu''(R_2 \cap [C])$, $\mu''(R'_2 \cap [C'])$, and $\mu''(M' \cap [C_{M'}])$ are disjoint. □

Theorem 5. *The class* RCM *is closed under intersection.*

Proof. Let $L_1 = \langle R_1, C_1, \mu_1 \rangle \subseteq \Sigma_1^\star$ and $L_2 = \langle R_2, C_2, \mu_2 \rangle \subseteq \Sigma_2^\star$, with $R_1 \subseteq \Gamma_1^\star$, $R_2 \subseteq \Gamma_2^\star$, $\mu_1 : \Gamma_1^\star \mapsto \Sigma_1^\star$ and $\mu_2 : \Gamma_2^\star \mapsto \Sigma_2^\star$.

Obviously, if a word w is in $L_1 \cap L_2$ then $w \in (\Sigma_1 \cap \Sigma_2)^\star$. Without loss of generality, we suppose that $\Sigma_1 = \Sigma_2 = \{\sigma_1, \ldots, \sigma_k\}$, otherwise we can consider the two languages $M_1 = \langle R_1, \hat{C}_1, \mu_1 \rangle \subseteq (\Sigma_1 \cap \Sigma_2)^\star$ and $M_2 = \langle R_2, \hat{C}_2, \mu_2 \rangle \subseteq (\Sigma_1 \cap \Sigma_2)^\star$ where

$$\hat{C}_1 = C_1 \bigwedge_{\gamma \in \Gamma_1, \mu_1(\gamma) \in \Sigma_1 \setminus \Sigma_2} (|w|_\gamma = 0),$$

$$\hat{C}_2 = C_2 \bigwedge_{\gamma \in \Gamma_2, \mu_2(\gamma) \in \Sigma_2 \setminus \Sigma_1} (|w|_\gamma = 0),$$

and notice that $L_1 \cap L_2 = M_1 \cap M_2$.

Let $A_1 = (Q_1, \Gamma_1, \delta_1, p_0, F_1)$ and $A_2 = (Q_1, \Gamma_2, \delta_2, q_0, F_2)$ be two deterministic automata recognizing R_1 and R_2, respectively. Without loss of generality, one suppose that $\Gamma_1 \cap \Gamma_2 = \emptyset$.

For all i with $1 \le i \le k$, let $\Upsilon_1(i) = \mu_1^{-1}(\sigma_i)$ and $\Upsilon_2(i) = \mu_2^{-1}(\sigma_i)$. We define a deterministic automaton $A = (Q, \Gamma, \delta, v_0, F)$ where

- $Q = Q_1 \times Q_2$;
- $\Gamma = \bigcup_{i=1}^k \Upsilon_1(i) \times \Upsilon_2(i) = \{(\beta, \gamma) | \beta \in \Gamma_1, \gamma \in \Gamma_2, \mu_1(\beta) = \mu_2(\gamma)\}$;
- $\delta((p_i, q_j), (\beta, \gamma)) = (p_s, q_t) \Leftrightarrow \delta_1(p_i, \beta) = p_s,\ \delta_2(q_j, \gamma) = q_t,\ \mu_1(\beta) = \mu_2(\gamma)$;
- $v_0 = (p_0, q_0)$;
- $F = F_1 \times F_2$.

For simplicity, we rename the symbols of Γ and let τ_{ij} denote the symbol (β_i, γ_j) of Γ, where β_i is the ith symbol of Γ_1 and γ_j is the jth symbol of Γ_2, respectively. So, we define the morphism $\mu : \Gamma^\star \mapsto \Sigma^\star$ by letting $\mu(\tau_{ij}) = \mu_1(\beta_i) = \mu_2(\gamma_j)$. Therefore, by construction, the language recognized by A consists of the words $\tau_{i_1 j_1} \cdots \tau_{i_n j_n}$ such that $\beta_{i_1}\beta_{i_2} \ldots \beta_{i_n} \in R_1$, $\gamma_{j_1}\gamma_{j_2} \ldots \gamma_{j_n} \in R_2$ and $\mu(\tau_{i_1 j_1} \cdots \tau_{i_n j_n}) = \mu_1(\beta_{i_1} \cdots \beta_{i_n}) = \mu_2(\gamma_{j_1} \cdots \gamma_{j_n})$. So, we set $R = L(A)$.

Now, we construct a system C of linear constraints as follows. Let $\hat{C}_1$ be the system of linear constraints obtained from C_1 by replacing $|w|_{\beta_i}$ with $\sum_{\tau \in \Upsilon_1(i) \times \Upsilon_2(i)} |w|_\tau$ in each linear constraint of C_1.

Indeed, if $w_1 \in R_1$ satisfies C_1 and $\mu_1(w_1) \in L_1 \cap L_2$ then there exists $w_2 \in R_2$ such that $(w_1, w_2) = \tau_{i_1 j_1} \cdots \tau_{i_k j_k}$ is in R and satisfies $\hat{C}_1$, and vice versa.

We proceed similarly for the constraints in C_2, obtaining a system $\hat{C}_2$. Lastly, let $C = \hat{C}_1 \wedge \hat{C}_2$ and notice that $(w_1, w_2) \in R$ satisfies C if and only if w_1 satisfies C_1 and w_2 satisfies C_2.

So, it remains to prove that the morphism μ defined above is injective on $R \cap [C]$. Indeed, suppose that there exist $(w_1, w_2), (\hat{w}_1, \hat{w}_2) \in R$ satisfying C and such that $\mu((w_1, w_2)) = \mu((\hat{w}_1, \hat{w}_2))$. Without loss of generality, let $w_1 \neq \hat{w}_1$. Since $w_1, \hat{w}_1 \in R_1 \cap [C_1]$ and μ_1 is injective on $R_1 \cap [C_1]$, one has $\mu_1(w_1) \neq \mu(\hat{w}_1)$, which is a contradiction as $\mu((w_1, w_2)) = \mu_1(w_1) = \mu((\hat{w}_1, \hat{w}_2)) = \mu_1(\hat{w}_1)$.

Therefore, the language $L_1 \cap L_2 = \langle R, C, \mu \rangle$ is in RCM. □

$L = \mu(R \cap [C])$, $\overline{L} = \overline{\mu(R)} \cup \mu(R \cap [C])$

In general, we can not state that the complement $\overline{L}$ of a language $L = \langle R, C, \mu \rangle$ in RCM is also in RCM. Note that $\overline{L} = L_1 \cup L_2$ where L_1 is the regular language $\overline{\mu(R)}$, which can be specified as $L_1 = \langle \overline{\mu(R)}, \bigvee_i (|w|_{\sigma_i} > 0), id \rangle$, and $L_2 = \mu(R \cap [\neg C])$. In general it is not true that a morphism that is injective on $R \cap [C]$ is also injective on $R \cap [\neg C]$, so we can not state that $L_2 \in$ RCM. Nevertheless, if μ turns out to be injective on R then the language $L_2 = \langle R, \neg C, \mu \rangle$ is in RCM, and so $\overline{L} \in$ RCM. For instance, the morphism used in Example 2 is injective on R, hence the complement of the language $\{a^{2i} b^{3j} a^i b^j | i, j > 0\}$ is in RCM.

A similar result holds for the set difference, since for any two languages A, B one has $A \setminus B = \bar{B} \cap A$. Therefore, we can state:

Corollary 6. *Let $L = \langle R, C, \mu \rangle$ be a language in* RCM *such that the morphism μ is injective on R. Then, the language $\overline{L}$ is in* RCM*, as well as the language $M \setminus L$, for all $M \in$* RCM.

The linear differential equation with polynomial coefficients satisfied by the generating function $\phi_L(x)$ of $L = \langle R, C, \mu \rangle \in$ RCM is the same equation satisfied by the generating function $\phi_{L'}(x)$ of the language $L' = R \cap [C]$ in LCL$_R$. Such an equation can be obtained from R and C through a (technical and quite tedious) symbolic computation that exploits the closure properties of the class of the holonomic functions. In [22] one can find the elimination process (in the Weyl Algebra) that given the annihilators of $f(x)$ and $g(x)$ determines the annihilator of $f(x)+g(x)$ ($f(x)\cdot g(x)$, $\int f(x)$, and so on). We refer to [17] for the details of the construction of the annihilator of the generating function of a language in LCL (or in LCL_R). We only point out that the degree of the annihilator may grow exponentially (in the number of linear constraints in C). So, if we are interested in solving the counting problem for L, where the complexity is measured with respect to the input size n (the length of the words we have to count), the cost of computing the annihilator can be considered as a (possibly large) constant.

We can also exploit the holonomicity of the generating functions of languages in RCM to prove some decidability results for RCM. Indeed, one has:

Theorem 7. *The following problems are decidable for* RCM*: equivalence, inclusion, disjointness, emptiness and universe.*

Proof. We recall that for any language $L=\langle R, C, \mu\rangle \in$ RCM, one can effectively compute a linear differential equation with polynomial coefficients satisfied by $\phi_L(x)$ by determining an annihilator of $\phi_{R\cap[C]}(x)$ as illustrated in [17].

Thus, consider two languages $L, M \in$ RCM and their intersection $T=L\cap M$, which is in RCM due to Theorem 5. Trivially, one has $L\subseteq M$ if and only if $L=T$. Moreover, it is immediate to see that $L=T$ if and only if $\phi_L(x)=\phi_T(x)$. So, we first determine an annihilator A_1 of $\phi_L(x)$ together with an annihilator A_2 of $\phi_T(x)$. Then, having as input A_1 and A_2, we compute an annihilator A_3 of the function $\phi_L(x)-\phi_T(x)$. If the order of A_3 is r, we compute the $2r$ coefficients $a_0, a_1, \ldots, a_{r-1}$ and $b_0, b_1, \ldots, b_{r-1}$, with $\phi_L(x)=\sum_{n\geq 0} a_n x^n$ and $\phi_T(x)=\sum_{n\geq 0} b_n x^n$, and we simply check that $a_i=b_i$ for $0\leq i<r$. This implies that $a_i=b_i$ for all $i\geq r$.

We proceed similarly for the other decision problems. Indeed, one has:

(equivalence) $L=M$ if and only if $L=T$ and $M=T$;
(disjointness) $L\cap M=\emptyset$ if and only if $\phi_{L\cap M}(x)=0$;
(emptiness) $L=\emptyset$ if and only if $\phi_L(x)=0$;
(universe) $L=\Sigma^\star$ if and only if $\phi_L(x)=\frac{1}{1-kx}$ where $k=|\Sigma|$ (note that any rational function satisfies a linear differential equation of the first order). □

As a matter of fact, the complexity of the above problems depends on the cost of computing the annihilators of languages in RCM (exponential in the number of linear constraints).

We point out that the decidability of the emptiness and disjointness problems for RCM can be proved in an alternative way. Indeed, in the next section we see that any language in RCM belongs also to NFCM$(k,0,1)$. Since emptiness and disjointness are decidable for NFCM(k,m,n) [13, Thm 3.1] the result follows.

4. Relationship with other classes

In this section we compare RCM to other classes of languages defined in Section 2 and recall some known result. We first show that the class RCM strictly includes LCL_R.

Theorem 8. $\mathsf{LCL}_R \subsetneq$ RCM *(or, equivalently,* $\mathcal{L}_{\mathsf{LPA}} \subsetneq$ RCM*).*

Proof. The relation $\mathsf{LCL}_R \subseteq$ RCM directly follows from the definition of RCM by taking the identity morphism. Furthermore, one can easily show that RCM strictly includes LCL_R by considering the Examples 4 or 5, where languages in RCM but not in $\mathcal{L}_{\mathsf{LPA}}$ are shown. □

The class RCM is not contained in the class of languages recognized by one-way deterministic reversal bounded counter machines. Indeed, one has the following:

Theorem 9. RCM $\not\subseteq \mathcal{L}_{\mathsf{DFCM}}$.

Proof. Let $\Sigma=\{a,b\}$ and consider the language $\Sigma\mathrm{ANBN}=\{wa^nb^n \mid n>0, w\in\Sigma^\star\}$, which is in $\mathcal{L}_{\mathsf{NFCM}}$ but not in $\mathcal{L}_{\mathsf{DFCM}}$ (due to [5, Lemma 3.15]). Thus, it is sufficient to show that $\Sigma\mathrm{ANBN}\in$ RCM. Consider the alphabet $\Gamma=\{a_1,a_2,b_1,b_2\}$, the language R described by the regular expression $(a_1+b_1)^\star a_2^+ b_2^+$, the system of linear constraints C given by $(|w|_{a_2}=|w|_{b_2})\wedge(|w|_{a_2}>0)$ and the morphism $\mu:\Gamma^\star\mapsto\Sigma$ defined by $\mu(a_1)=\mu(a_2)=a$ and $\mu(b_1)=\mu(b_2)=b$. Trivially, one has $\Sigma\mathrm{ANBN}=\langle R, C, \mu\rangle$ and we only need to show that μ is injective on $R\cap[C]$.

We argue by contradiction and suppose that there exist two different words $u, v\in R\cap[C]$ such that $\mu(u)=\mu(v)=w$. If there exist i and j, with $i\neq j$, such that $a_2^ib_2^i$ is a suffix of u and $a_2^jb_2^j$ is a suffix of v, then one has $\mu(u)\neq\mu(v)$. So, we necessarily have $i=j$ and $u=\alpha y$, $v=\beta y$, with $y=a_2^ib_2^i$ and $\alpha,\beta\in\{a_1,b_1\}^\star$. Lastly, by definition of μ, if $\alpha\neq\beta$ one has $\mu(\alpha)\neq\mu(\beta)$ and so $\mu(u)\neq\mu(v)$. □

Now, we consider the relationship between RCM and the class of languages recognized by one-way 1-reversal bounded nondeterministic counter machines. We recall that a language is recognized by a counter machine in NFCM if and only if it is recognized by a Parikh automaton in PA, that is, $\mathcal{L}_{\mathsf{PA}} = \mathcal{L}_{\mathsf{NFCM}}$ (see [5, Proposition 3.13]). Since PA is closed under homomorphisms [14, Property 4] and Proposition 1 holds, we may deduce that $\mathsf{RCM} \subseteq \mathcal{L}_{\mathsf{NFCM}}$.

This result is shown explicitly in the theorem below, which also provides an upper bound on the number of counters that are needed to recognize a language in RCM.

RCM ⊆ NFCM

Theorem 10. *Let $L = \langle R, C, \mu\rangle$ be a language in* RCM. *If the alphabet of R has p symbols and C consists of r linear constraints, Then, there exists $M \in \mathsf{NFCM}((p+1)\times r, 0, 1)$ such that $L = L(M)$.*

Proof. Let $\Sigma = \{\sigma_1, \dots, \sigma_h\}$ and $L = \langle R, C, \mu\rangle \subseteq \Sigma^\star$. Suppose that R is recognized by a DFA $A = (Q, \Gamma, \delta, q_0, F)$, with $\Gamma = \{\tau_1, \dots, \tau_p\}$, and that C consists of r linear constraints on the occurrences of symbols in Γ.

Since the value $|w|_{\tau_j}$ possibly appears in each of the r linear constraints, we provide r independent counters for each symbol $\tau_j \in \Gamma$. Furthermore, we add one counter for each linear constraint in C (used to verify that constraint). Thus, we have $(p+1)\times r$ counters, say $counter_{(j,i)}$, with $1 \le j \le p+1$ and $1 \le i \le r$.

We construct a nondeterministic counter machine

$$M = ((p+1)\times r, Q_M, \Sigma, \$, \delta_M, q_0, \{f\})$$

in $\mathsf{NFCM}((p+1)\times r, 0, 1)$ that works in two phases. First, it reads a word $w \in \Sigma^\star$ and simulates (nondeterministically) A. For any input symbol σ, M guesses which of the symbols in $\mu^{-1}(\sigma)$ causes the transition in A. Hence, it exploits the nondeterminism to increment the r counters associated with each symbol τ_j such that $\mu(\tau_j) = \sigma$. More formally, for each $q \in Q$ and for each $\tau_j \in \Gamma$ such that $\delta(q, \tau_j) = q'$, we define

$$\delta_M(q, \mu(\tau_j), c_1, \dots, c_{(p+1)\times r}) = (q', R, 0^{(j-1)\times r}1^r 0^{(p-j+1)\times r}).$$

In this phase, the automaton is in one of the states in Q and the transitions do not depend on the counters. Thus, we set $Q_M = Q \cup P \cup \{t, f\}$, where P is a set of states devoted to the second phase (i.e. when the linear constraints in C are verified), t is a trap state and f is the only final state.

Indeed, for each $q \in F$, let $\delta_M(q, \$, c_1, \dots, c_{(p+1)\times r}) = (p_0, S, 0^{(p+1)\times r})$. Thus, M on input w reaches the state $p_0 \in P$ if and only if there exists $w \in R$ such that $\mu(\hat{w}) = w$.

More precisely, in M there are as many computations leading to p_0 as words $\hat{w}$ such that $\mu(\hat{w}) = w$. Each of these computations uses the counters to store the Parikh vector of $\hat{w}$. So, in p_0, for all j with $1 \le j \le p$, one has $counter_{(j,1)} = counter_{(j,2)} = \dots = counter_{(j,r)} = |\hat{w}|_{\tau_j}$. Once in p_0, the machine verifies the system C of linear constraints. Without loss of generality, we suppose that C is in disjunctive normal form, $C = \bigvee_{i=1}^{l} \bigwedge_{g=1}^{g_i} v_{i,g}$.

See that M can easily check the tth linear constraint, say

$$v_{e,d} = \sum_{j\in J_1} c_{j,t}|w|_{\tau_j} \triangle \sum_{j\in J_2} c_{j,t}|w|_{\tau_j} \; + \; c_{p+1,t},$$

with $c_{p+1,t} \ge 0$, $\triangle \in \{=, >, \ge\}$, $c_{i,t} > 0$ for $i \in J_1 \cup J_2$, by accessing the counters $counter_{(j,t)}$, for $j \in J_1$, and computing $\sum_{j\in I_1} c_{j,t}|w|_{\tau_j}$ in $counter_{(p+1,t)}$. Then, M decrements $counter_{(p+1,t)}$ (step by step) by using $c_{p+1,t}$ and the values stored in $counter_{(j,t)}$, for $j \in J_2$. This implies that the number of states of P depends on r and p. Lastly, M enters the final state f if and only if there exists i such that $\bigwedge_{g=1}^{g_i} v_{i,g}$ is satisfied. Otherwise, no word $\hat{w} \in R$ such $\mu(\hat{w}) = w$ satisfies C, and M halts in the (non-final) state t. □

By considering the decidability results for the class RCM, one can easily prove that $\mathcal{L}_{\mathsf{NFCM}}$ strictly includes RCM.

Theorem 11. $\mathsf{RCM} \subsetneq \mathcal{L}_{\mathsf{NFCM}}$ *(or, equivalently,* $\mathsf{RCM} \subsetneq \mathcal{L}_{\mathsf{PA}}$*).*

Proof. By Theorem 7 the universe problem is decidable for RCM. So, if one had $\mathsf{RCM} = \mathcal{L}_{\mathsf{NFCM}}$, then the universe problem would be decidable also for $\mathcal{L}_{\mathsf{NFCM}}$, whereas it is known to be undecidable [1]. □

The previous result lets us prove that some well-known languages are not in RCM.

Example 6. The (non-context-free) language $\mathrm{COPY} = \{w\sharp w | w \in \Sigma^\star\}$ is not in RCM. Indeed, in [5, Proposition 3.7] it is shown that COPY is not in $\mathcal{L}_{\mathsf{NFCM}}$. Hence, $\mathsf{RCM} \subsetneq \mathcal{L}_{\mathsf{NFCM}}$ implies $\mathrm{COPY} \notin \mathsf{RCM}$.

It is also interesting to see the relationship between the holonomic functions and the generating functions of languages in $\mathcal{L}_{\mathsf{NFCM}}$. To this aim, one has the following:

Theorem 12. *For the class $\mathcal{L}_{\mathsf{NFCM}}$ exactly one of the following statements hold:*

- *there exists $L \in \mathcal{L}_{\mathsf{NFCM}}$ such that $\phi_L(x)$ is not holonomic;*
- *the generating function $\phi_L(x)$ of a language $L \in \mathcal{L}_{\mathsf{NFCM}}$ is holonomic but there is not an algorithm that computes the differential equation satisfied by $\phi_L(x)$ for any $L \in \mathcal{L}_{\mathsf{NFCM}}$.*

Proof. It is sufficient to note that if we could compute the differential equation with polynomial coefficients satisfied by the holonomic function $\phi_L(x)$, of any $L \in \mathcal{L}_{\mathsf{NFCM}}$, then the universe problem for $\mathcal{L}_{\mathsf{NFCM}}$ would we decidable. □

5. Conclusions and further work

We have presented a class of languages, called RCM, which is a subclass of the class of languages recognized by non-deterministic one-way 1-reversal bounded counter machines and admits several interesting properties: it is closed under union and intersection, most decision problems (equivalence, inclusion, emptiness, disjointness, universe) are decidable, and any language it contains has a holonomic generating function. Furthermore, RCM is not contained in the class $\mathcal{L}_{\mathsf{DFCM}}$.

Thus, given an integer k, one can ask whether there exists a relation between the class of languages recognized by a counter machine in $\mathsf{DFCM}(k, 0, 1)$ and the class RCM. For instance, is it possible to find a language L recognized by a machine in $\mathsf{DFCM}(k, 0, 1)$ such that $L \notin \mathsf{RCM}$? Does RCM include $\mathcal{L}_{\mathsf{DFCM}}$? Actually, we are almost certain that the languages recognized by machines in $\mathsf{DFCM}(1, 0, 1)$ are also in RCM, as we think that the content of one counter that can changes mode only once (from increasing to decreasing mode) can be simulated by defining a suitable set of constraints on a larger alphabet, together with a suitable injective morphism. This would be the first step in order to prove the following:

Conjecture 13. $\mathcal{L}_{\mathsf{DFCM}} \subsetneq \mathsf{RCM}$ *(as a consequence,* $\mathcal{L}_{\mathsf{DetPA}} \subsetneq \mathsf{RCM}$*).*

As far as we know, there is not a general result regarding the generating functions of languages recognized by suitable classes of counter machines or Parikh automata. Therefore, the previous conjecture is of particular interest since it would imply that the generating function of a language recognized by a counter machine in DFCM is holonomic. This result could be considered as the holonomic counterpart of the Chomsky–Schützenberger theorem for unambiguous context-free languages.

Lastly, we point out that the results we have obtained are interesting not only from a theoretical point of view. Indeed, they can be used to develop a system for solving the following problems for RCM: counting, equivalence, inclusion, universe, disjointness, emptiness. The design and implementation of such a system (as a package of a computer algebra system) is a challenge that we would like to tackle, as this would let us show examples that we can not deal with by means of hand computations.

Acknowledgements

The authors gratefully acknowledge the constructive comments and suggestions offered by the anonymous referees.

References

[1] B.S. Baker, R.V. Book, Reversal-bounded multipushdown machines, J. Comput. System Sci. 8 (3) (1974) 315–332.
[2] I.N. Bernstein, Modules over a ring of differential operators: study of the fundamental solutions of equations with constant coefficients, Funct. Anal. Appl. 5 (2) (1971) 89–101.
[3] I.N. Bernstein, The analytic continuation of generalized functions with respect to a parameter, Funct. Anal. Appl. 6 (4) (1972) 273–285.
[4] A. Bertoni, P. Massazza, N. Sabadini, Holonomic generating functions and context free languages, Internat. J. Found. Comput. Sci. 3 (2) (1992) 181–191.
[5] M. l Cadilhac, A. Finkel, P. McKenzie, Affine Parikh automata, RAIRO Theor. Inform. Appl. 46 (4) (2012) 511–545.
[6] N. Chomsky, M.P. Schützenberger, The algebraic theory of context-free languages, in: Computer Programming and Formal Systems, 1963, pp. 118–161.
[7] F. Chyzak, An extension of Zeilberger's fast algorithm to general holonomic functions, Discrete Math. 217 (13) (2000) 115–134.
[8] P. Duchon, On the enumeration and generation of generalized Dyck words, in: FPSAC'98, Discrete Math. 225 (1–3) (2000) 121–135.
[9] P. Flajolet, Ambiguity and transcendence, in: Automata, Languages and Programming, 12th Colloquium, in: Lect. Notes Comput. Sc., vol. 194, Springer, 1985, pp. 179–188.
[10] P. Flajolet, Analytic models and ambiguity of context-free languages, Theoret. Comput. Sci. 49 (1987) 283–309.
[11] F.R. Giles, W.R. Pulleyblank, Total dual integrality and integer polyhedra, Linear Algebra Appl. 25 (1979) 191–196.
[12] P. Henrici, Applied and Computational Complex Analysis, vol. 2, Wiley, 1977.
[13] O.H. Ibarra, Reversal-bounded multicounter machines and their decision problems, J. ACM 25 (1) (January 1978) 116–133.
[14] F. Klaedtke, H. Rueß, Parikh automata and monadic second-order logics with linear cardinality constraints, Technical report 177, Universität Freiburg, 2002.
[15] F. Klaedtke, H. Rueß, Monadic second-order logics with cardinalities, in: Automata, Languages and Programming, 30th International Colloquium, ICALP 2003, in: Lect. Notes Comput. Sc., vol. 2719, Springer, 2003, pp. 681–696.
[16] L. Lipshitz, D-finite power series, J. Algebra 122 (2) (1989) 353–373.
[17] P. Massazza, Holonomic functions and their relation to linearly constrained languages, RAIRO Theor. Inform. Appl. 27 (2) (1993) 149–161.
[18] P. Massazza, R. Radicioni, On computing the coefficients of bivariate holonomic formal series, Theoret. Comput. Sci. 346 (2–3) (2005) 418–438.
[19] M. Mishna, M. Zabrocki, Analytic aspects of the shuffle product, in: Susanne Albers, Pascal Weil (Eds.), STACS 2008, in: LIPIcs, vol. 1, 2008, pp. 561–572.

[20] B. Salvy, P. Zimmermann, GFUN: a Maple package for the manipulation of generating and holonomic functions in one variable, ACM Trans. Math. Software 20 (1994) 163–177.

[21] R.P. Stanley, Differentiably finite power series, European J. Combin. 1 (2) (1980) 175–188.

[22] D. Zeilberger, A holonomic systems approach to special functions identities, J. Comput. Appl. Math. 32 (3) (1990) 321–368.