$\mathbb{Z}$-polyregular functions

Thomas Colcombet
Université Paris Cité, CNRS, IRIF, F-75013, Paris, France

Gaëtan Douéneau-Tabot
Université Paris Cité, CNRS, IRIF, F-75013, Paris, France
Direction générale de l'armement - Ingénierie de projets, Paris, France

Aliaume Lopez
Université Paris Cité, CNRS, IRIF, F-75013, Paris, France
ENS Paris-Saclay, CNRS, LMF, France

Abstract

Given an monadic second-order formula φ with free variables $x_1, \dots, x_k$, one can define the function $\#\varphi$ mapping a word w to the number of valuations of $x_1, \dots, x_k$ satisfying φ in w. In this paper, we introduce the class of $\mathbb{Z}$-linear combinations of such functions, that we call $\mathbb{Z}$-polyregular functions. Indeed, it turns out to be closely related to the well-studied class of polyregular functions.

The main results of this paper solve two natural decision problems for $\mathbb{Z}$-polyregular functions. First, we show that one can minimise the number $k \geq 0$ of free variables which are needed to describe a function. Then, we show how to decide if a function can be defined using first-order formulas, by extending the notion of residual automaton and providing an original semantic characterisation based on aperiodicity. We also connect this class of functions to $\mathbb{Z}$-rational series.

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

Keywords and phrases Polyregular functions, monadic second-order logic, first-order logic, star free languages, residuals, factorization forests, rational series

Funding *Thomas Colcombet*: Supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No.670624) and the DeLTA ANR project (ANR-16-CE40-0007).

1 Introduction

Regular languages have been a conerstone of theoretical computer science since the 1950s. Over finite words, such languages can be described either in terms of finite automata, regular expressions, finite monoids, or monadic second-order (MSO) sentences. In this paper, we use MSO-formulas with free variables (as opposed to sentences, which define languages) as a building block towards a generalisation of regular languages in a quantitative setting. Given an MSO formula φ with free variables $x_1, \dots, x_k$, we define the function $\#\varphi$ mapping a finite word w to the number of valuations satisfying φ in w. We then consider the class of $\mathbb{Z}$-linear combinations of the counting functions $\#\varphi$, that we call $\mathbb{Z}$-polyregular functions.

Polyregular Functions. The class of *polyregular functions*, from finite words to finite words, can equivalently be computed by pebble transducers [2] or described by MSO-interpretations [4]. It has been intensively studied over the past few years. Polyregular functions with unary output (or equivalently, functions that output a nonnegative integer), that we call $\mathbb{N}$-polyregular functions, is also an active area of research (see, e.g., [8]).

$\mathbb{Z}$-polyregular functions are deeply connected to the class of polyregular functions: we claim in Section 2 that $\mathbb{Z}$-polyregular functions can be obtained as polyregular functions from A^* to $\{-1, 1\}^*$ post-composed with the sum operation. Note that results on polyregular or $\mathbb{N}$-polyregular functions cannot be directly translated to $\mathbb{Z}$-polyregular functions, since the presence of negative integers can introduce compensations in the output.

Rational series. Another well-studied class of functions from finite words to $\mathbb{Z}$ is the class of $\mathbb{Z}$-rational series (see, e.g., [1]). It forms the smallest class of functions containing the indicator functions $\mathbf{1}_L$ of regular languages L, and closed under taking external products by relative numbers, sums, Cauchy products, and Kleene stars. Several relations between MSO logics and rational series have already been investigated, e.g., [9, 12].

In Section 3 of this paper, we show that the class of $\mathbb{Z}$-polyregular functions corresponds to the subclass of $\mathbb{Z}$-rational series that can be obtained without using Kleene stars. This class also coincides with the class of $\mathbb{Z}$-rational series f which have *polynomial growth*, i.e. such that $|f(w)| = \mathcal{O}(|w|^k)$ for some $k \geq 0$, as described in Figure 1.

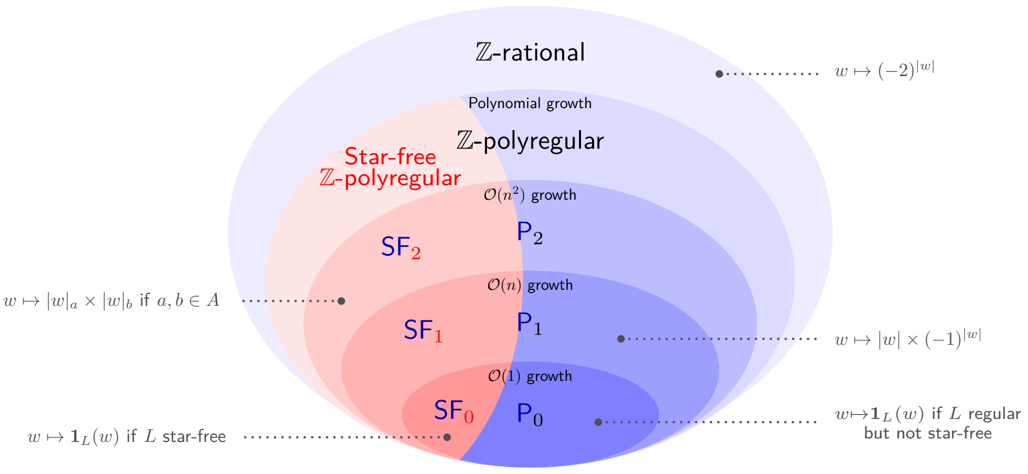

$\mathbb{Z}$-rational
$w \mapsto (-2)^{|w|}$
Polynomial growth
$\mathbb{Z}$-polyregular
Star-free $\mathbb{Z}$-polyregular
$\mathcal{O}(n^2)$ growth
SF_2
P_2
$w \mapsto |w|_a \times |w|_b$ if $a, b \in A$
$\mathcal{O}(n)$ growth
SF_1
P_1
$w \mapsto |w| \times (-1)^{|w|}$
$\mathcal{O}(1)$ growth
SF_0
P_0
$w \mapsto \mathbf{1}_L(w)$ if L star-free
$w \mapsto \mathbf{1}_L(w)$ if L regular but not star-free

Figure 1 The classes of functions studied in this paper.

Growth rate and variable minimisation. Since $\mathbb{Z}$-polyregular functions have polynomial growth, a very natural question is, given such a function f, to compute the least exponent $k \geq 0$ such that $|f(w)| = \mathcal{O}(|w|^k)$. It is known that this problem is decidable for polyregular functions [6]. Moreover, in this case, the least exponent k coincides with the least number of free variables needed to represent the function as an MSO-interpretation [3].

We prove in Section 4 that the growth rate $k \geq 0$ of a $\mathbb{Z}$-polyregular function is also computable. Furthermore, this k corresponds to the minimal integer such that the function lies in P_k, which is the class of $\mathbb{Z}$-linear combinations of $\#\varphi(x_1, \ldots, x_k)$; see Figure 1.

First-order definability. Regular languages definable by first order (FO) sentences are well-known under the name of star-free languages. Furthermore, it can be decided whether a regular language is star-free, e.g., by checking whether its minimal automaton is counter-free [14]. However, very little is known for first-order definability of functions over finite words. Indeed, contrary to languages, it is not known in general how to build a canonical object that describes a function (a notable exception is that of rational functions (not series!), for which first-order definability is decidable [10] and a canonical object can be built [11]).

In this paper, we define star-free $\mathbb{Z}$-polyregular functions as the natural restriction of $\mathbb{Z}$-polyregular functions to $\mathbb{Z}$-combinations of $\#\varphi$ where φ is a FO-formula. We show that the class of star-free $\mathbb{Z}$-polyregular function is the smallest class of functions containing the indicator functions $\mathbf{1}_L$ of star-free languages, and closed under taking external products by relative numbers, sums, and Cauchy products.

In Section 5, we show how to decide whether a $\mathbb{Z}$-polyregular function is star-free. The proof introduces a canonical object associated to the function, named its residual transducer, which can effectively be computed. Furthermore, we give a semantic characterisation of star-freeness for $\mathbb{Z}$-polyregular functions. By Schützenberger's celebrated theorem, it is well-known that a language L is star-free if and only if it is recognised by an aperiodic monoid. In other words, if $\mathbf{1}_L : A^* \to \{0,1\}$ is the characteristic function of L star-free, then for all $\alpha, u, \beta \in A^*$, $X \mapsto \mathbf{1}_L(\alpha u^X \beta)$ is a constant function for large enough X. Our semantic characterisation of star-free $\mathbb{Z}$-polyregular functions is roughly obtained by replacing "constant" by "a polynomial in X." It intuitively means that the function has no periodic behaviors. Finally, we show that this class enjoys a variable minimisation theorem: a star-free $\mathbb{Z}$-polyregular function with growth rate $k \geq 0$ lies in SF_k, which denotes the class of $\mathbb{Z}$-linear combinations of $\#\varphi(x_1, \ldots, x_k)$ for $\varphi \in \mathsf{FO}$; see Figure 1.

Outline. The relationship between $\mathbb{Z}$-polyregular functions, $\mathbb{N}$-polyregular functions, and and polyregular functions is studied in Section 2 through Example 2.5 and Proposition 2.8. We then prove in Section 3 that P is a decidable subclass of $\mathbb{Z}$-rational series, and provide an algebraic characterisation of SF (Theorems 3.1 and 3.4). In Section 4, we provide an effective algorithm to decide P_k inside P (Theorem 4.1), which is then leveraged in Section 5 to produce a residual automaton (Algorithm 1) allowing to decide SF_k inside P (Theorem 5.3).

2 Preliminaries

In the following, $\mathbb{Z}$ (resp. $\mathbb{N}$) denotes the set of integers (resp. nonnegative integers). If $i \leq j$, the set $[i{:}j]$ is $\{i, i{+}1, \ldots, j\} \subseteq \mathbb{N}$ (empty if $j < i$). The capital letter A denotes a fixed alphabet, i.e. a finite set of letters. A^* is the set of words over A. The empty word is $\varepsilon \in A^*$. If $w \in A^*$, let $|w| \in \mathbb{N}$ be its length, and for $1 \leq i \leq |w|$ let $w[i]$ be its i-th letter. If $a \in A$, let $|w|_a$ be the number of letters a occurring in w. We assume that the reader is familiar with the basics of automata theory, in particular the notions of monoid morphisms, monadic second-order (MSO) logic and first-order (FO) logic over finite words.

Counting valuations on finite words. Let MSO_k (resp. FO_k) be the set of MSO- (resp. FO-) formulas over the signature $(A, <)$ which have k free first-order variables.

▶ **Definition 2.1** (Counting). *Given $\varphi(x_1, \ldots, x_k) \in \mathsf{MSO}_k$, we let $\#\varphi : A^* \to \mathbb{N}$ be the function defined by $\#\varphi(w) = |\{\ (i_1, \ldots, i_k) \colon w \models \varphi(i_1, \ldots, i_k)\ \}|$.*

The value $\#\varphi(w)$ is the number of tuples which make the formula φ true in the model w. Notice that $|\#\varphi(w)| = \mathcal{O}(|w|^k)$ where k is the number of free variables in φ.

▶ **Example 2.2.** Let $A \coloneqq \{a, b\}$. Let $\varphi(x, y) \coloneqq a(x) \wedge b(y)$, then $\#\varphi(w) = |w|_a \times |w|_b$ for all $w \in A^*$. Let $\psi(x, y) \coloneqq \varphi(x, y) \wedge x > y$, then $\#\varphi(a^{n_0} b a^{n_1} \cdots a^{n_p} b) = \sum_{i=0}^{p} i \times n_i$.

If F is a subset of $A^* \to \mathbb{Z}$ and $S \subseteq \mathbb{Z}$, let $\mathsf{Span}_S(F) \coloneqq \{\ \sum_i a_i f_i \colon a_i \in S, f_i \in F\ \}$ be the set of *S-linear combinations* of the functions from F. The set $\mathsf{Span}_{\mathbb{N}}(\{\#\varphi : \varphi \in \mathsf{MSO}_k, k \geq 0\})$ has been recently studied in [8] under the name of *polyregular functions with unary output*. In the following, we call this class the $\mathbb{N}$-polyregular functions, since their output is in $\mathbb{N}$.

$\mathbb{Z}$-polyregular functions. The goal of this paper is to study the class of functions which are obtained by taking $\mathbb{Z}$-linear combinations of the basic $\#\varphi$ functions. In order to obtain a fine-grained description of this class, we decompose it in several subclasses depending on the number of free variables which are needed to build a function.

▶ **Definition 2.3** ($\mathbb{Z}$-polyregular). *Let* $\mathsf{P}_k := \mathsf{Span}_{\mathbb{Z}}(\{\#\varphi : \varphi \in \mathsf{MSO}_k\})$ *for* $k \geq 0$. *Let* $\mathsf{P} := \bigcup_k \mathsf{P}_k$ *be the class of* $\mathbb{Z}$-polyregular functions.

We also let $\mathsf{P}_{-1} := \{0\}$. It is easy to see that the set P_0 is exactly the set of functions of the form $\sum_i \delta_i \mathbf{1}_{L_i}$ where the $\delta_i \in \mathbb{Z}$ and the $\mathbf{1}_{L_i}$ indicator functions of regular languages. However, the sets P_k contain functions which do not have a finite image.

▶ **Example 2.4.** For all $k \geq 0$, the function $w \mapsto (-1)^{|w|} \times |w|^k$ is in P_k. Observe that it cannot be written as a single $\delta\#\varphi$ for some $\delta \in \mathbb{Z}$, $\varphi \in \mathsf{MSO}_\ell$, $\ell \geq 0$.

The use of negative coefficients has deep consequences on the expressive power of functions in P: Example 2.5 provides a non-negative $\mathbb{Z}$-polyregular function that cannot be expressed as a $\mathbb{N}$-polyregular function.

▶ **Example 2.5.** The function $f : w \mapsto (|w|_a - |w|_b)^2$ is in P_2. Moreover, it cannot be expressed as a $\mathbb{N}$-polyregular function, as $f^{-1}(\{0\})$ is not a regular language.

We say that a $\mathbb{Z}$-polyregular function is *star-free* if can be defined by using only FO-formulas. It naturally generalizes the notion of star-free languages.

▶ **Definition 2.6** (Star-free $\mathbb{Z}$-polyregular). *Let* $\mathsf{SF}_k := \mathsf{Span}_{\mathbb{Z}}(\{\#\varphi : \varphi \in \mathsf{FO}_k\})$ *for* $k \geq 0$. *Let* $\mathsf{SF} := \bigcup_k \mathsf{SF}_k$, *it is the class of* star-free $\mathbb{Z}$-polyregular functions.

We also let $\mathsf{SF}_{-1} := \{0\}$. It is easy to see that the set SF_0 is exactly the set of functions of the form $\sum_i a_i \mathbf{1}_{L_i}$ where the $a_i \in \mathbb{Z}$ and the $\mathbf{1}_{L_i}$ indicator functions of star-free languages.

▶ **Example 2.7.** The function $w \mapsto |w|_a \times |w|_b$ is in SF_1.

Now, let us observe that $\mathbb{Z}$-polyregular functions are deeply connected to the well-studied class of polyregular functions from finite words to finite words. This class can be described by several equivalent computation models, such as pebble transducers over finite words. The reader is invited to consult e.g. [2] for a formal definition of these functions, that we skip here. Let $\sigma : \{-1, 1\}^* \to \mathbb{Z}$ be the sum operation mapping w to $\sum_{i=1}^{|w|} w[i]$.

▶ **Proposition 2.8.** *The class of $\mathbb{Z}$-polyregular functions $A^* \to \mathbb{Z}$ is exactly the class of functions of the form $\sigma \circ f$ where $f : A^* \to \{-1, 1\}^*$ is polyregular.*

3 Relationship with rational series

The class of *$\mathbb{Z}$-rational series* is the smallest class of functions $A^* \to \mathbb{Z}$ which contains the indicator function $\mathbf{1}_L : A^* \to \{0, 1\}$ for all regular language $L \subseteq A^*$, and is closed under taking products with a scalar, sums, Cauchy products and Kleene stars. It is well known that it can be described by *$\mathbb{Z}$-weighted automata* (see, e.g., [1]). Let us recall the definition of the aforementioned combinators for $f, g : A^* \to \mathbb{Z}$ and $\delta \in \mathbb{Z}$:

- the *external product by a relative number* $\delta f : w \mapsto \delta \times f(w)$;
- the *sum* $f + g : w \mapsto f(w) + g(w)$;
- the *Cauchy product* $f \otimes g : w \mapsto \sum_{w=uv} f(v) \times g(w)$;
- if and only if $f(\varepsilon) = 0$, the *Kleene star* $f^* := \sum_{n \geq 0} f^n$ where $f^0 : \varepsilon \mapsto 1, w \neq \varepsilon \mapsto 0$ is neutral for Cauchy product and $f^{n+1} := f \otimes f^n$.

It follows from [8] that that the class of $\mathbb{N}$-polyregular functions coincides with the class of *$\mathbb{N}$-rational series* (defined as $\mathbb{Z}$-rational series where external products are only allowed by nonnegative integers) f which have polynomial growth, i.e. such that $|f(w)| = \mathcal{O}(|w|^k)$ for some $k \geq 0$. We intend to show an analogous result for $\mathbb{Z}$-polyregular functions.

▶ **Theorem 3.1.** *The following conditions classes of functions are equal:*
1. *$\mathbb{Z}$-polyregular functions;*
2. *$\mathbb{Z}$-rational series with polynomial growth;*
3. *the smallest class of functions containing the indicator functions of regular languages and closed under taking external products, sums and Cauchy products.*

The conversions are effective and one can decide if a $\mathbb{Z}$-rational series is $\mathbb{Z}$-polyregular.

▶ **Example 3.2.** Let $A \coloneqq \{a\}$ and let L_e (resp. L_o) be the language of words of even (resp. odd) length. Then $\mathbf{1}_{L_o} \otimes \mathbf{1}_{L_o} + \mathbf{1}_{L_e} \otimes \mathbf{1}_{L_e} - \mathbf{1}_{L_e} \otimes \mathbf{1}_{L_o} - \mathbf{1}_{L_o} \otimes \mathbf{1}_{L_e} : w \mapsto (-1)^{|w|}|w| \in \mathsf{P}_1$.

▶ Remark 3.3. There exists $\mathbb{Z}$-rational series which are not $\mathbb{Z}$-polyregular. Consider for instance the series $w \mapsto (-2)^{|w|} \notin \mathsf{P}$ over a unary input alphabet.

We also give a similar characterisation for star-free $\mathbb{Z}$-polyregular functions, that were defined using first-order logics, which justifies the "star-free" terminology.

▶ **Theorem 3.4.** *The class of star-free $\mathbb{Z}$-polyregular functions is the smallest class of functions containing the indicator functions of star-free languages and closed under taking external products, sums, and Cauchy products. The conversions are effective.*

▶ **Example 3.5.** Let $A = \{a, b\}$. Then $\mathbf{1}_{A^*a} \otimes \mathbf{1}_{A^*} : w \mapsto |w|_a \in \mathsf{SF}_1$. In a similar way, we can see that $\mathbf{1}_{A^*a} \otimes \mathbf{1}_{A^*} \otimes \mathbf{1}_{bA^*} + \mathbf{1}_{A^*b} \otimes \mathbf{1}_{A^*} \otimes \mathbf{1}_{aA^*} : w \mapsto |w|_a \times |w|_b \in \mathsf{SF}_2$.

▶ Remark 3.6. The class of (star-free) $\mathbb{Z}$-polyregular functions is also closed under Hadamard product (the componentwise product of functions, i.e. $f \times g(w) = f(w) \times g(w)$).

The rest of this section is devoted to the proof of the two aforementioned theorems. We first observe that the classes P and SF are closed under taking Cauchy products.

▷ Claim 3.7. Let $k, \ell \geq 0$. Let $f \in \mathsf{P}_k$ (resp. $f \in \mathsf{SF}_k$) and $g \in \mathsf{P}_\ell$ (resp. $g \in \mathsf{SF}_\ell$), then $f \otimes g \in \mathsf{P}_{k+\ell+1}$ (resp. $f \otimes g \in \mathsf{SF}_{k+\ell+1}$). The construction is effective.

When considering a regular (resp. star-free) language $L \subseteq A^*$ and $f \in \mathsf{P}_k$ (resp. $f \in \mathsf{SF}_k$), Claim 3.7 proves that $\mathbf{1}_L \otimes f \in \mathsf{P}_{k+1}$ (resp. $f \in \mathsf{SF}_{k+1}$). The following lemma states that such functions actually generate the whole space P_{k+1} (resp. SF_{k+1}).

▶ **Lemma 3.8.** *For every $k \geq 0$, $\mathsf{P}_{k+1} = \mathsf{Span}_{\mathbb{Z}}\left(\mathbf{1}_L \otimes f : L \text{ regular}, f \in \mathsf{P}_k\right)$ and similarly $\mathsf{SF}_{k+1} = \mathsf{Span}_{\mathbb{Z}}\left(\mathbf{1}_L \otimes f : L \text{ star-free}, f \in \mathsf{SF}_k\right)$. The conversions are effective.*

As an immediate consequence of Lemma 3.8, we obtain for all $k \geq 0$:

$$\begin{aligned} \mathsf{P}_k &= \mathsf{Span}_{\mathbb{Z}}\left(\{\mathbf{1}_{L_0} \otimes \cdots \otimes \mathbf{1}_{L_k} : L_0, \ldots, L_k \text{ regular languages}\}\right) \\ \text{and } \mathsf{SF}_k &= \mathsf{Span}_{\mathbb{Z}}\left(\{\mathbf{1}_{L_0} \otimes \cdots \otimes \mathbf{1}_{L_k} : L_0, \ldots, L_k \text{ star-free languages}\}\right). \end{aligned} \tag{1}$$

Now, we can conclude the proofs of the two theorems stated above.

Proof of Theorem 3.1 and Theorem 3.4. First note that Theorem 3.4 and (effective) equivalence between Item 1 and Item 3 in Theorem 3.1 follows from Claim 3.7 and Equation (1). By [1, pp. 103-104], we see that that Item 2 and Item 3 are equivalent, and furthermore one can decide whether a $\mathbb{Z}$-rational series has polynomial growth. However, [1] does not explain if its construction from Item 2 to Item 3 is effective. A simple way to recover effectiveness is the following: given a $\mathbb{Z}$-rational series f with polynomial growth, we enumerate all the $\mathbb{Z}$-rational series g which can be written under the form of Item 3, and check if $f = g$ (equality of $\mathbb{Z}$-rational series is decidable, see e.g. [1]). This process necessarily terminates. ◀

4 Variable minimisation and growth rate

In this section, we show that the growth of the output provides a semantic characterisation of P_k inside P. This statement is formalized in Theorem 4.1, which also provides both a decision procedure for this property and an effective conversion algorithm.

▶ **Theorem 4.1** (Variable minimisation). *Given $f \in \mathsf{P}$ and $k \geq 0$, we have $f \in \mathsf{P}_k$ if and only if $|f(w)| = \mathcal{O}(|w|^k)$. Furthermore, this property is decidable and the construction is effective.*

This result has its own interest, since it allows to "minimise" the number of free variables which are needed to represent a function. It is also a stepping stone towards computing the residual automaton of a function $f \in \mathsf{P}$, which is done in Section 5.

The rest of this section is devoted to the proof of Theorem 4.1. Let us first discuss the easy case $k = 0$. Assume that $f \in \mathsf{P}$ and $|f(w)| = \mathcal{O}(1)$, then $f(A^*)$ is finite. Moreover, for all $\delta \in \mathbb{Z}$, the language $L_\delta \coloneqq \{ w \in A^* \colon f(w) = \delta \}$ is regular, thus $f = \sum_{\delta \in f(A^*)} \delta \mathbf{1}_{L_\delta} \in \mathsf{P}_0$. Notice that if $f \in \mathsf{SF}$, then the L_δ are aperiodic and we have proven that $f \in \mathsf{SF}_0$. In the general case of $|f(w)| = \mathcal{O}(|w|^k)$ with $k \geq 1$, we crucially rely on a pumping lemma, arising from factorisations forests of bounded height associated to the function f. Namely we are going to prove the following result, from which Theorem 4.1 directly follows.

▶ **Lemma 4.2.** *Let $f \in \mathsf{P}_k$. The following three propositions are equivalent:*
1. $f \in \mathsf{P}_{k-1}$;
2. $|f(w)| = \mathcal{O}(|w|^{k-1})$;
3. $\forall \alpha_0, u_1, \alpha_1, \ldots, \alpha_{k-1}, u_k, \alpha_k \in A^*, |f(\alpha_0 \prod_{i=1}^{k} (u_i)^{X_i} \alpha_i)| = \mathcal{O}(|X_1 + \cdots + X_k|^{k-1})$.
Moreover this property can be decided and the construction is effective.

Given a morphism $\mu \colon A^* \to M$ into a finite monoid and $w \in A^*$, a μ-forest of w with respect to μ is a word over $\hat{A} \coloneqq A \uplus \{\langle, \rangle\}$ that is defined as follows.

▶ **Definition 4.3** (Factorisation forest [16]). *Given a monoid morphism $\mu \colon A^* \to M$ and $w \in A^*$, we say that F is a μ-forest of w when:*
- *either $F = a$, and $w = a \in A$;*
- *or $F = \langle F_1 \rangle \cdots \langle F_n \rangle$, $w = w_1 \cdots w_n$ and for all $1 \leq i \leq n$, F_i is a μ-forest of $w_i \in A^+$. Furthermore, if $n \geq 3$ then $\mu(w_1) = \cdots = \mu(w_n)$ is an idempotent of M.*

We write $\mathcal{F}^\mu$ to denote the set of μ-forests. Because forests can be seen as (ordered) trees, we will use the standard vocabulary to talk about the nodes, the sibling/parent relation, the root, the leaves and the depth of a forest. We let $\mathcal{F}^\mu_d$ be the set of forests with depth at most d. We are interested in building functions $\mathcal{F}^\mu_d \to \mathbb{Z}$ which are in P_k (since the set $\mathcal{F}^\mu_d$ is a regular language over $\hat{A}$, we can forget about the input words which are not in $\mathcal{F}^\mu_d$). Let $\mathsf{word} : \mathcal{F}^\mu_d \to A^*$ be the function mapping a μ-forest of $w \in A^*$ to w itself.

▶ **Example 4.4.** Let $M \coloneqq (\{-1, 1, 0\}, \times)$. A forest $F \in \mathcal{F}^\mu_5$ (where μ maps a word to the product of its elements) such that $\mathsf{word}(F) = (-1)(-1)0(-1)000000$ is depicted in Figure 2. Double lines denote idempotent nodes (i.e. nodes with more than 3 children).

When M is a finite monoid, it is known that any word in A^* has a μ-factorisation of depth at most $3|M|$. This celebrated result allows to ensure the existence of forests having numerous idempotent nodes over large enough words. Furthermore, this small factorisation can be computed by a simple class of *regular functions* from words to words (see e.g. [8]).

▶ **Theorem 4.5** (Simon, [16, 5]). *For all morphism $\mu \colon A^* \to M$ into a finite monoid M, one can build a* regular *function $\mathsf{forest} : A^* \to \mathcal{F}^\mu_d$ such that $\mathsf{word} \circ \mathsf{forest}$ is the identity function.*

Following the classical connections between MSO-formulas and regular languages, we claim that for every function $f \in \mathsf{P}_k$ there exists a monoid M and a morphism $\mu \colon A^* \to M$, such that $f(w)$ can be reconstructed using "simple" MSO-formulas which are evaluated along bounded-depth μ-factorisations of w.

Let us formalize this claim. If $\mu \colon A^* \to M$ and $d \in \mathbb{N}$ are fixed, we write $\mathsf{between}_m(x, y)$ for the (MSO-definable) predicate that evaluates to $\top$ if and only if $x \leq y$ and $\mu(w[x] \dots w[y]) = m$ where $w := \mathsf{word}(F)$. We add $\mathsf{left}_m(x)$ that evaluates to $\top$ if and only if $\mu(w[1] \dots w[x]) = m$ and $\mathsf{right}_m(x)$ that evaluates to $\top$ if and only if $\mu(w[x] \dots w[|w|]) = m$. Finally, we write $\mathsf{isleaf}(x)$ for the predicate asserting that x is a leaf in the forest F. We will be interested in a simple fragment INV of MSO over $\mathcal{F}^{\mu}_{d}$ obtained by restricting formulas to quantifier free formulas using only the predicates $\mathsf{between}_m$, left_m, and right_m where m ranges over M, and where every free variable x is guarded by the predicate $\mathsf{isleaf}(x)$.

▶ **Lemma 4.6.** *For all $f \in \mathsf{P}_k$, one can build a finite monoid M, a depth $d \in \mathbb{N}$, a surjective morphism $\mu \colon A^* \to M$, constants $\delta_i \in \mathbb{Z}$, formulas $\psi_i(x_1, \dots, x_k) \in \mathsf{INV}$, such that for every word $w \in A^*$, for every factorisation forest $F \in \mathcal{F}^{\mu}_{d}$ of w, $f(w) = \sum_{i=1}^{n} \delta_i \times \#\psi_i(F)$.*

In the rest of this section, we therefore focus on the number of free variables in $\mathbb{Z}$-linear combinations of $\#\psi_i$ where $\psi \in \mathsf{INV}$. The crucial idea is that one can leverage the structure of the forest $F \in \mathcal{F}^{\mu}_{d}$ to compute $\#\psi$ more efficiently, at the cost of building a non-INV formula. For that, we explore the structure of the forest F by extracting skeletons, a notion which was introduced in [8] for the study of $\mathbb{N}$-polyregular functions. Informally, the skeleton of a node is a subforest rooted at that node containing only its right-most and left-most children, in a recursive way.

▶ **Definition 4.7.** *Let $F \in \mathcal{F}^{\mu}$ and $\mathfrak{t} \in \mathsf{Nodes}(F)$, we define the skeleton of $\mathfrak{t}$ by:*

- *if $\mathfrak{t} = a \in A$ is a leaf, then $\mathsf{Skel}(\mathfrak{t}) := \{\mathfrak{t}\}$;*
- *otherwise if $\mathfrak{t} = \langle F_1 \rangle \cdots \langle F_n \rangle$, then $\mathsf{Skel}(\mathfrak{t}) := \{\mathfrak{t}\} \cup \mathsf{Skel}(F_1) \cup \mathsf{Skel}(F_n)$.*

Given a word w, a μ-factorisation F of w, and a node $\mathfrak{t} \in \mathsf{Nodes}(F)$, the set of nodes $\mathsf{Skel}(\mathfrak{t})$ defines a μ-factorisation of a subword u of w: the one obtained by concatenating the leaves of F that are in $\mathsf{Skel}(\mathfrak{t})$. See Figure 2 for an example of skeleton. A crucial property of $\mathsf{Skel}(\mathfrak{t})$ seen as a forest is that $\mu(\mathsf{word}(\mathsf{Skel}(\mathfrak{t}))) = \mu(F')$ where F' is the subforest of F rooted at $\mathfrak{t}$, because we only remove inner idempotent nodes.

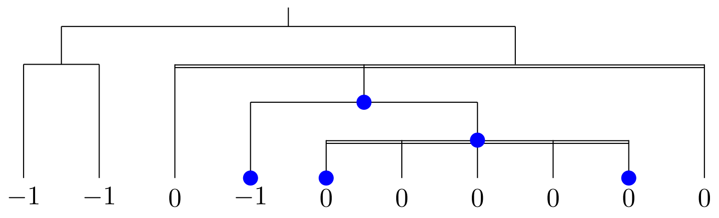


Figure 2 A forest F with $\mathsf{word}(F) = (-1)(-1)0(-1)000000$ together with a skeleton in blue.

▷ Claim 4.8. Let $x \in \mathsf{Leaves}(F)$, there exists $\mathfrak{t} \in \mathsf{Nodes}(F)$ such that $x \in \mathsf{Skel}(\mathfrak{t})$. Furthermore, if $\mathfrak{t}'$ is such that $x \in \mathsf{Skel}(\mathfrak{t}')$, then $\mathsf{Skel}(\mathfrak{t}) \subseteq \mathsf{Skel}(\mathfrak{t}')$ or $\mathsf{Skel}(\mathfrak{t}') \subseteq \mathsf{Skel}(\mathfrak{t})$.

It follows from Claim 4.8 that the set $\mathsf{Leaves}(F)$ can be partitioned depending on the maximal skeleton (for inclusion) which contains a leaf.

▶ **Definition 4.9** (Skeleton of a leaf). *Let $\mathsf{skel\text{-}root} : \mathsf{Leaves}(F) \to \mathsf{Nodes}(F)$ mapping a leaf x to the $\mathfrak{t} \in \mathsf{Nodes}(F)$ such that $x \in \mathsf{Skel}(\mathfrak{t})$ and $\mathsf{Skel}(\mathfrak{t})$ is maximal for inclusion.*

Skeletons enable us to introduce a notion of dependency for leaves, inspired by [8].

▶ **Definition 4.10** (Observation). *We say that* $\mathfrak{t}' \in \mathsf{Nodes}(F)$ *observes* $\mathfrak{t} \in \mathsf{Nodes}(F)$ *if either* $\mathfrak{t}'$ *is a parent of* $\mathfrak{t}$*, or the immediate left or right sibling of a parent of* $\mathfrak{t}$*, or an immediate sibling of* $\mathfrak{t}$*, or* $\mathfrak{t}' = \mathfrak{t}$*.*

▶ **Definition 4.11** (Dependent leaves). *In a forest* F*, a leaf* y *depends on a leaf* x *when* $\mathsf{skel\text{-}root}(y)$ *observes* $\mathsf{skel\text{-}root}(x)$*.*

Beware that this relation x depends-on y is not symmetric. This allows to ensure that the number of leaves y that depends on a fixed leaf x is uniformly bounded.

▷ Claim 4.12. Given $d \geq 0$, there exists a (computable) bound $N_d \in \mathbb{N}$ such that for every $F \in \mathcal{F}^{\mu}_{d}$ and every leaf $x \in \mathsf{Leaves}(F)$, there exists at most N_d leaves which depend on x.

It is a routine check that for every fixed d, one can define the predicate $\mathsf{sym\text{-}dep}(x, y)$ in MSO over $\mathcal{F}^{\mu}_{d}$ checking whether x depends on y or y depends on x, i.e., the *symmetrised* version of x depends-on y. We generalize this predicate to tuples $\vec{x} := (x_1, \ldots, x_k)$ via:

$$\mathsf{sym\text{-}dep}(\vec{x}) := \begin{cases} \top & \text{for } k = 0; \\ \top \text{ if and only if } x_1 \text{ is the root} & \text{for } k = 1; \\ \bigvee_{i \neq j} \mathsf{sym\text{-}dep}(x_i, x_j) & \text{otherwise.} \end{cases}$$

Notice that the independence (or dependence) of a tuple of leaves $\vec{x}$ only depends on the tuple $\mathsf{skel\text{-}root}(x_1), \ldots, \mathsf{skel\text{-}root}(x_n)$. The notion of dependent leaves is motivated by the fact that counting dependent leaves can be done with less variables, as shown in Lemma 4.13.

▶ **Lemma 4.13.** *Let* $d \geq 0$*,* $k \geq 1$*, and let* $\psi(x_1, \ldots, x_k)$ *be an* MSO *formula over* $\mathcal{F}^{\mu}_{d}$*, then* $\#(\psi(\vec{x}) \wedge \mathsf{sym\text{-}dep}(\vec{x})) : \mathcal{F}^{\mu}_{d} \to \mathbb{Z}$ *effectively belongs to* P_{k-1}*.*

Now, let $f \in \mathsf{P}_k$ verifying Item 3 in Lemma 4.17. Thanks to Lemma 4.6 and Theorem 4.5, there exists $\mu : A^* \to M$, $d \geq 0$, $\delta_i \in \mathbb{Z}$, $\psi_i(x_1, \ldots, x_k) \in \mathsf{INV}$ such that:

$$\begin{aligned} f &= \left(\sum_{i=1}^{n} \delta_i \# \psi_i\right) \circ \mathsf{forest} \\ &= \underbrace{\left(\sum_{i=1}^{n} \delta_i \#(\psi_i \wedge \mathsf{sym\text{-}dep})\right) \circ \mathsf{forest}}_{:= f_{\mathsf{dep}}} + \underbrace{\left(\sum_{i=1}^{n} \delta_i \#(\psi_i \wedge \neg\, \mathsf{sym\text{-}dep})\right) \circ \mathsf{forest}}_{:= f_{\mathsf{indep}}} \end{aligned} \tag{2}$$

Using Lemma 4.13, $f_{\mathsf{dep}} \in \mathsf{P}_{k-1}$. Since $\mathbb{Z}$-polyregular are closed under pre-composition by a regular function (it is a consequence of Proposition 2.8), it suffices to prove that $f_{\mathsf{indep}} = 0$ to conclude that $f \in \mathsf{P}_{k-1}$. Assuming that $f_{\mathsf{indep}} \neq 0$, we are going to build a "pumping family" of input forests so that f_{indep} exhibits a asymptotic behavior of $\Omega(|F|^k)$, which is a contradiction. For that, let us define how we are going to pump words.

▶ **Definition 4.14** (Pumping family). *A pumping family of size* $k \geq 1$ *is given by words* $\alpha_0, u_1, \alpha_2, \ldots, \alpha_{k-1}, u_k, \alpha_k \in A^*$*. Given* $\bar{X} = X_1, \ldots, X_k \geq 0$ *the family provides the word* $w^{\bar{X}} := \alpha_0 \prod_{i=1}^{k} (u_i)^{X_i} \alpha_i$ *together with a* μ*-factorisation* $F^{\bar{X}}$ *of* $w^{\bar{X}}$ *of depth a most* d*.*

▶ Remark 4.15. A pumping family of size k satisfies that $|w^{\bar{X}}| = \theta(X_1 + \cdots + X_k)$, and $|F^{\bar{X}}| = \theta(X_1 + \cdots + X_k)$ since its depth is bounded by d.

▶ **Lemma 4.16.** *Let f_{indep} be defined as in Equation (2). If $f_{\mathsf{indep}} \neq 0$, there exists a pumping family of size k such that $f(F^{\bar{X}})$ is asymptotically a $\mathbb{Z}$-polynomial in $X_1, \dots, X_k$ with a non-zero coefficient for $X_1 \dots X_k$. Moreover, one can decide whether $f_{\mathsf{indep}} = 0$.*

Now, we are almost ready to conclude the proof of Lemma 4.2. The only difficulty left is handled by the following technical lemma which enables to lift a bound on the asymptotic growth of polynomials to a bound on their respective degrees. It is also reused in Section 5.

▶ **Lemma 4.17** (Bounding polynomials). *Let $P, Q \in \mathbb{R}[X_1, \dots, X_n]$, $C, N_0 \geq 0$, be such that for all $x_1, \dots, x_n \in \mathbb{N}_{\geq N_0}$, $|P(x_1, \dots, x_n)| \leq C|Q(x_1, \dots, x_n)|$. Then $\deg(P) \leq \deg(Q)$.*

Proof of Lemma 4.2. The only non-trivial implication is Item 3 $\implies$ Item 1. Let $f \in \mathsf{P}_k$ verifying the conditions of Item 3. We can decompose this function following Equation (2). As observed above, we only need to show that $f_{\mathsf{indep}} = 0$.

Consider a pumping family $(w^{\bar{X}}, F^{\bar{X}})$ of size k, we have:

$$|f_{\mathsf{indep}}(F^{\bar{X}})| = |f(w^{\bar{X}}) - f_{\mathsf{dep}}(F^{\bar{X}})| = \mathcal{O}(|X_1 + \cdots + X_k|^{k-1}).$$

Assume by contradiction that $f_{\mathsf{indep}} \neq 0$, this provides us with a pumping family such that $f_{\mathsf{indep}}(F^{\bar{X}})$ is asymptotically a polynomial with non-zero coefficient for $X_1 \dots X_k$, in particular, it is of degree greater than k. However, this polynomial is bounded asymptotically by $(X_1 + \cdots + X_k)^{k-1}$, hence Lemma 4.17 yields a contradiction.

The construction of forest, f_{dep}, and f_{indep} are effective, therefore so is our procedure. Moreover, one can decide whether $f_{\mathsf{indep}} = 0$ thanks to Lemma 4.16. ◀

5 Deciding star-freeness

Now, we intend to show that given a $\mathbb{Z}$-polyregular function, we can decide if it is star-free. Furthermore, we provide a semantic characterisation of star-free $\mathbb{Z}$-polyregular functions thanks to the notion of ultimate periodicity, which is a generalisation of aperiodicity.

▶ **Definition 5.1.** *We say that $f : A^* \to \mathbb{Z}$ is ultimately polynomial if there exists $\omega \geq 0$ such that for all $\ell \geq 0$, for all $\alpha_0, u_1, \alpha_1, \dots, u_\ell, \alpha_\ell \in A^*$, the function $X_1, \dots, X_\ell \mapsto f(\alpha_0 u_1^{X_1} \alpha_1 \cdots u_\ell^{X_\ell} \alpha_\ell)$ is a polynomial of $\mathbb{Z}[X_1, \dots, X_\ell]$ for $X_1, \dots, X_\ell \geq \omega$.*

▶ **Example 5.2.** It is easy to see that $u \mapsto |u|_a \times |u|_b$ is ultimately polynomial. On the other hand, the function $u \mapsto (-1)^{|u|} \times |u|$ is not ultimately polynomial since $X \mapsto (-1)^X X$ is not a polynomial, even for X large enough.

We can now state the main result of this section.

▶ **Theorem 5.3.** *Let $k \geq 0$ and $f \in \mathsf{P}_k$. The following properties are equivalent:*
1. *f is star-free $\mathbb{Z}$-polyregular (i.e. $f \in \mathsf{SF}$);*
2. *$f \in \mathsf{SF}_k$;*
3. *f is ultimately polynomial.*

Furthermore, this property is decidable and the constructions are effective.

In order to show this result, we shall introduce the notion of residual transducer, a machine which describes a $\mathbb{Z}$-polyregular function in a somehow canonical way. Contrary to the residual automaton of a language, this machine may not be unique, but *any* residual transducer is canonical enough in our context.

Before giving to the proof of Theorem 5.3 in the rest of this section, we note that it implies an analogous of Theorem 4.1 for the classes SF_k. We are not aware of a direct proof of Corollary 5.4, since the proof of the variable minimisation result for MSO relies on factorisation forests (see Section 4), which cannot be defined in FO.

▶ **Corollary 5.4** (First-order variable minimisation). *Let $f \in \mathsf{SF}$, then $f \in \mathsf{SF}_k$ if and only if $|f(w)| = \mathcal{O}(|w|^k)$. This property is decidable and the construction is effective.*

Proof. Let $f \in \mathsf{SF}$ be such that $|f(w)| = \mathcal{O}(|w|^k)$. By Theorem 4.1 we get $f \in \mathsf{P}_k$, thus by Theorem 5.3, $f \in \mathsf{SF}_k$. All the steps are effective and decidable. ◀

5.1 Residuals of a function

We first introduce the notion of residual of a function $f : A^* \to \mathbb{Z}$ under a word $u \in A^*$.

▶ **Definition 5.5** (Residual). *Given $f : A^* \to \mathbb{Z}$ and $u \in A^*$, we define the function $u \triangleright f : A^* \to \mathbb{Z}, w \mapsto f(uw)$. We let $\mathsf{Res}(f) := \{u \triangleright f : u \in A^*\}$.*

It is easy to see that $u \mapsto u \triangleright f$ defines a monoid action of A^* and $A^* \to \mathbb{Z}$. Let us observe that this action (effectively) preserves the classes of functions P_k for $k \geq -1$.

▷ Claim 5.6. Let $k \geq 0$ and $f \in \mathsf{P}_k$, then $u \triangleright f \in \mathsf{P}_k$ for all $u \in A^*$. Furthermore $u \triangleright f$ can effectively be written as a linear combination of counting MSO_k formulas.

Note that if $L \subseteq A^*$ and $u \in A^*$, then $u \triangleright \mathbf{1}_L$ is the characteristic function of the well-known residual language $u^{-1}L := \{v \in A^* : uv \in L\}$. In particular, the set $\{u \triangleright \mathbf{1}_L : u \in A^*\}$ is finite. However, given $f \in \mathsf{P}_k$ for $k \geq 1$, the set $\{u \triangleright f : u \in A^*\}$ may not be finite in general (consider e.g. $f : u \mapsto |u|$). We now intend to show that this set is still finite, up to an identification of the functions whose difference is in P_{k-1}.

▶ **Definition 5.7.** *Given $k \geq -1$ and $f, g : A^* \to \mathbb{Z}$, we let $f \sim_k g$ if and only if $f - g \in \mathsf{P}_k$*

If $f, g \in \mathsf{P}$ and $k \geq 0$, then $f \sim_k g$ if and only if $|(f-g)(w)| = \mathcal{O}(|w|^k)$ and this property is decidable by Theorem 4.1. For $k = -1$, we have $f \sim_k g$ if and only if $f = g$. This property is also decidable since $\mathbb{Z}$-polyregular functions can effectively be transformed into $\mathbb{Z}$-rational series by Theorem 3.1, for which equality is decidable [1].

Now, let us observe that the $\triangleright$ action is compatible with the $\sim_k$ equivalence classes.

▷ Claim 5.8. For all $k \geq 0$, $u \in A^*$ and $f, g : A^* \to \mathbb{Z}$, if $f \sim_k g$, then $u \triangleright f \sim_k u \triangleright g$.

Finally, we describe an interaction between residuation and Cauchy product. Claim 5.9 will be especially useful since $\mathsf{P}_{k+1} = \mathsf{Span}_{\mathbb{Z}}(\{\mathbf{1}_L \otimes g : g \in \mathsf{P}_k, L \text{ regular}\})$.

▷ Claim 5.9. Let $k \geq -1$, $g \in \mathsf{P}_k$, L regular and $u \in A^*$. Then $u \triangleright (\mathbf{1}_L \otimes g) \sim_k (u \triangleright \mathbf{1}_L) \otimes g$.

Now, we have all the ingredients to show that for all $f \in \mathsf{P}_k$, $\mathsf{Res}(f)$ is finite up to identifying the functions in P_{k-1}. This is the purpose of Lemma 5.10.

▶ **Lemma 5.10.** *Let $k \geq 0$ and $f \in \mathsf{P}_k$, then $\mathsf{Res}(f)/\sim_{k-1}$ is finite.*

5.2 Residual transducers

Now we intend to show that a function $f \in \mathsf{P}_k$ can effectively be computed by a somehow *canonical* machine, whose states are based on finite set $\mathsf{Res}(f)/\sim_{k-1}$. First, let us introduce an abstract notion of transducer which can call functions on suffixes of its input.

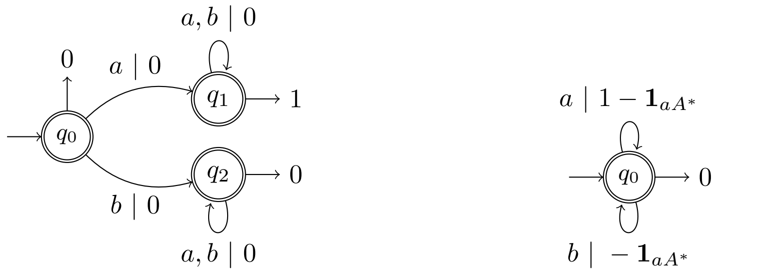
▶ **Definition 5.11** ($\mathcal{H}$-transducer). *Let $k \geq 0$ and $\mathcal{H}$ be a fixed subset of the functions $A^* \to \mathbb{Z}$. A* (deterministic) *$\mathcal{H}$-transducer $\mathcal{T} = (A, Q, q_0, \delta, \mathcal{H}, \lambda, F)$ consists of:*

- *a finite input alphabet A;*
- *a finite set of states Q with $q_0 \in Q$ initial;*
- *a transition function $\delta : Q \times A \to Q$;*
- *a labelling function $\lambda : Q \times A \to \mathcal{H}$;*
- *an output function $F : Q \to \mathbb{Z}$.*

The semantics of $\mathcal{T}$ is the following. Given $q \in Q$, we define by induction on $w \in A^*$ the value $\mathcal{T}_q(w) \in \mathbb{Z}$. For $w = \varepsilon$, we let $\mathcal{T}_q(w) := F(q)$. Otherwise let $\mathcal{T}_q(aw) := \mathcal{T}_{\delta(q,a)}(w) + \lambda(q,a)(w)$. Finally, the function computed by the $\mathcal{H}$-transducer $\mathcal{T}$ is $\mathcal{T}_{q_0} : A^* \to \mathbb{Z}$. All the functions $\mathcal{T}_q$ are total, since the output and transition functions are so.

▶ Remark 5.12. This model is somehow related to the *marble transducers* of [7]. The main differences is that it has no registers, and it calls the sub-functions from $\mathcal{H}$ on the suffix of its input instead of the prefix (but this is just a matter of symmetry).

▶ **Example 5.13.** We have depicted in Figure 3 a P_{-1}-transducer and a P_0-transducer computing the function $\mathbf{1}_{aA^*}$ for $A = \{a, b\}$. The first one can easily be identified with the minimal automaton of $\mathbf{1}_{aA^*}$ (up to considering that a state is final if it outputs 0). The second one has a single state and it "hides" its computation into the calls to P_0. One can check e.g. that $1 = \mathbf{1}_{aA^*}(aab) = (1 - \mathbf{1}_{aA^*}(ab)) + (1 - \mathbf{1}_{aA^*}(b)) - \mathbf{1}_{aA^*}(\varepsilon) + 0$.



(a) A P_{-1}-transducer computing $\mathbf{1}_{aA^*}$. **(b)** A P_0-transducer computing $\mathbf{1}_{aA^*}$

Figure 3 Two transducers computing $\mathbf{1}_{aA^*}$.

The reader may guess that every function $f \in \mathsf{P}_k$ can effectively be computed by a P_{k-1}-transducer. We provide a stronger result and show that f can be computed by some specific P_{k-1}-transducer whose transition function is uniquely defined by $\mathsf{Res}\,(f)/\sim_{k-1}$.

▶ **Definition 5.14.** *Let $k \geq 0$, let $\mathcal{T} = (A, Q, \delta, \lambda, F)$ be a P_{k-1}-transducer and $f : A^* \to \mathbb{Z}$. We say that $\mathcal{T}$ is a k-residual transducer of f if the following conditions hold:*
- *$\mathcal{T}$ computes f;*
- *$Q = \mathsf{Res}\,(f)/\sim_{k-1}$ and for all $w \in A^*$, $f_w \in \delta(q, w)$;*
- *$\lambda(Q, A) \subseteq \mathsf{Span}_{\mathbb{Z}}(\mathsf{Res}\,(f)) \cap \mathsf{P}_{k-1}$.*

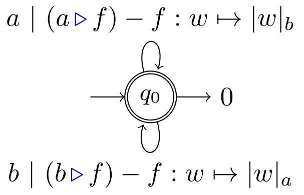
Given a regular language L, the 0-residual transducer of its indicator function $\mathbf{1}_L$ can easily be identified with the *minimal automaton* of the language L, like in Example 5.13. However, for $k \geq 1$, the k-residual transducer of $f \in \mathsf{P}_k$ may not be unique. More precisely, two k-residual transducer share the same underlying automaton (A, Q, δ, λ), but the labels λ of the transitions may not be the same.

▶ **Example 5.15.** The P_{-1}-transducer (resp. P_0-transducer) from Figure 3 is a 0-residual transducer (resp. 1-residual transducer) of $\mathbf{1}_{aA^*}$. Let us check it for the 1-residual transducer.

First note that $b \triangleright \mathbf{1}_{aA^*} \sim_0 a \triangleright \mathbf{1}_{aA^*} \sim_0 \mathbf{1}_{aA^*}$, hence $|\mathsf{Res}(\mathbf{1}_{aA^*})/\sim_0| = 1$. Thus a 1-residual transducer of $\mathbf{1}_{aA^*}$ has exactly one state q_0. Furthermore the labels of the transitions of our transducer belong to $\lambda(Q, A) \subseteq \mathsf{Span}_{\mathbb{Z}}(\mathsf{Res}_f(a))$ since $1 - \mathbf{1}_{aA^*} = (a \triangleright \mathbf{1}_{aA^*}) - f$.

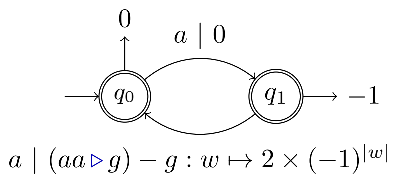
▶ **Example 5.16.** Let $A := \{a, b\}$. The function $f : w \mapsto |w|_a \times |w|_b \in \mathsf{P}_2$ has a single residual up to $\sim_1$-equivalence. A 2-residual transducer of f is depicted in Figure 4a.

▶ **Example 5.17.** Let $A := \{a\}$. The function $g : w \mapsto (-1)^{|w|} \times |w| \in \mathsf{P}_1$ has two residuals up to $\sim_0$-equivalence. A 1-residual transducer of g is depicted in Figure 4b.



$a \mid (a \triangleright f) - f : w \mapsto |w|_b$
q_0
0
$b \mid (b \triangleright f) - f : w \mapsto |w|_a$

(a) A 2-residual transducer of $w \mapsto |w|_a|w|_b$.



0
$a \mid 0$
q_0
q_1
-1
$a \mid (aa \triangleright g) - g : w \mapsto 2 \times (-1)^{|w|}$

(b) A 1-residual transducer of $u \mapsto (-1)^{|w|}|w|$.

Figure 4 Two residual transducers.

Now, let us describe how to build a k-residual transducer for any $f \in \mathsf{P}_k$.

▶ **Lemma 5.18.** *Let $k \geq 0$. Given $f : A^* \to \mathbb{Z}$ such that $\mathsf{Res}(f)/\sim_{k-1}$ is finite, one can build there exists a k-residual transducer computing f. Moreover, it can effectively be built given $f \in \mathsf{P}$.*

Proof. We apply Algorithm 1, which computes the set of residuals of f and the relations between them. The states of our machine are not be labelled by the equivalence classes of $\mathsf{Res}(f)/\sim_{k-1}$, but directly by some elements of $\mathsf{Res}(f)$. Remark that the labels on the transitions are of the form $wa \triangleright f - w \triangleright f$ when $wa \triangleright f \sim_{k-1} w \triangleright f$, hence are in $\mathsf{Span}_{\mathbb{Z}}(\mathsf{Res}(f)) \cap \mathsf{P}_{k-1}$ by definition. Now, let us justify the correctness of Algorithm 1.

First, we note that it maintains two sets O and Q such that $O \uplus Q \subseteq \mathsf{Res}(f)$ and for all $f, g \in O \uplus Q$ we have $f \neq g \implies f \not\sim_{k-1} g$. Hence the algorithm terminates since $\mathsf{Res}(f)/\sim_{k-1}$. At the end of its execution, we have for all $q \in Q$ and $a \in A$, that $\delta(q, a) \sim_{k-1} a \triangleright q$ and $\lambda(q, a) = a \triangleright q - \delta(q, a)$.

Now, let us show by induction on $n \geq 0$ that for all $a_1 \cdots a_n \in A^*$, if $q_0 \to^{a_1} q_1 \to^{a_2} \cdots \to^{a_n} q_n$ is the run labelled by $a_1 \cdots a_n$ in the underlying automaton (A, Q, q_0, δ), and $g_1 \cdots g_n$ are the functions which label the transitions, we have $q_n \sim_{k-1} a_1 \cdots a_n \triangleright f$ and for all $v \in A^*$, $f(a_1 \cdots a_n w) = \sum_{i=2}^{n} g_i(a_i \cdots a_n w) + q_n(w)$.

For $n = 0$ the result is obvious because $q_0 = f$. Now, assume that the result holds for some $n \geq 0$ and consider $a_1 \cdots a_n a_{n+1} \in A^*$. Let $q_0 \to^{a_1} q_1 \to^{a_2} \cdots \to^{a_n+1} q_{n+1}$ be the run and $g_1 \cdots g_{n+1}$ be the labels of the transitions. Since $q_n \sim_{k-1} a_1 \cdots a_n \triangleright f$ (by induction) we get $a_{n+1} \triangleright q_n \sim_{k-1} a_1 \cdots a_n a_{n+1} \triangleright f$ by Claim 5.8. Finally $q_{n+1} = \delta(q_n, a) \sim_{k-1} a_{n+1} \triangleright q_n$ (as observed in a previous paragraph) thus $q_{n+1} \sim_{k-1} a_1 \cdots a_n a_{n+1} \triangleright f$. Now, let us fix $w \in A^*$. We have $f(a_1 \cdots a_n a_{n+1} w) = \sum_{i=2}^{n} g_i(a_i \cdots a_n a_{n+1} u) + q_n(a_{n+1} w)$ by induction hypothesis. But since $g_{n+1} = a_{n+1} \triangleright q_n - \delta(q_n, a_{n+1}) = a_{n+1} \triangleright q_n - q_{n+1}$ (as observed in a previous paragraph) we get $q_n(a_{n+1} w) = g_{n+1}(w) + q_{n+1}(w)$.

We conclude the proof by considering $w = \varepsilon$ and the definition of F. ◀

▶ Remark 5.19. In Algorithm 1, we need to "choose" a way to range over the elements of O and the letters of A. Different choices may not lead to the same k-residual transducers.

Algorithm 1 Computing a k-residual transducer of $f \in \mathsf{P}_k$

```
$O \leftarrow \{f_\varepsilon\}$;
$Q \leftarrow \varnothing$;
while $O \neq \varnothing$ do
    choose $w \triangleright f \in O$;
    for $a \in A$ do
        if $wa \triangleright f \not\sim_{k-1} v \triangleright f$ for all $v \triangleright f \in O \uplus Q$ then
            $O \leftarrow O \uplus \{wa \triangleright f\}$;
            $\delta(w \triangleright f, a) = wa \triangleright f$;
            $\lambda(w \triangleright f, a) = 0$;
        else
            let $f_v \in O \uplus Q$ be such that $wa \triangleright f \sim_{k-1} v \triangleright f$;
            $\delta(w \triangleright f, a) = v \triangleright f$;
            $\lambda(w \triangleright f, a) = wa \triangleright f - v \triangleright f$;
        end
    end
    $O \leftarrow O \smallsetminus \{w \triangleright f\}$;
    $Q \leftarrow Q \uplus \{w \triangleright f\}$;
    for $w \triangleright f \in Q$ do
        $F(w \triangleright f) \leftarrow f(w)$;
    end
end
```

We deduce from Lemma 5.18 the that P_{k-1}-transducers describe exactly the class P_k.

▶ **Corollary 5.20.** *P_k is the class of functions which can be computed by a P_{k-1}-transducer. Furthermore, the conversions are effective.*

▶ **Corollary 5.21.** *For all $k \geq 0$, $\mathsf{P}_k = \{\, f : A^* \to \mathbb{Z} \colon \mathsf{Res}(f)/\sim_{k-1} \text{ is finite} \,\}$.*

5.3 Semantic characterization

Now, given $f \in \mathsf{P}_k$, we intend to use its k-residual transducer in order to decide whether $f \in \mathsf{SF}_k$. Indeed, this machine being defined in a semantical way from f, it contains somehow intrinsic information on its behavior. More precisely, we show that star-freeness faithfully translates to a counter-free property of the k-residual transducer, together with an inductive property of the functions which label its transitions.

▶ **Definition 5.22** ([14]). *A deterministic automaton (A, Q, q_0, δ) is counter-free if for all $q \in Q$, $u \in A^*$, $n \geq 1$, if $\delta(q, u^n) = q$ then $\delta(q, u) = q$.*

We say that a P_k-transducer is counter-free if its underlying automaton is so.

▶ **Example 5.23.** The P_0-transducer depicted in Figure 4b is not counter-free, since $\delta(q_0, a) \neq \delta(q_0, aa)$ and $\delta(q_0, aaa) = \delta(q_0, a)$.

The goal of this section is to show Theorem 5.24. Note that the semantic characterisation (Item 2) is not a side result: it is needed for the inductive proof of equivalence between the other items. The authors are not aware of a proof technique that would avoid it.

▶ **Theorem 5.24.** *Let $k \geq 0$ and $f \in \mathsf{P}_k$, the following conditions are equivalent:*

1. *f is star-free polyregular (i.e. $f \in \mathsf{SF}$);*
2. *f is ultimately polynomial;*
3. *any k-residual transducer of f is counter-free and has labels in SF_{k-1};*
4. *some k-residual transducer of f is counter-free and has labels in SF_{k-1};*
5. *$f \in \mathsf{SF}_k$.*

Furthermore, this property is decidable and the constructions are effective.

Before coming to the proof of Theorem 5.24, we present three technical lemmas which are needed to show it by induction on $k \geq 0$. First, we note that a counter-free transducer computes a star-free function (provided that the labels of its transitions are star-free).

▶ **Lemma 5.25.** *Let $k \geq 0$. A counter-free SF_{k-1}-transducer computes a function of SF_k. Furthermore, the conversion is effective.*

Next, we show that star-freeness implies ultimate polynomiality. The following result is shown by induction by using Lemma 3.8, and the base case is roughly a reformulation of the fact that a star-free language can be recognized by an aperiodic monoid.

▶ **Lemma 5.26.** *Let $f \in \mathsf{SF}$, then f is ultimately polynomial.*

Finally, we show that ultimate polynomiality is sufficient to show that any k-residual transducer is counter-free. Lemma 5.27 is the key ingredient for showing Theorem 5.24.

▶ **Lemma 5.27.** *Let $k \geq 0$. Let $f \in \mathsf{P}_k$ which is ultimately polynomial and $\mathcal{T}$ be a k-residual transducer of f. Then $\mathcal{T}$ is counter-free and its label functions are ultimately polynomial.*

Proof. Ultimate polynomiality is preserved under taking linear combinations and residuals. Hence the function labels of $\mathcal{T}$ are ultimately polynomial (by definition of a k-residual transducer). It remains to show that $\mathcal{T} = (A, Q, q_0, \delta, \lambda, F)$ is counter-free.

Let $\alpha, u \in A^*$ and suppose that $\delta(q_0, \alpha) = \delta(q_0, \alpha u^n)$ for some $n \geq 1$. We want to show that $\delta(q_0, \alpha u) = \delta(q_0, \alpha)$. Since $\delta(q_0, \alpha) = \delta(q_0, \alpha u^{nX})$ and $\delta(q_0, \alpha u) = \delta(q_0, \alpha u^{nX+1})$ for all $X \geq 1$, it is enough to show that $\delta(q_0, \alpha u^{nX+1}) = \delta(q_0, \alpha u^{nX})$ for some $X \geq 1$.

Let $\omega \geq 1$ given by the definition of ultimate polynomiality for f. We want to show that $(\alpha u^{n\omega+1} \triangleright f) \sim_{k-1} (\alpha u^{n\omega} \triangleright f)$, i.e. $|(\alpha u^{n\omega+1} \triangleright f)(u) - (\alpha u^{n\omega} \triangleright f)(u)| = \mathcal{O}(|u|^{k-1})$ since the residuals belong to P. For this, let us pick any $\alpha_0, u_1, \alpha_1, \cdots, u_k, \alpha_k \in A^*$. By Lemma 4.2, it is sufficient to show that:

$$|(\alpha u^{n\omega} \triangleright f - \alpha u^{n\omega+1} \triangleright f)(\alpha_0 u_1^{X_1} \cdots u_k^{X_k} \alpha_k))| = \mathcal{O}((X_1 + \cdots + X_k)^{k-1})$$

Because f is ultimately polynomial, for all $X, X_1, \cdots, X_k \geq \omega$, $f(\alpha u^X \alpha_0 u_1^{X_1} \cdots u_k^{X_k} \alpha_k)$ is a polynomial $P(X, X_1, \ldots, X_k)$. Our goal is to show that $|P(n\omega, X_1, \ldots, X_k) - P(n\omega + 1, X_1, \ldots, X_k)| = \mathcal{O}(|X_1 + \cdots + X_k|^{k-1})$. Since $f \in \mathsf{P}_k$, we have $|P(X, X_1, \ldots, X_k)| = \mathcal{O}((X + X_1 + \cdots + X_k)^k)$. Thus by Lemma 4.17, P has degree at most k, hence it can be rewritten under the form $P_0 + XP_1 + \cdots + X^k P_k$ where $P_i(X_1, \ldots, X_k)$ has degree at most $k - i$. Therefore:

$$\begin{aligned}
&|P(n\omega, X_1, \ldots, X_k) - P(n\omega + 1, X_1, \ldots, X_k)| \\
&= \left| \sum_{i=1}^{k} P_i(X_1, \ldots, X_k)((n\omega)^i - (n\omega + 1)^i) \right| \leq \sum_{i=1}^{k} |P_i(X_1, \ldots, X_k)|(n\omega + 1)^i
\end{aligned}$$

since the term P_0 vanishes when doing the subtraction. The result follows since the polynomials P_i for $1 \leq i \leq k$ have degree at most $k - 1$. ◀

Now, we have all the elements to give an inductive proof of Theorem 5.24.

Proof of Theorem 5.24. The (effective) equivalences are shown by induction on $k \geq 0$. For Item 1 $\implies$ Item 2 we apply Lemma 5.26. For Item 2 $\implies$ Item 3, we use Lemma 5.27 which shows that any k-residual transducer of f is counter-free and has ultimately polynomial labels. Since these labels are in P_{k-1}, then by induction hypothesis they belong to SF_{k-1}. For Item 4 $\implies$ Item 5 we use Lemma 5.25. The two remaining implications are obvious.

It remains to see that this property can be decided, which is also shown by induction on $k \geq 0$. Given $f \in \mathsf{P}_k$, we can effectively build a k-residual transducer of f by Lemma 5.18. If it is not counter-free, the function is not star-free polyregular. Otherwise, we can check by induction that the labels belong to SF_{k-1} (since they belong to P_{k-1}). ◀

6 Outlook

This paper describes a robust class of functions, which admits three characterizations in terms of logics, rational expressions and (canonical) transducers. Furthermore, two natural class membership problems (variable minimisation and first-order definability) are shown decidable. Interestingly, the variable minimisation result also holds for the subclass of star-free functions. We believe these results together with the technical tools introduced to prove them pave the way towards a better understanding of classes of $\mathbb{Z}$- and $\mathbb{N}$-polyregular functions. Now, let us discuss a few tracks which seem to be interesting.

Weaker logics. Boolean combination of existential first-order formulas define a well-known subclass of first-order logic, often denoted $\mathcal{B}(\exists\mathsf{FO})$. Over finite words, $\mathcal{B}(\exists\mathsf{FO})$-sentences describe the celebrated class of *piecewise testable languages* (see e.g. [15]). In our quantitative setting, one could define for all $k \geq 0$ the class of linear combinations of the counting formulas from $\mathcal{B}(\exists\mathsf{FO})_k$, as we did for P_k and SF_k. While this class seems to be a good candidate for *piecewise testable* $\mathbb{Z}$-polyregular functions, it do not admit a variable minimisation result depending on the growth rate of the functions. Indeed, let $A := \{a, b\}$ and consider the indicator function $\mathbf{1}_{aA^*} = \#\varphi$ for $\varphi(x) := a(x) \wedge \forall y. y \geq x \in \mathcal{B}(\exists\mathsf{FO})_1$. Even if $\mathbf{1}_{aA^*}$ has finite image, it cannot be written as a linear combination of counting formulas from $\mathcal{B}(\exists\mathsf{FO})_0$. Assume the converse, then $\mathbf{1}_{aA^*} = \sum_{i=1}^{n} \delta_i L_i$ for some piecewise testable languages L_i. This would imply that aA^* is piecewise testable, a contradiction.

Star-free $\mathbb{N}$-polyregular functions. A very natural question is, given a $\mathbb{N}$-polyregular function, to determine whether it can be rewritten as a *star-free* $\mathbb{N}$*-polyregular* function, i.e. an element of $\mathsf{Span}_{\mathbb{N}}(\#\varphi : \varphi \in \mathsf{FO}_k, k \geq 0)$. We conjecture that star-free $\mathbb{N}$-polyregular functions are exactly the $\mathbb{N}$-polyregular functions which are star-free $\mathbb{Z}$-polyregular.

However, the techniques of this paper cannot be applied directly to solve this problem, since a residual automaton (see Section 5) of a $\mathbb{N}$-polyregular function may need labels which are not $\mathbb{N}$-polyregular, or even not nonnegative. In other words, replacing the output group by an output monoid seems to prevent from building canonical representations.

Star-free $\mathbb{Z}$-rational series. In Figure 1, there is not equivalent of the class SF among the whole class of $\mathbb{Z}$-rational series. The authors are not aware of a way to define a class of *star-free* $\mathbb{Z}$-rational series, neither with logics nor with rational expressions. Indeed, allowing the use of Kleene star for rational series enables to build the indicator functions of all regular languages. However, it seems natural to say that $w \mapsto 2^{|w|}$ should be a star-free $\mathbb{Z}$-rational series, while $w \mapsto (-2)^{|w|}$ should not be star-free.

References

1 Jean Berstel and Christophe Reutenauer. *Rational series and their languages*, volume 12. Springer-Verlag, 1988.

2 Mikolaj Bojańczyk. Polyregular functions. *arXiv preprint arXiv:1810.08760*, 2018.

3 Mikolaj Bojańczyk. On the growth rate of polyregular functions, 2022.

4 Mikolaj Bojańczyk, Sandra Kiefer, and Nathan Lhote. String-to-string interpretations with polynomial-size output. In *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019*, pages 106:1–106:14, 2019. URL: `https://doi.org/10.4230/LIPIcs.ICALP.2019.106`, `doi:10.4230/LIPIcs.ICALP.2019.106`.

5 Thomas Colcombet. Green's relations and their use in automata theory. In *International Conference on Language and Automata Theory and Applications*, pages 1–21. Springer, 2011.

6 Gaëtan Douéneau-Tabot. Pebble transducers with unary output. In *46th International Symposium on Mathematical Foundations of Computer Science, MFCS 2021, August 23-27, 2021, Tallinn, Estonia*, 2021.

7 Gaëtan Douéneau-Tabot, Emmanuel Filiot, and Paul Gastin. Register transducers are marble transducers. In *45th International Symposium on Mathematical Foundations of Computer Science, MFCS 2020, August 24-28, 2020, Prague, Czech Republic*, 2020.

8 Gaëtan Douéneau-Tabot. Hiding pebbles when the output alphabet is unary. In *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022*, 2022.

9 Manfred Droste and Paul Gastin. Weighted automata and weighted logics. In *International Colloquium on Automata, Languages, and Programming*, pages 513–525. Springer, 2005.

10 Emmanuel Filiot, Olivier Gauwin, and Nathan Lhote. Aperiodicity of rational functions is pspace-complete. In *36th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2016)*. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016.

11 Emmanuel Filiot, Olivier Gauwin, Nathan Lhote, and Anca Muscholl. On canonical models for rational functions over infinite words. In *38th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 122. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018.

12 Paul Gastin and Benjamin Monmege. Adding pebbles to weighted automata: Easy specification & efficient evaluation. *Theoretical Computer Science*, 534:24–44, 2014.

13 J.A. Makowsky. Algorithmic uses of the feferman–vaught theorem. *Annals of Pure and Applied Logic*, 126(1):159–213, 2004. Provinces of logic determined. Essays in the memory of Alfred Tarski. Parts I, II and III. URL: `https://www.sciencedirect.com/science/article/pii/S0168007203001003`, `doi:https://doi.org/10.1016/j.apal.2003.11.002`.

14 Robert McNaughton and Seymour A Papert. *Counter-Free Automata.* The MIT Press, 1971.

15 Dominique Perrin and Jean-Eric Pin. First-order logic and star-free sets. *Journal of Computer and System Sciences*, 32(3):393–406, 1986.

16 Imre Simon. Factorization forests of finite height. *Theor. Comput. Sci.*, 72(1):65–94, 1990. URL: `https://doi.org/10.1016/0304-3975(90)90047-L`, `doi:10.1016/0304-3975(90)90047-L`.

A Proofs of Section 2

A.1 Proof of Proposition 2.8

Proof of $\Rightarrow$. Let $f \in \mathsf{P}_k$, it is written $f = \sum_{i=1}^{n} \delta_i \# \varphi_i$ with $\varphi_i \in \mathsf{MSO}_k$. We can decompose the sum into:

$$f = \underbrace{\sum_{i=1}^{n} \max(\delta_i, 0)\#\varphi_i}_{:=f_+} + \underbrace{\sum_{i=1}^{n} \min(\delta_i, 0)\#\varphi_i}_{:=f_-}$$

Note that f_+ and f_- are $\mathbb{N}$-polyregular functions, since they are $\mathbb{N}$-combinations of counting formulas. In particular, there exists a polyregular function $g_+ \colon A^* \to \{1\}^*$ such that $|g_+(w)| = f_+(w)$ and a polyregular function $g_- \colon A^* \to \{-1\}^*$ such that $|g_-(w)| = f_-(w)$. Now, let us write $g(w) = g_+(w) \cdot g_-(w)$. This function is polyregular and satisfies $\sigma \circ g = f$. ◀

Proof of $\Leftarrow$. Let us consider a polyregular function $g \colon A^* \to \{-1, 1\}^*$. The maps $g_+ \colon w \mapsto |g(w)|_1$ and $g_- \colon w \mapsto |g(w)|_{-1}$ are polyregular functions with unary output (since they correspond to a post-composition by the regular function which removes some letter, and polyregular functions are closed under post-composition by a regular function [2]). Hence g_- and g_+ are in P, and $\sigma \circ g = g_+ - g_-$ lies in P too. ◀

B Proofs of Section 3

B.1 Proof of Claim 3.7

Since $(\lambda f + \mu g) \otimes h = \lambda(f \otimes g) + \mu(g \otimes h)$ for $f, g, h : A^* \to \mathbb{Z}$ and $\lambda, \mu \in \mathbb{Z}$, it is sufficient to show the result for $f = \#\varphi$ and $g = \#\psi$ with $\varphi \in \mathsf{MSO}_k$ and $\psi \in \mathsf{MSO}_\ell$ (resp. $\varphi \in \mathsf{FO}_k$ and $\psi \in \mathsf{FO}_\ell$). This follows by expanding the definitions of $\#$ and $\otimes$.

B.2 Proof of Lemma 3.8

Let $\varphi(x_1, \ldots, x_k)$ be a formula, then:

$$\#\varphi(w) = \sum_{\emptyset \subsetneq P \subseteq \{1,\ldots,k\}} \sum_{w=uv} \#(\varphi \wedge \bigwedge_{i \in P} (x_i = |u|) \wedge (\bigwedge_{j \notin P} x_j > |u|))(uv).$$

This is obtained because every valuation has a set P of variables that are minimal with respect to the ordering of the word w. Using the celebrated Feferman-Vaught technique [13], we can (effectively) build a sentence φ_1^P and a formula φ_2^P such that $uv, \vec{x} \models \varphi \wedge \bigwedge_{i \in P}(x_i = |u|) \wedge (\bigwedge_{j \notin P} x_j > |u|)$ if and only if $u \models \varphi_1^P$ and $v, (x_i)_{i \notin P} \models \varphi_2^P$. As a consequence:

$$\#\varphi(w) = \sum_{i=1}^{k} \sum_{w=uv} \#\varphi_1(u) \times \#\varphi_2(v).$$

Notice that φ_1^P has no free variables, φ_2^P has $k - |P| \leq k - 1$ free-variables, and that the Feferman–Vaught theorem preserves FO (resp. MSO) formulas.

C Proofs of Section 4

C.1 Proof of Claim 4.12

The set $\mathsf{Skel}\,(\mathsf{skel\text{-}root}\,(x))$ is a binary tree of height at most d. Moreover, every node $\mathfrak{t}$ in F has at most d parents, 2 siblings, 2 siblings for its parent. As a consequence, x has at most $2^d \times 2^d \times (d+4)$ dependent leaves in any forest F of height at most d.

C.2 Proof of Lemma 4.13

First, we use the lexicographic order to find the first pair (x_i, x_j) that is dependent in the tuple $\vec{x}$. This allows to partition our set of valuations as follows:

$$\begin{aligned}
&\{\, \vec{x} \in \mathsf{Leaves}(F) \colon F, \vec{x} \models \psi \wedge \mathsf{sym\text{-}dep}\,(\vec{x}) \,\} \\
&= \biguplus_{1 \leq i < j \leq n} \{\, \vec{x} \in \mathsf{Leaves}(F) \colon F, \vec{x} \models \psi \wedge \mathsf{sym\text{-}dep}\,(x_i, x_j) \wedge \bigwedge_{(k,l) <_{\text{lex}} (i,j)} \neg\, \mathsf{sym\text{-}dep}\,(x_k, x_l) \,\} \\
&= \biguplus_{1 \leq i < j \leq n} \{\, \vec{x} \in \mathsf{Leaves}(F) \colon F, \vec{x} \models \underbrace{\psi \wedge x_j \mathrel{\mathsf{depends\text{-}on}} x_i \wedge \bigwedge_{(k,l) <_{\text{lex}} (i,j)} \neg\, \mathsf{sym\text{-}dep}\,(x_k, x_l)}_{:= \psi_{i \to j}(\vec{x})} \,\} \\
&\qquad \cup \{\, \vec{x} \in \mathsf{Leaves}(F) \colon F, \vec{x} \models \underbrace{\psi \wedge x_i \mathrel{\mathsf{depends\text{-}on}} x_j \wedge \bigwedge_{(k,l) <_{\text{lex}} (i,j)} \neg\, \mathsf{sym\text{-}dep}\,(x_k, x_l)}_{:= \psi_{i \leftarrow j}(\vec{x})} \,\}
\end{aligned}$$

As a consequence, $\#(\psi \wedge \mathsf{sym\text{-}dep}\,) = \sum_{1 \leq i < j \leq n} \#\psi_{i \to j} + \#\psi_{i \leftarrow j} - \#\psi_{i \to j} \wedge \psi_{i \leftarrow j}$. We can now rewrite this as an infinite sum of functions in P_{k-1}, using quantifiers of the form $\exists^{=l}\, x.\psi$ to denote the fact that there exists exactly l different values for x so that $\psi(x)$ holds (which is expressible at every fixed l).

$$\begin{aligned}
\#(\psi \wedge \mathsf{sym\text{-}dep}\,) = \sum_{l,p,q \geq 0} \sum_{1 \leq i < j \leq n} \quad & l \times \#(\exists^{=l}\, x_j.\psi_{i \to j}) \\
& + p \times \#(\exists^{=p}\, x_i.\psi_{i \leftarrow j}) \\
& - q \times \#(\exists^{=q}\, x_i.\psi_{i \to j} \wedge \psi_{i \leftarrow j})
\end{aligned}$$

Thanks to Claim 4.12, there exists a bound N_d over the maximal number of nodes that are dependent to a fixed node in a forest of depth at most d. Hence, the above sum can be truncated to $l, p, q \leq N_d$, which is a finite linear combination of elements in P_{k-1}.

C.3 Proof of Claim 4.8

First of all, given a leaf $x \in \mathsf{Leaves}(F)$, $\mathsf{Skel}\,(x) = \{x\}$ contains x. Hence, every leaf is contained in at least one skeleton.

Assume that $\mathfrak{t}$ and $\mathfrak{t}'$ are two nodes such that $\mathsf{Skel}\,(\mathfrak{t})$ and $\mathsf{Skel}\,(\mathfrak{t}')$ contains x. As $\mathsf{Skel}\,(\mathfrak{t})$ contains only children of $\mathfrak{t}$, one deduces that x is a children of $\mathfrak{t}$ and $\mathfrak{t}'$. Because F is a tree, parents of x are totally ordered by their height in the tree. As a consequence, without loss of generality, one can assume that $\mathfrak{t}$ is a parent of $\mathfrak{t}'$. Because $\mathsf{Skel}\,(\mathfrak{t})$ is a subforest of F containing x, it must contain $\mathfrak{t}'$. Now, by definition of Skel, whenever $\mathfrak{t}' \in \mathsf{Skel}\,(\mathfrak{t})$, we have $\mathsf{Skel}\,(\mathfrak{t}') \subseteq \mathsf{Skel}\,(\mathfrak{t})$.

C.4 Proof of Lemma 4.16

To construct such a pumping family, we will rely on the fact that independent tuples of leaves have a very specific behaviour with respect to the factorisation forest. Given a node $\mathfrak{t}$, we write $\mathsf{start}\,(\mathfrak{t}) := \min_{\leq}\{y \in \mathsf{Leaves}(F) \cap \mathsf{Skel}\,(\mathfrak{t})\}$ and $\mathsf{end}\,(\mathfrak{t}) := \max_{\leq}\{y \in \mathsf{Leaves}(F) \cap \mathsf{Skel}\,(\mathfrak{t})\}$.

▷ Claim C.1. Let $\vec{x}$ be an independent tuple of $k \geq 1$ leaves in a forest $F \in \mathcal{F}_d^{\mu}$ factorising a word w. Let $\vec{\mathfrak{t}}$ be the vector of nodes such that $\mathfrak{t}_i := \mathsf{skel\text{-}root}\,(x_i)$. One can order the leaves according to their position in the word w so that $1 < \mathsf{start}\,(\mathfrak{t}_1) \leq \mathsf{end}\,(\mathfrak{t}_1) < \cdots < \mathsf{start}\,(\mathfrak{t}_k) \leq \mathsf{start}\,(\mathfrak{t}_k) < |w|$.

Proof. Assume by contradiction that there exists a pair $i < j$ such that $\mathsf{start}\,(\mathfrak{t}_j) \geq \mathsf{end}\,(\mathfrak{t}_i)$. Let us write s_j the first leaf of $\mathsf{Skel}\,(\mathfrak{t}_j)$ and e_i the last leaf of $\mathsf{Skel}\,(\mathfrak{t}_i)$. We know that $x_i \leq s_j \leq e_i \leq x_j$. This implies that $\mathsf{skel\text{-}root}\,(x_j)$ observes $\mathsf{skel\text{-}root}\,(x_i)$ or $\mathsf{skel\text{-}root}\,(x_j)$ observes $\mathsf{skel\text{-}root}\,(x_i)$, which contradicts the independence of $\vec{x}$.

Assume by contradiction that there exists i such that $\mathsf{start}\,(\mathfrak{t}_i) = 1$ (resp. $\mathsf{end}\,(\mathfrak{t}_i) = |w|$). Then $\mathsf{skel\text{-}root}\,(x_i)$ is the root of F, hence $\vec{x}$ is a dependent tuple, which is absurd. ◁

Given an independent tuple $\vec{x}$, with $\mathsf{skel\text{-}root}\,(\vec{x}) = \vec{nod}$, ordered by their position in the word, let us define $m_0 := \mu(w_1 \ldots w_{\mathsf{start}\,(\mathfrak{t}_1)-1})$, $m_k := \mu(w_{\mathsf{end}\,(\mathfrak{t}_k)+1} \ldots w_{|w|})$ and $m_i := \mu(w_{\mathsf{end}\,(\mathfrak{t}_k)+} \ldots w_{\mathsf{start}\,(\mathfrak{t}_{i+1})-1})$ for $1 \leq i \leq k-1$.

▶ **Definition C.2** (Type of and indpendent valuation). *Let $F \in \mathcal{F}_d^{\mu}$ factorising a word w, $\vec{x}$ be an independent tuple of leaves in F, and $\vec{\mathfrak{t}} = \mathsf{skel\text{-}root}\,(\vec{x})$. Without loss of generality assume that the nodes are ordered by* start *. The type* s-type $(\vec{\mathfrak{t}})$ *in the forest F is given by the tuple $(m_0, [\mathsf{Skel}\,(\mathfrak{t}_1)], m_1, \ldots, m_{k-1}, [\mathsf{Skel}\,(\mathfrak{t}_k)], m_k)$, where $[\mathsf{Skel}\,(\mathfrak{t}_i)]$ is the subforest of F that $\mathsf{Skel}\,(\mathfrak{t}_i)$ represents, coloured with the elements of the monoid corresponding to each node.*

At depth d, there are finitely many possible types for tuples of k nodes, which we collect in the set $\mathsf{Types}_{\,d,k}$. Moreover, given a type $T \in \mathsf{Types}_{\,d,k}$, one can build the MSO formula $\mathsf{s\text{-}type}_{\,=T}(\vec{\mathfrak{t}})$ over $\mathcal{F}_d^{\mu}$ that tests whether a tuple of *nodes* $\mathfrak{t}_1, \ldots, \mathfrak{t}_k$ is of type T, and can be obtained as $\mathsf{skel\text{-}root}\,(\vec{x})$ for some tuple $\vec{x}$ of independent leaves. The key property of types is that counting types is enough to count independent valuations for a formula $\psi \in \mathsf{INV}$, as precised in the following lemma.

▶ **Lemma C.3.** *Let $\psi(x_1, \ldots, x_k) \in \mathsf{INV}$. There are computable coefficients $\lambda_T \in \mathbb{Z}$, such that*

$$\#(\psi \wedge \neg\,\mathsf{sym\text{-}dep}\,) = \sum_{T \in \mathsf{Types}_{\,d,k}} \lambda_T \#(\mathsf{s\text{-}type}_{\,=T}(\vec{\mathfrak{t}}))$$

Proof.

▷ Claim C.4. Let $T \in \mathsf{Types}_{\,d,k}$, $F \in \mathcal{F}_d^{\mu}$, $\vec{x}$ and $\vec{y}$ be two k-tuples of independent leaves of F such that $\mathsf{s\text{-}type}\,(\mathsf{skel\text{-}root}\,(x_1), \ldots, \mathsf{skel\text{-}root}\,(x_k)) = \mathsf{s\text{-}type}\,(\mathsf{skel\text{-}root}\,(y_1), \ldots, \mathsf{skel\text{-}root}\,(y_k)) = T$. There exists a bijection $\sigma \colon L_1 \to L_2$, where $L_1 := \mathsf{Leaves}(F) \cap \bigcup_{i=1}^{k} \mathsf{Skel}\,(\mathsf{skel\text{-}root}\,(x_i))$ and $L_2 := \mathsf{Leaves}(F) \cap \bigcup_{i=1}^{k} \mathsf{Skel}\,(\mathsf{skel\text{-}root}\,(y_i))$, such that for every $z \in L_1^k$, for every formula $\psi \in \mathsf{INV}_k$, $F \models \psi(z)$ if and only if $F \models \psi(\sigma(z))$.

Proof. Because of the type equality, we know that $\mathsf{Skel}\,(\mathsf{skel\text{-}root}\,(x_i))$ and $\mathsf{Skel}\,(\mathsf{skel\text{-}root}\,(y_i))$ are isomorphic for $1 \leq i \leq k$. As the skeletons are disjoint in an independent tuple, this automatically provides the desired bijection σ.

Let us now prove that σ preserves the semantics of invariant formulas. Notice that this property is stable under disjunction, conduction and negation. Hence, it suffices to check the property for the following three formulas $\mathsf{between}_m(x, y)$, $\mathsf{left}_m(x)$ and $\mathsf{right}_m(y)$.

We know that the relative ordering of the nodes is preserved by σ, and skeletons are obtained by removing inner idempotent nodes, hence the evaluation of $\mu(w_i \dots w_j)$ only depends on the skeleton of w_i, the one of w_j, and the element of the monoid separating them, which is contained in the type T. ◁

Using the claim, we can now proceed to prove Lemma C.3.

$$\begin{aligned}
\#\psi \wedge \neg\,\mathsf{sym\text{-}dep}\,(F) &= \sum_{\vec{x}\ \mathrm{indep}} \mathbf{1}_{F \models \psi(\vec{x})} \\
&= \sum_{T \in \mathsf{Types}_{d,k}} \sum_{\vec{\mathfrak{t}} \in \mathsf{Nodes}(F)} \sum_{\vec{x}\ \mathrm{indep}} \mathbf{1}_{F \models \psi(\vec{x})} \mathbf{1}_{\vec{\mathfrak{t}} = \mathsf{skel\text{-}root}\,(\vec{x})} \mathbf{1}_{\mathsf{s\text{-}type}_{=T}(\vec{\mathfrak{t}})} \\
&= \sum_{T \in \mathsf{Types}_{d,k}} \sum_{\vec{\mathfrak{t}} \in \mathsf{Nodes}(F)} \mathbf{1}_{\mathsf{s\text{-}type}_{=T}(\vec{\mathfrak{t}})} \left(\sum_{\vec{x}\ \mathrm{indep}} \mathbf{1}_{F \models \psi(\vec{x})} \mathbf{1}_{\vec{\mathfrak{t}} = \mathsf{skel\text{-}root}\,(\vec{x})} \right) \\
&= \sum_{T \in \mathsf{Types}_{d,k}} \sum_{\vec{\mathfrak{t}} \in \mathsf{Nodes}(F)} \mathbf{1}_{\mathsf{s\text{-}type}_{=T}(\vec{\mathfrak{t}})} \lambda_T \\
&= \sum_{T \in \mathsf{Types}_{d,k}} \lambda_T \#(\mathsf{s\text{-}type}_{=T}(\vec{\mathfrak{t}}))
\end{aligned}$$ ◀

The behaviour of the formulas $\mathsf{s\text{-}type}_{=T}$ is much more regular and allows us to extract pumping families that clearly distinguishes different types.

▶ **Lemma C.5** (Pumping Lemma). *Let $T \in \mathsf{Types}_{d,k}$. There exists a pumping family $(w^{\bar{X}}, F^{\bar{X}})$, such that for every type $T' \in \mathsf{Types}_{d,k}$, $\#(\mathsf{s\text{-}type}_{=T'}(\vec{\mathfrak{t}}))(F^{\bar{X}})$ is asymptotically a $\mathbb{Z}$-polynomial in $\bar{X}$ that has non-zero coefficient for $X_1 \dots X_n$ if and only if $T = T'$.*

Proof. Let T be a type, it is obtained by considering some forest $F \in \mathcal{F}^{\mu}_{d}$ factorising a word w. And some tuple $\vec{x}$ of independent leaves in F. Let us construct $F^{\bar{X}}$ obtained by replacing the subtree of $\mathfrak{t}_i := \mathsf{skel\text{-}root}\,(x_i)$ by the repetition of X_i times $S_i := \mathsf{Skel}\,(\mathsf{skel\text{-}root}\,(x_i))$.

Recall that $\mu(\mathsf{word}\,(S_i)) = \mu(\mathsf{word}\,(F_i))$ where F_i is the subforest of F rooted in $\mathfrak{t}_i$ as we only remove inner idempotent nodes. As a consequence, replacing the subforest F_i by S_i still provides a valid factorisation forest for a subword of w.

Now, as $\mathsf{skel\text{-}root}\,(x_i)$ cannot be the root of the forest F and is the highest parent of x_i that is not a leftmost or rightmost child, it must be the immediate inner child of an idempotent node in F. As a consequence, $\mu(\mathsf{word}\,(S_i))$ is an idempotent, and one can repeat 0 or more times the forest S_i in F while preserving the fact that it is a valid factorisation forest. Because the leaves are independent, these repetitions are done over disjoint *factors* of the word w, hence preserving the validity of the global factorisation forest.

Notice that $F^{\bar{X}}$ is the factorisation forest of the word $w^{\bar{X}} := \alpha_0 (u_1)^{X_1} \alpha_1 \dots \alpha_{k-1} (u_k)^{X_k} \alpha_k$ where $u_i = \mathsf{word}\,(S_i)$, $\alpha_i = w_{\mathsf{end}\,(\mathfrak{t}_i)+1} \dots w_{\mathsf{start}\,(\mathfrak{t}_i)-1}$ for $2 \leq i \leq k-1$, $\alpha_0 = w_0 \dots w_{\mathsf{start}\,(\mathfrak{t}_1)-1}$, and $\alpha_k = w_{\mathsf{end}\,(\mathfrak{t}_k)+1} \dots w_{|w|}$ are non-empty factors of w.

Let $T' \in \mathsf{Types}_{d,k}$ be another type and let us count the number of valuations for $\mathsf{s\text{-}type}_{=T'}$ in $F^{\bar{X}}$. Let us write E for the set of nodes in $F^{\bar{X}}$ that are not appearing in any of the X_i repetitions of S_i, for $1 \leq i \leq k$. The set E has a size bounded independently of $X_1, \dots, X_k$. To a tuple $\vec{\mathfrak{t}}$ satisfying $\mathsf{s\text{-}type}_{=T'}$, one can associate the mapping $\rho_{\vec{\mathfrak{t}}} \colon \{1, \dots, k\} \to \{1, \dots, k\} \uplus E$, so that $\rho_{\vec{\mathfrak{t}}}(i) = \mathfrak{t}_i$ when $\mathfrak{t}_i \in E$, and $\rho_{\vec{\mathfrak{t}}}(i) = j$ when $\mathfrak{t}_i$ is a node appearing in one of the X_j repetitions of the skeleton S_i (there can be at most one j satisfying this property).

Notice that if $\vec{\mathfrak{t}}$ satisfies $\text{s-type}_{=T'}$ and that $\rho_{\vec{\mathfrak{t}}}(i) = j$, then $\mathfrak{t}_i$ must be the root of one of the X_j copies of S_j in $F^{\bar{X}}$. Indeed, $\vec{\mathfrak{t}}$ is obtained as $\text{skel-root}(\vec{y})$ for some independent tuple $\vec{y}$ of leaves. Hence, $\mathfrak{t}_i = \text{skel-root}(y_i)$ which belong to some copy of S_j, hence $\mathfrak{t}_i$ must be the root of this copy of S_j.

Given a map $\rho\colon \{1,\dots k\} \to \{1,\dots,k\} \uplus E$ one can build the set C_ρ of all vectors $\vec{\mathfrak{t}}$ satisfying $\text{s-type}_{=T'}$ such that $\rho_{\vec{\mathfrak{t}}} = \rho$. This allows us to rewrite the number of such vectors as a finite sum:

$$\#(\text{s-type}_{=T'}(\vec{\mathfrak{t}}))(F^{\bar{X}}) = \sum_{\rho\colon \{1,\dots,k\}\to\{1,\dots,k\}\uplus E} \#C_\rho$$

▷ Claim C.6. For every $\rho\colon \{1,\dots,k\} \to \{1,\dots,k\} \uplus E$, C_ρ is asymptotically a $\mathbb{Z}$-polynomial in $\bar{X}$, with a non-zero coefficient for $X_1 \dots X_k$ if and only if $\rho(i) = i$ for $1 \le i \le k$.

Proof. Assume that C_ρ is non-empty. Then choosing a vector $\vec{\mathfrak{t}} \in C_\rho$ is done by fixing the image of $\mathfrak{t}_i$ when $\rho(i) \in E$, and selecting $p_j := |\rho^{-1}(\{j\})|$ non consecutive copies of S_j among among the X_j copies available.

The number of ways one can select p non consecutive nodes in among X nodes is (for large enough X) the binomial number $\binom{X-p+1}{p}$, as it is the same as selecting p positions among $X - p + 1$ and then adding $p - 1$ separators.

As a consequence, the size of C_ρ is asymptotically a product of $\binom{X_j-p_j+1}{p_j}$ for the non-zero p_j, which is a $\mathbb{Z}$-polynomial in $X_1, \dots, X_k$. Moreover, it has a non-zero coefficient for $X_1 \dots X_k$ if and only if $p_j \neq 0$ for $1 \le j \le k$, which is precisely when $\rho(i) = i$. ◁

We have proven that $\#(\text{s-type}_{=T'}(\vec{\mathfrak{t}}))(F^{\bar{X}})$ is a $\mathbb{Z}$-polynomial in $X_1, \dots, X_k$, and that the only term possibly having a non-zero coefficient for $X_1 \dots X_k$ is C_{id}. Notice that if C_{id} is non-zero, we immediately conclude that $T = T'$. ◀

We now have all the ingredients to prove Lemma 4.16, allowing us to pump functions built by counting independent tuples of invariant formulas.

Let $k \ge 1$, and f_{indep} be a linear combination of $\#\psi_i \wedge \neg\,\text{sym-dep}$, where $\psi_i \in \text{INV}$ has k free variables. Assume moreover that $f_{\text{indep}} \neq 0$.

Thanks to Lemma C.3, every $\#\psi_i \wedge \neg\,\text{sym-dep}$ can be written as a linear combination of $\#\text{s-type}_{=T}(\vec{t})$, hence $f_{\text{indep}} = \sum_{T\in\text{Types}_{d,k}} \lambda_T \#\text{s-type}_{=T}(\vec{t})$, and the coefficients λ_T are computable.

Since $f_{\text{indep}} \neq 0$, there exists $T \in \text{Types}_{d,k}$ such that $\lambda_T \neq 0$. Using Lemma C.5, there exists a pumping family $(w^{\bar{X}}, F^{\bar{X}})$ for T. In particular, $f(F^{\bar{X}})$ is asymptotically a $\mathbb{Z}$-polynomial in $X_1 \dots X_k$, and its coefficient in $X_1 \dots X_k$ is the sum of the coefficients in $X_1 \dots X_k$ of the polynomials $\#\text{s-type}_{=T'}(F^{\bar{X}})$ which is non-zero if and only if $T = T'$. Hence, $f(F^{\bar{X}})$ is asymptotically a $\mathbb{Z}$-polynomial with a non-zero coefficient for $X_1 \dots X_k$.

As a side result, we have proven that a linear combination of $\#\text{s-type}_{=T}$ is the constant function 0 if and only if all the coefficient are 0, which is decidable.

C.5 Proof of Lemma 4.17

We first introduce the following claim

▷ Claim C.7. Let $P \in \mathbb{R}[X_1, \dots, X_n]$ which evaluates to 0 over $\mathbb{N}^n$, then $P = 0$.

Proof. The proof is done by induction on the number n of variables. If P has one variable and $P_{|\mathbb{N}} = 0$, then P has infinitely many roots and $P = 0$. Now, let P having $n+1$ variables, and such that $P(x_1, \dots, x_n, x_{n+1}) = 0$ for all $(x_1, \dots, x_{n+1}) \in \mathbb{N}^{n+1}$. By induction hypothesis, $P(X_1, \dots, X_n, x_{n+1}) = 0$ for all $x_{n+1} \in \mathbb{N}$, hence $P = 0$. ◁

Now, we are ready to prove Lemma 4.17. If $P = 0$, then $\deg P \leq \deg Q$. Otherwise, let us write $P = P_1 + P_2$ with P_1 containing all the terms of degree exactly $\deg(P)$ in P.

Because P_1 is a non-zero polynomial, there exists a tuple $(x_1, \dots, x_n) \in \mathbb{N}$ such that $\alpha \coloneqq P_1(x_1, \dots, x_n) \neq 0$ (Claim C.7). Let us now consider $R(Y) \coloneqq P(Ux_1, \dots, Ux_n)$, and $S(Y) \coloneqq Q(Yx_1, \dots, Yx_n)$. Notice that $R(Y)$ is a polynomial in Y of degree exactly $\deg(P)$ whose term of degree $\deg(P)$ is $\alpha Y^{\deg(P)}$. Furthermore, $S(\lambda)$ is a polynomial in Y of degree at most $\deg(Q)$. Since $|R(Y)| \sim_{+\infty} |\alpha| Y^{\deg(P)}$, and $C|S(Y)| \sim_{+\infty} C' Y^{\deg(S)} \leq C' Y^{\deg(Q)}$, we conclude that $\deg(P) \leq \deg(Q)$.

D Proofs of Section 5

D.1 Proof of Claim 5.9

By expanding the definitions we note that $a \triangleright (\mathbf{1}_L \otimes g) = (a \triangleright \mathbf{1}_L) \otimes g + \mathbf{1}_L(\varepsilon) \times (a \triangleright g)$ for all $a \in A$. By Claim 5.6 we get $a \triangleright g \in \mathsf{P}_k$, hence $a \triangleright (\mathbf{1}_L \otimes g) \sim_k (a \triangleright \mathbf{1}_L) \otimes g$. The result follows by induction since $a \triangleright \mathbf{1}_L = \mathbf{1}_{a^{-1}L}$ and by Claim 5.8.

D.2 Proof of Lemma 5.10

We first note that $u \triangleright (\delta f + \eta g) = \delta(u \triangleright f) + \eta(u \triangleright g)$, for all $f, g : A^* \to \mathbb{Z}$, $\delta, \eta \in \mathbb{Z}$ and $u \in A^*$. Hence it suffices to show that Lemma 5.10 holds on a set S of functions such that $\mathsf{Span}_{\mathbb{Z}}(S) = \mathsf{P}_k$. For $k = 0$, we can chose $S \coloneqq \{\mathbf{1}_L : L \text{ regular}\}$. As observed above, we have $u \triangleright \mathbf{1}_L = \mathbf{1}_{u^{-1}L}$ and the result holds since regular languages have finitely many residual languages. For $k \geq 1$, we can choose $S \coloneqq \{\mathbf{1}_L \otimes g : g \in \mathsf{P}_{k-1}, L \text{ regular}\}$ by Lemma 3.8. Let $\mathbf{1}_L \otimes g \in S$. Then by Claim 5.9 we get $u \triangleright (\mathbf{1}_L \otimes g) \sim_{k-1} (u \triangleright \mathbf{1}_L) \otimes g = \mathbf{1}_{u^{-1}L} \otimes g$. Since a regular language has finitely many residual languages, there are finitely many $\sim_{k-1}$-equivalence classes for the residuals of $\mathbf{1}_L \otimes g$.

D.3 Proof of Corollary 5.20

Lemma 5.18 shows that any function from P_k is computed by its k-residual transducer. Conversely, given a P_{k-1}-transducer, it is easy to write its function as a linear combination of elements of the form $\mathbf{1}_L \otimes g$, where g is the label of a transition, thus in P_{k-1}.

D.4 Proof of Corollary 5.21

Every map in P_k has finitely many residuals up to $\sim_{k-1}$ thanks to Lemma 5.10. We now prove the converse implication. Let f such that $\mathsf{Res}(f)/\sim_{k-1}$ is finite. By Lemma 5.18 one can compute a k-residual transducer of f. In particular, the transitions of this transducer are in $\mathsf{Span}_{\mathbb{Z}}(\mathsf{Res}(f)) \cap \mathsf{P}_{k-1} \subseteq \mathsf{P}_{k-1}$. As a consequence, this is a P_{k-1}-transducer. Thanks to Corollary 5.20, $f \in \mathsf{P}_k$.

D.5 Proof of Lemma 5.25

Let $\mathcal{T} = (A, Q, q_0, \delta, \lambda)$ be such a transducer computing a function $f \in \mathsf{P}_k$. Since the deterministic automaton (A, Q, q_0, δ) is counter-free, then by [14] for all $q \in Q$ the language $L_q \coloneqq \{u : \delta(q_0, u) = q\}$ is star-free. So is $L_q a$ for all $a \in A$. Now observe that:

$$f = \sum_{\substack{q \in Q \\ a \in A}} \mathbf{1}_{L_q a} \otimes \lambda(q, a).$$

We conclude thanks to Equation (1).

D.6 Proof of Lemma 5.26

It is easy to see that ultimate polynomiality is preserved under taking sums and external products. Hence by Equation (1), it is sufficient to show that for all $k \geq 0$ and all $L_0, \dots, L_k$ star-free languages, the function $\mathbf{1}_{L_0} \otimes \cdots \otimes \mathbf{1}_{L_k}$ is ultimately polynomial. We show this result by induction on $k \geq 0$. For the base case, let $\mu : A^* \to M$ be a morphism into an aperiodic monoid which recognizes L_0. Let ω be the idempotence index of M, then for all $m \in M, X, X' \geq \omega$, $m^X = m^{X'}$. Now, let $\ell \geq 0$ and $\alpha_0, u_1, \alpha_1, \dots, u_\ell, \alpha_\ell \in A^*$. Then for $X_1, \dots, X_\ell \geq \omega$ we have that $\mu(\alpha_0 u_1^{X_1} \alpha_1 \cdots u_\ell^{X_\ell} \alpha_\ell)$ is constant. Hence $\mathbf{1}_{L_0}$ is ultimately polynomial (in fact "ultimately constant" since the former expression is constant).

Now, let us consider $g \coloneqq f \otimes \mathbf{1}_L$ where $f : A^* \to \mathbb{Z}$ are ultimately polynomial and L is star-free. By induction hypothesis and thanks to the base case, there exists $\omega \geq 0$ such that for all $\ell \geq 0$, $\alpha_0, u_1, \alpha_1, \dots, u_\ell, \alpha_\ell \in A^*$, and $X_1, \dots, X_\ell \geq \omega$, $\mathbf{1}_L(\alpha_0 u_1^{X_1} \alpha_1 \cdots u_\ell^{X_\ell} \alpha_\ell)$ is constant and $f(\alpha_0 u_1^{X_1} \alpha_1 \cdots u_\ell^{X_\ell} \alpha_\ell)$ is a polynomial in $X_1, \dots, X_\ell$. Let $\ell \geq 0$ and $\alpha_0, u_1, \alpha_1, \dots, u_\ell, \alpha_\ell \in A^*$, then for $X_1, \dots, X_\ell \geq 0$ we have:

$$
\begin{aligned}
&g(\alpha_0 u_1^{X_1} \alpha_1 \cdots u_\ell^{X_\ell} \alpha_\ell) = f(\alpha_0 u_1^{X_1} \alpha_1 \cdots u_\ell^{X_\ell} \alpha_\ell) \mathbf{1}_L(\varepsilon) \\
&+ \sum_{j=0}^{\ell} \sum_{i=0}^{|\alpha_j|-1} f(\alpha_0 u_1^{X_1} \alpha_1 \cdots u_j^{X_j} (\alpha_j[1{:}i])) \times \mathbf{1}_L((\alpha_j[i{+}1{:}|\alpha_j|]) u_{j+1}^{X_{j+1}} \cdots \alpha_\ell) \\
&+ \sum_{j=1}^{\ell} \sum_{i=0}^{|u_j|-1} \sum_{Y=0}^{X_j-1} f(\alpha_0 u_1^{X_1} \alpha_1 \cdots u_j^{Y} (u_j[1{:}i])) \times \mathbf{1}_L((u_j[i{+}1{:}|u_j|]) u_j^{X_j-Y-1} \cdots \alpha_\ell)
\end{aligned}
$$

The two first terms are (sums of) polynomials when $X_1, \dots, X_\ell \geq \omega$. Let focus on the last term. Let us fix some $1 \leq j \leq \ell$ and $0 \leq i \leq |u_j| - 1$. We only need to show that given $\beta, \gamma \in A^*$, then the following term is a polynomial for $X_1, \dots, X_\ell \geq 2\omega$:

$$
\sum_{Y=1}^{X_j-1} f(\alpha_0 u_1^{X_1} \alpha_1 \cdots u_j^{Y} \beta) \mathbf{1}_L(\gamma u_j^{X_j-Y-1} \cdots \alpha_\ell)
$$

The 2ω terms (since $X_j \geq 2\omega$) for $Y < \omega$ or $X_j - Y - 1 \leq \omega$ (i.e. $X_j - 1 \geq Y > X_j - \omega$) can be removed and treated separately. It remains:

$$
\begin{aligned}
&\sum_{Y=\omega}^{X_j-\omega} f(\alpha_0 u_1^{X_1} \alpha_1 \cdots u_j^{Y} \beta) \mathbf{1}_L(\gamma u_j^{X_j-Y-1} \cdots \alpha_\ell) \\
&= c \times \sum_{Y=\omega}^{X_j-\omega} f(\alpha_0 u_1^{X_1} \alpha_1 \cdots u_j^{Y} \beta) \\
&= c \times \sum_{Y=\omega}^{X_j-\omega} P(X_1, \dots, X_{j-1}, Y) \\
&= \sum_{Y=\omega}^{X_j-\omega} \sum_{d=0}^{D} Y^d P_d(X_1, \cdots, X_{j-1}) = \sum_{d=0}^{D} P_d(X_1, \cdots, X_{j-1}) \left(\sum_{Y=\omega}^{X_j-\omega} Y^d \right)
\end{aligned}
$$

where $C \in \{0, 1\}$, $D \geq 0$, and P, P_d for $0 \leq d \leq D$ are polynomials. It is well-known that $\sum_{Y=1}^{X_j} Y^d$ is a polynomial in X_j. So is $\sum_{Y=\omega}^{X_j-\omega} Y^d$, which concludes the proof.