On Parikh Images of Higher-Order Pushdown Automata (Extended Abstract)[1]

Wong Kariante
Lehrstuhl für Informatik VII, RWTH Aachen, D-52056 Aachen, Germany
e-mail: karianto@informatik.rwth-aachen.de

ABSTRACT

We introduce the notion of semi-polynomial sets, generalizing the notion of semi-linear sets, and show that each semi-polynomial set is the Parikh image of level 2 pushdown automata, which represent a special class of higher-order pushdown automata.

Keywords: Parikh mapping, semi-linear sets, polynomials, higher-order pushdown automata

1. Introduction

The Parikh mapping, which gives information on the distribution of symbols in a word or a language, respectively, has turned out to be a quite useful tool in the study of formal languages. In particular, in the context of context-free languages it provides a bridge between formal language theory and number theory: Parikh's theorem [8] asserts that the Parikh image (that is, the image under the Parikh mapping) of any context-free language is always a semi-linear set of vectors of natural numbers, and moreover, given a representation of a context-free language, its (semi-linear) Parikh image can be effectively constructed. Given this fact, for instance, the decidability of the emptiness problem for context-free languages follows immediately.

There are several automaton models known in the literature that generalize pushdown automata, which precisely recognize context-free languages. Higher-order pushdown automata (HOPDA) represent such a model; an HOPDA, essentially, is a pushdown automaton whose infinite store is a (multiply) nested pushdown stack, that is, a stack of stacks of . . . of stacks. For an exposition of this model, see, for instance, [7, 4, 3]. In current research, this model is of interest in model checking by its strong decidability properties. Not only the emptiness problem for HOPDA's is decidable, but also the monadic second-order theory of the transition graph of any HOPDA [2].

Despite these nice properties and the tight connection to pushdown automata, surprisingly, HOPDA's have not been much studied in terms of their Parikh images. In particular, a precise characterization of the Parikh images of HOPDA's is still missing. In this work, we explore this issue for a fairly small class of HOPDA's, namely for the level 2 HOPDA's (2-PDA's), which are pushdown automata with a stack of stacks as infinite store. We show that 2-PDA's can generate polynomials in a sense to be defined more precisely later on. Although we have not succeeded in identifying a class of vectors of natural numbers that captures the Parikh images of 2-PDA's yet, our result might suggest which ingredient is needed for such a class.

Following this introduction, in Sect. 2 we fix our notation and propose the notion of semi-polynomial sets as a generalization of semi-linear sets. In Sect. 3 we outline the idea of showing

[1]This work is part of an ongoing joint work with Aloys Krieg and Wolfgang Thomas.

that each semi-polynomial set can be generated as the Parikh image of a 2-PDA. Section 4 concludes with some remarks.

2. Semi-Polynomial Sets

We denote the set of natural numbers by $\mathcal{N}$ and the set of vectors of natural numbers of dimension $n \geq 1$ by $\mathcal{N}^n$. Recall that a subset A of $\mathcal{N}^n$, $n \geq 1$, is *linear* if there are vectors $\bar{u}_0, \bar{u}_1, \ldots, \bar{u}_m \in \mathcal{N}^n$, $m \geq 0$, such that $A = \{\bar{u}_0 + k_1\bar{u}_1 + \ldots + k_m\bar{u}_m \mid k_1, \ldots, k_m \in \mathcal{N}\}$. The set A is *semi-linear* if it is a finite union of linear sets.

Let $n \geq 1$ and $\Sigma = \{a_1, \ldots, a_n\}$ be an alphabet. The *Parikh mapping* $\Phi\colon \Sigma^* \to \mathcal{N}^n$ is defined by $\Phi(w) = (|w|_{a_1}, \ldots, |w|_{a_n})$, for each $w \in \Sigma^*$. The *Parikh image* (or *commutative image*) of a language $L \subseteq \Sigma^*$ is the set $\Phi(L) := \{\Phi(w) \mid w \in L\} \subseteq \mathcal{N}^n$.

A natural generalization of semi-linear sets is the following:

Definition 1 A subset A of $\mathcal{N}^n$, $n \geq 1$, is a *polynomial set of degree* $d \geq 1$ if there is a vector $\bar{u}_0 \in \mathcal{N}^n$ and a family of vectors $(\bar{u}_{i,j})_{1 \leq i \leq m, 1 \leq j \leq d}$ in $\mathcal{N}^n$, for some $m \geq 0$, such that

$$\begin{aligned} A = \{\bar{u}_0 &+ k_1\bar{u}_{1,1} + k_1^2\bar{u}_{1,2} + \cdots + k_1^{d-1}\bar{u}_{1,d-1} + k_1^d\bar{u}_{1,d} \\ &+ \cdots + k_m\bar{u}_{m,1} + k_m^2\bar{u}_{m,2} + \cdots + k_m^{d-1}\bar{u}_{m,d-1} + k_m^d\bar{u}_{m,d} \mid k_1, \ldots, k_m \in \mathcal{N}\} . \end{aligned}$$

The set A is a *semi-polynomial set of degree* d if it is a finite union of polynomial sets of degree d. The set A is a *polynomial* (resp. *semi-polynomial*) set if it is a polynomial (resp. semi-polynomial) set of degree d for some $d \geq 1$. Occasionally, we refer to (semi-)polynomial sets of degree 2 as (semi-)quadratic sets.

Clearly, each (semi-)linear set is (semi-)polynomial.

Given the generators of a polynomial set A as in the definition, one can decide whether a given vector $\bar{u} = (u_1, \ldots, u_n)$ belongs to A; it suffices to check the k_i-values up to $\max(u_1, \ldots, u_n)$. Hence, the membership problem for a semi-polynomial set is decidable.

Example 2 The set $A_1 := \{(u_1, u_2) \in \mathcal{N}^2 \mid u_2 = u_1^2\}$ is quadratic since it coincides with $A_1 = \{(0,0) + k(1,0) + k^2(0,1) \mid k \in \mathcal{N}\}$. Furthermore, it is not difficult to show that A_1 is not semi-linear, for instance, by using a simple growth rate argument.

One can also show that the set $\{(u_1, u_2) \in \mathcal{N}^2 \mid u_2 = u_1^{d+1}\}$ is not semi-polynomial of degree d, for each $d \geq 1$, and that the set $A_2 := \{(u_1, u_2) \in \mathcal{N}^2 \mid u_2 = 2^{u_1}\}$ is not semi-polynomial.

It is worth noting that the product relation, such as the set $A_3 := \{(u, v, uv) \mid u, v \in \mathcal{N}\} \subseteq \mathcal{N}^3$, is not semi-polynomial. For this, the simple comparison of growth rates does not suffice, and some deeper structural analysis is needed.

3. Level 2 Pushdown Automata

The purpose of this section is to show that a simple extension of pushdown automata suffices to generate (via the Parikh mapping) all semi-polynomial sets. More precisely, we consider *level 2 pushdown automata (2-PDA)*, which are a special case of *higher-order pushdown automata*. Roughly speaking, A 2-PDA is a finite automaton augmented with a pushdown stack whose elements are again pushdown stacks. The model of 2-PDA is known to be equivalent to the indexed grammars of [1], so the languages recognized by 2-PDA are precisely the indexed languages [4].

We now introduce 2-PDA more precisely, following [2]. We use Γ as stack alphabet and $\bot \in \Gamma$ as initial stack symbol. A *level 1 pushdown stack (1-stack)* over Γ is a sequence of stack symbols denoted by $[Z_m \cdots Z_1]$, $m \geq 0$, where Z_m is considered as the topmost symbol. A *level 2 pushdown stack (2-stack)* is a sequence $[s_r, \ldots, s_1]$ of $r \geq 1$ 1-stacks. Note that a 2-stack always contains at least one 1-stack. The empty 1-stack is denoted by $[\varepsilon]$ while the empty 2-stack, which contains only one single empty 1-stack, is denoted by $[[\varepsilon]]$. During any computation, only the topmost symbol of the topmost 1-stack can be accessed. The set Instr of the *instructions* that can be executed on a 2-stack comprises: (1) pushing a stack symbol Z to the topmost 1-stack, (2) copying the topmost 1-stack completely and placing it on top of the 2-stack, (3) removing the topmost symbol of the topmost 1-stack, and (4) removing the topmost 1-stack completely. Note that the latter instruction can only be executed if the resulting stack is again a 2-stack.

A *level* 2 *pushdown automaton (2-PDA)* is of the form $\mathcal{A} := (Q, \Sigma, \Gamma, \delta, q_0, \bot)$, where Q is a finite, nonempty set of states, Σ the input alphabet, Γ the stack alphabet, $\delta\colon Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \rightarrow \mathcal{P}(Q \times \text{Instr})$ the transition function, $q_0 \in Q$ the initial state, and $\bot \in \Gamma$ the initial stack symbol. A *configuration* of $\mathcal{A}$ is a pair (q, s), where q is a state in Q , and s is a 2-stack. The *initial configuration* of $\mathcal{A}$ is $(q_0, [[\bot]])$. The 2-PDA $\mathcal{A}$ can reach a configuration (q', s') from a configuration (q, s) by reading $a \in \Sigma \cup \{\varepsilon\}$ if $\delta(q, a, \text{top(s)})$ contains (q', instr), where top(s) denotes the topmost symbol of the topmost 1-stack of the 2-stack s, and $\text{instr(s)} = \text{s}'$. The 2-PDA $\mathcal{A}$ accepts a word $w \in \Sigma^*$ if $\mathcal{A}$ reaches from the initial configuration a configuration $(q, [[\varepsilon]])$, for some $q \in Q$, after reading w. The language recognized by $\mathcal{A}$ is denoted by $L(\mathcal{A})$.

Example 3 The language $L := \{a^k b^{k^2} \mid k \in \mathcal{N}\} \subseteq \{a, b\}^*$ is 2-PDA recognizable. We give an informal description of a 2-PDA $\mathcal{A}$ which recognizes L.

The stack alphabet of $\mathcal{A}$ is $\Gamma := \{\bot, Z, Z_2\}$; On an input word $w := a^k b^{k^2}$, $\mathcal{A}$ reads the a^k-prefix of w while pushing $(2k)$-many Zs into the stack and a Z_2. The resulting 2-stack is $[[Z_2 Z^{2k} \bot]]$. Then, $\mathcal{A}$ copies the topmost 1-stack and removes two symbols which are not $\bot$ from the stack, resulting in the stack $[[Z^{2k-1}\bot], [Z_2 Z^{2k}\bot]]$. The last step is repeated until the topmost 1-stack contains only one Z. Now, the resulting stack is $[[Z\bot], [Z^3\bot], \ldots, [Z^{2k-1}\bot], [Z_2 Z^{2k}\bot]]$. The number of Zs which lie above Z_2 is $\sum_{i=0}^{k-1}(2i+1)$, which yields k^2. Now $\mathcal{A}$ just pops the Zs one by one while reading bs until Z_2 is seen, then empties the stack, and accepts.

Note that the quadratic set A_1 of Example 2 is the Parikh image of the language L of Example 3. In other words, the set A_1 can be generated as the Parikh image of a 2-PDA recognizable language. Exploiting this idea, we show that each semi-polynomial set is the Parikh image of a 2-PDA recognizable language. At the core of our construction, we use a 2-PDA that generates, starting from a top stack content with $2k$ symbols Z, the values $k, k^2, \ldots, k^d$ via the Parikh mapping, thereby returning to the initial stack content and not touching the stacks below. Given this preparation, we can then show our main result:

Theorem 4 *Let $n \geq 1$. Every semi-polynomial subset of $\mathcal{N}^n$ is the Parikh image of a language recognizable by a 2-PDA.*

Proof. (sketch) Without loss of generality, we restrict ourselves to polynomial sets.

Let $A \subseteq \mathcal{N}^n$ be a polynomial set of degree $d \geq 1$, given by its constant vector $\bar{u}_0$ and its periods $\bar{u}_{i,j}$ $(1 \leq i \leq m,\ 1 \leq j \leq d)$, for some $m \geq 0$. We take $\Sigma := \{a_1, \ldots, a_n\}$ and assign to each generator vector of A a word in $a_1^* \cdots a_n^*$ such that the Parikh image of this word yields the corresponding generator vector. Let us call these words w_0 and $w_{i,j}$ $(1 \leq i \leq m,\ 1 \leq j \leq d)$.

We construct a 2-PDA $\mathcal{A}$ such that the Parikh image of $L(\mathcal{A})$ yields A. More precisely, $L(\mathcal{A})$ will contain the following words, for $k_1, \ldots, k_m \in \mathcal{N}$:

$$w_0 \quad w_{1,1}^{k_1} w_{1,2}^{k_1^2} \cdots w_{1,d-1}^{k_1^{d-1}} w_{1,d}^{k_1^d} \quad \cdots \quad w_{m,1}^{k_m} w_{m,2}^{k_m^2} \cdots w_{m,d-1}^{k_m^{d-1}} w_{m,d}^{k_m^d}$$

The construction of $\mathcal{A}$ generalizes the basic idea of Example 3 and is omitted in this abstract. The full proof can be found in [5]. □

The Parikh images of 2-PDA recognizable languages give a much larger class than just the semi-polynomial sets. For example, the language $\{a^k b^{2^k} \mid k \in \mathcal{N}\}$, whose Parikh image is not semi-polynomial (see the set A_2 above), is 2-PDA recognizable. A nice 2-PDA construction[2] uses bits as top stack symbols, combined to binary representations of numbers. For example, in the case of $k = 4$ the number 12 with binary representation 1100 is coded by the 2-stack

$$\begin{aligned}&[[0\bot],\\&[0Z\bot],\\&[1ZZ\bot],\\&[1ZZZ\bot]]\quad.\end{aligned}$$

It is not difficult to implement the counting process from 0 to $2^k - 1$ using this structure and process input b^{2^k}, starting from a 1-stack of length k, which is produced upon input a^k.

The 2-PDA of Example 3 does not require the special symbol Z_2; it can be turned into a 'level 2 counter automaton' with stack symbols $Z, \bot$ only. This model also suffices to recognize $\{a^m b^n c^{mn} \mid m, n \in \mathcal{N}\}$, whose Parikh image is the (non-semi-polynomial) product relation A_3 (construct a stack of length m and copy it n times, generating a stack of size mn). So even level 2 counter automata can generate sets which are not semi-polynomial.

4. Concluding Remarks

In this work, we looked at the power of 2-PDA's in term of their Parikh images. Although we have not succeeded in characterizing these sets, we have seen some of the ingredients needed for such characterization: polynomial terms as well as (one-fold) exponential terms.

In a recent work of Lisovik and Karnaukh [6], a related result is shown in the framework of indexed grammars, however aiming at the representability of unary functions $f\colon \mathcal{N} \to \mathcal{N}$ via indexed grammars over a unary terminal alphabet. Our treatment covers relations and thus functions of higher arity, and the model of 2-PDA's used here seems to give a more direct insight into the underlying computations.

Acknowledgements

I would like to thank Wolfgang Thomas for supervising this work.

References

[1] A.V. Aho: Indexed grammars—an extension of context-free grammars. Journal of the ACM **15** (1968) 647–671

[2] A. Carayol, S. Wöhrle: The Caucal hierarchy of infinite graphs in terms of logic and higher-order pushdown automata. In Proc. FSTTCS 2003. LNCS 2914. Springer (2003) 112–123

[3] W. Damm, A. Goerdt: An automata-theoretic characterization of the OI-hierarchy. In Proc. ICALP 1982. LNCS 140. Springer (1982) 141–153

[4] J. Engelfriet: Iterated pushdown automata and complexity classes. In Proc. STOC 1983. ACM Press (1983) 365–373

[5] W. Karianto: Parikh automata with pushdown stack. Diploma thesis, RWTH Aachen (2004)

[6] L.P. Lisovik, T.A. Karnaukh: A class of functions computable by index grammars. Cybernetics and Systems Analysis **39** (2003) 91–96

[7] A.N. Maslov: Multilevel stack automata. Problems of Information Transmission **12** (1976) 38–42

[8] R.J. Parikh: On context-free languages. Journal of the ACM **13** (1966) 570–581

[2] The underlying idea is due to Carayol and Wöhrle and is mentioned here with their kind permission.