Iterative Learning of Simple External Contextual Languages

Leonor Becerra-Bonache⋆[1] John Case[2] Sanjay Jain⋆⋆[3] Frank Stephan⋆⋆⋆[4]

[1] Department of Computer Science, Yale University,
New Haven, CT 06520-8285, United States of America, leonor.becerra-bonache@yale.edu
[2] Department of Computer and Information Sciences, University of Delaware,
Newark, DE 19716-2586, United States of America, case@cis.udel.edu
[3] Department of Computer Science, National University of Singapore,
Singapore 117417, Republic of Singapore. sanjay@comp.nus.edu.sg
[4] Department of Computer Science and Department of Mathematics, National University of Singapore,
Singapore 117543, Republic of Singapore. fstephan@comp.nus.edu.sg

Abstract. It is investigated for which choice of a parameter q, denoting the number of contexts, the class of simple external contextual languages is iteratively learnable. On the one hand, the class admits, for all values of q, polynomial time learnability provided an adequate choice of the hypothesis space is given. On the other hand, additional constraints like consistency and conservativeness or the use of a one-one hypothesis space changes the picture — iterative learning limits the long term memory of the learner to the current hypothesis and these constraints further hinder storage of information via padding of this hypothesis. It is shown that if $q > 3$, then simple external contextual languages are not iteratively learnable using a class preserving one-one hypothesis space, while for $q = 1$ it is iteratively learnable, even in polynomial time. It is also investigated for which choice of the parameters, the simple external contextual languages can be learnt by a consistent and conservative iterative learner.

1 Introduction

Consider the following model of learning a class of languages: a learner M is shown any listing of the strings of a language L in the class and M outputs a sequence of hypotheses as it sees successively more and more strings of L. M, eventually, stops changing its mind and the final output is a grammar that correctly generates L. This model of learning is called "explanatory learning" as the final grammar can be seen as an "explanation for the language to be learnt." Explanatory learning from positive data and its variants are frequently used to model important scenarios such as language acquisition of children [18].

Within the scenario of natural language acquisition, the formal study of this phenomenon requires answering the following question: Which kind of formal languages is adequate to model

⋆ Supported by a Marie Curie International Fellowship within the 6th European Community Framework Programme.
⋆⋆ Supported in part by NUS grant number R252-000-308-112.
⋆⋆⋆ Supported in part by NUS grant number R252-000-308-112.

natural languages? This question has been a subject of debate for a long time. This debate started soon after the publication of [11], and it was focused on determining whether natural languages are context-free (CF) or not. Nevertheless, in the late 80's, linguists seemed to agree finally that natural languages are not CF; there were discovered, in many natural languages, convincing examples of non-CF constructions [6, 12, 37], such as so-called *multiple agreements*, *crossed agreements* and *duplication structures*. Besides, these works suggested that more generative capacity than CF is necessary to describe natural languages. The following example needs some context-sensitivity:

Bill, Dick, Harry, ... gave Hillary, Pat, Bess, ...
candy bars, chocolates, flowers, ..., respectively.

This involves a (mentioned above) multiple-agreement construction and can be essentially modeled by the well-known, context-sensitive, not context-free set $\{a^n b^n c^n \mid n > 0\}$.

The difficulty of working more generally with context-sensitive languages has forced researchers to find other ways to generate CF and non-CF constructions, but keeping under control the generative power. This idea has led to the notion of a *mildly context-sensitive* (MCS) family of languages, introduced by Joshi [19].

In the literature, different definitions of MCS have been presented. In this paper, by a *mildly context-sensitive* family of languages we mean a family $\mathcal{L}$ of languages that satisfies the following conditions:

(1) each language in $\mathcal{L}$ is semilinear [32];
(2) for each language in $\mathcal{L}$ the membership problem is solvable in deterministic polynomial time;
(3) $\mathcal{L}$ contains the following three non-context-free languages:
- *multiple agreements*: $L_1 = \{a^n b^n c^n \mid n \geq 0\}$;
- *crossed agreements*: $L_2 = \{a^n b^m c^n d^m \mid n, m \geq 0\}$;
- *duplication*: $L_3 = \{ww \mid w \in \{a, b\}^*\}$.

Some authors [20, 35, 39] consider that such a family contains all CF languages and present mechanisms that fabricate mildly context-sensitive families which fully cover the CF but not the CS level of the Chomsky Hierarchy. However, taking into account the linguistic motivation of the concept of MCS, the following question arises: is it necessary that such a formalism generates all CF languages? As some authors [2, 3, 21, 25] pointed out, natural languages could occupy an orthogonal position in the Chomsky Hierarchy. In fact, we can find some examples of natural language constructions that are neither regular (REG) nor CF and also some REG or CF constructions that do not appear naturally in sentences. Therefore, it is justified to give up the requirement of generating all CF languages, and we strive for formalisms which generate MCS languages in the above sense and occupy an orthogonal position in the Chomsky Hierarchy. Furthermore, Gold [17] showed that any language class containing an infinite language and all its finite sublanguages (such as CF) is not explanatory learnable from positive data.

One example of a mechanism with these desirable linguistic properties is the *Simple External Contextual grammars* ($\mathrm{SEC}_{p,q}$ grammars, where p, q are parameters discussed below). Note that, on the one hand, the corresponding class $\mathrm{SEC}_{p,q}$ is, for $p, q > 1$, a mildly context-sensitive class.

So the context-sensitive structures that led to the non-context-freeness of natural languages (multiple agreement, crossed agreement and duplication) can be covered by such grammars [2], as we shall see in Section 2 below:

– $\{a^n b^n c^n \mid n \geq 0\}$ is in $\mathrm{SEC}_{2,1}$;
– $\{a^m b^n c^m d^n \mid n, m \geq 0\}$ and $\{ww \mid w \in \{a, b\}^*\}$ are in $\mathrm{SEC}_{2,2}$.

On the other hand, such classes $\mathrm{SEC}_{p,q}$ are incomparable with the families REG and CF, but included in CS [2] — that is, they occupy an orthogonal position in the Chomsky Hierarchy. So, due to their properties, the $\mathrm{SEC}_{p,q}$'s may be appropriate candidates to model some aspects of the syntax of natural languages. Moreover, the $\mathrm{SEC}_{p,q}$ grammar mechanism is (technically) quite simple and intuitively could be explained as follows: In the sentence "Anton learns English" one could add more objects and obtain "Anton learns English, French, German and Spanish." Similarly the sentence "Gerd goes to France, Spain and Holland on Thursday, Saturday and Sunday, respectively" can be extended by expanding the list of countries and corresponding days, but for each new country also a new day has to be added. So, the idea is to start with an easy base sentence and then add new parts at several places in a consistent manner. One can think of the parameter p as the number of positions in a base where additions can be inserted and the parameter q as the number of various contexts which can be inserted. In Section 2 below, we present the rigorous definitions.

Becerra-Bonache and Yokomori [3] made the first attempt to learn these $\mathrm{SEC}_{p,q}$ grammars from only positive data; they show that for each choice of parameters $p, q \geq 1$, $\mathrm{SEC}_{p,q}$ is explanatorily learnable from positive data. They employ Shinohara's results [38]. However, the learning algorithm derived from their main result was not time-efficient. In [28], efficient learning of $\mathrm{SEC}_{p,q}$ for some small values of the parameters p, q is considered.

The $\mathrm{SEC}_{p,q}$ classes have their roots in contextual grammars [26], which were introduced with linguistic motivations (to model some natural aspects, such as, for example, the acceptance of a word only in certain contexts). For an overview on contextual grammars the reader is referred to [33]. Fernau and Holzer [13] investigated learnability of classes of external contextual languages different from those of the present paper.

Human memory for past data seen seems to have limitations, see, for example, [7, 8, 14, 24, 30, 40] for results featuring memory-limited learners. The present paper is about a nicely *memory-limited* form of explanatory learning from positive data called *iterative learning*. Each output grammar/hypothesis of an *iterative learner* depends only on the just prior, if any, hypothesis it output and on the string currently seen from an input listing of a language.

Our main positive results (Theorems 8 and 21, the latter together with Remark 23, below in Sections 3 and 6, respectively) actually feature polynomial time learnability. This roughly means that the update time of the associated learner M is polynomial in the size of its previous hypothesis and the latest datum. Here the size of the hypotheses themselves is bounded by a polynomial in the size of all the input data seen so far (thus the learner is *fair* and runs in time polynomial in all the data seen so far). In the prior literature on polynomial time explanatory learning (e.g., [22, 34, 42]), there are a number of suggestions on how to rule out unfair delaying tricks — unfair delaying tricks such as waiting for a long datum to have enough time to output an hypothesis actually based on a much shorter earlier datum. Fortunately, iterative learning (as in the present paper) is one of the best ways to rule out such delaying tricks. Intuitively, this is because the learner M does not have available for employment its whole history of past data. Theorem 8 says that, for each $p, q \geq 1$, the class $\mathrm{SEC}_{p,q}$ is iteratively learnable in polynomial time using a class-preserving hypothesis space.

Of course, an iterative M can pad up its conjectured hypotheses to store a limited amount of past data seen. For example, some dummy information not affecting the semantics of an hypothesis can be added to it to, in effect, code in some bounded information about past data. In fact the proof of the positive result Theorem 8, depends on such a padding trick. It is thus interesting to see if such a result can still hold if we outlaw padding tricks in some natural ways.

One way to outlaw padding is to require the hypothesis space to be one-one, that is, to require that there is exactly one hypothesis available per relevant language. Another main result (Theorem 16 below in Section 4) says that *for class-preserving one-one hypothesis spaces*, for $p \geq 1$ and $q \geq 4$, $\mathrm{SEC}_{p,q}$ is not iteratively learnable. By contrast, Theorem 21, together with Remark 23, provides, for each p, for a class-preserving one-one hypothesis space, polynomial time iterative learnability of $\mathrm{SEC}_{p,1}$.

For a *consistent learner*, every hypothesis conjectured by the learner must generate all the data seen to that point. A *conservative learner* revises its hypothesis only if a current datum is inconsistent with it. Iterative learners which are both consistent and conservative are restricted in how much padding they can use. Another main result is that, for $p \geq 1, q \geq 2$, $\mathrm{SEC}_{p,q}$ is not learnable by consistent and conservative iterative learners using a class-preserving hypothesis space.

In the remainder of the present paper, for the values of q not covered in each of the just above two paragraphs, we provide some partial results.

In a recent paper Yoshinaka [42] suggests some other possible requirements to place on a learner, such as consistency, conservativeness, polynomial number of mind changes before convergence to final hypothesis (on arbitrary texts) and so on. Most of these additional requirements are not satisfied by our positive results, except that in some cases we are able to achieve consistency and/or conservativeness. Here note that, for the learner for $\mathrm{SEC}_{p,1}$ in Theorem 21, for each $L \in \mathrm{SEC}_{p,1}$, there exists a subset D_L of cardinality at most 3, such that the presence of D_L in the input already implies convergence to the correct hypothesis by the learner.

2 Notation and Preliminaries

For any unexplained recursion theoretic notation, the reader is referred to the textbooks of Rogers [36] and Odifreddi [29]. The symbol $\mathbb{N}$ denotes the set of natural numbers, $\{0, 1, 2, 3, \ldots\}$. For S a finite, non-empty subset of $\mathbb{N}$, $\gcd(S)$ denotes the greatest common divisor of the elements in S. Σ denotes a finite alphabet set. Subsets of Σ^* are referred to as languages. The symbols $\emptyset$, $\subseteq$, $\subset$, $\supseteq$ and $\supset$ denote empty set, subset, proper subset, superset and proper superset, respectively. The cardinality of a set S is denoted by $|S|$. Let $|x|$ denote the length of the string x, where we take, then, $x = x(0)x(1)\ldots x(|x|-1)$. For $n \leq |x|$ let $x[n]$ denote the string formed from the first n characters of x. For i, j with $i \leq j < |x|$ let $x[i,j]$ denote the substring $x(i)x(i+1)\ldots x(j)$; if $j < i$ or $i \geq |x|$ or $j \geq |x|$ then $x[i,j] = \epsilon$, the empty string. Furthermore $x \cdot y$ or just xy denotes the concatenation of the strings x and y.

We often use regular expressions to define languages: For example, $A + B$ denotes the union $A \cup B$, x denotes $\{x\}$, $A - x$ denotes the set $A - \{x\}$, $A \cdot B = \{x \cdot y : x \in A, y \in B\}$. For example, $aa(bb+cc)^* = \{a \cdot a \cdot x : x \in \{b \cdot b, c \cdot c\}^*\}$ and $a^3 \cdot a^* = \{a^n : n \geq 3\}$.

We now present concepts from language learning theory. Sets of the form $\{x : x < n\}$, for some n, are called initial segments of $\mathbb{N}$. The next definition introduces the concept of a *sequence* of data.

Definition 1. (a) A (*finite*) *sequence* σ is a mapping from an initial segment of $\mathbb{N}$ into $(\Sigma^* \cup \{\#\})$. The empty sequence is denoted by λ.

(b) The *content* of a sequence σ, denoted $\mathrm{ctnt}(\sigma)$, is the set of elements occurring in σ that are different from $\#$.

(c) The *length* of σ, denoted by $|\sigma|$, is the number of elements in σ. So, $|\lambda| = 0$.

(d) For $n \leq |\sigma|$, the initial sequence of σ of length n is denoted by $\sigma[n]$. So, $\sigma[0]$ is λ.

Intuitively, $\#$'s represent pauses in the presentation of data. We let σ, τ and γ range over finite sequences.

There are two types of concatenation: $\diamond$ is the symbol for concatenation of sequences (including those consisting of one string only) while $\cdot$ denotes the concatenation of strings or sets of strings. So $a^3 \diamond a^5 \diamond a^8$ is the sequence a^3, a^5, a^8; while $a^3 \cdot a^5 \cdot a^8$ is the string a^{16}.

Definition 2 (Gold [17]). (a) A *text* T for a language L is a mapping from $\mathbb{N}$ into $(\Sigma^* \cup \{\#\})$ such that L is the set of all strings occurring in the range of T. $T(i)$ represents the $(i+1)$-st element in the text.

(b) The *content* of a text T, denoted by $\mathrm{ctnt}(T)$, is the set of elements occurring in T that are different from $\#$; that is, the language which T is a text for.

(c) $T[n]$ denotes the finite initial sequence of T with length n.

Definition 3. (Gold [17], Case and Lynes [9], Wiehagen [41] and Lange and Zeugmann [24]). Let $H_0, H_1, H_2, \ldots$ be the underlying hypothesis space.

(a) An *iterative learner* or, in this paper, just a *learner*, is a total recursive mapping M from $(\mathbb{N} \cup \{?\}) \times (\Sigma^* \cup \{\#\})$ to $\mathbb{N} \cup \{?\}$. The definition of M is extended to finite sequences by

$$\begin{aligned} \forall n\ \forall x_0, x_1, \ldots, x_n \in \Sigma^* \cup \{?\}\ [M(x_0 \diamond x_1 \diamond \ldots \diamond x_n) \ &= \\ M(M(\ldots M(M(?, x_0), x_1), \ldots), x_n)&] \end{aligned}$$

and every expression $M(\sigma)$, for a finite sequence σ, refers to this extension.

(b) M learns a language L from text T iff there is an index e with $H_e = L$ and $M(T[n]) = e$ for almost all n.

(c) A class $\mathcal{L}$ of languages is *iteratively learnable* iff there is an iterative learner M such that M learns every language $L \in \mathcal{L}$ from every text T for L.

Intuitively, an iterative learner [24, 41] is a learner whose hypothesis depends only on its last conjecture and current input. That is, for $n \geq 0$, $M(T[n+1])$ can be computed algorithmically from $M(T[n])$ and $T(n)$. Here, note that $M(T[0]) = ?$.

Furthermore, note that we, in part (a) of the above definition, required M to be *total recursive* instead of just partial recursive, where, for successful iterative learning of a language L, M would have to be defined on the initial segments on any text for L. It is a folklore result that this makes a difference as to what can be iteratively learned (see [10] for a proof). Our positive results happen to hold for total iterative learners M and our negative results would also hold if we removed the totality restriction. Therefore, it is for expository convenience that we consider herein, for all of our results, total iterative learners.

Definition 4 (L. and M. Blum and Fulk [5, 16]). A finite sequence σ is said to be a *stabilizing sequence* for M on L iff

- $\mathrm{ctnt}(\sigma) \subseteq L$ and
- $M(\sigma \diamond \tau) = M(\sigma)$ for all τ with $\mathrm{ctnt}(\tau) \subseteq L$.

Furthermore, σ is said to be a *locking sequence* for M on L iff both σ is a stabilizing sequence for M on L and $M(\sigma)$ is an index of L (in the hypothesis space used by M).

If M learns L, then every stabilizing sequence for M on L is a locking sequence for M on L. Furthermore, one can show that if M learns L, then for every σ such that $\mathrm{ctnt}(\sigma) \subseteq L$, there exists a locking sequence for M on L which extends σ, see [5, 16].

Definition 5 (Angluin and L. and M. Blum [1, 5]). Let $H_0, H_1, H_2, \ldots$ be the hypothesis space used by M.

(a) M is said to be *consistent* on σ if $M(\sigma) = ?$ or $\mathrm{ctnt}(\sigma) \subseteq H_{M(\sigma)}$. M is *consistent* on text T iff M is consistent on $T[n]$, for all n. M is *consistent* on L if M is consistent on each text for L. M is *consistent* on $\mathcal{L}$ if M is consistent on each $L \in \mathcal{L}$.

(b) M is said to be *conservative* on T if for all $n \in \mathbb{N}$, $M(T[n]) \neq ?$ and $\mathrm{ctnt}(T[n+1]) \subseteq H_{M(T[n])}$ implies $M(T[n+1]) = M(T[n])$. M is *conservative* on L if M is conservative on each text for L. M is *conservative* on $\mathcal{L}$ if M is conservative on each $L \in \mathcal{L}$.

For consistent / conservative learning of a class $\mathcal{L}$, we require the learners to be consistent / conservative on each language in $\mathcal{L}$.

Kudlek, Martín-Vide, Mateescu and Mitrana [21] introduced and studied a mechanism to fabricate MCS families of languages called p-dimensional external contextual grammars.

Let $p \geq 1$ be a fixed integer. A p-word is a p-tuple $(w_1, w_2, \ldots, w_p)$ of strings. A p-context is a $2p$-word. An $\mathrm{SEC}_{p,q}$ language can be represented by an $\mathrm{SEC}_{p,q}$ grammar defined as follows.

Definition 6 (Becerra-Bonache and Yokomori [3]). Fix Σ. A *simple external contextual grammar with parameters p and q* (an $\mathrm{SEC}_{p,q}$ grammar) is a pair $G = (base, C)$, where $base$ is a p-word over Σ, and C is a set of p-contexts of cardinality at most q.

Given a p-word $w = (w_1, w_2, \ldots, w_p)$ and a p-context $u = (u_1, u_2, \ldots, u_{2p-1}, u_{2p})$, $\mathrm{gen}(w, u)$ is the p-word $(u_1 w_1 u_2, u_3 w_2 u_4, u_5 w_3 u_6, \ldots, u_{2p-1} w_p u_{2p})$. We generalize the definition of gen to multiple contexts by saying $\mathrm{gen}(w, C) = \{\mathrm{gen}(w, u) : u \in C\}$.

Suppose that a p-word *base* and a set C of p-contexts are given. Then we obtain $Lang(base, C)$ by considering the smallest set S satisfying the following two conditions:

- $base \in S$;
- If p-word $w \in S$ and p-context $u \in C$, then $\mathrm{gen}(w, u) \in S$.

By Kleene's Minimal Fixed-Point Theorem [36], such a set S uniquely exists and is recursively enumerable. Now

$$Lang(base, C) = \{w_1 w_2 \ldots w_p : (w_1, w_2, \ldots, w_p) \in S\}.$$

We also refer to $(base, C)$ as grammar for $Lang(base, C)$. Let $\mathcal{G}_{p,q} = \{(base, C) : base$ is a p-word and C is a set of at most q p-contexts$\}$. Let $\mathrm{SEC}_{p,q} = \{Lang(G) : G \in \mathcal{G}_{p,q}\}$. Furthermore, let $\mathrm{SEC}_{p,*} = \bigcup_{j \in \{1,2,\ldots\}} \mathrm{SEC}_{p,j}$ and $\mathrm{SEC}_{*,q} = \bigcup_{j \in \{1,2,\ldots\}} \mathrm{SEC}_{j,q}$.

For example, $\{a^n b^n c^n \mid n \geq 0\}$ is generated by the $\mathrm{SEC}_{2,1}$ grammar $G_{2,1} = (base = (\epsilon, \epsilon), context = \{(a, b, c, \epsilon)\})$.

$\{ww \mid w \in \{a, b\}^*\}$ is generated by the $\mathrm{SEC}_{2,2}$ grammar $G_{2,2} = (base = (\epsilon, \epsilon), context = \{(\epsilon, a, \epsilon, a), (\epsilon, b, \epsilon, b)\})$.

$\{a^n b^m c^n d^m : n, m \geq 0\}$ is generated by the $\mathrm{SEC}_{2,2}$ grammar $G_{2,2} = (base = (\epsilon, \epsilon), context = \{(a, \epsilon, c, \epsilon), (\epsilon, b, \epsilon, d)\})$.

Thus, for any $p \geq 2, q \geq 2$, $\mathrm{SEC}_{p,q}$ is a mildly context-sensitive family. For p or q being 1, the class $\mathrm{SEC}_{p,q}$ is not a mildly context-sensitive family.

We define the *size* of various objects of importance. The size of a string w is the length of the string w. The size of a p-word is the sum of p and the sizes of the strings in it. The size of a context set C is the sum of the cardinality of C and the sizes of the contexts in it. The size of a grammar G is the size of its base plus the size of its context set. The size of a finite sequence $x_0 \diamond x_1 \diamond x_2 \diamond \ldots \diamond x_n$ is the sum of $n+1$ and the size of all strings $x_0, x_1, x_2, \ldots, x_n$ in this finite sequence.

For us, an iterative learner M *runs in polynomial time* iff both its update-function $M(e, x)$ runs in time polynomial in the size of e and x and the size of $M(\sigma)$ is polynomially bounded in the size of σ for every finite sequence σ.

We say a learner which learns a class $\mathcal{L}$ is *class-preserving* iff its underlying hypothesis space does not generate any languages outside $\mathcal{L}$ [23].

3 $\mathrm{SEC}_{p,q}$ is Consistently Iteratively Learnable

Kudlek, Martín-Vide, Mateescu and Mitrana [21] noted that the membership question for languages in $\mathrm{SEC}_{p,q}$ is decidable in polynomial time.

Proposition 7. *Fix p, q and let $(base, C)$ be a member of $\mathcal{G}_{p,q}$. Given a string x, it can be decided in polynomial time (in the size of $x, base, C$) whether $x \in Lang(base, C)$. The degree of the polynomial is linear in p and independent of q.*

Proof. Let S_i be the set of p-words defined as follows. $S_0 = \{base\}$. $S_{i+1} = \{\text{gen}(w, u) : u \in C, w \in S_i\}$. Let VT be the set of all tuples of form $(i_1, j_1, \ldots, i_p, j_p)$ such that, for all $r \in \{1, 2, \ldots, p\}$, $0 \leq i_r \leq j_r + 1 \leq |x|$ and $r < p \Rightarrow j_r < i_{r+1}$. Note that the cardinality of VT is at most $(|x| + 1)^{2p}$. Let X_k denote the set of tuples $(i_1, j_1, \ldots, i_p, j_p)$ in VT such that some $(w_1, w_2, \ldots, w_p) \in S_k$ satisfies $w_r = x[i_r, j_r]$, for r with $1 \leq r \leq p$.

For each tuple of the form $(i_1, j_1, \ldots, i_p, j_p)$, its membership in X_0 can be tested in time $O(|x|)$. Using X_k, one can determine if a tuple $(i_1, j_1, \ldots, i_p, j_p) \in VT$ belongs to X_{k+1} by checking if $(x[i_1, j_1], \ldots, x[i_p, j_p]) = gen(x[i'_1, j'_1], \ldots, x[i'_p, j'_p], u)$, for some $u \in C$ and $(i'_1, j'_1, \ldots, i'_p, j'_p) \in X_k$. Thus, one can compute X_{k+1} from X_k in time $O(size(C) * (|x| + 1)^{4p+1})$. It follows that one can compute X_r in time $O(size(C) * (|x| + 1)^{4p+1} * r)$.

Now $x \in Lang(base, C)$ iff $x = x[i_1, j_1]x[i_2, j_2] \ldots x[i_p, j_p]$ for some $(i_1, j_1, \ldots, i_p, j_p) \in X_r$, for some $r \leq |x|$. $\square$

Theorem 8. *For each $p, q \in \{1, 2, 3, \ldots\}$, $SEC_{p,q}$ has a polynomial time iterative consistent learner M which uses a class-preserving hypothesis space. The runtime of M (measured in terms of the size of the previous hypothesis and current input-datum) and the size of M's conjecture (measured in terms of the size of all input data seen so far) are each bounded by a polynomial of degree linear in pq.*

Proof. A pair $(base, C)$ of a base and a context set is said to *minimally generate* a set Z iff $Z \subseteq Lang(base, C)$ and there is no $C' \subset C$ with $Z \subseteq Lang(base, C')$.

For a fixed p, q and a finite set Z, let X_Z denote the set of elements of $\mathcal{G}_{p,q}$ which minimally generate Z. For any string x, one can determine $X_{\{x\}}$, by considering all possible $(base, C)$, such that

- $base = (w_1, w_2, \ldots, w_p)$ where each w_i is a substring of x,
- for each member $(u_1, u_2, \ldots, u_{2p-1}, u_{2p})$ of C, each u_i is a substring of x,
- C contains at most q contexts,
- $(base, C)$ minimally generates $\{x\}$.

Note that this can be done in polynomial time in the length of x, as the number of possible substrings of x is at most $(|x| + 1)^2$, and thus the number of possible grammars $(base, C)$ is at most $((|x| + 1)^2)^{p+2pq}$. For this, if the number of contexts is less than q, one could consider the rest of the contexts as "empty."

For finding $(base, C)$ which minimally generate $Z \cup \{x\}$, note that if $(base, C)$ minimally generates $Z \cup \{x\}$, then for some $C', C'' \subseteq C$, $(base, C')$ minimally generates Z and $(base, C'')$ minimally generates $\{x\}$ and $C = C' \cup C''$. Additionally, note that it must be the case that no proper subset C_s of C satisfies that $(base, C_s)$ generates $Z \cup \{x\}$.

In other words, for a nonempty set Z, given X_Z and $X_{\{x\}}$, note that $X_{Z \cup \{x\}}$ consists of grammars $(base, C) \in \mathcal{G}_{p,q}$ such that

– for some C', C'' satisfying $C = C' \cup C''$ it holds that $(base, C') \in X_Z$ and $(base, C'') \in X_{\{x\}}$,
– no $C''' \subset C$ satisfies the property above (for the given $base$ with C''' in place of C).

Thus, one can determine $X_{Z\cup\{x\}}$ from X_Z and x in time polynomial in $|x|$ and the size of X_Z. Furthermore, the size of X_Z is polynomial in the size of Z (as each string in the base and in the set of contexts is a substring of one of the strings in Z and there are at most q contexts in each $(base, C) \in X_Z$). Thus, one can compute $X_{Z\cup\{x\}}$ in time polynomial in the size of $Z \cup \{x\}$, given X_Z and x.

Now consider an arbitrary text T. Then we claim that $\lim_{n\to\infty} X_{\mathrm{ctnt}(T[n])}$ converges. This can be seen as follows. One may assume without loss of generality that T does not contain any # (as input # does not lead to modification of $X_{\mathrm{ctnt}(T[n])}$). Now consider a forest formed as follows. F_1 consists of $|X_{\mathrm{ctnt}(T[1])}|$ roots corresponding to each member of $X_{\mathrm{ctnt}(T[1])}$ (labeled using the corresponding member). By induction we will have that $X_{\mathrm{ctnt}(T[n])}$ would be a subset of the set of leaves of the forest F_n. F_{n+1} is constructed by possibly adding some children to leaves of F_n as follows. If $(base, C) \in X_{\mathrm{ctnt}(T[n+1])} - X_{\mathrm{ctnt}(T[n])}$, then pick a $(base, C') \in X_{\mathrm{ctnt}(T[n])}$ such that $C' \subset C$ (there exists such a $(base, C')$ by construction). Add $(base, C)$ as a child of $(base, C')$. As the depth of the forest is at most q and the number of roots and the branching factor at any node is finite, the sequence $F_1, F_2, \ldots$ converges.

Now one considers iterative learning of $\mathrm{SEC}_{p,q}$. Let $g(\cdot)$ be a one-one polynomial time computable and polynomial time invertible coding of all finite sets of grammars over $\mathcal{G}_{p,q}$. The learner uses a class-preserving hypothesis space H such that $H_{g(X)}$ is for a minimal language in $\{Lang(G) : G \in X\}$ (for $X = \emptyset$, we let $g(X)$ to be a grammar for $\{\epsilon\}$). Note that such a class-preserving hypothesis space H can be constructed by letting, for $X \neq \emptyset$, $x \in H_{g(X)}$ iff $x \in Lang(base, C)$ for all $(base, C) \in X$ such that for all y length lexicographically smaller than x it holds that $y \in Lang(base, C) \Leftrightarrow y \in H_{g(X)}$. In the construction of $H_{g(X)}$, a minimal language instead of an intersection of languages is used to have a class-preserving hypothesis space rather than class-comprising one (a *class-comprising* hypothesis space can also have hypotheses for languages not in the class of languages under consideration [23]).

The learner, on input $T[n]$, outputs the hypothesis $g(X_{\mathrm{ctnt}(T[n])})$, if $\mathrm{ctnt}(T[n]) \neq \emptyset$. Otherwise, the learner outputs ?. Note that $X_{\mathrm{ctnt}(T[n+1])}$ can be iteratively computed using $T(n)$ and $X_{\mathrm{ctnt}(T[n])}$ (which can be obtained from $g(X_{\mathrm{ctnt}(T[n])})$). Here note that, if the input language belongs to $\mathrm{SEC}_{p,q}$, then (a) every grammar in $\lim_{n\to\infty} X_{\mathrm{ctnt}(T[n])}$ contains the input language, (b) there is a grammar for the input language in $\lim_{n\to\infty} X_{\mathrm{ctnt}(T[n])}$. Thus, for large enough n, $H_{g(X_{T[n]})}$ is the input language. □

Remark 9. Note that the membership question for the hypothesis space $H_{g(X)}$ in the preceding proof may not be decidable uniformly in polynomial time (as finding the minimal language in X may not be easy). On the other hand, if one had chosen the class comprising hypothesis space obtained by taking $H_{g(X)} = \bigcap_{(base,C)\in X} Lang(base, C)$, then the membership problem is uniformly decidable in polynomial time, though the hypothesis space is no longer class preserving. Alternatively, using the following hypothesis space one could still make the hypothesis space class preserving, with the membership question being uniformly decidable in polynomial time, though the convergence to a final hypothesis would be delayed somewhat. For this, we consider the hypothesis space given by $H_{g(X,r)} = L(G)$, where G is the $(r+1)$-st element of X, in the lexicographic ordering of grammars in X. Now, if the previous hypothesis of the learner (that is the output of the learner on input $T[n]$) is $g(X_{T[n]}, r)$ and the new input is $T(n)$, then the output of the learner on input $T[n+1]$ is $g(X_{T[n+1]}, r')$, where r' is defined as follows. Let G be the $(r+1)$-st grammar in the lexicographic ordering of $X_{T[n]}$.

$$r' = \begin{cases} 0, & \text{if } X_{T[n+1]} \neq X_{T[n]}; \\ r, & \text{if } X_{T[n+1]} = X_{T[n]} \text{ and} \\ & L(G) \cap \{x : |x| \leq \frac{\log(|T(n)|)}{|X_{T[n]}|}\} = \\ & (\bigcap_{G' \in X_{T[n]}} L(G')) \cap \{x : |x| \leq \frac{\log(|T(n)|)}{|X_{T[n]}|}\}; \\ r' = (r+1) \bmod |X_{T[n]}|, & \text{otherwise.} \end{cases}$$

Example 10. To explain the working of the algorithm, we now consider an example. Suppose the class being learnt is in $\mathrm{SEC}_{2,1}$. Suppose the input text T is for the language $\{a^n b^n c^n : n \geq 0\}$.

Suppose the initial example is $a^2b^2c^2$. Then, $X_{\{a^2b^2c^2\}}$ will contain all the $\mathrm{SEC}_{2,1}$ grammars which minimally generate $a^2b^2c^2$. These will include,

$((\epsilon, \epsilon), \{(\epsilon, a, b, c)\})$, $((\epsilon, \epsilon), \{(a, \epsilon, b, c)\})$, $((\epsilon, \epsilon), \{(a, b, \epsilon, c)\})$,
$((\epsilon, \epsilon), \{(a, b, c, \epsilon)\})$, $((ab, c), \{(a, b, c, \epsilon)\})$, $((ab, c), \{(a, b, \epsilon, c)\})$,
$((a, bc), \{(a, \epsilon, b, c)\})$, $((a, bc), \{(\epsilon, a, b, c)\})$,

and several other grammars such as

$((a, bc), \{(\epsilon, ab, \epsilon, c)\})$, $((ab, c), \{(a, \epsilon, bc, \epsilon)\})$, $((a, b), \{(a, b, \epsilon, cc)\})$,
$((a, \epsilon), \{(\epsilon, abb, c, c)\})$, $((aabbcc, \epsilon), \emptyset)$, $((aab, bcc), \emptyset)$ and so on.

Suppose the next example is abc. Then, $X_{\{abc\}}$ will contain all the $\mathrm{SEC}_{2,1}$ grammars which minimally generate abc. These will include,

$((\epsilon, \epsilon), \{(\epsilon, a, b, c)\})$, $((\epsilon, \epsilon), \{(a, \epsilon, b, c)\})$, $((\epsilon, \epsilon), \{(a, b, \epsilon, c)\})$,
$((\epsilon, \epsilon), \{(a, b, c, \epsilon)\})$, $((a, bc), \emptyset)$, $((ab, c), \emptyset)$,

as well as several other grammars such as

$((a, b), \{(\epsilon, \epsilon, \epsilon, c)\})$, $((a, \epsilon), \{(\epsilon, b, \epsilon, c)\})$ and so on.

This will then result in $X_{\{a^2b^2c^2, abc\}}$ containing

$((\epsilon, \epsilon), \{(\epsilon, a, b, c)\})$, $((\epsilon, \epsilon), \{(a, \epsilon, b, c)\})$, $((\epsilon, \epsilon), \{(a, b, \epsilon, c)\})$,
$((\epsilon, \epsilon), \{(a, b, c, \epsilon)\})$, $((ab, c), \{(a, b, c, \epsilon)\})$, $((ab, c), \{(a, b, \epsilon, c)\})$,
$((a, bc), \{(a, \epsilon, b, c)\})$, $((a, bc), \{(\epsilon, a, b, c)\})$,

as well as

$((a, bc), \{(\epsilon, ab, \epsilon, c)\})$, $((ab, c), \{(a, \epsilon, bc, \epsilon)\})$.

Note that grammars such as $((aabbcc, \epsilon), \emptyset)$, $((aab, bcc), \emptyset)$, $((aa, cc), \{(\epsilon, b, b, \epsilon)\})$, in $X_{\{a^2b^2c^2\}}$, will not have any grammar in $X_{\{abc\}}$ with the same base. Grammars such as $((a, b), \{(a, b, \epsilon, cc)\})$ will have a corresponding grammar in $X_{\{abc\}}$ with the same base — however, the combination of the contexts will give a context set of size more than 1 and thus would not be considered.

Now suppose ϵ is the next example. Then, $X_{\{\epsilon\}}$ will consist of $((\epsilon, \epsilon), \emptyset)$, which will then lead to $X_{\{a^2b^2c^2, abc, \epsilon\}}$ containing only the grammars:

$((\epsilon, \epsilon), \{(\epsilon, a, b, c)\})$, $((\epsilon, \epsilon), \{(a, \epsilon, b, c)\})$, $((\epsilon, \epsilon), \{(a, b, \epsilon, c)\})$,
$((\epsilon, \epsilon), \{(a, b, c, \epsilon)\})$.

The learner will not change $X_{T[n]}$ beyond the above for any further examples from the set $\{a^n b^n c^n : n \geq 0\}$.

In this particular example all the grammars in $\lim_{n\to\infty} X_{T[n]}$ were equivalent. However this may not be the case in general. For example, if one considers text T for $\{a^{2n} : n \in \mathbb{N}\}$, then one would have $\lim_{n\to\infty} X_{T[n]}$ containing 10 grammars for $\{a^{2n} : n \in \mathbb{N}\}$:

$((\epsilon, \epsilon), \{(aa, \epsilon, \epsilon, \epsilon)\})$, $((\epsilon, \epsilon), \{(a, a, \epsilon, \epsilon)\})$, $((\epsilon, \epsilon), \{(a, \epsilon, a, \epsilon)\})$,
$((\epsilon, \epsilon), \{(a, \epsilon, \epsilon, a)\})$, $((\epsilon, \epsilon), \{(\epsilon, aa, \epsilon, \epsilon)\})$, $((\epsilon, \epsilon), \{(\epsilon, a, a, \epsilon)\})$,
$((\epsilon, \epsilon), \{(\epsilon, a, \epsilon, a)\})$, $((\epsilon, \epsilon), \{(\epsilon, \epsilon, aa, \epsilon)\})$, $((\epsilon, \epsilon), \{(\epsilon, \epsilon, a, a)\})$,
$((\epsilon, \epsilon), \{(\epsilon, \epsilon, \epsilon, aa)\})$.

Furthermore $\lim_{n\to\infty} X_{T[n]}$ would contain the four grammars for $\{a^n : n \geq \mathbb{N}\}$:

$((\epsilon, \epsilon), \{(a, \epsilon, \epsilon, \epsilon)\})$, $((\epsilon, \epsilon), \{(\epsilon, a, \epsilon, \epsilon)\})$, $((\epsilon, \epsilon), \{(\epsilon, \epsilon, a, \epsilon)\})$,
$((\epsilon, \epsilon), \{(\epsilon, \epsilon, \epsilon, a)\})$.

These grammars are then all those which will be there in $\lim_{n\to\infty} X_{T[n]}$, and the final hypothesis will be equivalent to one of the first 10.

Corollary 11. *Suppose $\mathcal{G}$ is a recursive subset of $\mathcal{G}_{p,q}$ such that*

- *for all $C' \subseteq C$, $(base, C) \in \mathcal{G} \Rightarrow (base, C') \in \mathcal{G}$,*
- *for all p-words base, $(base, \emptyset) \in \mathcal{G}$,*
- *for all $G, G' \in \mathcal{G}$, one can effectively check whether $Lang(G) \subseteq Lang(G')$.*

Then,

(a) *$\mathcal{L} = \{Lang(G) : G \in \mathcal{G}\}$ is conservatively iteratively learnable using a class-preserving hypothesis space (this learner however may not be consistent);*

(b) *for each $G \in \mathcal{G}$, one can effectively find a finite $D(G) \subseteq Lang(G)$ such that, for all $G' \in \mathcal{G}$, if $D(G) \subseteq Lang(G')$, then $Lang(G) \subseteq Lang(G')$.*

Proof. This proof is obtained by a slight modification of the proof of Theorem 8 above. To define X_Z, as in the proof of Theorem 8, we use only grammars from $\mathcal{G}$. Also, g is a coding for all finite sets of grammars from $\mathcal{G}$. The hypothesis space H' used by the learner is defined by using $H'_{2g(X_{\mathrm{ctnt}(T[n])})}$ to be $H_{g(X_{\mathrm{ctnt}(T[n])})}$, where H is as defined in proof of Theorem 8 (for the modified g). We let $H'_{1+2g(X_{\mathrm{ctnt}(T[n])})}$ contain just the shortest element generated by all the grammars in $X_{\mathrm{ctnt}(T[n])}$. On input $T[n]$, the learner outputs $2g(X_{\mathrm{ctnt}(T[n])})$ if $X_{\mathrm{ctnt}(T[n])}$ contains a grammar G such that, for all $G' \in X_{\mathrm{ctnt}(T[n])}$, $Lang(G) \subseteq Lang(G')$. Otherwise, the learner outputs $1 + 2g(X_{\mathrm{ctnt}(T[n])})$.

The above learner is conservative: if the previous hypothesis output was $2g(X_{\mathrm{ctnt}(T[n])})$ or $1 + 2g(X_{\mathrm{ctnt}(T[n])})$, and the new input $T(n)$ belongs to the corresponding language, then $T(n)$ belongs to all $Lang(G)$, $G \in X_{\mathrm{ctnt}(T[n])}$ and thus $X_{\mathrm{ctnt}(T[n+1])} = X_{\mathrm{ctnt}(T[n])}$.

Define $D(G)$ as follows. Consider a text T for $Lang(G)$. Then, one could iteratively construct $X_{\mathrm{ctnt}(T[n])}$ until an n is found such that, for all $G' \in X_{\mathrm{ctnt}(T[n])}$, $Lang(G) \subseteq Lang(G')$. Then $D(G) = \mathrm{ctnt}(T[n])$ satisfies the requirements for part (b). $\square$

Note that if the subclass $\mathcal{G}$ consists of only grammars for regular languages (along with one being able to find effectively the DFA for the corresponding language), then it satisfies the third condition in Corollary 11. This is what is utilized in the applications of the above corollary in Proposition 12 and Section 7 below. It would be interesting to explore other important subclasses of $\mathrm{SEC}_{p,q}$ where the above corollary is applicable.

A tell-tale set [1] of a language $L \in \mathcal{L}$ is a finite set $S \subseteq L$ such that, for all $L' \in \mathcal{L}$, if $S \subseteq L'$, then $L' \not\subset L$. Note that $D(G)$ as defined in Corollary 11 is a tell-tale set.

Suppose $\mathcal{G} \subseteq \mathcal{G}_{p,q}$ satisfies the preconditions as in Corollary 11. Suppose further that, for all $G, G' \in \mathcal{G}$, $Lang(G) \not\subseteq Lang(G')$ implies there exists an x of length at most polynomial in the size of G, G' such that $x \in Lang(G) - Lang(G')$. Then the proof of Corollary 11 can be used to give a tell-tale set of polynomial size, as the branching factor of the forest formed in the proof of Theorem 8 would then be polynomially bounded in the size of G, when one considers an increasing text T for the input language.

If $|\Sigma| = 1$ then $\mathrm{SEC}_{p,q} = \mathrm{SEC}_{1,q}$, as the order of words in the base and the contexts does not matter. Furthermore, $Lang(G)$ is regular for each $G \in \mathcal{G}_{1,q}$ (where the automata for accepting $Lang(G)$ can be effectively obtained from G). Thus, for $|\Sigma| = 1$, $p, q \in \mathbb{N}$, $\mathrm{SEC}_{p,q}$ is conservatively iteratively learnable. The following proposition generalizes this to $\mathrm{SEC}_{p,*}$.

Proposition 12. *Fix $\Sigma = \{a\}$. Then, $SEC_{1,*}$ is iteratively learnable using a class preserving hypothesis space. The learner can be made consistent or conservative* (*but not both simultaneously*).

Proof. We first define $D(G)$, effectively obtainable from G, such that, for all $G' \in \mathcal{G}_{1,*}$, if $D(G) \subseteq Lang(G')$, then $Lang(G) \subseteq Lang(G')$. For ease of notation, we consider $G \in \mathcal{G}_{1,*}$ to be of the form (a^n, S), with $Lang(G) = a^n S^*$, where S is a finite set of strings.

Consider $G = (a^n, S)$. If $S = \emptyset$, then $D(G) = \{a^n\}$. If $S = \{a^{i_1}, a^{i_2}, \ldots, a^{i_r}\}$ is not empty, then let $m = \gcd(\{i_1, i_2, \ldots, i_r\})$. Thus, $Lang(G)$ is a finite variant of $a^n(a^m)^*$. Let i be minimal such that a^{n+im} and $a^{n+im+m} \in Lang(G)$. Now, for any set S',

- If $a^n \in a^s(S')^*$, then $s \leq n$;
- if $\{a^{n+im}, a^{n+im+m}\} \subseteq a^s(S')^*$, then $a^s\{a^{n+im-s}, a^{n+im+m-s}\}^* \subseteq a^s(S')^*$;
- for $s \leq n$, $Lang(G) - a^s\{a^{n+im-s}, a^{n+im+m-s}\}^*$ is finite and one can effectively (from s) find this set.

Let $D(G) = \{a^n, a^{n+im}, a^{n+im+m}\} \cup \bigcup_{s \leq n}[Lang(G) - a^s\{a^{n+im-s}, a^{n+im+m-s}\}^*]$. It follows that any language in $\mathrm{SEC}_{1,*}$ containing $D(G)$ also contains $Lang(G)$.

Now we can use the methods in the proofs of Theorem 8 and Corollary 11 to obtain an iterative consistent or an iterative conservative learner. Note that the existence of $D(G)$ as above is enough to guarantee the convergence of $X_{\mathrm{ctnt}(T[n])}$ for texts T for $Lang(G)$, as needed for learnability (as $X_{\mathrm{ctnt}(T[n])} = X_{D(G)}$, for all n such that $D(G) \subseteq \mathrm{ctnt}(T[n])$). □

The following non-learnability result holds even for the more general notions of explanatory learning [17] and behaviourally correct learning [4, 9, 31]. In the following theorem we use the term "learnable" to denote any of these notions of learnability.

Theorem 13. *Suppose that $\{a, b\} \subseteq \Sigma$. Then $SEC_{1,*}$ is not learnable.*

Proof. Let $L_i = b^* \cdot \{ab^j : j \leq i\}^*$. Let $H = \{a, b\}^*$. Note that $L_0 \subset L_1 \subset L_2 \subset \ldots \subset H$ and $H = \bigcup_{i \in \mathbb{N}} L_i$. Furthermore, $L_0, L_1, L_2, \ldots, H \in \mathrm{SEC}_{1,*}$. Thus, $\mathrm{SEC}_{1,*}$ is not learnable, by a result of Gold [17]. □

4 Padding is Necessary

Padding naturally needs that there are several hypotheses for at least some of the languages involved. Therefore it is natural to ask how learnability is affected in the case that there is only one grammar for each language in the class to be learnt. In such a situation, it is of course also needed to consider class-preserving hypothesis spaces as otherwise hypotheses for languages outside the class could be used to store information intermediately. For the following, since we are considering one-one hypothesis spaces, we often identify the language with its grammar.

Note that this assumption is adequate as $\mathrm{SEC}_{p,q}$ has a one-one hypothesis space. The reasons are that one can effectively decide the membership question for any language in $\mathrm{SEC}_{p,q}$ and any r.e. family of recursive functions can be made one-one.

Remark 14. Let a class-preserving one-one hypothesis space $H_0, H_1, H_2, \ldots$ of some class and an iterative learner M for this class be given. Then M is conservative. One can even show the following stricter variant:

$$(\forall \sigma)(\forall x \in H_{M(\sigma)})[M(\sigma \diamond x) = M(\sigma)].$$

If this condition would fail for some σ and $x \in H_{M(\sigma)}$, the learner M would not learn $H_{M(\sigma)}$ from any text T containing infinitely many x. This holds as for every n with $T(n) = x$, either $M(T[n]) \neq M(\sigma)$ or $M(T[n+1]) \neq M(\sigma)$, although $M(\sigma)$ is the only index of $H_{M(\sigma)}$.

Remark 15. Suppose M is an iterative learner for the class of languages $\mathcal{L}$ using a class preserving one-one hypothesis space. Suppose that M on σ outputs H and that T is a text for $L \supseteq H$, where $H, L \in \mathcal{L}$. Then M converges on $\sigma \diamond T$ to L.

To see this, consider a locking sequence τ for M on H. As $M(\sigma) = M(\tau)$, M converges on $\sigma \diamond T$ and $\tau \diamond T$ to the same hypothesis. As $\tau \diamond T$ is also a text for L, M converges on $\sigma \diamond T$ to an hypothesis for L as well.

Theorem 16. *Suppose that $p \in \{1, 2, 3, \ldots, *\}$ and $q \in \{4, 5, 6, \ldots, *\}$. Then $SEC_{p,q}$ is not iteratively learnable using a class preserving one-one hypothesis space.*

Proof. Let a be a member of the alphabet Σ. Suppose by way of contradiction that some iterative learner M learns $\mathrm{SEC}_{p,q}$ using a class preserving one-one hypothesis space. Let H be the set described by the hypothesis $M(\sigma_1)$ where $\sigma_1 = a^4 \diamond a^5$ (if $M(\sigma_1) = ?$, then clearly M cannot distinguish between $\sigma \diamond T$ and T where T is a text for $a^6 a^*$). The set H is not empty and thus has a shortest element, let n be its length. Now consider the following cases where Case i is only taken if no Case j with $j < i$ applies.

- Case 1: $H \not\subseteq a^*$. Let $x \in H - a^*$ and let T be a text for $a^* x^* - \{x\}$. This language is in $\mathrm{SEC}_{1,4}$ as one can generate it with base ϵ and the four contexts (a, ϵ), (a, x), (ϵ, x^2) and (ϵ, x^3). As M is conservative, $M(\sigma_1 \diamond x) = M(\sigma_1)$; so M converges on the text $\sigma_1 \diamond x \diamond T$ and the text $\sigma_1 \diamond T$ to the same hypotheses although these are texts for the different languages $a^* x^*$ and $a^* x^* - \{x\}$, respectively.
- Case 2: $n < 4$. Let T be a text for $a^{n+1} \cdot a^*$. As M is conservative, it converges to the same hypothesis on the texts $\sigma_1 \diamond T$ and $\sigma_1 \diamond a^n \diamond T$, which are texts for the different languages, $a^{n+1} a^*$ and $a^n a^*$, both of which are in $\mathrm{SEC}_{1,4}$.
- Case 3: $n > 4$. In this case let σ_2 be a stabilizing sequence for M on H. Let T be a text for $a^5 a^*$. Then, M on $\sigma_2 \diamond T$ as well as on $\sigma_1 \diamond T$ converges to the hypothesis for $a^5 a^*$, though $\sigma_1 \diamond T$ is a text for $a^4 a^*$.
- Case 4: $n = 4$ and $a^5 \notin H$. In this case, let T be a text for $a^4(a^2 + a^3)^*$ and let σ_3 be a locking sequence for M on H. Now the learner converges to the same grammar on $\sigma_3 \diamond T$ as on $\sigma_1 \diamond T$, though these are texts for $a^4(a^2 + a^3)^*$ and $a^4 a^*$ respectively.
- Case 5: $H = a^4 a^*$. As M is conservative, $M(\sigma_1 \diamond a^6) = M(\sigma_1)$. Now let T be a text for $a(a^3 + a^4)^*$. The learner M converges on $\sigma_1 \diamond a^6 \diamond T$ and on $\sigma_1 \diamond T$ to the same grammar, though these are, respectively, the texts for the languages $a(a^3 + a^4 + a^5)^*$ and $a(a^3 + a^4)^*$.

Hence, in all five cases, the learner M fails to infer some language it should infer. Note that if H contains a^4 and a^5 (but not a^n for $n < 4$), then the base for H is a^4, and one of the contexts is a; thus $H = a^4 a^*$. Hence, the case-distinction is exhaustive. So, the theorem follows. □

Another method to hinder padding is to require that a learner is consistent and conservative. Consistency enforces that the learner has to incorporate new data in a reasonable way so that no padding can be done by choosing a bogus hypothesis, conservativeness rules out updating done for data-storage purposes only.

Proposition 17. *Suppose that $p \in \{1, 2, 3, \ldots\}$ and $q \in \{2, 3, 4, \ldots, *\}$. Then $SEC_{p,q}$ has no consistent and conservative iterative learner using a class-preserving hypothesis space.*

Proof. Suppose by way of contradiction that $\mathrm{SEC}_{p,q}$ has a consistent and conservative iterative learner M. Suppose M on input $a^4 \diamond a^5$ outputs the hypothesis H. (If M on input $a^4 \diamond a^5$ outputs ?, then M cannot distinguish between the inputs $a^4 \diamond a^5 \diamond T$ and T, where T is a text for $a^6 a^*$.)

Note that every language $H \in \mathrm{SEC}_{p,q}$ which contains a^4, a^5 also contains either a^6 or a string $x \notin a^4 a^*$.

- Case 1: H contains a^6. Let σ be such that $\mathrm{ctnt}(\sigma) \subseteq a(a^3 + a^4)^*$ and $M(a^4 \diamond a^5 \diamond \sigma)$ is a grammar for $a(a^3 + a^4)^*$. Note that there exists such a σ as M identifies $a(a^3 + a^4)^*$. But then $M(a^4 \diamond a^5 \diamond \sigma) = M(a^4 \diamond a^5 \diamond a^6 \diamond \sigma)$ (as M is iterative and $M(a^4 \diamond a^5 \diamond a^6) = M(a^4 \diamond a^5)$ due to the conservativeness of M). Thus M is not consistent on $a^4 \diamond a^5 \diamond a^6 \diamond \sigma$, which is an initial segment of a text for a^*.
- Case 2: H contains $x \notin a^4a^*$. Let σ be such that $\mathrm{ctnt}(\sigma) \subseteq a^4a^*$ and $M(a^4 \diamond a^5 \diamond \sigma)$ is a grammar for a^4a^*. Note that there exists such a σ as M identifies a^4a^*. But then $M(a^4 \diamond a^5 \diamond \sigma) = M(a^4 \diamond a^5 \diamond x \diamond \sigma)$ (as M is iterative and $M(a^4 \diamond a^5 \diamond x) = M(a^4 \diamond a^5)$ due to the conservativeness of M). Thus M is not consistent on $a^4 \diamond a^5 \diamond x \diamond \sigma$ which is an initial segment of a text for $(a + x)^*$.

The theorem follows from above cases. □

5 Learnability and the Unary Alphabet

The previous section leaves open whether $\mathrm{SEC}_{p,1}$, $\mathrm{SEC}_{p,2}$ and $\mathrm{SEC}_{p,3}$ can be iteratively learnt using a class-preserving one-one hypothesis space. While this question will be answered positively for $\mathrm{SEC}_{p,1}$ by Theorem 21, together with Remark 23, below, it remains open for $\mathrm{SEC}_{p,2}$ and $\mathrm{SEC}_{p,3}$. The main purpose of this section is to partially address this gap for the case that the alphabet has size 1 and, hence, the situation is easier to clarify.

Note that the proof of Theorem 16 needs $q \geq 4$ only in Case 1. In the other cases, the languages considered ($a^n\, a^*$, $a^4(a^2 + a^3)^*$, $a(a^3 + a^4)^*$, $a(a^3 + a^4 + a^5)^*$) are all in $\mathrm{SEC}_{1,3}$. Hence, for $\Sigma = \{a\}$, one has that $\mathrm{SEC}_{p,3}$ is also not iteratively learnable using a class-preserving one-one hypothesis space.

Corollary 18. *Suppose that $|\Sigma| = 1$. Then $SEC_{p,3}$ is not iteratively learnable using a class preserving one-one hypothesis space.*

Remark 19. The following question was originally claimed to be solved in the conference version of the current paper, but is still open: Assuming $\Sigma = \{a\}$, is $\mathrm{SEC}_{p,2} = \mathrm{SEC}_{1,2}$ iteratively learnable using class preserving one-one hypothesis space?

Remark 20. Note that, if $|\Sigma| = 1$, then $\mathrm{SEC}_{p,1} = \mathrm{SEC}_{1,1}$ and $\mathrm{SEC}_{p,1}$ has a consistent and conservative iterative learner which uses a class preserving one-one hypothesis space. If the input language is a singleton a^r, then the learner conjectures $\{a^r\}$. Otherwise, the learner conjectures $a^r(a^s)^*$, where a^r is the shortest string seen so far and $s = \gcd(\{i : i \neq 0, a^{r+i}$ is seen in the input so far$\})$. Note that, for nonempty S with $0 \notin S$, $\gcd(\{j\} \cup S) = \gcd(\{j, \gcd(S)\})$. Also, $\gcd(\{j\} \cup \{i + j : i \in S\}) = \gcd(\{j, \gcd(S)\})$. Thus, $s = \gcd(\{i : i \neq 0, a^{r+i}$ is seen in the input so far$\})$, can always be computed using the new datum and the previous hypothesis.

6 Classes with One Context Only

For arbitrary alphabet size, we do not yet know if $\mathrm{SEC}_{p,1}$ can be consistently iteratively learnt using a class preserving one-one hypothesis space. However, we show in this section that one can do so if the consistency requirement is dropped.

For the theorems in this section, for ease of notation, when we say that a language equals $\{z_0, z_1, z_2, z_3, \ldots\}$, we assume that the z_i's are listed in the length non-decreasing order, that is, for all i, $|z_i| \leq |z_{i+1}|$.

In the theorem below, the degree of the polynomial bounding the runtime of M depends linearly on p. The size of the hypothesis (measured in terms of the size of all input data seen so far) is linear. In this section, "language" means "language in $\mathrm{SEC}_{p,1}$."

Theorem 21. *Suppose $p \in \{1, 2, 3, \ldots, *\}$. Then $SEC_{p,1}$ is iteratively learnable in polynomial time using the hypothesis space $\mathcal{G}_{p,1}$.*

Proof. Note that, for any language $L \in \mathrm{SEC}_{p,1}$ and $v, w \in L$ with $v \neq w$, there is a language $L' \in \mathrm{SEC}_{p,1}$ such that v, w are the two shortest words in L' (as, if L has *base* $= (b_1, b_2, \ldots, b_p)$ and context $\{(u_1, u_2, \ldots, u_{2p-1}, u_{2p})\}$, $v = u_1^{k_1} b_1 u_2^{k_1} u_3^{k_1} b_2 u_4^{k_1} \ldots u_{2p-1}^{k_1} b_p u_{2p}^{k_1}$ and $w = u_1^{k_2} b_1 u_2^{k_2} u_3^{k_2} b_2 u_4^{k_2} \ldots u_{2p-1}^{k_2} b_p u_{2p}^{k_2}$, with $k_1 < k_2$, then we could take L' to have *base* $= (u_1^{k_1} b_1 u_2^{k_1}, \ldots, u_{2p-1}^{k_1} b_p u_{2p}^{k_1})$ and context $\{(u_1^{k_2-k_1}, u_2^{k_2-k_1}, \ldots, u_{2p-1}^{k_2-k_1}, u_{2p}^{k_2-k_1})\}$).

The iterative learning algorithm starts with ?, makes no mind change on # and updates on a new datum x according to the first case which applies:

- Case 1: If the current hypothesis is ?, then the new hypothesis is $\{x\}$.
- Case 2: If the current hypothesis is $\{y\}$ for a word $y \neq x$, then the learner conjectures an $H \in \mathrm{SEC}_{p,1}$ such that the two shortest words in H are x, y.
- Case 3: If the current hypothesis is $L = \{y_0, y_1, y_2, \ldots\}$, $|x| < |y_1|$ and $x \neq y_0$, then the learner conjectures an $H \in \mathrm{SEC}_{p,1}$ such that the two shortest words in H are x, y_0.
- Case 4: If the current hypothesis is $L = \{y_0, y_1, y_2, \ldots\}$, $x \notin L$ and within time $|x|^3$ one can "find an $H \in \mathrm{SEC}_{p,1}$ and verify that H is the only such hypothesis containing x, with y_0 and y_1 being the two shortest words in H," then the learner outputs this H as the new hypothesis. Here note that, the number of possible grammars $(base, C) \in \mathcal{G}_{p,1}$ which generate y_0 and y_1 as the two shortest elements are finite and can be effectively found. Furthermore, for each such grammar, *base* $= (b_1, b_2, \ldots, b_p)$ and $C = \{(u_1, u_2, \ldots, u_{2p})\}$, the membership of $x \in Lang(base, C)$ can be tested in time linear in $|x|$. This is so, because $x \in Lang(base, C)$ iff $x = u_1^k b_1 u_2^k u_3^k b_2 u_4^k \ldots u_{2p-1}^k b_p u_{2p}^k$, where $k = (|x| - |y_0|)/(|y_1| - |y_0|)$.
- Case 5: Otherwise the learner repeats its old hypothesis.

This algorithm has the following properties:

- It is an invariant of the construction that the hypothesis is of the form $\{y_0\}$ iff exactly one word, y_0, has shown up so far; otherwise the hypothesis is of the form $\{y_0, y_1, y_2, \ldots\}$ such that y_0 and y_1 are the two shortest words seen so far.
- If the current hypothesis is $L = \{y_0, y_1, y_2, \ldots\}$ with y_0, y_1 being the two shortest words of the language to be learnt and a datum x longer than $(|y_1| + 2)^3$ appears in the input which is not in L, then, by Theorem 22 below, there is a unique language $H \in \mathrm{SEC}_{p,1}$ such that y_0, y_1 are the shortest words of H and $x \in H$; hence the hypothesis is updated to the correct one (unless it is already correct). Note that Theorem 22 below could be formulated for arbitrary alphabets besides the alphabet $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ used in the proof.

– Note that the number of languages in $\mathrm{SEC}_{p,1}$ which have y_0, y_1 as their shortest two elements is bounded by $3^{|y_1|}$ and all such hypotheses can be enumerated in time depending only on $|y_1|$ (this is exponential in $|y_1|$ and thus we need an x whose length is exponential in $|y_1|$ to be able to check in polynomial time whether a unique H generates x with y_0, y_1 being its shortest two elements, even though there exists a polynomial size x which determines the uniqueness of the hypothesis by Theorem 22). This is so, as each character in y_1 could be marked as a member of a constant part in the base word or as the starting character of a part to be included n-fold in the $(n+1)$-st word of the language or a continuing member of such a part. There are $3^{|y_1|}$ ways to mark the characters of y_1 and each language in $\mathrm{SEC}_{p,1}$ containing y_1 as the second smallest word can be represented this way. For each such language, checking whether y_0, y_1 are the shortest two words in the language, and whether x belongs to the language can be done in time linear in $|x|$, for $|x| > |y_1|$. Thus, there exists (and one can indeed find such) unique language in $\mathrm{SEC}_{p,1}$, as needed for the fourth case of the algorithm.

This completes the verification. □

Theorem 22. *Let $V = \{v_0, v_1, v_2, \ldots\}$ and $W = \{w_0, w_1, w_2, \ldots\}$ be languages in $SEC_{*,1}$ over the alphabet $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$. If $v_0 = w_0$, $v_1 = w_1$ and there are m, n with $v_n = w_m \wedge |v_n| > (|v_1| + 2)^3$ then $V = W$.*

Proof. Assume that the precondition of the statement is satisfied. Since $v_0 = w_0$, $v_1 = w_1$, and there is only one context, there are constants r, t such that v_k, w_k have both the length $rk + t$, hence the m, n in the precondition of the theorem have to be the same number. The proof is now based on the following correspondence: for every language $L = \{u_0, u_1, u_2, \ldots\}$ there is a polynomial f_L with all coefficients being rational numbers such that $u_k = f_L(10^k)$ interpreted as a string, where 10^k represents ten to the power of k (not to be confused with sequences of digits as above). To see this, consider the example where

$$L = \{212 \cdot (322)^n \cdot 512 \cdot (1111)^n \cdot 33 : n \in \mathbb{N}\}.$$

Then the following polynomial produces on input 10^k a natural number which has u_k as decimal representation:

$$\begin{aligned} f_L(10^k) = 21200000 \cdot (10^k)^7 + \tfrac{32200000}{999} \cdot ((10^k)^7 - (10^k)^4) \ + \\ 51200 \cdot (10^k)^4 + \tfrac{111100}{9999} \cdot ((10^k)^4 - (10^k)^0) + 33 \cdot (10^k)^0. \end{aligned}$$

It follows that

$$f_L(10^5) = 2123223223223223225121111111111111111111133.$$

In general a k-fold repetition of i digits $e_1 e_2 \ldots e_i$ with $ck + d$ digits after them will contribute

$$e_1 e_2 \ldots e_i \cdot (10^d) \cdot (10^k)^c \cdot \tfrac{(10^k)^i - 1}{10^i - 1}$$

to the overall sum and that a constant part $e_1e_2e_3\dots e_j$ with $c'k+d'$ digits after them will contribute

$$e_1e_2\dots e_j\cdot 10^{d'}\cdot(10^k)^{c'}$$

to the sum of the polynomial f_L.
In general, f_L for the language $\{x_1(y_1)^t x_2(y_2)^t\dots x_r(y_r)^t : t\in\mathbb{N}\}$ can be expressed as follows:

$$\begin{aligned} f_L(10^k) = & \sum_{i\in\{1,2,\dots,r\}} (x_i\cdot(10)^{\sum_{j\in\{i+1,i+2,\dots,r\}}|x_j|+k\sum_{j\in\{i,i+1,\dots,r\}}|y_j|}) \\ & + \sum_{i\in\{1,2,\dots,r\}} (y_i\cdot[(10)^{\sum_{j\in\{i+1,i+2,\dots,r\}}(|x_j|+k|y_j|)}]\cdot\tfrac{(10^{|y_i|})^k-1}{10^{|y_i|}-1}), \end{aligned}$$

where x_i and y_i are interpreted as decimal numbers above. It is then easy to verify that $f_L(10^k)$, read as a string, is the k-th word u_k. Now assume that $v_0=w_0$, $v_1=w_1$ and $v_m=w_m$ with $|v_m|>(|v_1|+2)^3$. Note that m can be computed from v_0,v_1,v_m. The idea is now to check whether the polynomials f_V and f_W are the same on input 10^m. Instead of f_V,f_W one considers the polynomials g_V,g_W where

$$g_L(10^k)=[f_L(10^k)+f_L(10)\cdot\sum_{s\in\{0,1,2,\dots,|f_L(10)|\}}(10^k)^s]\cdot\prod_{s\in\{1,2,\dots,|f_L(10)|\}}(10^s-1).$$

Note that the product $\prod_{s\in\{1,2,\dots,|f_L(10)|\}}(10^s-1)$ is used to get rid of the denominator and the sum $f_L(10)\cdot\sum_{s\in\{0,1,2,\dots,|f_L(10)|\}}(10^k)^s$ is used to get rid of potential negative coefficients of some of the powers of 10^k in $f_L(10^k)$.

Now, $f_V(10)$ and $f_W(10)$ are just the (equal) words v_1 and w_1. The resulting polynomials g_V and g_W have coefficients in the natural numbers and each coefficient is at most $2\cdot10^{|v_1|}\cdot\prod_{s=1,2,\dots,|f_L(10)|}(10^s-1)$. So the coefficients are values between 0 and $10^{(|v_1|+2)^2}$. Hence, for $k>(|v_1|+2)^2$, the coefficient of the h-th power of the variable of the polynomial coincides with the decimal digits $e_{hk+(k-1)}e_{hk+(k-2)}\dots e_{hk+2}e_{hk+1}e_{hk}$ interpreted as a decimal number from the overall decimal representation $e_\ell e_{\ell-1}e_{\ell-2}\dots e_2e_1e_0$ of the value of $g_L(10^k)$. If now $v_k=w_k$ then the coefficients of g_V and g_W coincide, hence $g_V=g_W$ and $f_V=f_W$ and $V=W$. □

At this point we do not know if the n chosen above could be made smaller. In particular, we do not know whether there exist distinct $V,W\in\mathrm{SEC}_{*,1}$ such that $v_0=w_0, v_1=w_1$ and $v_n=w_n$, for some $n\geq 2$.

Remark 23. One can use the polynomial f_L in normal form instead of a grammar for representing L — this leads to one hypothesis per language which could be found from y_0,y_1 and y_n for any $n>(|y_1|+2)^2$ in polynomial time; note that one can check whether a polynomial generates a language of the desired form and therefore one can use this class-preserving one-one hypothesis space.

Remark 24. Let ω be the first transfinite ordinal. The ordinal mind change complexity [15] of the algorithm in Theorem 21 is $\omega+1$: The learner starts with the ordinal counter $\omega+1$, changes it to ω, when it makes a conjecture of the form $\{z\}$ and then to $2 \cdot (|y_0| + |y_1|)$, when an hypothesis of the form $\{y_0, y_1, y_2, \ldots\}$ is first output. Now whenever a new word x with $|x| < |y_1|$ arrives then the sum of the two shortest words goes down by 1; furthermore, in between each such mind change, there can be at most one mind change due to some sufficiently large x being seen in the input, which causes an update to the unique hypothesis which contains x and has the two shortest words identical to the previous hypothesis. This gives us the required bound.

7 A Special Case

In this section, a subclass of $\mathrm{SEC}_{p,q}$ consisting of regular sets only will be considered which is defined as follows: $\mathrm{R}_{p,q} = \{x_1 \cdot Y_1^* \cdot x_2 \cdot Y_2^* \cdot \ldots \cdot x_p \cdot Y_p^* : x_1, x_2, \ldots, x_p \in \Sigma^*, Y_1, Y_2, \ldots, Y_p \subseteq \Sigma^*, |Y_1| + |Y_2| + \ldots + |Y_p| \leq q\}$.

This subcase is obtained by permitting only right contexts and by requiring that the strings in every context are different from ϵ at only one place.

Remark 25. Note that $\mathrm{R}_{q+1,q} = \mathrm{R}_{*,q}$ and for unary alphabet $\mathrm{SEC}_{p,q} = \mathrm{R}_{1,q}$.

Remark 26. As one can construct, for each language in $\mathrm{R}_{p,q}$, a finite automaton accepting it, the inclusion-problem for these languages is decidable. Thus, by Corollary 11,

- $\mathrm{R}_{p,q}$ is conservatively learnable and
- there is a recursive function which computes for every $L \in \mathrm{R}_{p,q}$ (given in adequate form) a finite subset $D(L)$ of L such that, for all $H \in \mathrm{R}_{p,q}$, $D(L) \subseteq H \Rightarrow L \subseteq H$.

Remark 27. For $q \in \{6, 7, 8, \ldots, *\}$, $p \in \{1, 2, 3, \ldots\}$, one cannot iteratively learn $\mathrm{R}_{p,q}$ using a one-one class-preserving hypothesis space. This holds as the learner on input a^4, a^5 either produces an hypothesis $L \subseteq a^*$, which allows us to do the diagonalization as shown in Theorem 16 or it produces an hypothesis L which contains some $x \notin a^*$. In the latter case, the learner cannot distinguish the language $\{a, x\}^*$ from the language $\{a, xx, xxx, ax, xa, xax\}^* = \{a, x\}^* - \{x\}$.

If $(p, q) \in \{(2, 4), (3, 4), (4, 4), (5, 4), (2, 5), (3, 5), (4, 5), (5, 5), (6, 5)\}$, then a similar proof can be made. For the proof, one would present texts of subsets of a^* to the learner as in the proof of Theorem 16, but, when facing a conjecture containing some $x \notin a^*$, one presents either x or $\#$ to the learner which does not make a mind change when receiving this datum. After that, the learner receives all data for the language $\{a, ax\}^* \cdot \{xx, xxx\}^*$. Therefore the learner converges on some texts of $\{a, ax\}^* \cdot x^*$ and $\{a, ax\}^* \cdot \{xx, xxx\}^*$, respectively, to the same hypothesis although one of these languages contains x and the other one not. Both languages are in $\mathrm{R}_{p,q}$ but at least one of them is not learnt.

Note that it is not possible to use any of the two proofs above for the cases of $\mathrm{R}_{1,4}$ nor in $\mathrm{R}_{1,5}$ as both proofs utilize sets which are not in these classes. Furthermore, it is open at present whether $\mathrm{R}_{p,3}$, $\mathrm{R}_{1,2}, \mathrm{R}_{2,2}$ are iteratively learnable using class preserving one-one hypothesis space. $\mathrm{R}_{p,1}$ is iteratively learnable using class preserving one-one hypothesis space, as can be shown using essentially the same proof idea as used for $\mathrm{SEC}_{p,1}$ (see Theorem 21 and and Remark 23). We consider learnability of $\mathrm{R}_{3,2}$ below.

If $\Sigma = \{a\}$, then we do not yet know if one can iteratively learn $\mathrm{R}_{p,2}$ using a class-preserving one-one hypothesis space (as for $\Sigma = \{a\}$, $\mathrm{R}_{p,2}$ is same as the class $\mathrm{SEC}_{1,2}$).

Our next result shows that one can iteratively learn $\mathrm{R}_{p,2}$, for $p \geq 3$, in the case that $|\Sigma| \geq 3$, cf. Remark 25. In the following, let $S_{c,a,b,n} = c^n a^n b^n c\{a,b\}^* c b^n a^n c^n$.

Proposition 28. *Suppose $L \in R_{3,2}$ and u, v are two distinct strings in L.*
(a) *Suppose $u = au'$ and $v = bv'$, where $a \neq b$. Let $c \notin \{a,b\}$ and $n > \max\{|u|,|v|\}$. Then, $S_{c,a,b,n} \cap L = \emptyset$.*
(b) *Suppose $u = u'a$ and $v = v'b$, where $a \neq b$. Let $c \notin \{a,b\}$ and $n > \max\{|u|,|v|\}$. Then, $S_{c,a,b,n} \cap L = \emptyset$.*
(c) *Suppose both u, v do not start or end with a c. Let a, b be two distinct characters different from c, and $n > \max\{|u|,|v|\}$. Then, $S_{c,a,b,n} \cap L = \emptyset$ or for some x', y' with $|y'| > |x'|$, $S_{c,a,b,n} \cap L = \{y'\}$, x' does not start or end with a c and ($L = \{x', y'\}^*$ or $L = (x')^*(y')^*$ or $L = (y')^*(x')^*$).*

Proof. Suppose $L = \alpha x^* \gamma y^* \beta$ or $L = \alpha\{x,y\}^*\beta$, where x, y are suitable contexts and α, γ, β are constant strings. Note that any $L \in \mathrm{R}_{3,2}$ which contains at least 2 strings can be expressed as such. Without loss of generality, we assume that $x \neq \epsilon$, and if $y = \epsilon$, then so is γ.

(a) Let u, v, a, b, c, n be as in the hypothesis. Clearly we have that $\alpha = \epsilon$.

- Case 1: $\gamma \neq \epsilon$. Note that the strings u, v must start with either x or γ. Thus, one of x and γ starts with an a and the other with a b. Thus all strings of L start with either a or b. On the other hand all strings in $S_{c,a,b,n}$ start with a c. Thus, $S_{c,a,b,n} \cap L = \emptyset$.
- Case 2: $\gamma = \epsilon$. If $\beta \notin c^*$, then clearly no string in L belongs to $S_{c,a,b,n}$ because all strings in L end in β, $n > \max\{|u|,|v|\} > |\beta|$ and β is not a suffix of c^n.
  On the other hand if $\beta \in c^*$, then one of u, v starts with an x and the other with a y. Thus, one of x, y starts with an a and the other with a b. Thus, all strings in L, except β, start with either an a or a b. Furthermore, $\beta \notin S_{c,a,b,n}$ as $|\beta| \leq \max\{|u|,|v|\} < n$. It follows that $L \cap S_{c,a,b,n} = \emptyset$.

(b) This part can be proven similarly to part (a).

(c) Let u, v, a, b, c, n be as in the hypothesis. If $\alpha \neq \epsilon$ then all strings in L do not start with a c, whereas all strings in $S_{c,a,b,n}$ start with a c. Thus $S_{c,a,b,n} \cap L = \emptyset$. Similarly if $\beta \neq \epsilon$ then all strings in L do not end with a c; whereas, all strings in $S_{c,a,b,n}$ end with a c. Thus $S_{c,a,b,n} \cap L = \emptyset$. Thus, for the following assume that $\alpha = \beta = \epsilon$.

- Case 1: $\gamma \neq \epsilon$. Suppose at least one of u, v starts with an x and at least one of u, v ends with a y. Then, x does not start with a c and y does not end with a c. It follows that all strings in L except γ do not belong to $S_{c,a,b,n}$. As $|\gamma| < \max\{|u|,|v|\} \leq n$, we also have that $\gamma \notin S_{c,a,b,n}$. Thus, $L \cap S_{c,a,b,n} = \emptyset$.
  Suppose both u, v do not end with an y. Then, we have that x does not start with a c (as at least one of u, v starts with an x) and γ does not end with a c (as both u, v end with γ). Thus, strings in $L \cap S_{c,a,b,n}$ could only be of the form γy^+. But γ does not end with a c and $|\gamma| < n$. Thus, γ is not a prefix of c^n and thus not a prefix of any string in $S_{c,a,b,n}$. It follows

that $L \cap S_{c,a,b,n} = \emptyset$. One can similarly argue for the case that both u, v do not start with an x.

– Case 2: $\gamma = \epsilon$. In this case, $L = x^* y^*$ or $L = \{x, y\}^*$.
  Suppose at least one of u, v starts with an x or ends with an x, and at least one of u, v starts with a y or ends with a y. Then, all strings starting with x or y contain a character different from c, and both x and y are of length at most n. Thus, $L \cap S_{c,a,b,n} = \emptyset$.
  Now suppose u and v both start and end with an x (case of y is similar). Thus, $|x| < n$ and x does not start or end with a c. It follows that $L \cap S_{c,a,b,n}$ can only consist of strings which start and end with a y.
– Case 2a: $y \in c^*$. In this case, as $|x| < n$, we have that $a^n b^n$ is not a substring of any string in $\{x, y\}^*$ and thus $L \cap S_{c,a,b,n} = \emptyset$.
– Case 2b: $y \notin c^*$. If y does not start or end with c^n, then we immediately have that $L \cap S_{c,a,b,n} = \emptyset$. So assume that y starts and ends with c^n. Thus, for any string which starts and ends with a y to be a member $S_{c,a,b,n}$, we must have that y starts with $c^n a^n b^n$ and ends with $b^n a^n c^n$. It follows that y can be the only string in $L \cap S_{c,a,b,n}$. Note that $|y| > |x|$ follows as $|x| < n$.

This case-distinction completes the proof. ☐

Let S_L be the union of all $S_{c,a,b,n}$, where a, b, c are distinct and there exist distinct $u, v \in L$ such that $n > \max\{|u|, |v|\}$, and (i) u and v start with a and b respectively or (ii) u and v end with a and b respectively or (iii) u, v both do not start or end with a c.

Note that $S_L \cap L$ is either $\emptyset$ or $\{y\}$, where the second case applies only if, for some x not starting or ending with a c, ($L = \{x, y\}^*$ or $L = x^* y^*$ or $L = y^* x^*$) and $y = c^m a^m b^m czcb^m a^m c^m$ with $z \in \{a, b\}^*$, $m > |x|$ and a, b, c being distinct.

Proposition 29. *Suppose $L, L' \in R_{3,2}$, ($L = \{x, y\}^*$ or $L = x^* y^*$ or $L = y^* x^*$) and $L - S_L \subseteq L'$. Then, $L \subseteq L'$.*

Proof. Without loss of generality assume $|y| \geq |x|$. If $S_L \cap L = \emptyset$, then the proposition is trivial. So suppose $S_L \cap L \neq \emptyset$. Then, by Proposition 28, we have that $y = c^m a^m b^m czcb^m a^m c^m$, for some $m > |x|$, $z \in \{a, b\}^*$, a, b, c being distinct and x not starting or ending with a c.

As $\epsilon \in L - S_L$, we have that L' is of the form $(x')^*(y')^*$ or $\{x', y'\}^*$ for some x', y'. As x belongs to $L - S_L$ and thus L', we have that the context used in the leftmost position in the generation of x in L' cannot start with a c and the context used in the rightmost position in the generation of x in L' cannot end with a c. If these contexts in the leftmost and rightmost positions are different, then yy cannot be generated in L' as both x' and y' are of length at most $|x|$, and neither would be a prefix of c^m and thus of yy, which belongs to $L - S_L$. Thus, the same context is used to start and end the generation of x in L'. Let this context be x'. Note that the generation of yy in L' must start (end) with a context which starts (ends) with a c. Thus, y' must start and end with a c and y' is a prefix as well as suffix of yy. If $y' \in c^*$ then we further have that x', y' cannot generate yy as $a^m b^m$ is not a substring of $(x')^*$ for any x' with $|x'| \leq |x| < m$. Thus, we have that $y' \notin c^*$. As generation of yy in L' must start and end with y', we immediately have that $c^m a$ is a prefix of y' and ac^m is a suffix of y'. As generation of yy

in L' starts and ends with y', and $c^m a$ only appears at the beginning of y and ac^m only appears at the end of y, it follows that either $y' = y$ or $y' = yy$. If $y' = yy$, then $yyy \in L - S_L$ cannot be generated by L'. Thus, y' must be y. It follows that $y \in L'$ and thus $L \subseteq L'$. □

Corollary 30. *There exists a finite subset Z of $L - S_L$ such that, for all $L' \in R_{3,2}$, $Z \subseteq L'$ implies $L \subseteq L'$.*

Proof. Consider the construction of $D(G)$ as in the proof of Corollary 11 for a grammar G for L, where $\mathcal{G}$ is taken to be $\mathrm{R}_{3,2}$. However, instead of using a text T for L, one uses a text T for $L - S_L$. Again, it can be argued that (i) $\lim_{n\to\infty} X_{\mathrm{ctnt}(T[n])}$ converges, (ii) every grammar G' in $\lim_{n\to\infty} X_{\mathrm{ctnt}(T[n])}$ satisfies $L - S_L \subseteq Lang(G')$. Thus, $L \subseteq Lang(G')$ by Proposition 28 and Proposition 29. It follows that $D(G)$ constructed as such will satisfy the requirements of Z as needed. □

Remark 31. For any $m \in \mathbb{N}$, $c^m a^m b^m c\{a, b\}^* cb^m a^m c^m$ is not a proper subset of any language in $\mathrm{R}_{3,2}$.

Proof. Suppose $L \in \mathrm{R}_{3,2}$ contains $c^m a^m b^m c\{a, b\}^* cb^m a^m c^m$. Then, one of the contexts in L must be a as otherwise L cannot contain $c^m a^m b^m c\{a\}^* cb^m a^m c^m$. Similarly, the second context in L must be b, as otherwise L cannot contain $c^m a^m b^m c\{b\}^* cb^m a^m c^m$. Furthermore, as $c^m a^m b^m c\{ab\}^* cb^m a^m c^m$ is contained in L, we have that L is of the form $\alpha\{a, b\}^*\beta$. It follows that $\alpha\beta = c^m a^m b^m ccb^m a^m c^m$. It remains to look at the cases where $c^m a^m b^m cc$ is a prefix of α or $ccb^m a^m c^m$ is a suffix of β. In these cases, L does not contain $c^m a^m b^m cacb^m a^m c^m$. It follows that $\alpha = c^m a^m b^m c$ and $\beta = cb^m a^m c^m$. □

Next we will prove our result on the learnability of $\mathrm{R}_{3,2}$ when the alphabet size is at least 3.

Intuitively, in the learning algorithm below, the learner tries to output the "minimal" hypothesis in $\mathrm{R}_{3,2}$ which contains the input data seen so far. If this is not possible, then the learner tries to remember the data seen (and implied) by using a coded output language which is

$$c^m a^m b^m c\{a, b\}^* cb^m a^m c^m$$

for some appropriate m, where we take the coding language such that the learner can recognize from relevant data that it is a coding language and do the updates accordingly. It is not always possible to do the coding perfectly and the learner may lose some data due to iterativeness. However, the learner misses out at most one element, as given by Proposition 28 (also see Claim 33 below). This missing element is not critical as shown by Proposition 29 and Corollary 30 above, hence the coding can be done without losing too much information.

We now proceed formally and give the algorithm outlined. Please recall Remark 26 for the definition of $D(H)$ used below.

Theorem 32. *If $|\Sigma| \geq 3$, then $R_{3,2}$ can be iteratively learnt using a class preserving one-one hypothesis space.*

Proof. One can use the following iterative algorithm for learning $R_{3,2}$ employing a class preserving one-one hypothesis space. On input # do not change the hypothesis. If the input is a string w, then proceed according to which of the following cases is applicable.

1. If the previous hypothesis was ?, then output a grammar for $\{w\}$.
2. If the previous hypothesis contains w, then repeat the hypothesis.
3. If the previous hypothesis H does not contain w, then let

$$Q = \begin{cases} D(H) \cup \{w\}, & \text{if } H \text{ is not of the form } c^m a^m b^m c\{a,b\}^* c b^m a^m c^m \\ & \text{for any } m > 0; \\ Q_H \cup \{w\}, & \text{if } H \text{ is of the form } c^m a^m b^m c\{a,b\}^* c b^m a^m c^m \\ & \text{for some } m > 0, \text{ where } Q_H \text{ is the} \\ & \text{finite set coded by } k' = \max\{k'' : 2^{k''} \text{ divides } m\}. \end{cases}$$

and output the hypothesis

$$H = \begin{cases} H', & \text{if there exists a (necessarily unique)} \\ & H' \in R_{3,2} \text{ such that } Q \subseteq H' \text{ and} \\ & Q \subseteq L' \Leftrightarrow H' \subseteq L' \text{ for all } L' \in R_{3,2}; \\ c^n a^n b^n c\{a,b\}^* c b^n a^n c^n, & \text{where } n = 2^k(2m+1), \\ & \text{for } k \text{ being the code for the finite set} \\ & Q,\ m \text{ being the length of the second} \\ & \text{smallest string in } Q \text{ and pairwise} \\ & \text{distinct } a,b,c \in \Sigma \text{ being chosen to} \\ & \text{satisfy at least one of (i) to (iii):} \\ & \text{Let } u,v \text{ be two shortest strings in } Q; \\ & \text{(i) } u = au', v = bv'; \\ & \text{(ii) } u = u'a, v = v'b; \\ & \text{(iii) } u \text{ and } v \text{ do not start or end with a } c. \end{cases}$$

Note that one can always choose a, b, c as needed above for any two strings u, v.

Note that we used $n = 2^k(2m+1)$, rather than k itself so that the requirements of Proposition 28 are satisfied.

Intuitively, Q above recovers (approximately) the data seen in the past. The data lost due to the iterativeness of the algorithm is not critical as our proof shows.

Now we show that the above algorithm would learn the class $R_{3,2}$ iteratively. For a finite set Q, let $X(Q)$ denote $\bigcap_{L' \in R_{3,2}: Q \subseteq L'} L'$. That is $X(Q)$ is the intersection of the languages in $R_{3,2}$ which contain Q.

Suppose the input text T is for $L \in R_{3,2}$. If L contains only one element, then clearly the learner iteratively learns L. So assume that L contains at least two (and thus infinitely many) elements. Without loss of generality assume that T does not contain # (as the above algorithm ignores #).

Let H_n denote the hypothesis of the learner above after seeing input $T[n]$ (that is just after it sees $T(n-1)$, where $H_0 = ?$). Define Q_n as follows. $Q_0 = \emptyset$, $Q_1 = \{T(0)\}$. For $n \geq 2$, if step 2 was executed after seeing input $T(n)$, then let $Q_{n+1} = Q_n$. Otherwise, let Q_{n+1} be Q as defined in step 3. Note that if the hypothesis H_t is produced via step 1 or via the first clause in definition of H in step 3, then $H_t = X(Q_t)$. In particular, this is true if H_t is not of the form $c^m a^m b^m c\{a,b\}^* c b^m a^m c^m$, for any $m > 0$.

Claim 33. *For all n,*
(a) $Q_n \subseteq L$.
(b) $\text{ctnt}(T[n]) - X(Q_n) \subseteq S_L$.

We can show the claim by induction on n. For $n = 0, 1$, clearly, (a), (b) hold. Suppose (a) and (b) hold for $n = t$. Then we show it for $n = t + 1$.

Suppose step 2 is executed on input $T(t)$. Thus $Q_{t+1} = Q_t$ and part (a) follows using induction hypothesis. Note that by Remark 31, $H_t \subseteq L$ iff H_t was produced via first clause in the definition of H in step 3 (on some input $T[t']$ with $t' \leq t$, where step 2 was used on inputs $T(t'), T(t'+1), \ldots, T(t)$; here note that if H_t is produced via second clause in the definition of H in step 3, then $L \not\subseteq H_t$.) Thus, if $H_t \subseteq L$, then $H_t = X(Q_t)$; on the other hand, if $H_t \not\subseteq L$, then $H_t \subseteq S_L$. Thus, part (b) follows by induction and $T(t) \in H_t$.

Suppose step 3 is executed on input $T(t)$. Then, Q_{t+1} is Q as defined in step 3.

If Q in step 3 is defined via first clause then $H_t = X(Q_t)$ and $Q_{t+1} = D(H_t) \cup \{T(t)\}$. Thus, $D(H_t) \subseteq H_t = X(Q_t) \subseteq L$ (as $Q_t \subseteq L$, by induction). Thus, $Q_{t+1} \subseteq L$ and (a) follows. Moreover,

$$\begin{aligned}
\text{ctnt}(T[t+1]) &= \text{ctnt}(T[t]) \cup \{T(t)\} \\
&\subseteq S_L \cup X(Q_t) \cup \{T(t)\} \text{ (by induction)} \\
&= S_L \cup H_t \cup \{T(t)\} \ \ \text{(as } H_t = X(Q_t)) \\
&= S_L \cup X(D(H_t)) \cup \{T(t)\} \ \ \text{(by definition of } D(H_t)) \\
&\subseteq S_L \cup X(D(H_t) \cup \{T(t)\}) \\
&= S_L \cup X(Q_{t+1}).
\end{aligned}$$

Thus (b) follows.

If Q in step 3 is defined via second clause, then $Q_{t+1} = Q_t \cup \{T(t)\} \subseteq L$ by induction and thus part (a) follows. Furthermore,

$$\begin{aligned}
\text{ctnt}(T[t+1]) &= \text{ctnt}(T[t]) \cup \{T(t)\} \\
&\subseteq S_L \cup X(Q_t) \cup \{T(t)\} \text{ (by induction)} \\
&\subseteq S_L \cup X(Q_{t+1}).
\end{aligned}$$

This proves the claim.

Now let $Z \subseteq L - S_L$ be as in Corollary 30. Let t be large enough so that $\text{ctnt}(T[t]) - S_L \supseteq Z$. Then it follows from Claim 33(a) that $Q_t \subseteq L$ and from Claim 33(b) that $X(Q_t) \supseteq Z$. Hence $X(Q_t) = L$. It follows that $H_t = L$. Clearly, the learner never changes its mind once it outputs the correct hypothesis. Thus the learner iteratively learns L. □

8 Conclusion

In this paper we considered the iterative learnability of simple external contextual languages, called $\mathrm{SEC}_{p,q}$, where q denotes the number of contexts and p denotes the number of parts in the base. We showed that $\mathrm{SEC}_{p,q}$ is consistently and iteratively learnable, by a polynomial time algorithm, for each value of the parameters p and q, as long as a suitable class preserving hypothesis space is allowed. On the other hand, for $q \geq 2$, $\mathrm{SEC}_{p,q}$ is not consistently and conservatively iteratively learnable using a class preserving hypothesis space.

The positive result above needed some padding. It was shown that if one considers only one-one class preserving hypothesis spaces then $\mathrm{SEC}_{p,q}$ is not iteratively learnable using a one-one hypothesis space if $q \geq 4$, though $\mathrm{SEC}_{p,1}$ is polynomial time iteratively learnable using a one-one hypothesis space. The problem of iteratively learning $\mathrm{SEC}_{p,q}$ using a one-one hypothesis space, for $q = 2$ or $q = 3$, is open at present; we only know that for unary alphabet, $\mathrm{SEC}_{p,3}$ is not iteratively learnable using a class preserving one-one hypothesis space. We also considered a special case of allowing only right contexts, and showed that $\mathrm{R}_{3,2}$ (the restricted version of $\mathrm{SEC}_{3,2}$) is iteratively learnable using a one-one hypothesis space.

Acknowledgements. We thank Samuel E. Moelius III and the anonymous referees of TCS and ALT 2008 for several useful comments.

References

1. Dana Angluin. Inductive inference of formal languages from positive data. *Information and Control*, 45:117–135, 1980.
2. Leonor Becerra-Bonache. *On the Learnability of Mildly Context-Sensitive Languages Using Positive Data and Correction Queries*. PhD thesis, Rovira i Virgili University, 2006.
3. Leonor Becerra-Bonache and Takashi Yokomori. Learning mild context-sensitiveness: toward understanding children's language learning. *Grammatical Inference: Algorithms and Applications*, Seventh International Colloquium, ICGI 2004. Springer LNCS, 3264:53-64, 2004.
4. Janis Bārzdiņš. Two theorems on the limiting synthesis of functions. *Theory of Algorithms and Programs*, Volume 1, Latvian State University, Riga, 210:82–88, 1974.
5. Lenore Blum and Manuel Blum. Toward a mathematical theory of inductive inference. *Information and Control*, 28:125–155, 1975.
6. Joan Bresnan, Ronald M. Kaplan, Stanley Peters and Annie Zaenen. Cross-serial dependencies in Dutch. Linguistic Inquiry 13:613-635, 1982.
7. Lorenzo Carlucci, John Case, Sanjay Jain and Frank Stephan. Results on memory-limited U-shaped learning. *Information and Computation*, 205:1551–1573, 2007.
8. John Case, Sanjay Jain, Steffen Lange, Thomas Zeugmann. Incremental concept learning for bounded data mining. *Information and Computation*, 152:74–110, 1999.
9. John Case and Christopher Lynes. Inductive inference and language identification. *Ninth International Colloquium on Automata, Languages and Programming*, ICALP 1982. Springer LNCS, 140:107–115, 1982.

10. John Case and Samuel E. Moelius. Parallelism increases iterative learning power. *Theoretical Computer Science*, 410:1863–1875, 2009.
11. Noam Chomsky. Three models for the description of language. *IRE Transactions on Information Theory*, 3:113–124, 1956.
12. Christopher Culy. The complexity of the vocabulary of Bambara. *Linguistics and Philosophy* 8:345-351, 1985.
13. Henning Fernau and Markus Holzer. External contextual and conditional languages. *Recent Topics in Mathematical and Computational Linguistics, Papers in Honor of Solomon Marcus on the Occasion of his 75th Birthday.* Editura Academiei Româe, Bucuresti, pages 104–120, 2000.
14. Rūsiņš Freivalds, Efim Kinber and Carl Smith. On the intrinsic complexity of learning. *Information and Computation*, 123:64–71, 1995.
15. Rūsiņš Freivalds and Carl Smith: On the role of procrastination for machine learning, *Information and Computation*, 107:237–271, 1993.
16. Mark Fulk. Prudence and other conditions on formal language learning. *Information and Computation*, 85:1–11, 1990.
17. E. Mark Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.
18. Sanjay Jain, Daniel Osherson, James Royer and Arun Sharma. *Systems that Learn: An Introduction to Learning Theory.* MIT Press, Cambridge, Massachusetts, second edition, 1999.
19. Aravind K. Joshi. Tree adjoining grammars: How much context-sensitivity is required to provide reasonable structural descriptions? *Natural Language Parsing*, pages 206–250, Cambridge University Press, 1985.
20. Aravind K. Joshi, Yves Schabes. Tree-adjoining grammars. *Handbook of Formal Languages*, volume 3, pages 69–123. Springer, 1997.
21. Manfred Kudlek, Carlos Martín-Vide, Alexandru Mateescu and Victor Mitrana. Contexts and the concept of mild context-sensitivity. *Linguistics and Philosophy*, 26:703–725, 2003.
22. Steffen Lange and Rolf Wiehagen. Polynomial time inference of arbitrary pattern languages. *New Generation Computing*, 8:361–370, 1991.
23. Steffen Lange and Thomas Zeugmann. Language learning in dependence on the space of hypotheses. *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, COLT 1993. ACM Press, 127–136, 1993.
24. Steffen Lange and Thomas Zeugmann. Incremental learning from positive data. *Journal of Computer and System Sciences*, 53:88–103, 1996.
25. Alexis Manaster-Ramer. Some uses and abuses of mathematics in linguistics. *Issues in Mathematical Linguistics*, pages 73–130. John Benjamins, Amsterdam, 1999.
26. Solomon Marcus. Contextual grammars. *Revue Roumaine des Mathématiques Pures et Appliquées*, 14:1525–1534, 1969.
27. Solomon Marcus, Gheorghe Paun and Carlos Martín-Vide. Contextual grammars as generative models of natural languages. *Computational Linguistics*, 24(2):245–274, 1998.

28. Tim Oates, Tom Armstrong, Leonor Becerra-Bonache and Mike Atamas. Inferring grammars for mildly context-sensitive languages in polynomial-time. *Grammatical Inference: Algorithms and Applications*, Eighth International Colloquium, ICGI 2006. Springer LNAI, 4201:137–147, 2006.
29. Piergiorgio Odifreddi. *Classical Recursion Theory.* North-Holland, Amsterdam, 1989.
30. Daniel N. Osherson, Michael Stob and Scott Weinstein. *Systems That Learn, An Introduction to Learning Theory for Cognitive and Computer Scientists.* Bradford — The MIT Press, Cambridge, Massachusetts, 1986.
31. Daniel N. Osherson and Scott Weinstein. Criteria of language learning. *Information and Control*, 52:123–138, 1982.
32. Rohit J. Parikh. On context-free languages. *Journal of the ACM*, 13:570–581, 1966.
33. Gheorghe Paun. *Marcus Contextual Grammar.* Kluwer Academic Publishers, Netherlands, 1997.
34. Lenny Pitt. Inductive inference, DFAs, and computational complexity. *Analogical and Inductive Inference*, Second International Workshop, AII 1989. Springer LNAI 397:18–44, 1989.
35. Kelly Roach. Formal properties of head grammars. In A. Manaster-Ramer, editor, *Mathematics of Language*, pages 293–348. John Benjamins, Amsterdam, 1987.
36. Hartley Rogers. *Theory of Recursive Functions and Effective Computability.* McGraw-Hill, New York, 1967.
37. Stuart M. Shieber. Evidence against the context-freeness of natural language. *Linguistics and Philosophy*, 8:333–343, 1985.
38. Takeshi Shinohara. Rich classes inferable from positive data: Length-bounded elementary formal systems. *Information and Computation*, 108:175–186, 1994.
39. Mark Steedman. Dependency and coordination in the grammar of Dutch and English. *Language*, 61:523–568, 1985.
40. Kenneth Wexler and Peter W. Culicover. *Formal Principles of Language Acquisition.* Cambridge, Massachusetts, The MIT Press, 1980.
41. Rolf Wiehagen. Limes-Erkennung rekursiver Funktionen durch spezielle Strategien. *Elektronische Informationsverarbeitung und Kybernetik*, 12:93–99, 1976.
42. Ryo Yoshinaka. Learning efficiency of very simple grammars from positive data. *Theoretical Computer Science*, 410:1807–1825, 2009.