Learning Correction Grammars

Lorenzo Carlucci[1], John Case[2], and Sanjay Jain *[3]

[1] Università di Roma "La Sapienza," Department of Computer Science, Via Salaria 112, 00198 Roma, carlucci@di.uniroma1.it; and Scuola Normale Superiore di Pisa, Classe di Lettere, Piazza dei Cavalieri 7, 56126 Pisa.
[2] Department of Computer and Information Sciences, University of Delaware, Newark, DE 19716-2586, U.S.A.
case@cis.udel.edu
[3] Department of Computer Science,
National University of Singapore, Singapore 117417, Republic of Singapore.
sanjay@comp.nus.edu.sg

Abstract. We investigate a new paradigm in the context of learning in the limit, namely, learning *correction grammars* for classes of *computably enumerable* (*c.e.*) languages. Knowing a language may feature a representation of it in terms of *two* grammars. The second grammar is used to make corrections to the first grammar. Such a pair of grammars can be seen as a single description of (or grammar for) the language. We call such grammars *correction grammars.* Correction grammars capture the observable fact that people *do* correct their linguistic utterances during their usual linguistic activities.
We show that learning correction grammars for classes of c.e. languages in the **TxtEx**-model (i.e., converging to a single correct correction grammar in the limit) is sometimes more powerful than learning ordinary grammars even in the **TxtBc**-model (where the learner is allowed to converge to infinitely many syntactically distinct but correct conjectures in the limit). For each $n \geq 0$, there is a similar learning advantage, again in learning correction grammars for classes of c.e. languages, but where we compare learning correction grammars that make $n+1$ corrections to those that make n corrections.
The concept of a correction grammar can be extended into the constructive transfinite, using the idea of counting-down from notations for transfinite constructive ordinals. This transfinite extension can also be conceptualized as being about learning Ershov-descriptions for c.e. languages. For u a notation in Kleene's general system $(O, <_o)$ of ordinal notations for constructive ordinals, we introduce the concept of an u-correction grammar, where u is used to bound the number of corrections that the grammar is allowed to make. We prove a general hierarchy result: if u and v are notations for constructive ordinals such that $u <_o v$, then there are classes of c.e. languages that can be **TxtEx**-learned by conjecturing v-correction grammars but not by conjecturing u-correction grammars.

* Lorenzo Carlucci and John Case were supported in part by NSF Grant number CCR-0208616 at the University of Delaware. Lorenzo Carlucci was also supported in part by grant number 13393 of the John Templeton Foundation, and by a Telecom Italia "Progetto Italia" Fellowship at the Scuola Normale Superiore di Pisa. Sanjay Jain was supported in part by NUS grant numbers R252-000-308-112 and R252-000-212-112.

Surprisingly, we show that — above "ω-many" corrections — it is not possible to strengthen the hierarchy: **TxtEx**-learning u-correction grammars of classes of c.e. languages, where u is a notation in O for *any* ordinal, can be simulated by **TxtBc**-learning w-correction grammars, where w is any notation for the smallest infinite ordinal ω.

1 Introduction and Motivation

We investigate a new model in the context of Gold-style learning theory (see [24, 26]): learning "correction grammars". Burgin [9] suggested that knowing a language may feature a representation of the language in terms of *two* sets of rules, i.e., two grammars, say g_1 and g_2: g_2 is used to "edit" errors of (make corrections to) g_1. In set-theoretic terms, the language L is represented as the difference $(L_1 - L_2)$, where L_i is the language generated by the grammar g_i. The pair $\langle g_1, g_2\rangle$ can thus be seen as a single description of (or "grammar" for) the language L. Burgin called these grammars *grammars with prohibition.* We prefer to call them *correction grammars.* The idea is readily formalizable in a recursion-theoretic context: p is a *correction grammar* for L if and only if $p = \langle i, j\rangle$ and $L = (W_i - W_j)$.[4] In general, one cannot effectively convert correction grammars $\langle i, j\rangle$ for a language L to a standard grammar for L, even when L is computable (i.e., computably decidable) (this can be proved using standard techniques and also follows from proof of Theorem 17).

It is natural to consider extensions of the correction grammars paradigm. The concept is readily generalizable to descriptions of c.e. languages as finite differences of c.e. languages. This idea formalizes the concept of a finite, fixed number of successive editings for errors. For $k > 0$, a *k-correction grammar* for a c.e. language L is a number p such that $p = \langle i_1, \ldots, i_k\rangle$ and $L = W_{i_1} - (W_{i_2} - \cdots (W_{i_{k-1}} - W_{i_k}) \cdots)$.

Correction grammars — besides being mathematically interesting — can be seen as capturing the observable fact that people *do* correct their linguistic utterances during their usual linguistic activities (both oral and written). As will be seen later, the formal concept of a *general* correction grammar can be equivalently expressed in terms of algorithms that initially exclude all candidate items but are allowed, for each candidate item, a finite number of mind-changes about whether to include in or exclude from the language that candidate item (in other words, correction grammars are algorithms for limiting computable functions — that initially output 0 for exclusion). For example, a correction grammar $\langle g_1, g_2\rangle$ for the language $L = L_1 - L_2$, as described above, can be equivalently thought of as an algorithm that initially excludes each item x and, then, can change its mind about x's inclusion or exclusion up to *twice* on the way to giving its final, correct answer as to whether x is included in or excluded from L.[5] More generally, a k-correction grammar is equivalent to an algorithm that initially excludes each item x and, then, it can change its mind

[4] W_i is the i-th c.e. set, where i codes a program for generating or for accepting W_i.

[5] Of course, an ordinary grammar (c.e. index) g for a c.e. language L can be thought of as an algorithm that initially excludes each item x and, then, it can change its mind about x's inclusion or exclusion up to *once* on the way to giving its final, correct answer as to whether x is included or excluded in L.

about x's inclusion or exclusion up to k times on the way to giving its final, correct answer as to whether x is included or excluded.

This description of correction grammars can be seen as modeling the observable, multiply self-correcting behaviour of humans.

The observable need for self-correction might be blamed on an error in the rule itself or on a performance error, i.e., an error of rule application. The idea of correction grammars explores the theoretical possibility that the cause of self-correcting behaviour depends on the form of the rules themselves, rather than, e.g., on an error in the rule or in rule application.

If it is the case that the self-correcting behaviour of humans depends on the fact that humans internally represent the target language as a correction grammar and not as a standard grammar, it is natural to ask: is there some *learning* gain that compensates for the need of self-corrections?

We investigate a formal version of the latter question, in the context of computability-theoretic learning theory (or learning in the limit) [24, 26]: a learning machine (an algorithmic device) receives as input a sequence $e_1, e_2, \ldots$ of all and only the elements of a c.e. language L (any such sequence is called a *text* for L) and outputs a corresponding infinite sequence $g_1, g_2, \ldots$ of grammars that *may* generate L.

Several criteria of successful learning of a language can and have been studied. The most basic one is *Explanatory Learning from Text* (**TxtEx**-learning): the machine is required to output, past some point, one and the same correct grammar for the input language.

A more liberal (and more powerful) criterion is *Behaviourally Correct Learning from Text* (**TxtBc**-learning): the machine is required to output, past some point, only correct grammars, though possibly infinitely many syntactically distinct ones.

Both these criteria feature learning *in the limit* (the machine does not know if and when it has converged) and require success of the learner on *any* order of presentation of the data. Since learning a single c.e. language is trivial in this model, the simultaneous learning of *classes* of c.e. languages is studied. Note that the conjectures of a learning machine as just above are standard type-0 grammars [25], or, equivalently, c.e. indices for the language [40]. The formal version of the above question is now the following: what is the power of an algorithmic learning machine that outputs correction grammars instead of c.e. indices for c.e. languages?

One of the main results of the present paper (Theorem 21, Section 3.2) implies that learning correction grammars for classes of c.e. languages is more powerful than learning ordinary c.e. indices in the **TxtEx**-model. The increase in power is so strong that there are classes of *computable* languages that are **TxtEx**-learnable by a machine that outputs correction grammars but not by any **TxtBc**-learner conjecturing c.e. indices — and this even if the learner is presented with informant, instead of text, for the languages in the class (an *informant* for a language is the graph of the characteristic function of the language). An informal, intuitive explanation for this result, as well as (*mutatis mutandis*) for its generalizations described below, is that, while ordinary grammars or c.e. indices provide information on how to list the corresponding c.e. language, correction grammars do not, in general, provide this information, so, one might expect that correction grammars, being, in *this* respect less informative, may be easier to learn.[6]

Actually, Theorem 21 provides that learning $(k+1)$-correction grammars is more powerful than learning k-correction grammars. So a next mathematically natural step is to extend the concept of a correction grammar into the *constructive transfinite*. We explain briefly. *Ordinals* are canonical representatives of well-orderings. We know from set theory that each well-ordered set (finite or infinite) is isomorphic to a unique ordinal. The *transfinite ordinals* are the ordinals beyond the finite ones — i.e., which are not isomorphic to any of the well-ordered finite sets. A *constructive ordinal* is the order-type of some computable well-ordering of the set of non-negative integers [40]. Constructive ordinals are equivalent to those that have a program, called a *notation*, which specifies how to effectively build them or lay them out end-to-end (see *infra* for further details). All constructive ordinals are countable. ω denotes the first (smallest) ordinal which is not isomorphic to a finite well-ordering. The ordinal ω is isomorphic to the standard well-ordering of the natural numbers $0 < 1 < 2 < \ldots$. The ordinal ω is constructive and transfinite, in fact the smallest transfinite ordinal. Our concept of a general correction grammar is roughly based on the idea of using constructive ordinals to bound the number of corrections that the grammar can make. To do this rigorously, we explain and use below the idea of counting-down from notations for (countable, constructive) transfinite ordinals. *For example*, it can be shown that counting down corrections allowed from any notation for ω is equivalent to declaring algorithmically, at the time a first correction is made, a finite number bound on the number of *further* corrections to be allowed. This is more powerful than just initially setting a *fixed, finite* number of corrections allowed. As another example consider using the ordinal $\omega + \omega$ (two copies of ω laid end to end), also constructive and transfinite: in this case, at the time the first correction is made, the algorithm declares a finite bound on the number of further corrections it is going to make; this bound is, however, allowed to be changed once, at a later time. For constructive ordinal $\omega + \omega + \omega$, the algorithm is allowed to update the bound twice. If one considers the constructive ordinal ω^2, then the algorithm is allowed to make a finite number of changes to the bound, where the maximum number of changes allowed to the bound is announced at the time the algorithm makes the first correction!

To expand a bit on our description above, the constructive ordinals are just those that have a notation (in some system) which specifies how to build them from below using notations for smaller ordinals and basic effective operations (typically successor and constructive limit). We will employ, as our system of notations, Kleene's general system O [29–31, 40]. This system has at least one notation for each constructive ordinal and comes with Kleene's standard, useful order relation $<_o$ on the notations in O. The order relation $<_o$ naturally embeds into the ordering of

[6] Furthermore, it is shown in [14] that correction grammars for *suitable* finite (respectively, co-finite) languages can be $\emptyset^{(1)}$ (respectively $\emptyset^{(2)}$) more succinct than minimal size ordinary grammars; hence, *these* correction grammars may carry less information than the corresponding minimal size ordinary grammars. *However*, here is a subtlety. It is also shown in [14] that one can prove some global properties of suitable c.e. languages L from suitable correction grammars for L that one cannot prove from any ordinary grammars for L. This provides a different and contrasting sense in which some correction grammars for some c.e. languages carry more information than any corresponding ordinary grammars.

the corresponding constructive ordinals (i.e., if u is a notation for α and v is a notation for β and $u <_o v$, then $\alpha < \beta$).

The idea of using constructive notations to perform *algorithmic* count-down from transfinite (constructive) ordinals is widely used in Proof Theory (e.g., to measure the strength of formal systems and classify their provably total functions) [43, 39], and in Term Rewriting (e.g., to prove termination of rewrite systems) [8, 44]. In Learning Theory, this idea has been introduced by Freivalds and Smith in [23]. They used ordinal notations, for example, for algorithmically counting down the mind-changes of inductive inference procedures. This allowed for handling and studying constructive, "transfinite" bounds on mind-changes of inductive inference machines.

In the present paper, we use ordinal notations to bound the number of corrections allowed to a correction grammar. Equivalently, a correction grammar is a total procedure for a limiting computable 0–1 function that initially excludes each item x (outputs 0 for exclusion) and that uses an ordinal notation to bound the number of later mind-changes about inclusion or exclusion of an element in the conjectured language. The well-orderedness of the ordinals ensures that such a procedure converges. (See [1] for another use of ordinal notations in the context of inductive inference *of functions.*) We formalize the concept of a u-correction grammar, where u is a notation in O for some constructive ordinal. To do this, we, in effect, use *descriptions* of sets in levels of the Ershov Hierarchy [18–20]. The Ershov Hierarchy is based on effective iteration of set-theoretic differences of c.e. sets *including up into the constructive transfinite.* A correction grammar for a c.e. set is, in fact, a "description" of the c.e. set as belonging to some level of the Ershov Hierarchy. We will build on recent work by Case and Royer [14] who obtained succinctness results for correction grammars and developed *useful*, *uniform* numberings (i.e., programming systems) for the relevant classes of the Ershov Hierarchy. Our using these programming systems has the following advantage: it is shown in [14] that these programming systems are *acceptable* [40, 32].[7] Since our results are easily seen to be *independent* of which acceptable systems are used for the Ershov classes, from our employment in our proofs of these useful acceptable systems from [14], we can conclude that our results hold for *all* acceptable systems.

From the perspective of the Cognitive Science interpretation of the present investigation, the following should be observed. Although the definition of general correction grammars makes use of the notion of (constructive) transfinite ordinals, the algorithmic count-down always consists of a *finite* number of steps down along a (possibly very intricate) countable *well-ordering.* Thus, the general notion of correction grammar still reflects the behaviour of a learner that is allowed to correct its linguistic output *a finite number* of times.

On the other hand, the correction grammar model has no pretension of modeling human self-correcting behaviour in more detail. For example, there is no control on *how late* a correction may occur, and it should be observed that this is already true for the base case of correction grammars, i.e., grammars that are allowed only one correction on each candidate element of the language.

[7] By definition, the *acceptable* programming systems for a class are those which contain a universal simulator and into which all other universal programming systems for the class can be compiled. Acceptable systems are characterized as universal systems with an algorithmic substitutivity principle called S-m-n [40, 32, 41, 14]. Acceptable systems also satisfy self-reference principles such as Recursion Theorems [40, 32, 41, 14].

The existence of a hierarchy of more and more powerful learning criteria, increasing in the ordinal notation used to count corrections of a correction grammar is one of the main results of the present paper (Theorem 19, Section 3.1). We show that, if u and v are two notations for two constructive ordinals α and β, respectively, such that $u <_o v$ (which implies $\alpha < \beta$), then **TxtEx**-learning correction grammars that count-down from v is more powerful than **TxtEx**-learning correction grammars that count-down from u. That is, there are classes of c.e. languages, even classes of computable languages, that can be **TxtEx**-learned by conjecturing correction grammars that count-down from v but not by conjecturing correction grammars that count-down from u — and this even if the learner is presented with informant, instead of text, for the languages in the class.

Surprisingly, Theorems 29 and 35 show that the following collapse occurs: any class of c.e. languages that is **TxtEx**-learnable or **TxtBc**-learnable by a learner outputting u-correction grammars, for any notation u for a transfinite ordinal, is already **TxtBc**-learnable by a learner outputting w-correction grammars, where w is a notation for the smallest infinite ordinal ω.[8] Hence, there is a learning power tradeoff between, on the one hand, employing u-corrections, where u is a notation for a very large transfinite ordinal, with **TxtEx**-learning which nicely features only *one* correct correction grammar in the limit and, on the other hand, stopping at "ω" corrections but, then, paying the price of infinitely many distinct correction grammars in the limit. Theorems 32 and 35 show similar collapsing result when (unbounded or bounded) finite number of errors are allowed in the grammars (eventually) output by the **TxtBc** learner.

The paper is organized as follows. In Section 2 we introduce notations and definitions needed for the rest of the paper. This section includes general recursion-theoretic notation (Section 2.1), a quick treatment of Kleene's O (Section 2.2), the definition of the Ershov Hierarchy (Section 2.3), and the basics of Gold-style Learning Theory, culminating in the formal definition of learning general correction grammars (Section 2.4).

Next, in Section 3, we first prove our general hierarchy result (Theorem 19): **TxtEx**-learning correction grammars that count-down from (constructive notations for) transfinite ordinals yields an infinite hierarchy of more and more powerful learning criteria.

We then show (Theorem 21) that, for each finite level of this hierarchy, the result can be strengthened as follows: **TxtEx**-learning $n+1$-correction grammars is sometimes more powerful than **TxtBc**-learning n-correction grammars for every natural number n.

In Section 4, we present some surprising collapsing results, showing that the hierarchy results of Section 3 are best possible. For example, we show (Theorem 29) that every class that is **TxtEx**-learnable by conjecturing u-correction grammars, where u is a notation for any constructive ordinal, is already **TxtBc**-learnable using correction grammars that count-down from any notation for ω, the least infinite ordinal.

Section 5 contains some miscellaneous further results regarding learning slightly anomalous programs (Section 5.1) and program size complexity restricted learning (Section 5.2).

[8] We note, by way of informally and very *approximately* explaining this collapse that in contexts of learning descriptions of *functions*, e.g., in the proofs of [15, Theorem 3.10 (Harrington)], [12, Theorem 14], and [4, Theorem 7(d)], one gets collapsings of some different sorts with cases of *suitably* twice iterating either limits or $\forall^\infty$ quantifiers, where $\forall^\infty$ denotes 'for all but finitely many'.

Section 6 contains concluding remarks, announcement of further results, open problems, and suggestions for future work.

2 Preliminaries

This section contains terminology for the rest of the paper, a presentation of Kleene's ordinal notation system O, and general background on the Ershov Hierarchy. These technical concepts and tools are then used to define formally the criteria of learning correction grammars that count-down from (notations for constructive) transfinite ordinals.

2.1 Notation and Recursion Theory Background

Any unexplained recursion theoretic notation is from [40]. The symbol $\mathbf{N}$ denotes the set of natural numbers, $\{0, 1, 2, 3, \ldots\}$. The symbols $\emptyset$, $\subseteq$, $\subset$, $\supseteq$, $\supset$, and $\mathbf{\Delta}$ denote empty set, subset, proper subset, superset, proper superset, and symmetric difference, respectively. The cardinality of a set S is denoted by $\mathrm{card}(S)$. $\mathrm{card}(S) \leq *$ denotes that S is finite. We use the convention $n < *$ for all $n \in \mathbf{N}$. The maximum and minimum of a set are denoted by $\max(\cdot), \min(\cdot)$, respectively, where $\max(\emptyset) = 0$ and $\min(\emptyset) = \infty$. $L_1 =^n L_2$ means that $\mathrm{card}(L_1 \mathbf{\Delta} L_2) \leq n$ and L_1 and L_2 are called n-variants. $L_1 =^* L_2$ means that $\mathrm{card}(L_1 \mathbf{\Delta} L_2) \leq *$, i.e., is finite; in this case L_1 and L_2 are called finite-variants.

We let $\langle \cdot, \cdot \rangle$ stand for Cantor's computable, bijective mapping $\langle x, y \rangle = \frac{1}{2}(x+y)(x+y+1)+x$ from $\mathbf{N} \times \mathbf{N}$ onto $\mathbf{N}$ [40]. Note that $\langle \cdot, \cdot \rangle$ is monotonically increasing in both of its arguments. We define $\pi_1(\langle x, y \rangle) = x$ and $\pi_2(\langle x, y \rangle) = y$.

By φ we denote a fixed *acceptable* programming system for the partial computable functions mapping $\mathbf{N}$ to $\mathbf{N}$ [40]. By φ_i we denote the partial computable function computed by the program number i in the φ-system. We assume that multiple arguments are coded in some standard way [40] and suppress the explicit coding. By Φ we denote an arbitrary fixed Blum complexity measure [7, 25] for the φ-system. A partial computable function $\Phi(\cdot, \cdot)$ is said to be a Blum complexity measure for φ, if and only if the following two conditions are satisfied:

(a) for all i and x, $\Phi(i, x)\downarrow$ if and only if $\varphi_i(x)\downarrow$.

(b) One can effectively (in i, x, t) decide whether $\Phi(i, x) \leq t$.

By convention we use Φ_i to denote the partial computable function $x \rightarrow \Phi(i, x)$. Intuitively, $\Phi_i(x)$ may be thought as the number of steps it takes to compute $\varphi_i(x)$. $\varphi_{i,s}$ denotes the complexity-bounded version of φ_i, that is, $\varphi_{i,s}(x) = \varphi_i(x)$, if $x < s$ and $\Phi_i(x) < s$; $\varphi_{i,s}(x)$ is undefined otherwise.

Formally, W_i denotes domain(φ_i). That is, W_i is the set of all numbers on which the φ-program i halts. This treats i as an *acceptor* program for W_i [25]. By $W_{i,s}$ we denote the set domain$(\varphi_{i,s}) = \{x < s \mid \Phi_i(x) < s\}$. We say that p is a limiting computable program for a total function f if φ_p is a total function, and for all x, $\lim_{t\to\infty} \varphi_p(x, t) = f(x)$.

Every subset of $\mathbf{N}$ is considered to be language. The symbol $\mathcal{E}$ will denote the set of all c.e. languages. The symbol L ranges over $\mathcal{E}$. By $\overline{L}$, we denote the complement of L, that is $\mathbf{N} - L$. χ_L denotes the characteristic function of L. The symbol $\mathcal{L}$ ranges over subsets of $\mathcal{E}$.

As noted above, $\forall^\infty$ denotes for all but finitely many.

2.2 Constructive Ordinals and Kleene's *O*

We proceed informally (for a detailed treatment see [40, 3]). The exposition in this and next subsection (Section 2.3) is based closely on [14]. A *system of notation* S is a collection of programs (*S-notations*) each of which specifies a structured algorithmic description of some ordinal. Specifically, the notations are programs for building, or laying down end-to-end, the denoted ordinal. An ordinal is called *constructive* when it has a notation in some system of notation. An ordinal is called a *successor* ordinal, if it is of the form $\alpha + 1$, for some ordinal α. An ordinal is a *limit* ordinal if it is neither 0 nor a successor ordinal.

A *system of notation* S consists of a subset N_S of $\mathbf{N}$ (the set of S-notations), and a mapping $S[\cdot]$ from N_S to an initial segment of the ordinals, such that:

- For $x \in N_S$, the properties of being a notation for 0, a notation for a successor ordinal and a notation for a limit ordinal are computably decidable.
- There is a partial computable function p (the *predecessor* function) such that, if $x \in N_S$ is a notation for a successor ordinal, then $p(x)$ is defined and is a notation for the immediate predecessor of the ordinal $S[x]$ denoted by x. I.e., $S[p(x)] + 1 = S[x]$.
- There exists a partial computable function q (the *fundamental sequence* function) such that for every notation $x \in N_S$ for a limit ordinal, the function mapping y to $q(x, y)$ is total and $S[q(x, 1)] < S[q(x, 2)] < \ldots$ has limit $S[x]$.

A system of notation S is *acceptable* if any other system is computably order-preservingly embeddable in it. Formally, if, for any other system of notation S', there exists a partial computable τ such that, for all S'-notation x, $\tau(x)$ is defined and is an S-notation, and $S'[x] \leq S[\tau(x)]$; Furthermore, if $x <_{S'} y$, then $\tau(x) <_S \tau(y)$, where $<_{S'}$ and $<_S$ are ordering relations among S' and S notation, respectively. Each acceptable system of notation assigns at least one notation to every constructive ordinal. A system of notation S is *univalent* if $S[\cdot]$ is injective. It is known that every acceptable system *fails* to be univalent (see [40]).

Kleene [29–31, 40] developed a general acceptable system of notation O. Every constructive ordinal has at least one notation in O. O is endowed with a relation $<_o$ on notations that naturally embeds in the ordering of the corresponding constructive ordinals: for all O-notations u, v, if $u <_o v$ then $O[u] < O[v]$.

We will not need much of the particular features of O in what follows, but it is nonetheless necessary to refer to *some* system of notations to define precisely our general concept of learning by correction grammars. The use of O is advantageous in that it provides a general system containing at least one notation for each constructive ordinal. This will allow us to state our general Hierarchy Theorem with satisfactory generality and uniformity.

In Kleene's system O, 2^0 is (by definition) the *notation* for the *ordinal* 0. If u is a notation for the immediate predecessor of a successor ordinal, then a notation for that successor ordinal is (by definition) 2^u. We omit the details of the definition of $<_o$. We now consider notations for limit ordinals. Suppose $\varphi_p(0), \varphi_p(1), \varphi_p(2), \ldots$ are each notations in $<_o$ order (see [40]). Suppose, then, that the corresponding ordinals are longer and longer initial segments of some limit ordinal which is their supremum. For example, some such p generates the respective notations for $0, 1, 2, \ldots$ in $<_o$ order, and ω is the supremum of this sequence. In general, then, p essentially describes how to build the limit ordinal which is the supremum of the ordinals with notations $\varphi_p(0), \varphi_p(1), \varphi_p(2), \ldots$. A notation for this limit ordinal is (by definition) $3 \cdot 5^p$. A sequence of notations for larger and larger ordinals is also called a *fundamental sequence* for their supremum. The assignment of fundamental sequences to limit ordinals is an essential ingredient of an ordinal notation system. Clearly limit ordinals have infinitely many notations, different ones for different generating p's. Nothing else is a notation. We define '$x =_o y$' to mean '$x, y \in O$ and $x = y$'. As in the literature on constructive ordinals, we use '$x \leq_o y$' for '$x <_o y \vee x =_o y$', '$x \geq_o y$' to mean '$y \leq_o x$ and '$x >_o y$' to mean '$y <_o x$'. We also recall that the mapping $O[\cdot] : O \to$ the set of (constructive) ordinals, is defined as follows [29, 31, 40]

$$\begin{aligned} O[1] &= 0; \\ O[2^u] &= O[u] + 1; \\ O[3 \cdot 5^p] &= \lim_{n \to \infty} O[\varphi_p(n)], \end{aligned}$$

where, for the latter, for each n, $\varphi_p(n) <_o \varphi_p(n+1) \in O$.

For all $x, y \in O$, it is true that, if $x <_o y$ then $O[x] < O[y]$. It is also true that, for all $y \in O$, if $O[y] = \beta$, then for every $\alpha < \beta$, there is an x such that $x <_o y$ and $O[x] = \alpha$. If $u \in O$ and $O[u] = \alpha$, then we say that u is *for* α.

We shall use the following properties of O in later proofs.

Lemma 1 (Some properties of O, [40]).

1. *For every $n \in \mathbf{N}$ there exists a unique O-notation for n. This notation will be denoted by $\underline{n}$.*
2. *For every $v \in O$, $\{u \mid u <_o v\}$ is a univalent system of notations for the corresponding initial segment of the ordinals.*
3. *There exists a c.e. set Z such that $\{u \mid u <_o v\} = \{u \mid \langle u, v \rangle \in Z\}$, for each $v \in O$.*
4. *There exists a computable mapping $+_o : \mathbf{N} \times \mathbf{N} \longrightarrow \mathbf{N}$ such that, for every $u, v \in O$,* (i) *$u +_o v \in O$,* (ii) *$O[u +_o v] = O[u] + O[v]$, and* (iii) *if $v \neq 0$ then $u <_o u +_o v$.*

In the rest of this paper, u, v, w denote elements in O.

2.3 The Ershov Hierarchy

In the present subsection we introduce the Ershov Hierarchy [18–20], and give the definition of a *particular* acceptable universal programming system W^u for each level of the hierarchy, due to Case and Royer in [14].[9]

[9] This system was created in part to make sure there *is* such an acceptable system. The construction of the W^u's is also nicely uniform in $u \in O$.

Our presentation of the Ershov Hierarchy is in terms of count-down functions from O-notations for constructive ordinals. Similar presentations, differing from but equivalent to the one originally given by Ershov, can be found in [3, 17].

Definition 2 (Count-Down Function). A computable function $F : \mathbf{N} \times \mathbf{N} \to O$ is a *count-down function* if for all x and t, $F(x, t+1) \leq_o F(x, t)$.

For a binary function $h(\cdot, \cdot)$ we write $h(x, \infty)$ for the limit $\lim_{t\to\infty} h(x, t)$. For a set A, we denote by χ_A the characteristic function of A.

Definition 3 (Ershov Hierarchy). $A \in \Sigma^{-1}_u$ if and only if there exists a computable function $h : \mathbf{N} \times \mathbf{N} \to \{0, 1\}$ and a count-down function F such that, for all $x, t \in \mathbf{N}$,

(i) $\chi_A(x) = h(x, \infty)$,
(ii) $h(x, 0) = 0$ and $F(x, 0) \leq_o u$,
(iii) $h(x, t+1) \neq h(x, t) \Rightarrow F(x, t+1) <_o F(x, t)$.

In this case we say that h and F *witness* $A \in \Sigma^{-1}_u$.

Note that $\Sigma^{-1}_{\underline{0}} = \{\emptyset\}$. Definition 3 immediately implies that $u <_o v \Rightarrow \Sigma^{-1}_u \subseteq \Sigma^{-1}_v$. The containment is in fact proper, so that one speaks of the Ershov Hierarchy.

The next lemma spells out the correspondence between a description of a set in terms of finite differences of c.e. sets and in terms of count-down functions.

Lemma 4 (Case and Royer, [14]). *$X \in \Sigma^{-1}_{u+_o\underline{1}}$ if and only if there exists $Y \in \mathcal{E}$, $\exists S \in \Sigma^{-1}_u$ such that $S \subseteq Y$ and $X = Y - S$.*

Corollary 5. *Let $n \geq 1$. $X \in \Sigma^{-1}_{\underline{n}}$ if and only if there exist c.e. sets $Y_1 \supseteq \ldots \supseteq Y_n$ such that $X = Y_1 - (Y_2 - (\ldots - (Y_{n-1} - Y_n) \ldots))$.*

We will now proceed to define, for each $u \in O$, an acceptable universal numbering or programming system W^u for Σ^{-1}_u. This numbering is due to Case and Royer [14]. The proof that it gives an acceptable numbering is omitted here.

Let us denote by φ^{TM} an acceptable programming system for the partial computable functions based on a coding of deterministic multi-tape Turing Machines. By standard results [40], φ^{TM} is an acceptable programming system for the partial computable functions.

For each x and i in $\mathbf{N}$, let $\Phi^{\mathrm{TM}}_i(x)$ be the runtime of Turing Machine i on input x. It is easy to check that $\Phi^{\mathrm{TM}}_i(x)$ is a Blum Complexity Measure [7] for φ^{TM}. It is easy to arrange the numerical TM coding so that $\{\langle x, i, t\rangle \mid \Phi^{\mathrm{TM}}_i(x) \leq t\}$ is primitive recursively decidable (see, for example, [41]).

From Lemma 1 part 3, we know that there exists a c.e. set Z such that

$$\{u \mid \langle u, v\rangle \in Z\} = \{u \mid u <_o v\},$$

for each $v \in O$. In what follows, let z_0 be a fixed φ^{TM} program for accepting Z.

Definition 6 (Convenient Function). Let F be a count-down function. Then F is *convenient* (relative to z_0) if

$$(\forall x)(\forall t)[F(x,t+1) <_o F(x,t) \Rightarrow \Phi^{\mathrm{TM}}_{z_0}(\langle F(x,t+1), F(x,t)\rangle) \le t].$$

We say that h and F *conveniently witness* $A \in \Sigma^{-1}_u$ when h and F witness $A \in \Sigma^{-1}_u$ and F is convenient.

Let ψ be a *standard* programming system for the primitive recursive functions, (i.e., one for which the S-m-n Theorem and the Recursion Theorems all hold) [41].

Definition 7. Let $u \in O$. We say that i, j, x are *u-consistent through t* when $\psi_i(x,0) = 0$, $\psi_j(x,0) =_o u$ and, for each $t' < t$,

(i) $\psi_i(x,t'+1) \in \{0,1\}$,
(ii) $\psi_j(x,t'+1) \ne \psi_j(x,t') \Leftrightarrow \Phi^{\mathrm{TM}}_{z_0}(\psi_j(x,t'+1), \psi_j(x,t')) \le t'$,
(iii) $\psi_i(x,t'+1) \ne \psi_i(x,t') \Rightarrow \psi_j(x,t'+1) <_o \psi_j(x,t')$.

Definition 8. Let $u \in O$. For each i, j, x, t let

$$h^u(i,j,x,t) = \begin{cases} 0 & \text{if } i,j,x \text{ are not } u\text{-consistent through } 0; \\ \psi_i(x,t') & \text{otherwise, where} \\ & t' \text{ is the greatest number } \le t \\ & \text{such that } i,j,x \text{ are } u\text{-consistent through } t'. \end{cases}$$

For each $i, j \in \mathbf{N}$, let

$$W^u_{\langle i,j\rangle} = \{x \mid h^u(i,j,x,\infty) = 1\}.$$

We call p a *u-correction grammar* for W^u_p, and we are herein interested in such grammars for c.e. (and computable) languages.

We observe that $h^u(i,j,x,t)$ is double recursive (see [37]) as a function of u, i, j, x, t, while, for each u, $h^u(i,j,x,t)$ is primitive recursive as a function of i, j, x, t.[10]

Theorem 9 (Case and Royer, [14]).

(a) *W^u is an acceptable universal numbering for Σ^{-1}_u.*
(b) *The S-m-n and the Recursion Theorems hold for W^u.*

In particular, by the previous Theorem, the Kleene Recursion Theorem holds in the W^u-system.[11] This means that, given the specification of a W^u-system task p, there exists an e such that program e in the W^u-system makes a self-copy (i.e., computes a copy of its own code) and applies that task p to this self-copy (and, of course, to its external input). In proofs below, for convenience, we will give the description of what such an e does with its self-copy in an *informal*

[10] As noted in [14], it is possible, using methods from [41], to make $h^u(i,j,x,t)$ exponential-time computable as a function of u, i, x, t and, for every u, to make $h^u(i,j,x,t)$ linear-time computable as a function of i, j, x, t.

[11] For the φ-system the Kleene Recursion Theorem is described in [40, Page 214]. Herein we employ this self-reference principle for the W^u systems, $u \in O$.

system. In particular we will describe task-relevant functions h and F, *each informally in terms of* e, such that F is a count-down function and h and F witness that $\{x \mid h(x,\infty) = 1\} \in \Sigma_u^{-1}$ (Definition 3). In each such case, we make sure this latter set *is* u-c.e. Implicitly, then, we invoke the acceptability of W^u to obtain a translation of the informal description involving h and F into the W^u-system to get what e really does in the formal W^u-system with its self-copy (and its external input). In practice, though, we will merely describe informally what such a formal e does with its self-copy, and not mention again the invoking of acceptability to get a translation of what such an e does more formally with its self-copy, etc.

We adopt, for the sake of readability, the following notational convention:

$$\theta_p^u(x,t) = h^u(\pi_1(p), \pi_2(p), x, t).$$

We now have all the tools needed to define the general learning criteria of learning u-correction grammars.

2.4 Learning Criteria

We now present concepts from language learning theory (see [26]) and then formally define learning correction grammars.

The next definition introduces the concept of a *sequence* of data.

Definition 10 (Sequence).
(a) A *sequence* σ is a mapping from an initial segment of $\mathbf{N}$ into $(\mathbf{N} \cup \{\#\})$. The empty sequence is denoted by λ.
(b) The *content* of a sequence σ, denoted content(σ), is the set of natural numbers in the range of σ.
(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, the pause-symbol # represents a pause in the presentation of data. We let σ, τ and γ range over finite sequences. We denote the sequence formed by the concatenation of τ at the end of σ by $\sigma \diamond \tau$ or by $\sigma\tau$. Sometimes we abuse the notation and use σx to denote the concatenation of sequence σ and the sequence of length 1 which contains the element x. SEQ denotes the set of all finite sequences.

Definition 11 (Texts, [24]).
(a) A *text* T for a language L is a mapping from $\mathbf{N}$ into $(\mathbf{N} \cup \{\#\})$ such that L is the set of natural numbers 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 content(T), is the set of natural numbers in the range of T; that is, the language which T is a text for.
(c) $T[n]$ denotes the finite initial sequence of T with length n.

Definition 12 (Learning Machine, [24]). A *learning machine* (or just *learner*) is an algorithmic device which computes a mapping from SEQ into $\mathbf{N}$.

We let $\mathbf{M}$ range over learning machines. We note that, without loss of generality, for all criteria of learning discussed in this paper, a learner $\mathbf{M}$ may be assumed to be total (see [33, Lemma 4.2.2B] for the proof for the criteria **TxtEx**; same proof works for the criteria **TxtBc** also). $\mathbf{M}(T[n])$ denotes the hypothesis of the learner $\mathbf{M}$ after it has seen the first n members of T. $\mathbf{M}(T) = e$ denotes that $\mathbf{M}$ converges on T to e, that is $\mathbf{M}(T[n]) = e$, for all but finitely many n.

There are several criteria for a learning machine to be successful on a language. Below we define some of them.

Definition 13 (Explanatory Learning, [13, 24]). Suppose $a \in \mathbf{N} \cup \{*\}$.

(a) $\mathbf{M}$ $\mathbf{TxtEx}^a$*-identifies a text* T just in case $(\exists i \mid W_i =^a \text{content}(T))$ $(\forall^\infty n)[\mathbf{M}(T[n]) = i]$.

(b) $\mathbf{M}$ $\mathbf{TxtEx}^a$*-identifies a c.e. language* L (written: $L \in \mathbf{TxtEx}^a(\mathbf{M})$) just in case $\mathbf{M}$ $\mathbf{TxtEx}^a$-identifies each text for L.

(c) $\mathbf{M}$ $\mathbf{TxtEx}^a$*-identifies a class* $\mathcal{L}$ *of c.e. languages* (written: $\mathcal{L} \subseteq \mathbf{TxtEx}^a(\mathbf{M})$) just in case $\mathbf{M}$ $\mathbf{TxtEx}^a$-identifies each language from $\mathcal{L}$.

(d) $\mathbf{TxtEx}^a = \{\mathcal{L} \subseteq \mathcal{E} \mid (\exists \mathbf{M})[\mathcal{L} \subseteq \mathbf{TxtEx}^a(\mathbf{M})]\}$.

Definition 14 (Behaviourally Correct Learning, [13, 34]). Suppose $a \in \mathbf{N} \cup \{*\}$.

(a) $\mathbf{M}$ $\mathbf{TxtBc}^a$*-identifies a text* T just in case $(\forall^\infty n \in \mathbf{N})[W_{\mathbf{M}(T[n])} =^a \text{content}(T)]$.

(b) $\mathbf{M}$ $\mathbf{TxtBc}^a$*-identifies a c.e. language* L (written: $L \in \mathbf{TxtBc}^a(\mathbf{M})$) just in case $\mathbf{M}$ $\mathbf{TxtBc}^a$-identifies each text for L.

(c) $\mathbf{M}$ $\mathbf{TxtBc}^a$*-identifies a class* $\mathcal{L}$ *of c.e. languages* (written: $\mathcal{L} \subseteq \mathbf{TxtBc}^a(\mathbf{M})$) just in case $\mathbf{M}$ $\mathbf{TxtBc}^a$-identifies each language from $\mathcal{L}$.

(d) $\mathbf{TxtBc}^a = \{\mathcal{L} \subseteq \mathcal{E} \mid (\exists \mathbf{M})[\mathcal{L} \subseteq \mathbf{TxtBc}^a(\mathbf{M})]\}$.

For learning criteria $\mathbf{I}$, $a = 0$, we often write $\mathbf{I}$ instead of $\mathbf{I}^a$. It is well-known that $\mathbf{TxtEx}^a \subset \mathbf{TxtBc}^a$ (see [13]). With an abuse of terminology we sometimes refer to any criterion that requires syntactic (resp. semantic) convergence in the limit as **Ex**- (resp. **Bc**-) learning.

We collect some well-known facts about learning with anomalies in the following theorem.

Theorem 15 (Basic Results [13]). Suppose $n, m \in \mathbf{N}$.

(a) $\mathbf{TxtEx} \subset \mathbf{TxtEx}^1 \subset \ldots \subset \mathbf{TxtEx}^*$,

(b) $\mathbf{TxtEx}^* - \bigcup_{n \in \mathbf{N}} \mathbf{TxtEx}^n \neq \emptyset$,

(c) $\mathbf{TxtBc} \subset \mathbf{TxtBc}^1 \subset \ldots \subset \mathbf{TxtBc}^*$,

(d) $\mathbf{TxtEx}^*$ and $\mathbf{TxtBc}$ are incomparable,

(e) $\{L \mid L =^{2m+1} \mathbf{N}\} \in (\mathbf{TxtEx}^{2m+1} - \mathbf{TxtBc}^m)$,

(f) $\mathbf{TxtEx}^{2m} \subseteq \mathbf{TxtBc}^m$.

When we only require that a learner is successful when fed the graph of the characteristic function of the language instead of any text, we obtain the concept of *learning from informant* (see [24]). For an informant I, we denote by $I[n]$, the first n elements of I. A canonical informant for a language L is $(0, \chi_L(0)), (1, \chi_L(1)), (2, \chi_L(2)), \ldots$. We often identify χ_L with the canonical informant for L. For a characteristic function f, we use $f[n]$ to denote the initial segment $(0, f(0)), (1, f(1)), \ldots, (n-1, f(n-1))$.

Using **Inf** instead of **Txt** in the name of any learning criterion indicates that the requirement of learning *from texts* is substituted by the requirement of learning *from informant*. It is well-known that more can be learned from informant than from text (see [26]).

We can now formally introduce learning by correction grammars. Intuitively, a $\mathbf{Cor}^u\mathbf{I}$-learner, where **I** is any learning criterion, is a successful **I**-learner when its conjectures are interpreted as u-correction grammars.

Definition 16 (Learning Correction Grammars). Let $u \in O$, $a \in \mathbf{N} \cup \{*\}$.

(a) $\mathbf{Cor}^u\mathbf{TxtEx}^a$ is the collection of all classes $\mathcal{L}$ of *c.e.* languages such that there exists an $\mathbf{M}$ such that $(\forall L \in \mathcal{L})(\forall \text{ texts } T \text{ for } L)(\exists i)[W_i^u =^a L \wedge (\forall^\infty n)[\mathbf{M}(T[n]) = i]]$ — in this case we say that $\mathcal{L} \subseteq \mathbf{Cor}^u\mathbf{TxtEx}^a(\mathbf{M})$ or $\mathcal{L}$ is $\mathbf{Cor}^u\mathbf{TxtEx}^a$-identified by $\mathbf{M}$.

(b) $\mathbf{Cor}^u\mathbf{TxtBc}^a$ is the collection of all classes $\mathcal{L}$ of c.e. languages such that there exists an $\mathbf{M}$ such that $(\forall L \in \mathcal{L})(\forall \text{ texts } T \text{ for } L)(\forall^\infty n)[W^u_{\mathbf{M}(T[n])} =^a L]$ — in this case we say that $\mathcal{L} \subseteq \mathbf{Cor}^u\mathbf{TxtBc}^a(\mathbf{M})$ or $\mathcal{L}$ is $\mathbf{Cor}^u\mathbf{TxtBc}^a$-identified by $\mathbf{M}$.

It is important to note that, while the Ershov Hierarchy goes well beyond the c.e. languages, we are interested primarily in the *c.e.* languages and their learnability with respect to u-correction grammars. This implies that, for example, a $\mathbf{Cor}^u\mathbf{TxtEx}$-learner outputs "descriptions" of a *c.e.* language as a member of the u-th level of the Ershov Hierarchy (whereas the latter level also contains non-c.e. sets, if u denotes an ordinal larger than 1).

3 Hierarchy Results

In this section we prove some hierarchy results about learning correction grammars. Each of these separation results is witnessed by a class of computable languages.

Our first main result (Theorem 19) shows that an increase in learning power is obtained — in the context of **TxtEx**-learning correction grammars — when the number of corrections allowed is counted by (notations for) larger and larger constructive transfinite ordinals.

Next (Theorem 21) we prove a strengthening of this hierarchy for all finite levels: for all $n \in \mathbf{N}$, there are classes of *computable* languages that can be **TxtEx**-learned by a learner conjecturing $\underline{k+1}$-correction grammars that cannot be **TxtBc**-learned by any learner conjecturing $\underline{k}$-correction grammars (not even from informant). We will show in Section 4 that this strengthening is best possible: it cannot be extended beyond the ω-th level of the hierarchy.

3.1 The General Cor^{u}TxtEx Hierarchy

We will prove that for all $u, v \in O$ such that $u <_o v$ there exist classes of computable languages that are learnable by a **TxtEx**-learner that outputs v-correction grammars but such that no **TxtEx**-learner can learn those classes using u-correction grammars, even if presented with informants instead of texts. The general case will follow from the following result on the successor case.

Notation: We use $h(\cdot, s)$ to denote the function which maps x to $h(x, s)$.

Theorem 17. *For all $n \in \mathbf{N}$, $u \in O$, $\mathbf{Cor}^{u+_O\underline{1}}\mathbf{TxtEx} - \mathbf{Cor}^{u}\mathbf{InfEx}^{n} \neq \emptyset$.*

Proof. Let $\mathcal{L} = \{L \text{ computable} \mid L \neq \emptyset \wedge W^{u+_O\underline{1}}_{\min(L)} = L\}$. Clearly $\mathcal{L} \in \mathbf{Cor}^{u+_O\underline{1}}\mathbf{TxtEx}$. Suppose by way of contradiction that $\mathbf{M}$ $\mathbf{Cor}^{u}\mathbf{InfEx}^{n}$-identifies $\mathcal{L}$. Without loss of generality assume that all grammars e output by $\mathbf{M}$ satisfy that $\pi_1(e), \pi_2(e), x$ are u-consistent through t, for all x and t.

By the Kleene Recursion Theorem in the system $W^{u+_O\underline{1}}$ there exists an e such that $W^{u+_O\underline{1}}_e = \{x \mid h(x,\infty) = 1\}$, where h is a function informally defined in stages below (we will have that $e = \min(\{x \mid h(x,\infty) = 1\})$). Along with h we informally define another function F, such that F is a count-down function and h and F witness that $\{x \mid h(x,\infty) = 1\} \in \Sigma^{-1}_{u+_O\underline{1}}$ (Definition 3).

Initially, $h(x,0) = 0$ for all x; $h(y,1) = 0$, for $y < e$, and $h(y,1) = 1$, for $y \geq e$. $F(y,0) = u+_O\underline{1}$ and $F(y,1) = u$, for all y. Let $x_1 = e+1$. Go to stage 1 (we start with stage 1 for ease of notation).

We will have the invariants that, at the start of stage s,

(1) for $x > x_s + n$, $h(x,s) = 1$ and $F(x,s) = u$.
(2) for $x < x_s$, for all $t > s$, $h(x,t) = h(x,s)$.
(3) For all x such that $x_s \leq x \leq x_s + n$, either
  (3a) for $i = \mathbf{M}(h(\cdot,s)[x_s])$,
    $h(x,s) = 1 - \theta^u_i(x,s)$, and
    $F(x,s) = \psi_{\pi_2(i)}(x,s)$, or
  (3b) $h(x,s) = 1$, $F(x,s) = u$ (in this case $x_s \neq x_{s-1}$, where we take $x_0 = 0$).

Intuitively, we want the diagonalizing language to be $L = \{x \mid \lim_{t\to\infty} h(x,t) = 1\}$. Let L_s denote the diagonalizing language as at the beginning of stage s, that is $L_s = \{x \mid h(x,s) = 1\}$. At the start of stage s, we have settled the membership question for L on inputs $x < x_s$ (this is given by invariant (2) above). In stage s, we check (based on 'current' value L_s of L), whether $\mathbf{M}$ makes a mind change on χ_{L_s} beyond $\chi_{L_s}[x_s]$ (for computability reasons, we do this check only upto $\chi_{L_s}[s]$). If so, we fix the membership question for L on appropriate inputs so as to preserve the mind change (see step 1 in the construction below); otherwise, we make sure that the current conjecture of $\mathbf{M}$ on L (that is $\mathbf{M}(\chi_L[x_s])$) is wrong (as viewed at time $s+1$) on inputs x with $x_s \leq x \leq x_s + n$ (see step 2 in the construction). To achieve this latter property, we use invariant 3 above, which allows us to make sure that L is different from the conjecture $\mathbf{M}(\chi_L[x_s])$ on inputs x with $x_s \leq x \leq x_s + n$. Invariant (1) just says that the $x > x_s + n$ are 'unused' and available for future diagonalization if and when we observe a mind change by $\mathbf{M}$ on χ_L beyond $\chi_L[x_s]$.

Stage s

1. If there exists a z, $x_s + n < z \leq s$, such that $\mathbf{M}(h(\cdot,s)[z]) \neq \mathbf{M}(h(\cdot,s)[x_s])$, then
    Let $x_{s+1} = z$.
    For all x, let $h(x,s+1) = h(x,s)$ and $F(x,s+1) = F(x,s)$.
    Go to stage $s+1$.

```
2. Else,
    2.1  Let $i = \mathbf{M}(h(\cdot, s)[x_s])$.
    2.2  For $x_s \leq x \leq x_s + n$, let
             $h(x, s+1) = 1 - \theta_i^u(x, s+1)$, and
             $F(x, s+1) = \psi_{\pi_2(i)}(x, s+1)$.
         (Note that above change is valid, based on invariant 3 above).
    2.3  For $x < x_s$ or $x > x_s + n$, let
             $h(x, s+1) = h(x, s)$, and
             $F(x, s+1) = F(x, s)$.
    2.4  Let $x_{s+1} = x_s$.
         Go to stage $s+1$.
End stage $s$
```

We note that the invariants hold by induction on s. Clearly, x_s is monotonically non-decreasing in s. Invariants (1) and (2) are satisfied by induction as either $x_{s+1} > x_s$ and $h(x, s+1) = h(x, s)$ for all x (see step 1) or $x_{s+1} = x_s$ and $h(x, s+1) \neq h(x, s)$ only for x with $x_s \leq x \leq x_s + n$, if any (see step 2).

For invariant (3) note that if $x_{s+1} = x_s$, then for $x_s \leq x \leq x_s + n$, $h(x, s+1) \neq h(x, s)$ implies $\theta_i^u(x, s+1) \neq \theta_i^u(x, s)$ (by inductive property (3a)); if $x_{s+1} \neq x_s$, then $x_{s+1} > x_s + n$, and thus by invariant (1), (3b) holds (using $s+1$ instead of s in the property).

We now consider two cases.

Case 1: $\lim_{s\to\infty} x_s$ is infinite.

In this case, clearly, the function mapping x to $\lim_{t\to\infty} h(x, t)$ is a computable function, and on this function $\mathbf{M}$ makes infinitely many mind-changes. Furthermore, clearly, $\lim_{t\to\infty} h(x, t)$ is a characteristic function for a language in $\mathcal{L}$, as $e = \min(\{x \mid h(x, \infty) = 1\})$ and $W_e^{u+_o\underline{1}} = \{x \mid h(x, \infty) = 1\}$.

Case 2: $\lim_{s\to\infty} x_s$ is finite.

Suppose $\lim_{t\to\infty} x_t = z = x_s$. Again, the function mapping x to $\lim_{t\to\infty} h(x, t)$ is a computable function, and a characteristic function for a language (say L) in $\mathcal{L}$. Let χ_L denote the characteristic function of L and let $\mathbf{M}(\chi_L)$ denote $\mathbf{M}$'s final conjecture when the input informant is χ_L. We have that $\mathbf{M}(\chi_L) = \mathbf{M}(\chi_L[z])$, as the condition in step 1 did not succeed beyond stage s. Furthermore, using invariant (3), $\mathbf{M}(\chi_L[z])$ makes errors on inputs x, for $x_s \leq x \leq x_s + n$.

From both the above cases, we have that e is a $(u +_o \underline{1})$-correction grammar for a language in $\mathcal{L}$ which is not $\mathbf{Cor}^u\mathbf{InfEx}^n$-identified by $\mathbf{M}$. □

Corollary 18. *For all $n \in \mathbf{N}$, for all $v \in O$ such that v is a notation for a limit ordinal, for all $u <_o v$, $\mathbf{Cor}^v\mathbf{TxtEx} - \mathbf{Cor}^u\mathbf{InfEx}^n \neq \emptyset$.*

Proof. If v is a notation for a limit ordinal and $u <_o v$ then $u +_o \underline{1} <_o v$. But $\mathbf{Cor}^u\mathbf{TxtEx} \subset \mathbf{Cor}^{u+_o\underline{1}}\mathbf{TxtEx}$ by Theorem 17, and obviously $\mathbf{Cor}^{u+_o\underline{1}}\mathbf{TxtEx}$ is included in $\mathbf{Cor}^v\mathbf{TxtEx}$. □

By Theorem 17 and Corollary 18, we have the following Hierarchy Theorem. As a corollary we obtain an even stronger version.

Theorem 19. *For all $u, v \in O$, if $u <_o v$ then for all $n \in \mathbf{N}$*
(a) $\mathbf{Cor}^u\mathbf{TxtEx}^n \subset \mathbf{Cor}^v\mathbf{TxtEx}^n$.
(b) $\mathbf{Cor}^u\mathbf{InfEx}^n \subset \mathbf{Cor}^v\mathbf{InfEx}^n$.

Corollary 20. *For all $v \in O$, $\mathbf{Cor}^v\mathbf{TxtEx} - \bigcup_{u<_o v} \mathbf{Cor}^u\mathbf{InfEx}^n \neq \emptyset$.*

Proof. Let $\mathcal{L}_u$ denote $\mathcal{L}$ as defined in the proof of Theorem 17 (for u as in the statement of Theorem 17). Let $\mathcal{L}'_u = \{\{\langle u, x\rangle \mid x \in L\} \mid L \in \mathcal{L}_u\}$. Consider $\mathcal{L} = \bigcup_{u<_o v} \mathcal{L}'_u$.

Note that, for all u one can effectively (in u) find a $\mathbf{Cor}^{u+_o\underline{1}}\mathbf{TxtEx}$ learner for $\mathcal{L}_u$. Furthermore, for all u, and a $(u +_o \underline{1})$-grammar (for L), one can effectively (from u and the $(u +_o \underline{1})$-grammar) find a v-correction grammar for the set $\{\langle u, x\rangle \mid x \in L\}$. It follows that $\mathcal{L} \in \mathbf{Cor}^v\mathbf{TxtEx}$. Suppose now that $\mathcal{L} \in \bigcup_{u<_o v} \mathbf{Cor}^u\mathbf{InfEx}^n$. Let $u <_o v$ be such that $\mathcal{L} \in \mathbf{Cor}^u\mathbf{InfEx}^n$. Then, $\mathcal{L}'_u \in \mathbf{Cor}^u\mathbf{InfEx}^n$, and thus $\mathcal{L}_u \in \mathbf{Cor}^u\mathbf{InfEx}^n$. This contradicts Theorem 17. □

3.2 The Finite Levels: a Strong Hierarchy

In order to measure the increase in learning power unveiled by the previous results, it is natural to ask: are there classes that can be **TxtEx**-learned by guessing a $(u +_o \underline{1})$-correction grammar but such that no learner guessing u-correction grammars can learn those classes even if it is allowed to conjecture infinitely many syntactically distinct but correct conjectures in the limit? Our next result shows that the answer is positive for all the finite levels of the correction-grammars hierarchy. In the next section we will show that it is *impossible* to obtain the analogous strengthening of the hierarchy for all levels of the $\mathbf{Cor}^u\mathbf{TxtEx}$-hierarchy.

Theorem 21. *For $k \in \mathbf{N}$, $\mathbf{Cor}^{\underline{k+1}}\mathbf{TxtEx} - \mathbf{Cor}^{\underline{k}}\mathbf{InfBc} \neq \emptyset$.*

Proof. Let $\mathcal{L} = \{L \text{ computable} \mid L \neq \emptyset \wedge W^{\underline{k+1}}_{\min(L)} = L\}$.

Clearly, $\mathcal{L} \in \mathbf{Cor}^{\underline{k+1}}\mathbf{TxtEx}$. Now suppose by way of contradiction that $\mathcal{L} \in \mathbf{Cor}^{\underline{k}}\mathbf{TxtBc}$ as witnessed by $\mathbf{M}$.

By the Kleene Recursion Theorem in the system $W^{\underline{k+1}}$, there exists an e such that $W^{\underline{k+1}}_e = \{x \mid h(x, \infty) = 1\}$, where h can be informally defined in stages as follows. We will ensure that $h(x, \cdot)$ changes its mind for any x at most $k+1$-times. Thus, the definition of a function F such that h and F witness $\{x \mid h(x, \infty) = 1\} \in \Sigma^{-1}_{\underline{k+1}}$ is implicit in our construction.

We will also define finite sets $S^0 \subseteq S^1 \ldots$. Intuitively, these sets denote the values of x whose membership in $L = W^{\underline{k+1}}_e$ has already been fixed. In other words, for all s, for all $x \in S^s$, for all $t \geq s$, $h(x, t) = h(x, s)$.

Notation: Let $MC(h, x, s) = \text{card}(\{t < s \mid h(x, t) \neq h(x, t+1)\})$ (thus, $MC(h, x, s)$ denotes the number of mind changes in the sequence $h(x, 0), h(x, 1), \ldots, h(x, s)$). Similarly, let $MCP(i, x, s) = \text{card}(\{t < s \mid \theta^{\underline{k}}_i(x, t) \neq \theta^{\underline{k}}_i(x, t+1)\})$.

We will have the following invariants for each s.
(1) For all $x \notin S^{s+1}$, $MC(h, x, s+1) \leq 1 + MCP(i, x, s)$, where $i = \mathbf{M}(h(\cdot, s+1)[x])$.
(2) For all $x \in S^{s+1}$, $MC(h, x, s+1) \leq k+1$.

(3) If $x \notin S^{s+1}$ and $x \leq s$, then $h(x, s+1) \neq \theta^{\underline{k}}_{\mathbf{M}(h(\cdot,s+1)[x])}(x, s)$.

(4) $S^s \subseteq S^{s+1}$ and for all $x \in S^s$, for all $t \geq s$, $h(x,t) = h(x,s)$.

Here is intuitive idea of the proof. Intuitively, for any particular value x, for the diagonalizing language L, one could ensure that $W^{\underline{k}}_{\mathbf{M}(\chi_L[x])}$ is different from L with respect to membership of x. To do this, one can initially have x to be a member of L (whereas, initially the $\theta^{\underline{k}}_{\mathbf{M}(\chi_L[x])}(x, 0) = 0$). Then, each time $\theta^{\underline{k}}_{\mathbf{M}(\chi_L[x])}(x, \cdot)$ makes a mind change, one could do a corresponding correction for membership of x in L. As $\theta^{\underline{k}}_{\mathbf{M}(\chi_L[x])}(x, \cdot)$ makes at most k mind changes, there would be at most k-corrections (in addition to the initial change from 0 to 1) regarding membership of x in L.

However, for **Bc**-learning we need to do the above kind of diagonalization for infinitely many x, which is problematic, as a diagonalization for larger x needs to be able to fix the value of χ_L for all smaller values of x. To address this, we dovetail all the diagonalizations. For a diagonalizing step for x_1, we spoil a diagonalization for x_2 if: (a) $x_2 < x_1$, and the corrections made for x_2 are $<$ than the corrections made for x_1 or (b) $x_1 < x_2 \leq s$ (where all $x > s$, have not yet been used for any diagonalization). This would allow us to still argue that, for the largest $k' \leq k+1$ such that infinitely many diagonalizing points x used k'-corrections, all points at which k' corrections are made are valid diagonalizing points (modulo some finite number of such points). Furthermore, this would also allow us to argue that the language so constructed is computable. We now proceed formally.

Initially, $h(x,0) = 0$, for all x; $h(x,1) = 0$, for $x < e$ and $h(x,1) = 1$ for all $x \geq e$. Let $S^0 = \emptyset$. Let $S^1 = \{x \mid x \leq e\}$. Clearly, the invariants are satisfied in the beginning. Go to stage $s = 1$ (we start with stage 1, for ease of notation).

```
Begin Stage $s$:
1. If there exists an $x \leq s$, $x \notin S^s$ such that $\theta^{\underline{k}}_{\mathbf{M}(h(\cdot,s)[x])}(x,s) = h(x,s)$, then pick the least such
     $x$ and go to step 2. Otherwise, go to step 3.
   (* For $i = \mathbf{M}(h(\cdot,s)[x])$, note that invariant (1) implies that $MC(h,x,s) \leq 1+MCP(i,x,s-1) \leq 1 + MCP(i,x,s)$. Thus, $\theta^{\underline{k}}_{\mathbf{M}(h(\cdot,s)[x])}(x,s) = h(x,s)$, implies, $MC(h,x,s) \leq$
     $MCP(i,x,s)$. Thus, step 2 modification of $h(x,s+1)$ preserves invariant (1). *)
2. Let $h(x,s+1) = 1 - h(x,s)$. For $y \neq x$, let $h(y,s+1) = h(y,s)$.
   Let $S^{s+1} = S^s \cup \{y < x \mid MC(h,y,s+1) < MC(h,x,s+1)\} \cup \{y \mid x < y \leq s\}$.
   (* Intuitively, $\{y < x \mid MC(h,y,s+1) < MC(h,x,s+1)\}$ is added to $S^{s+1}$, as these $y$'s had
     too few mind changes, and we need to freeze them to maintain computability of $W^{\underline{k+1}}_{e}$. Set
     $\{y \mid x < y \leq s\}$ is added to $S^{s+1}$ as the diagonalizations done up to now for these $y$ are no
     longer valid due to a change in the membership of $x$; thus, to maintain invariant (1) and
     (3) we need to place such $y$ into $S^{s+1}$. *)
   Go to stage $s+1$.
3. For all $x$, let $h(x,s+1) = h(x,s)$, and let $S^{s+1} = S^s$.
   Go to stage $s+1$.
End Stage $s$
```

It is easy to verify that the invariants are satisfied. Also, using invariants (1), (2) we have that $\{x \mid h(x,\infty) = 1\} \in \Sigma^{-1}_{\underline{k+1}}$. Let L be the language for which h is the limiting characteristic function. We will show below that L is computable. Thus, $L \in \mathcal{L}$. We now argue that L is not $\mathbf{Cor}^{\underline{k}}\mathbf{TxtBc}$-identified by $\mathbf{M}$.

Let $k' \leq k+1$ be maximal such that there are infinitely many inputs x for which $MC(h,x,\infty) = \lim_{t\to\infty} MC(h,x,t) = k'$. Let s be the largest stage such that $MC(h,z,s+1) > MC(h,z,s) \geq k'$ for some z. Such a largest stage s exists by the maximality of k'.

Note that if $x > z$, and $MC(h,x,t+1) = k' > MC(h,x,t)$, for some $t > s$, then for all $y < x$, for all $t' > t$, $h(y,t') = h(y,t)$, as either $MC(h,y,t) \geq k'$, or y will be placed in S^{t+1} at stage t. It follows that all such x are not in $\bigcup_{s'\in\mathbf{N}} S^{s'}$, and thus $\theta^{\underline{k}}_{\mathbf{M}(\chi_L[x])}(x) \neq h(x,\infty)$, by invariant (3). Thus, $L \notin \mathbf{Cor}^{\underline{k}}\mathbf{InfBc}$.

We now argue that $L = W^{\underline{k+1}}_e$ is computable. Let w be maximal such that $\mathrm{card}(\{t \mid h(w,t) \neq h(w,t+1)\}) > k'$. Now, for each $x > w$, either $x \in \bigcup_{s\in\mathbf{N}} S^s$, or $\mathrm{card}(\{t \mid h(x,t) \neq h(x,t+1)\}) = k'$. It follows that L is computable, as for each $x > w$, $h(x,t) = h(x,s_x)$ for all $t \geq s_x$, where s_x is the least number such that $x \in S^{s_x}$ or $\mathrm{card}(\{t \leq s_x - 1 \mid h(x,t) \neq h(x,t+1)\}) = k'$. Thus, one can effectively decide membership in L. □

The following corollary is immediate.

Corollary 22. *$\mathbf{Cor}^{\underline{k+1}}\mathbf{TxtBc} - \mathbf{Cor}^{\underline{k}}\mathbf{InfBc} \neq \emptyset$.*

We observe that the proof of Theorem 21 essentially also shows the following.

Theorem 23. *For $k \in \mathbf{N}$, $\mathbf{Cor}^{\underline{k+1}}\mathbf{TxtEx} - \mathbf{Cor}^{\underline{k}}\mathbf{InfEx}^* \neq \emptyset$.*

As an obvious corollary, we have a strong hierarchy with respect to $\mathbf{TxtEx}^*$-learning: $\mathbf{Cor}^{\underline{1}}\mathbf{TxtEx}^* \subset \mathbf{Cor}^{\underline{2}}\mathbf{TxtEx}^* \subset \ldots$.

The proof of Theorem 21 can also be generalized to show the following.

Theorem 24. *For $k \in \mathbf{N}$, for all $m \in \mathbf{N}$, $\mathbf{Cor}^{\underline{k+1}}\mathbf{TxtEx} - \mathbf{Cor}^{\underline{k}}\mathbf{InfBc}^m \neq \emptyset$.*

Proof. (Sketch) Let $\mathcal{L}$ be as defined in Theorem 21. For any L, let $L' = \{\langle x,y\rangle \mid x \in L, y < (m+1)(k+1)\}$. Let $\mathcal{L}' = \{L' \mid L \in \mathcal{L}\}$. It is easy to verify that $\mathcal{L}'$ is in $\mathbf{Cor}^{\underline{k+1}}\mathbf{TxtEx}$ (as $\mathcal{L} \in \mathbf{Cor}^{\underline{k+1}}\mathbf{TxtEx}$). Also, one can show that if $\mathcal{L}' \in \mathbf{Cor}^{\underline{k}}\mathbf{InfBc}^m$, then $\mathcal{L} \in \mathbf{Cor}^{\underline{k}}\mathbf{InfBc}$, contradicting Theorem 21. □

In Section 4 we will show that — surprisingly — the strengthenings of the general hierarchy Theorem 17 given by Theorems 21, 23, and 24 cannot be generalized to the transfinite levels of the hierarchy!

3.3 TxtBc*-Learning Correction Grammars: a Partial Result

We close this section by showing a partial result on the existence of a hierarchy of learning criteria for finite correction grammars with respect to $\mathbf{TxtBc}^*$-learnability. The results of Section 3.2

imply, among other things, the existence of hierarchies $\mathbf{Cor}^{\underline{1}}\mathbf{TxtEx}^* \subset \mathbf{Cor}^{\underline{2}}\mathbf{TxtEx}^* \subset \ldots$ and $\mathbf{Cor}^{\underline{1}}\mathbf{TxtBc}^m \subset \mathbf{Cor}^{\underline{2}}\mathbf{TxtBc}^m \subset \ldots$ for every $m \in \mathbf{N}$ (recall that $\mathbf{TxtEx}^*$ and $\mathbf{TxtBc}$ are incomparable). $\mathbf{TxtBc}^*$-learning is one of the most powerful learning paradigms, allowing the learner to converge in the limit to infinitely many syntactically distinct grammars, each for some finite variant of the target language.

Our next result shows that, in this model, learning $\underline{2}$-correction grammars is more powerful than learning c.e. indices. It is open whether this result generalizes to all finite correction grammars. However, we will show in the next section that there can be no transfinite hierarchy for $\mathbf{Cor}^u\mathbf{TxtBc}^*$ learning above any notation for ω.

Theorem 25. $\mathbf{Cor}^{\underline{2}}\mathbf{TxtBc}^* - \mathbf{Cor}^{\underline{1}}\mathbf{TxtBc}^* \neq \emptyset$.

Proof. Let $L_i = \{\langle i, x\rangle \mid x \in \mathbf{N}\}$. We assume some ordering $\sigma_0, \sigma_1, \ldots$ of finite sequences. We abuse notation slightly to say $\sigma < n$ when $\sigma = \sigma_i$ for some $i < n$. Let $\mathbf{M}_0, \mathbf{M}_1, \ldots$ denote a computable enumeration of total learning machines such that for all $\mathcal{L} \in \mathbf{TxtBc}^*$, there exists an i such that $\mathcal{L} \subseteq \mathbf{TxtBc}^*(\mathbf{M}_i)$ (one can show that such an enumeration exists by essentially using the same argument as for $\mathbf{TxtEx}$ learning done in [33, Lemma4.2.2B]).

Let P_i be the following predicate:

$$(\forall \sigma \mid \mathrm{content}(\sigma) \subseteq L_i)(\exists \tau \mid \mathrm{content}(\tau) \subseteq L_i)(\exists y)(\forall x \geq y)[x \notin W_{\mathbf{M}_i(\sigma\tau)}].$$

Let $\mathcal{L}_i = \{L_i\}$, if P_i; $\mathcal{L}_i = \{S \mid \emptyset \subset S \subseteq L_i, \mathrm{card}(S) < \infty\}$, otherwise. Let $\mathcal{L} = \bigcup_{i \in \mathbf{N}} \mathcal{L}_i$.

By definition $\mathcal{L}_i \not\subseteq \mathbf{TxtBc}^*(\mathbf{M}_i)$: if P_i holds then, $\mathbf{M}_i$ does not have a $\mathbf{TxtBc}^*$-locking sequence for L_i, a necessary requirement for $\mathbf{M}_i$ to $\mathbf{TxtBc}^*$-identify L_i (see [6, 26] for details about locking sequences); if P_i does not hold, then $\mathbf{M}_i$ does not $\mathbf{TxtBc}^*$-identify one of the finite subsets, content(σ), of L_i — the σ which witnessed failure of P_i.

We now show $\mathcal{L} \in \mathbf{Cor}^{\underline{2}}\mathbf{TxtBc}^*$. Given i, define $g_i^n(z, 0) = 0$. $g_i^n(z, t+1) = 1$, if and only if $(\forall \sigma \leq n \mid \mathrm{content}(\sigma) \subseteq L_i)(\exists \tau \leq z \mid \mathrm{content}(\tau) \subseteq L_i)(\exists y \leq z)(\forall x \mid y \leq x \leq t)[x \notin W_{\mathbf{M}(\sigma\tau),t}]$.

Note that $g_i^n(z, \cdot)$, for any z, changes its mind from 1 to 0 at most once, and never from 0 to 1 (except the initial change at $g_i^n(z, 1)$). Thus, grammar for computing g_i^n is — essentially — a 2-correction grammar.

Furthermore, if P_i holds, then for all n, for all but finitely many z, $g_i^n(z, t+1) = 1$ (for any z which exceeds the τ, y witnessing the P_i corresponding to each $\sigma \leq n$). On the other hand, if P_i is false, then let σ witness this failure. For each τ, y, let $t_{\tau,y}$ be the corresponding value such that for some x between y and $t_{\tau,y}$, $x \in W_{\mathbf{M}_i(\sigma\tau), t_{\tau,y}}$. Then, for all $n \geq \sigma$, for all z, for the value of t being the maximal of $t_{\tau,y}$, for $\tau \leq z$ and $y \leq z$, we will have that $g_i^n(z, t+1)$ becomes 0.

Now by outputting g_i^n on inputs of length n, whose content is a non-empty subset of L_i, one has that $\mathcal{L} \in \mathbf{Cor}^{\underline{2}}\mathbf{TxtBc}^*$. $\square$

4 Collapsing Results

In this section we show that Theorem 17 cannot be improved, along the lines of Section 3.2, for the transfinite levels of the $\mathbf{Cor}^u\mathbf{TxtEx}$-hierarchy: every $\mathbf{Cor}^u\mathbf{TxtEx}$-learnable class is already *behaviourally* learnable by a learner outputting grammars that make at most ω mind-changes. In the rest of the section we show analogous collapsing results involving $\mathbf{TxtEx}^*$, $\mathbf{TxtBc}^a$, for $a \in \mathbf{N} \cup \{*\}$.

For proofs of theorems in this section, intuitively, for a given program p for a limiting computable function, we need to find the minimal (1-correction) grammar for the set $\{x \mid \lim_{t\to\infty} \varphi_p(x,t) = 1\}$. To this end, we define g_r^t as in lemma below, which approximates this search. Intuitively, g_r^t denotes r-th approximation for the minimal grammar i such that $W_{i,t} \subseteq \{x \mid \lim_{s\to\infty} \varphi_p(x,s) = 1\}$ and $\{x < t \mid \lim_{s\to\infty} \varphi_p(x,s) = 1\} \subseteq W_i$ (note that, for any fixed t, one can find such an i in the limit). Furthermore, for large enough t, such an i is the minimal grammar for $\{x \mid \lim_{s\to\infty} \varphi_p(x,s) = 1\}$. For technical reasons, we additionally need that the sequence $g_0^t, g_1^t, g_2^t, \ldots$ is non-increasing.

Lemma 26. *Suppose L is a c.e. language. Given a program p for limiting computably computing χ_L, one can effectively (in p, i, t) define numbers g_i^t, such that*

(a) *for all but finitely many t, for all but finitely many i, g_i^t is the minimal φ-grammar for L, and*
(b) *for all t, the sequence $g_0^t, g_1^t, \ldots$ is a non-increasing sequence starting with t (that is, $g_0^t = t$, and $g_{r+1}^t \leq g_r^t$, for all r, t).*

Proof. Given a limiting computable program p, let $g_0^t = t$ and let $g_{r+1}^t = \min(\{g_r^t\} \cup \{i < g_r^t \mid W_{i,t} \subseteq \{x \mid \varphi_p(x, t+r) = 1\}$ and $\{x < t \mid \varphi_p(x, t+r) = 1\} \subseteq W_{i,r}\})$.

Now suppose p is a limiting computable program for χ_L and j is the minimal grammar for L. Let $t > j$ be a large enough number such that for all $i < j$, there exists an $x < t$, such that (i) $\varphi_p(x,\cdot)$ does not make a mind-change beyond t (that is for all $t' > t$, $\varphi_p(x,t') = \varphi_p(x,t)$), and (ii) $x \in W_i$ if and only if $x \in W_{i,t}$ and (iii) W_i and L differ at x.

It is then easy to verify that for all $t' > t$, $\lim_{r\to\infty} g_r^{t'}$ converges to j. □

Corollary 27. *Let w be any O-notation for ω, and $u \in O$. There exists a computable function $h(\cdot,\cdot)$ such that, for any W^u-grammar q for a c.e. language L, for all but finitely many n, $h(q,n)$ is a W^w-grammar for L.*

Proof. $h(q,n)$ is defined as follows. Let p be such that p is a limiting computable program for χ_L. Note that p can be obtained effectively from q. Let g_i^t be as defined in Lemma 26 for p.

Let e_n be such that $\varphi_{e_n}(x,0) = 0$ and $\varphi_{e_n}(x,s+1) = 1$, if and only if $x \in W_{g_s^n,s}$. Thus, by Lemma 26, for all but finitely many n, e_n is a limiting computable program for χ_L, and $\varphi_{e_n}(x,\cdot)$ changes its mind at most $2n+2$ times.

By acceptability of W^w, one can effectively get a W^w-grammar i_n (from e_n, and thus from q, n) for $\{x \mid \lim_{t\to\infty} \varphi_{e_n}(x,t) = 1\}$. We are now done by defining $h(q,n) = i_n$ as above. □

Corollary 28. *Let w be any O-notation for ω, and $u \in O$. There exists a computable function $h(\cdot)$ such that, for any W^u-grammar q for a c.e. language L, $h(q)$ is a W^w-grammar for a finite variant of L.*

Proof. $h(q)$ is defined as follows. Let p be such that p is a limiting computable program for χ_L. Note that p can be obtained effectively from q. Let g_i^t be as defined in Lemma 26 for p.

Let e be such that $\varphi_e(x, 0) = 0$ and $\varphi_e(x, s+1) = 1$, if and only if $x \in W_{g_s^x, s}$. Thus, by Lemma 26, for all but finitely many x, $\varphi_e(x, \infty) = \chi_L(x)$, and $\varphi_e(x, \cdot)$ changes its mind at most $2x + 2$ times.

By acceptability of W^w, one can effectively get a W^w-grammar i (from e, and thus from q) for $\{x \mid \lim_{t\to\infty} \varphi_e(x, t) = 1\}$. We are now done by defining $h(q) = i$ as above. □

Theorem 29. *For all $u \in O$, for all O-notation w for ω,* $\mathbf{Cor}^u\mathbf{TxtEx} \subseteq \mathbf{Cor}^w\mathbf{TxtBc}$.

Proof. Let h be as defined in Corollary 27. Let $\mathbf{M}$ be $\mathbf{Cor}^u\mathbf{TxtEx}$-learner for $\mathcal{L}$. Let $\mathbf{M}'(T[n]) = h(\mathbf{M}(T[n]), n)$. Theorem now follows from Corollary 27. □

We now generalize Theorem 29 to show the following result.

Theorem 30. *For all $u \in O$, for all O-notation w for ω, for all $m \in \mathbf{N}$,* $\mathbf{Cor}^u\mathbf{TxtEx}^{2m} \subseteq \mathbf{Cor}^w\mathbf{TxtBc}^m$.

Proof. This uses a trick similar to the one used by [13] to show $\mathbf{TxtEx}^{2m} \subseteq \mathbf{TxtBc}^m$.

Let h be as defined in Corollary 27. Let *pat* be a computable function such that $W^w_{pat(q,S,S')} = (W^w_q \cup S) - S'$, where S and S' are finite sets.

Let $\mathbf{M}$ be $\mathbf{Cor}^u\mathbf{TxtEx}$-learner for $\mathcal{L}$. Let $\mathbf{M}'(T[n])$ be defined as follows. Let $S_n = \text{content}(T[n])$. Let S'_n be a set of the least m elements in $\{x \mid \theta^u_{\mathbf{M}(T[n])}(x, n) = 1\} - S_n$. Then let $\mathbf{M}'(T[n]) = pat(h(\mathbf{M}(T[n]), n), S_n, S'_n)$.

Suppose T is a text for a language L which is $\mathbf{Cor}^u\mathbf{TxtEx}^{2m}$-identified by $\mathbf{M}$. Suppose $\mathbf{M}(T)$ converges to q. Let $X = W^u_q - L$ and $Y = L - W^u_q$. Now, clearly, for all but finitely many n, the least m elements of X belong to S'_n (if X consists of $\leq m$ elements, then $X \subseteq S'_n$, for all but finitely many n). Also, by Corollary 27, for all but finitely many n, $h(q, n)$ is a W^w-grammar for W^u_q.

Case 1: X contains $> m$ elements. In this case, for all but finitely many n, $W^w_{pat(h(\mathbf{M}(T[n]),n),S_n,S'_n)} = (W^u_q \cup Y) - X'$, where X' contains the least m elements of X. Thus, for all but finitely many n, $pat(h(\mathbf{M}(T[n]), n), S_n, S'_n)$ is a W^w-grammar for a card$(X - X')$ (which is $\leq m$) variant of L.

Case 2: X contains $\leq m$ elements. In this case, for all but finitely many n, $W^w_{pat(h(\mathbf{M}(T[n]),n),S_n,S'_n)} = (W^u_q \cup Y) - S'_n$, where $X \subseteq S'_n$. Thus, for all but finitely many n $pat(h(\mathbf{M}(T[n]), n), S_n, S'_n)$ is a W^w-grammar for an $(m - \text{card}(X))$-variant of L.

From the above cases it follows that $\mathbf{M}'$ $\mathbf{Cor}^w\mathbf{TxtBc}^m$-identifies L. □

Note that we have a hierarchy with respect to $\mathbf{Ex}^*$ for all finite levels, by Theorem 23: $\mathbf{Cor}^{\underline{1}}\mathbf{TxtEx}^* \subset \mathbf{Cor}^{\underline{2}}\mathbf{TxtEx}^* \subset \ldots$. The next result shows that the hierarchy collapses at level ω.

Theorem 31. *For all $u \in O$, for all O-notation w for ω,* $\mathbf{Cor}^u\mathbf{TxtEx}^* \subseteq \mathbf{Cor}^w\mathbf{TxtEx}^*$.

Proof. Let h be as defined in Corollary 28. Let $\mathbf{M}$ be $\mathbf{Cor}^u\mathbf{TxtEx}$-learner for $\mathcal{L}$. Let $\mathbf{M}'(T[n]) = h(\mathbf{M}(T[n]))$. Theorem now follows from Corollary 28. □

The same construction as above gives an analogous collapsing result with respect to $\mathbf{TxtBc}^*$. It should be noted here that it is open whether there exists a hierarchy of learning $\underline{n}$-correction grammars, for $n \in \mathbf{N}$, with respect to the $\mathbf{TxtBc}^*$-model. The following result shows that there can be no hierarchy above the level ω.

Theorem 32. *For all $u \in O$, for all O-notation w for ω, $\mathbf{Cor}^u\mathbf{TxtBc}^* \subseteq \mathbf{Cor}^w\mathbf{TxtBc}^*$.*

One can also show the following.

Theorem 33. *For all $u \in O$, $\mathbf{Cor}^u\mathbf{TxtEx}^* \subseteq \mathbf{TxtBc}^*$.*

Proof. Suppose $\mathbf{M}$ is a $\mathbf{Cor}^u\mathbf{TxtEx}^*$-learner for $\mathcal{L}$. Then, on input $T[s]$, the learner $\mathbf{M}'$ first determines a p_s such that $\{x \mid \lim_{t\to\infty} \varphi_{p_s}(x,t) = 1\} = W^u_{\mathbf{M}(T[s])}$. Note that such a p_s can be found effectively from $T[s]$. Then, $\mathbf{M}'$ determines g^t_i as given by Lemma 26, for $p = p_s$. Then $\mathbf{M}'$ outputs a grammar p'_s for the language $\{x \mid x \in W_{g^s_x}\}$.

Suppose T is a text for L which is $\mathbf{Cor}^u\mathbf{TxtEx}$-identified by $\mathbf{M}$. Then $\mathbf{M}(T)$ converges to a W^u-grammar q for a finite variant of L. Thus, by Lemma 26, for all but finitely many s, for all but finitely many i, g^s_i, as constructed on input $T[s]$ above, is a grammar for W^u_q. Thus, for all but finitely many s, for all but finitely many x, $x \in W_{p'_s}$ if and only if $x \in L$. □

We now prove a collapsing result at level ω for $\mathbf{TxtBc}$-learning correction grammars. We also include anomalies in the treatment, to obtain a stronger result. Note that, by Theorem 24, for every $m \in \mathbf{N}$ there is a hierarchy $\mathbf{Cor}^{\underline{1}}\mathbf{TxtBc}^m \subset \mathbf{Cor}^{\underline{2}}\mathbf{TxtBc}^m \subset \ldots$ of learning *finite* correction grammars with respect to $\mathbf{TxtBc}$-learning with anomalies.

Lemma 34. *Fix $m \in \mathbf{N}$. There exists a computable function f such that for all z and texts T such that $W_z =^{2m} \mathrm{content}(T)$, for all but finitely many t, $f(z, T[t])$ is a grammar for an m-variant of $\mathrm{content}(T)$.*

Proof. This is based on a technique of [13]. By S-m-n theorem, there exists a computable f such that $f(z, T[n])$ may be defined as follows. Let $S_n = \mathrm{content}(T[n])$. Let S'_n be the set of least m elements in $W_{z,n} - S_n$ (if $W_{z,n} - S_n$ contains less than m elements then $S'_n = W_{z,n} - S_n$). Let $W_{f(z,T[n])} = (W_z \cup S_n) - S'_n$.

Let $Y = \mathrm{content}(T) - W_z$ and $X = W_z - \mathrm{content}(T)$. We now consider two cases.

Case 1: $\mathrm{card}(X) \geq m$. Let X' be the set of least m elements of X. Now, for all but finitely many t, $Y \subseteq S_t$ and $S'_t = X'$; thus, $W_{h(z,T[t])}\boldsymbol{\Delta}\mathrm{content}(T) = X - X'$, which is of cardinality at most m.

Case 2: $\mathrm{card}(X) \leq m$. Now, for all but finitely many t, $Y \subseteq S_t$ and $X \subseteq S'_t$; thus, $W_{h(z,T[t])}\boldsymbol{\Delta}\mathrm{content}(T) = S'_t - X$, which is of cardinality at most m.

The Lemma follows from above cases. □

Theorem 35. *For all $u \in O$, for all O-notation w for ω, for all $m \in \mathbf{N}$, $\mathbf{Cor}^u\mathbf{TxtBc}^m \subseteq \mathbf{Cor}^w\mathbf{TxtBc}^m$.*

Proof. Suppose $\mathbf{M}$ is given. Define $\mathbf{M}'$ as follows. Suppose T is a text for a language L which is $\mathbf{Cor}^u\mathbf{TxtBc}^m$-identified by $\mathbf{M}$. Let

$$U_p^t = \{x \mid (\exists t' \geq t)[\theta_p^u(x,t') = 1]\},$$

$$U_{p,r}^t = \{x \mid (\exists t' \leq r)[\theta_p^u(x,t+t') = 1]\}.$$

Intuitively, $(U_p^t)_{t=0\to\infty}$ is an approximation from above for the language W_p^u (that is $U_p^t \supseteq U_p^{t+1}$, for all t, and $\bigcap_{t\in\mathbf{N}} U_p^t = W_p^u$). $(U_{p,r}^t)_{r=0\to\infty}$ is an approximation from below of U_p^t.

Below, let p_i denote $\mathbf{M}(T[i])$. Let

$$g_r^{t,S,S'} = \min(\{t\} \cup \{j' < t \mid j' \in S \ \wedge \ (\forall i \in S')[\mathrm{card}(W_{j',t} - U_{p_i,r}^t) \leq m]\}).$$

Note that for fixed t, S, S', $g_r^{t,S,S'}$ is monotonically non-increasing in r.

Here is an intuitive, but not entirely correct, idea of the proof of the Theorem. Suppose S is a finite collection of grammars, such that S contains the minimal grammar for the input language L, and each member of S is a grammar for a superset of L. Suppose S' is a finite collection of numbers i such that p_i is a u-correction grammar for a superset of L, where at least one such p_i is a u-correction grammar for an m-variant of L. For S, S' as above, for large enough t, $\lim_{r\to\infty} g_r^{t,S,S'}$ would converge to a grammar $j' \leq$ minimal grammar for L, such that j' is a grammar for a $2m$-variant of L. This would suffice to get a $\mathbf{TxtBc}^m$-learner using techniques similar to that used in Proof of Theorem 30. We are not exactly able to get (even in the limit) S, S' as required above. *However*, as the analysis below shows using appropriate approximations S_t, S_t' suffices (see statement (*) below). We now proceed formally.

Now for any t, let $S_{t,r} = \{i \leq t \mid \mathrm{content}(T[t]) \subseteq W_{i,r}\}$, and $S_{t,r}' = \{i \leq t \mid \mathrm{card}(\mathrm{content}(T[t]) - U_{p_i,r}^t) \leq m\}$. It is easy to verify that $S_{t,r}$ and $S_{t,r}'$ are monotonically non-decreasing (in r) and bounded in cardinality by $t+1$. Let $S_t = \lim_{r\to\infty} S_{t,r} = \{i \leq t \mid \mathrm{content}(T[t]) \subseteq W_i\}$, and $S_t' = \lim_{r\to\infty} S_{t,r}' = \{i \leq t \mid \mathrm{card}(\mathrm{content}(T[t]) - U_{p_i}^t) \leq m\}$.

We will later show that,

(*) for all but finitely many t, $\lim_{r\to\infty} g_r^{t,S_t,S_t'}$ converges to a grammar $z \leq j$ such that $W_z =^{2m} L$, where j is the minimal grammar for L. Here, z may be different for different (large enough) t's; however, all these z's are $\leq j$.

Let $i_r^t = g_r^{t,S_{t,r},S_{t,r}'}$. Note that $\lim_{r\to\infty} i_r^t = \lim_{r\to\infty} g_r^{t,S_{t,r},S_{t,r}'} = \lim_{r\to\infty} g_r^{t,S_t,S_t'}$. Thus, for all but finitely many t, $\lim_{r\to\infty} i_r^t = i^t$ is a grammar for a $2m$-variant of L, and is bounded by the minimal grammar for L. Furthermore, for a fixed t, $\mathrm{card}(\{r \mid i_r^t \neq i_{r+1}^t\}) \leq (t+1)*(t+1)*(t+1)$.

Let f be as defined in Lemma 34. Define p_t' such that $\varphi_{p_t'}(x,0) = 0$, and for $t > 0$, $\varphi_{p_t'}(x,r) = 1$, if and only if $x \in W_{f(i_r^t,T[t]),r}$, and then let $\mathbf{M}'(T[t])$ be a W^w-grammar for the language $\{x \mid \varphi_{p_t'}(x,\infty) = 1\}$. By acceptability of W^w, such a W^w-grammar can be obtained effectively from p_t'. Now for all but finitely many t, for all but finitely many r, i_r^t is a grammar for a $2m$-variant of L which is bounded by the minimal grammar for L. Thus, by Lemma 34, for all but finitely many t, for all but finitely many r, $f(i_r^t, T[t])$ is a grammar for an m-variant of L. Theorem thus follows from the below proof of (*).

We now show that (*) holds. Let j be the minimal grammar for L. Let k be minimal such that p_k is a u-correction grammar for an m-variant of L. Let t be so large that the following holds.

(a) $t > j$, (thus $j \in S_t$).
(b) $t > k$, (thus $k \in S'_t$).
(c) for all $i < j$, such that $W_i \not\supseteq L$, content$(T[t]) \not\subseteq W_i$. Thus, $i \in S_t$ and $i < j$, implies $L \subset W_i$.
(d) for all i such that card$(L - \{x \mid \theta^u_{p_i}(x, \infty) = 1\}) > m$, we have $i < t$ and card(content$(T[t]) - U^t_{p_i}) > m$; thus, $i \notin S'_t$. Hence, S'_t consists only of i such that card$(L - W^u_{p_i}) \leq m$. Thus, $(\forall i \in S'_t)[\text{card}(W_{j,t} - U^t_{p_i}) \leq m]$.
(e) for all $i < j$, such that $W_i \supset L$ and card$(W_i - L) > 2m$, there exists at least $(m + 1)$ x's in $W_{i,t}$ such that, for all $t' \geq t$, $\theta^u_{p_k}(x, t') = 0$.

It follows from (a) and (d) that $\lim_{r\to\infty} g_r^{t,S_t,S'_t}$ converges to a grammar $i \leq j$. Furthermore, using (b), (c), and (e), it follows that $W_i \supseteq L$, and card$(W_i - L) \leq 2m$. □

5 Other Results

5.1 Trade-offs with Anomalies

In Section 4 we have proved a number of collapsing results for learning correction grammars. These results have shown that the transfinite hierarchy of learning correction grammars criteria is specific to the **TxtEx** model of learning. Now we consider another question: are there trade-offs between learning c.e. indices, learning u-correction grammars and the number of anomalies allowed? The following theorem shows that learning c.e. indices with more anomalies gives dramatic increase of learning power compared to learning correction grammars of any ordinal complexity with less anomalies allowed. The proofs are straightforward lifts from other contexts (see, e.g., [12]).

Theorem 36. *For all $u \in O$, for all $n \in \mathbf{N}$,*
(a) $\mathbf{TxtEx}^{2n+1} - \mathbf{Cor}^u\mathbf{TxtBc}^n \neq \emptyset$.
(b) $\mathbf{TxtBc}^{n+1} - \mathbf{Cor}^u\mathbf{TxtBc}^n \neq \emptyset$.
(c) $\mathbf{TxtEx}^{n+1} - \mathbf{Cor}^u\mathbf{TxtEx}^n \neq \emptyset$.
(d) $\mathbf{TxtEx}^* - \bigcup_{n\in\mathbf{N}} \mathbf{Cor}^u\mathbf{TxtBc}^n \neq \emptyset$.
(e) $\mathbf{Cor}^u\mathbf{InfEx}^* \subseteq \mathbf{Cor}^u\mathbf{InfBc}$.
(f) *For w an O-notation for ω, $\mathcal{E} \in \mathbf{Cor}^w\mathbf{InfBc}$.*

Proof. The class $\mathcal{L} = \{L \in \mathcal{E} \mid L =^{2n+1} \mathbf{N}\}$ witnesses (a) and (b). The class $\mathcal{L} = \{L \in \mathcal{E} \mid L =^{n+1} \mathbf{N}\}$ witnesses (c). The class $\mathcal{L} = \{L \in \mathcal{E} \mid L =^* \mathbf{N}\}$ witnesses (d). (e) can be proven by patching the input, along the same lines as $\mathbf{InfEx}^* \subseteq \mathbf{InfBc}$ (see [13]). Harrington's proof of $\mathcal{R}$, the class all computable functions, being in $\mathbf{Bc}^*$ (see [15]) can also be used to show (f). We omit the details. □

Essentially, the anomaly hierarchies $\{\mathbf{TxtI}^n\}_{\underline{n}\in\mathbf{N}}$, with $\mathbf{I}\in\{\mathbf{Ex},\mathbf{Bc}\}$ are very stable:[12] e.g., (b) and (c) show that the extra learning power of allowing *one* more anomaly in the final conjecture overplays the power of learning correction grammars of *any* transfinite ordinal complexity u.

5.2 Learning Succinct Correction Grammars

In scientific inference, parsimony of explanations is considered highly desirable. Grammar size is one of many ways to measure parsimony of grammars [21, 27]. It is known, for computability-theoretic inductive inference, that requiring the final and correct grammars to be minimal size [42] is highly restrictive on inferring power [21, 22] (and that the resultant inferring or learning power is dependent on which acceptable programming system is employed). Also known is the adverse effect on learning power of requiring the final and correct grammars to be merely within a computable parsimony factor of minimal size grammars [28, 16] (but that the resulting inferring power is independent of the underlying acceptable programming system [21]). Hence, parsimony restrictions of even the weaker kind described just above limit inferring power.

For c.e. L, let $\mathrm{MinGram}(L)$ be the minimal i such that $W_i = L$. For $\mathcal{L}\in\mathbf{TxtEx}$ as witnessed by $\mathbf{M}$, if there is a computable function g such that, for every $L\in\mathcal{L}$, for all texts T for L, $\mathbf{M}(T)\le g(\mathrm{MinGram}(L))$, then we say $\mathcal{L}\in\mathbf{TxtMEx}$ (as witnessed by $\mathbf{M}$ and g). In this setting we call g a *parsimony factor*. The final grammars of a **TxtMEx**-learner are, in a certain sense, *nearly minimal-size*. Kinber [28] was the first to show that $\mathbf{TxtMEx}\subset\mathbf{TxtEx}$. For example, the class $\mathrm{Zero}^* = \{\{\langle x, f(x)\rangle \mid x\in\mathbf{N}\} \mid f$ is a computable function and $(\forall^\infty x)[f(x)=0]\}$ witnesses this separation [28]. Chen [16] later showed that Zero^* is not even in $\mathbf{TxtMEx}^n$ for every $n\in\mathbf{N}$.

By contrast, the next Theorem shows that the class Zero^* is learnable with nearly-minimal-size final conjectures if the learner uses basic correction grammars (i.e., $\underline{2}$-correction grammars) instead of standard grammars.

Theorem 37. *There exists a learner* $\mathbf{M}$ *which* $\mathbf{Cor}^{\underline{2}}\mathbf{TxtEx}$*-identifies* Zero^**, and for some computable function* g*, for all texts* T *for* $L\in\mathrm{Zero}^*$*,* $\mathbf{M}(T)\le g(\mathrm{MinGram}(L))$*.*

Proof. Let e_1 be minimal grammar for $Y=\{\langle x,0\rangle \mid x\in\mathbf{N}\}$. Let h be computable, 1–1 and increasing function such that $h(i,j)$ is a $W^{\underline{2}}$ grammar for $W_i\boldsymbol{\Delta}W_j = (W_i\cup W_j)-(W_i\cap W_j)$ (as one can effectively, from i and j, find grammars for $W_i\cup W_j$ and $W_i\cap W_j$, using s-m-n theorem [40] such an h can be easily constructed).

Define a learner $\mathbf{M}$ as follows. Given a text T for $L\in\mathrm{Zero}^*$, $\mathbf{M}$ finds, in the limit, the finite set S of elements in $L\boldsymbol{\Delta}Y$ (note that this S can be determined, in the limit, as for all $x\in\mathbf{N}$, L contains a unique y such that $\langle x,y\rangle\in L$). From this S, $\mathbf{M}$ computes, in the limit, the minimal grammar e_2 for S. (Note that for finite sets, one can determine minimal φ-grammars in the limit). Then $\mathbf{M}$ on T converges to $h(e_1,e_2)$.

[12] An inspection of the original proofs shows that virtually any criterion that requires convergence in the limit to some kind of correct grammar for the target language can be beaten by allowing one more anomaly in the limit.

For $L \in \text{Zero}^*$, it is easy to verify that $\text{MinGram}(L\boldsymbol{\Delta}Y) \le h'(\text{MinGram}(L))$, for some computable function h', and thus $\mathbf{M}(T) \le h(e_1, h'(\text{MinGram}(L))) \le g(\text{MinGram}(L))$, for some computable function g. $\square$

Note that in Theorem 37 above, the final conjecture of $\mathbf{M}$ is within a computable factor of minimal grammars, not of minimal correction grammars. As a corollary, we have that one can learn very succinct correction grammars, when compared to ordinary grammars.

This can also be compared with the result mentioned above by Case and Royer [14]: correction grammars for some c.e. languages can be computable-in-$\emptyset^{(1)}$ more succinct than ordinary grammars for them. However, it should be observed that the latter result is not the reason for our Theorem 37. The fact that no machine can learn nearly minimal-size c.e. indices for Zero^* but some machine can learn correction grammars that are within a computable factor of the minimal c.e. index has to do with the learner's inability to get hold of succinct grammars, rather than with the existence of succinct grammars.

6 Conclusions

In the present paper we have investigated a new learning paradigm in which the learner outputs correction grammars instead of ordinary c.e. indices.

We have shown that learning correction grammars of larger and larger complexity enhances learning power over learning c.e. indices. *Perhaps*, humans may be making use of this by learning correction grammars instead of ordinary grammars. In the context of **TxtEx**-learning, an infinite hierarchy of more and more powerful learning criteria is obtained (Theorem 19). The increase in learning power is measured by notations for constructive transfinite ordinals used for algorithmic count-down of corrections. If u and v are notations in O for transfinite ordinals and u is smaller than v in the notation system, then one can learn more — in the **TxtEx**-model — by conjecturing v-correction grammars than by conjecturing u-correction grammars. Some partly hedged, "information-theoretic" intuition is given for possibly explaining these results and other such mathematical results.

For correction grammars with a fixed finite number of corrections, we have shown that the hierarchy can be strengthened: there are classes of languages that are **TxtEx**-learnable with $\underline{k+1}$-correction grammars such that no **Bc**-learner can learn those classes using $\underline{k}$-correction grammars, not even from the characteristic function of the graph of the language.

Surprisingly, we have shown that some interesting collapsing phenomena occur at the first transfinite level: for example, every class in $\mathbf{Cor}^u\mathbf{TxtEx}$ for $u \in O$ is already in $\mathbf{Cor}^w\mathbf{TxtBc}$ with w a notation for ω. The same is also true for every class in $\mathbf{Cor}^u\mathbf{TxtBc}$.

From the Cognitive Science perspective, our hierarchy results can be read as *suggesting* that the drawback of using correction grammars instead of standard ones (i.e., the need of self-corrections) may be compensated by an increase of learning power.

In this respect we note that many of our results can be adapted to vacillatory learning [11]. $\mathbf{TxtFex}^a_b$ with $b, a \in \mathbf{N} \cup \{*\}$ is the vacillatory learning criterion allowing the learner to vacillate between $\le b$ a-variants of the target language, $\mathbf{Cor}^u\mathbf{TxtFex}^a_b$ is the version using correction grammars, $\mathbf{Cor}^u\mathbf{InfFex}_b^a$ is the version with informants. The hierarchy proof from [11] can be adapted to show that we have a hierarchy $\mathbf{Cor}^u\mathbf{Fex}_1^a \subset \mathbf{Cor}^u\mathbf{Fex}_2^a \subset \ldots$ of vacillatory learning correction grammars. A modification of the proof of Theorem 17 can be used to show that $\mathbf{Cor}^{u+_o\underline{1}}\mathbf{TxtEx} - \mathbf{Cor}^u\mathbf{InfFex}_*^n \neq \emptyset$. Similarly, proof of Theorem 21 (see also Theorem 23) yields $\mathbf{Cor}^{\underline{n+1}}\mathbf{TxtEx} - \mathbf{Cor}^{\underline{n}}\mathbf{InfFex}_*^* \neq \emptyset$. Proof of Theorem 31 yields $\mathbf{Cor}^u\mathbf{InfFex}_b^* \subseteq \mathbf{Cor}^w\mathbf{InfFex}_b^*$ and $\mathbf{Cor}^u\mathbf{TxtFex}_b^* \subseteq \mathbf{Cor}^w\mathbf{TxtFex}_b^*$, for w a notation for ω.[13]

The question whether there exists a hierarchy of learning *finite* correction grammars with respect to $\mathbf{TxtBc}^*$-learning is open. We showed that $\mathbf{TxtBc}^* \subset \mathbf{Cor}^2\mathbf{TxtBc}^*$ and conjecture that the general result will require different methods of proofs.

Also of interest for the future is the investigation of "complexity" results along the lines of Theorem 37.

7 Acknowledgements

Preliminary version of the paper appeared in [10]. We thank the referees of COLT' 2007 and the *Journal of Symbolic Logic* for several helpful comments.

References

1. A. Ambainis, J. Case, S. Jain and M. Suraj. Parsimony hierarchies for inductive inference. *Journal of Symbolic Logic*, 69(1):287–327, 2004.
2. D. Angluin. Inductive inference of formal languages from positive data. *Information and Control*, 45:117–135, 1980.
3. J. Ash and J. F. Knight. Recursive structures and Ershov's hierarchy. *Mathematical Logic Quarterly*, 42:461–468, 1996.
4. G. Baliga, J. Case, S. Jain, and M. Suraj. Machine learning of higher order programs. *Journal of Symbolic Logic*, 59(2):486–500, 1994.
5. J. M. Bārzdiņš. Inductive inference of automata, functions and programs. In R. D. James, editor, *Proceedings of the 20th International Congress of Mathematicians*, pages 455–460, 1974.
6. L. Blum and M. Blum. Toward a mathematical theory of inductive inference. *Information and Control*, 28:125–155, 1975.
7. M. Blum. A machine-independent theory of the complexity of recursive functions. *Journal of the ACM*, 14:322–336, 1967.
8. W. Buchholz. Proof-theoretic analysis of termination proofs. *Annals of Pure and Applied Logic*, 75:57–65, 1995.
9. M. Burgin. Grammars with prohibition and human-computer interaction. In J. Hill, editor, *Proceedings of the 2005 Business and Industry Symposium and the 2005 Military, Government, and Aerospace Simulation Symposium*, pages 143–147. Society for Modeling and Simulation, 2005.

[13] For $u > 1$, $u, b \in \mathbf{N}$, the exact relationship among different $\mathbf{Cor}^u\mathbf{InfFex}_b$ is open.

10. L. Carlucci, J. Case and S. Jain. Learning correction grammars. In N. Bshouty and C. Gentile, editors, *Learning Theory: 20th Annual Conference on Learning Theory, Proceedings*, pages 203–217. Lecture Notes in Artificial Intelligence 4539. Spring Verlag, 2007.
11. J. Case. The power of vacillation in language learning. *SIAM Journal on Computing*, 28(6):1941–1969, 1999.
12. J. Case, S. Jain and A. Sharma. On learning limiting programs. *International Journal of Foundations of Computer Science*, 3(1):93–115, 1992.
13. J. Case and C. Lynes. Machine inductive inference and language identification. In M. Nielsen and E. M. Schmidt, editors, *Proceedings of the 9th International Colloquium on Automata, Languages and Programming*, pages 107–115. Lecture Notes in Computer Science 140. Springer Verlag, 1982.
14. J. Case and J. Royer. Program size complexity of correction grammars. *Working paper*, 2008.
15. J. Case and C. Smith. Comparison of identification criteria for machine inductive inference. *Theoretical Computer Science*, 25:193–220, 1983.
16. K. Chen. Tradeoffs in inductive inference of nearly minimal sized programs. *Information and Control*, 52:68–86, 1982.
17. R. L. Epstein, R. Haas and R. Kramer. Hierarchies of sets and degrees below $\mathbf{0}'$. In M. Lerman, J. H. Schmerl and R. I. Soare, editors, *Logic Year 1979-1980*, pages 32–48. Lecture Notes in Mathematics 859. Springer Verlag, 1981.
18. Y. L. Ershov. A hierarchy of sets I. *Algebra and Logic*, 7:23–43, 1968.
19. Y. L. Ershov. A hierarchy of sets II. *Algebra and Logic*, 7:212–232, 1968.
20. Y. L. Ershov. A hierarchy of sets III. *Algebra and Logic*, 9:20–31, 1970.
21. R. Freivalds. Minimal Gödel numbers and their identification in the limit. In, Jiri Becvár, editor, *Mathematical Foundations of Computer Science, 4th Symposium, Proceedings*, pages 219–225. Lecture Notes in Computer Science 32. Springer Verlag 1975.
22. R. Freivalds. Inductive inference of minimal programs. In M. Fulk and J. Case, editors, *Proceedings of the Third Annual Workshop on Computational Learning Theory*, pages 3–20. Morgan Kaufmann Publishers, Inc., 1990.
23. R. Freivalds and C. Smith. On the role of procrastination in machine learning. *Information and Computation*, 107(2):237–271, 1993.
24. E. M. Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.
25. J. Hopcroft and J. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, 1979.
26. S. Jain, D. Osherson, J. Royer and A. Sharma. *Systems that Learn: An Introduction to Learning Theory.* MIT Press, second edition, 1999.
27. S. Jain and A. Sharma. Program size restrictions in computational learning. *Theoretical Computer Science*, 127:351–386, 1994.
28. E. B. Kinber. On the synthesis in the limit of almost minimal Gödel numbers. In, J. M. Bārzdiņš, editor, *Theory Of Algorithms and Programs*, Latvian State University, Riga, 1:221–223, 1974.

29. S. C. Kleene. On notation for ordinal numbers. *Journal of Symbolic Logic*, 3:150–155, 1938.
30. S. C. Kleene. On the forms of predicates in the theory of constructive ordinals. *American Journal of Mathematics*, 66:41–58, 1944.
31. S. C. Kleene. On the forms of predicates in the theory of constructive ordinals (second paper). *American Journal of Mathematics*, 77:405–428, 1955.
32. M. Machtey and P. Young. *An Introduction to the General Theory of Algorithms.* North Holland, New York, 1978.
33. D. Osherson, M. Stob, and S. Weinstein. *Systems that Learn: An Introduction to Learning Theory for Cognitive and Computer Scientists.* MIT Press, 1986.
34. D. Osherson and S. Weinstein. Criteria for language learning. *Information and Control*, 52:123–138, 1982.
35. D. Osherson and S. Weinstein. A note on formal learning theory. *Cognition*, 11:77–88, 1982.
36. D. Osherson and S. Weinstein. Learning theory and natural language. *Cognition*, 17:1–28, 1984.
37. R. Peter. *Recursive Functions.* Academic Press, New York, 1967.
38. S. Pinker. Formal models of language learning. *Cognition*, 7:217–283, 1979.
39. M. Rathjen. The realm of ordinal analysis. In S. Cooper and J. Truss, editors, *Sets and Proofs*, pages 219–279. Cambridge University Press, 1999.
40. H. Rogers. *Theory of Recursive Functions and Effective Computability.* McGraw-Hill, 1967. Reprinted by MIT Press in 1987.
41. J. Royer and J. Case. *Subrecursive Programming Systems: Complexity & Succinctness.* Birkhäuser, 1994.
42. M. Schaefer. A guided tour of minimal indices and shortest descriptions. *Archive for Mathematical Logic*, 18:521–548, 1998.
43. G. Takeuti. *Proof Theory.* Second Edition, North Holland, 1987.
44. A. Weiermann. Proving termination for term rewriting systems. In E. Börger, G. Jäger, H. K. Büning and M. M. Richter editors, *Computer Science Logic, 5th Workshop*, pages 419–428. Lecture Notes in Computer Science 626, 1992.
45. K. Wexler and P. Culicover. *Formal Principles of Language Acquisition.* MIT Press, 1980.
46. K. Wexler. On extensional learnability. *Cognition*, 11:89–95, 1982.