Generalized Notions of Mind Change Complexity

Arun Sharma *
School of Computer
Science and Engineering
University of New South Wales
Sydney, NSW, 2052, Australia
arun@cse.unsw.edu.au

Frank Stephan †
Mathematisches Institut
Universität Heidelberg
Im Neuenheimer Feld 294
69120 Heidelberg, Germany
fstephan@math.uni-heidelberg.de

Yuri Ventsov ‡
School of Computer
Science and Engineering
University of New South Wales
Sydney, NSW, 2052, Australia
ventsov@cse.unsw.edu.au

Abstract

Speed of convergence in Gold's identification in the limit model can be measured by deriving bounds on the number of mind changes made by a learner before the onset of convergence. Two approaches to date are bounds given by constants (referred here as Type 1) and bounds expressed as constructive ordinals (referred as Type 2). The use of ordinals has recently been successfully employed to measure the mind change complexity of learning rich concept classes such as unions of pattern languages, elementary formal systems and logic programs. Motivated by these applications, the present work introduces two more general approaches to bounding mind changes. These are based on counting by going down in a linearly ordered set (Type 3) and on counting by going down in a partially ordered set (Type 4). In both cases the set must not contain infinite, descending, and computable sequences. These four types of mind changes yield a hierarchy and there are identifiable classes that cannot be learned with the most general mind change bound of Type 4.

It is shown that existence of Type 2 bound is equivalent to the existence of a learning algorithm which converges on every (also noncomputable) input function and the existence of Type 4 is shown to be equivalent to the existence of a learning algorithm which converges on every computable function. A partial characterization of Type 3 yields a result of independent interest in recursion theory.

The interplay between mind change complexity and choice of hypothesis space is investigated. It is established that for certain concept classes, a more expressive hypothesis space can sometimes reduce mind change complexity of learning these classes. This insight is likely to be useful in design of empirical learning systems.

The notion of mind change bound for *behaviorally correct learning* is indirectly addressed by employing the above four types to restrict the number of predictive errors of commission in *finite error next value learning* (NV$''$) — a model equivalent to behaviorally correct learning. Again, natural characterizations for Type 2 and Type 4 bounds are derived. Their naturalness is further illustrated by characterizing them in terms of branches of uniformly recursive families of binary trees.

1 Introduction

The learning model considered in this paper is Gold's identification in the limit [4, 7] usually referred to as Ex-identification. In this model, a learner M — a computable device — receives increasing segments of the values, $f(0), f(1), \ldots$, of a computable function f one element at a time. As it is receiving these values, M outputs a sequence of computer programs $p_0, p_1, \ldots$. M is said to Ex-*identify* f just in case the sequence of programs converges to a program for f. Ex denotes the class of collections of functions, S, for which there is a machine that learns each function in S.

Identification in the limit is a very general model which has turned out to be useful in analyzing learnability of rich concept classes (e.g., those which cannot be expressed in propositional logic) for which only negative results can be obtained using more restricted learning models. However, it has long been acknowledged that a more realistic model requires some additional information about the speed of convergence. One approach to measuring the speed of convergence is requiring a bound on the number of mind changes a learner is allowed to make before the onset of convergence.

The obvious approach to bounding the number of mind changes is requiring that the learner make no more than a given fixed number of mind changes before it con-

*Supported by the Australian Research Council grant A49600456.

†Supported by the Deutsche Forschungsgemeinschaft (DFG) grant Am 60/9-1.

‡Supported by the Australian Research Council grant A49600456.

verges. This approach was first considered by Barzdins and Freivalds [3] and investigated extensively by Case and Smith [5]. However, this notion turns out to be too restrictive as it places the same constant bound on each concept in the class being learned. It does not allow the possibility that a more "complex" concept may require a larger bound on the number of mind changes than a "simpler" concept.

Motivated by such limitations, Freivalds and Smith [6] introduced the use of constructive ordinals [23] to bound the number of mind changes. This notion turns out to be more natural as it allows a learner to decide on a mind change bound based on the data received. This approach also allows the possibility that the learner may wish to revise its current bound based on additional data. These ideas are best illustrated with the help of a few examples. A class of concepts that can be identified with a bound of ω means that there exists a learner that, after examining some element(s) of the concept, announces an upper bound on the number of mind changes it will make before the onset of successful convergence. Proceeding on, a bound of 2ω means that there is a learner that after examining some element(s) of the concept announces an upper bound on the number of mind changes, but reserves the right to revise this upper bound once. Similarly, in the case of 3ω, the machine reserves the right to revise its upper bound twice, and so on. A bound of ω^2 means that the machine announces an upper bound on the number of times it may revise its conjectured upper bound on the number of mind changes, and so on.

The use of ordinals to model mind change complexity has recently been applied by Jain and Sharma [9] and by Ambainis, Jain and Sharma [1] to measure the complexity of learning pattern languages, unions of pattern languages and elementary formal systems (a logic programming system on strings). More recently, these techniques have directly been applied to measure the mind change complexity of learning logic programs from positive facts and from both positive and negative facts [10].

Motivated by the above applications, this paper asks the question: "Are there more general ways of bounding the number of mind changes than ordinals?" We give an affirmative answer to this question and provide two natural extensions. We describe these notions as part of a unifying framework for mind change bounds.

The general idea of identification in the limit with bounded mind changes is that during the inference process, a learner is allowed to replace its old hypothesis by a new one. To respect this bound, the learner may be thought of as being equipped with a counter such that a mind change is only permitted if the counter at the same time is decreased, i.e., changed to a lower number. In the concrete case of a natural number as a mind change bound, the counter starts with a natural number, say 5, and is decremented every time a mind change takes place, e.g., from 5 to 4, next time from 4 to 3 until it reaches 0 and can no longer be decreased at which point the machine should have emitted its converged hypothesis. The more abstract realization of Freivalds and Smith [6] uses counters which range over constructive ordinals.

We propose that there are at least two additional levels which can be shown to lie between identification with ordinal mind change bound and identification in the limit with no mind change bound. We define these four types via the set Q of values which a mind change counter may take. It is required that Q be enumerable and the ordering of the values of Q be recursive in Q.

First type: constant bound.
Q is a finite set (e.g., any set like $\{0, 1, \ldots, c\}$ with the ordering of the natural numbers). This is the original notion of Barzdins and Freivalds [3].

Second type: ordinal bound.
The set Q is well-ordered; that means Q has no infinite descending chain. This is the notion of Freivalds and Smith [6].

Third type: linear bound.
The set Q is a linear ordered set which does not contain any infinite descending recursive sequence. (But it may contain nonrecursive ones.)

Fourth type: general bound.
The set Q is only partially ordered but does not have an infinite descending recursive sequence.

The reader may view the lack of any restriction on the mind change bound as the fifth type. The reader should also note that for the first, second and third types, Q can always be chosen as a set of rationals with its natural order. In this work, we investigate mind change complexity for both functions and languages, but report on only function learning results due to space restrictions. We briefly highlight some of the results before a more formal presentation.

Properties and Characterizations of Mind Change Bound Types: The first natural question is if the four types of mind change bounds form a hierarchy. We answer this question in the affirmative. We also establish that these four types are closed under union in the sense that if two classes S_1 and S_2 are learnable respecting a certain type of mind change bound, then the class $S_1 \cup S_2$ are learnable respecting a mind change bound of the same type.

A number of interesting characterizations gives evidence that these generalized notions of mind change bounds are quite natural. For example, we establish that a class of functions can be learned with a mind change bound of the fourth type iff it can be learned by a learner which converges on every computable function. More interestingly, we also observe that a class of functions can be learned with a mind change bound of the second type iff it can be learned by a learner which converges on every function — computable or noncomputable. This later result is a counterpart of a related result established by Ambainis, Jain and Sharma [1] for learning r.e. languages with ordinal mind change bound.

Unfortunately, there is no such nice characterization for the the mind change bound of the third type. However, we are able to establish a sufficient condition and slightly weak necessary condition which leads to an insight that is of independent interest in recursive function theory. We show that if a class S of functions is learnable by a machine that converges on every function which is computable relative to the halting problem, then S can be learned with a mind change bound of the third type. As a counterpart to this sufficient condition, we also establish a necessary condition that if a machine M learns a class with a mind change bound of the third type, then M converges on every function of hyperimmune-free Turing degree.

Mind Changes and Hypothesis Space: Our next set of results is about the interplay between mind change bounds and hypothesis space. We establish that the choice of a richer hypothesis space can lead to mind change complexity advantages. As an extreme case of this phenomena, we establish that there is a collection of functions S that can be learned with a mind change bound of the first type with respect to the standard hypothesis space of an acceptable programming system, but if some machine M learns S with respect to a class preserving hypothesis space then M does not respect mind change bound of any type.

However, we also show that mind change bounds of the second and fourth type are well behaved with respect to class preserving learning in the following way. Suppose $\{f_0, f_1, \ldots\}$ and $\{g_0, g_1, \ldots\}$ are any two uniformly recursive enumeration for a class of functions S. Now, if S can be learned with respect to the hypothesis space $\{f_0, f_1, \ldots\}$ respecting a mind change bound of the second (fourth) type, then S can also be learned with respect to the hypothesis space $\{g_0, g_1, \ldots\}$ respecting a mind change bound of the same type.

Mind Changes and BC Learning: The notion of convergence in Ex-identification is *syntactic*, i.e., the learner is required to output the same hypothesis from some point onwards. Behaviourally correct (BC) identification (see Case and Smith [5]) is a generalization of Ex learning in which the notion of convergence is *semantic*, i.e., the learner is required to output hypotheses, which may differ syntactically from each other, for the same concept after some point onwards. This weakened definition of convergence allows larger collections of functions to be learned (see Case and Smith [5]).

Instead of directly attempting to adapt mind change complexity to BC learning, we incorporate mind change bounds in the model of "finite error next value prediction" NV'' — a model which Podnieks [22] proved to be equivalent to BC (see also Case and Smith [5]). In this model a learner is allowed to be partial recursive. We say that learner M is successful on a class of functions S just in case for each $f \in S$, M predicts f almost everywhere, that is, for almost all x M is defined on the initial segment $f(0)f(1)\ldots f(x)$ and outputs the value $f(x+1)$. The learner M may either diverge or make false predictions on finitely many initial segments of each $f \in S$ as well as on every initial segment for functions outside the class S.

In the context of Ex learning, a mind change bound gives a "measure" of the "incorrect" activity that a learner indulges in before it is successful. For NV'' (and hence equivalently for BC), the portion of the input where the learner makes a false prediction or no prediction at all may be viewed similarly. Hence, it is not inappropriate to view a false prediction or the lack of a prediction as a mind change. We distinguish between two kinds of mind changes:

- If a learner is defined on an initial segment but makes a false prediction, we count it as a *hard* mind change.
- If M is undefined then we count it as a *weak* mind change.

We employ the four types of bounding mind changes to restrict the number of hard mind changes of a learner[1].

As a further proof of the naturalness of the second and the fourth types of mind change bounds, we observe that for BC-identification, a similar characterization to that of Ex-identification holds. Surprisingly, the notions of Ex-identification and BC-identification turn out to be incomparable for all four types of mind change bounds.

We also provide characterizations of the second and fourth types of mind change bound, for both Ex and BC, in terms of branches of uniformly recursive families of binary trees in the style of Merkle and Stephan [19].

Language Learning and Mind Changes: We also consider the generalized mind change bound for language learning from positive data (texts). We show that the four types of bounds can be adapted to this scenario and discuss the results that lift from the function case to the language case and also the ones that do not.

In the sequel, we proceed formally, and assume that the reader is familiar with basic notions of inductive inference of functions (e.g., see [5]).

[1]Bounding both hard and weak mind changes only gives subclasses of Ex.

2 Learning Functions

This section deals with characterizations of the various types of mind changes via the classes of functions on which the learners converge. While the second and fourth type have nice natural characterizations, that one for the third type is partial. But since this partial characterization implies the result on the second type, it is convenient to start with the third type.

Theorem 2.1 *Let M be a computable learner for S which converges on every function which is computable relative to the halting problem K. Then S can be learned with bounded mind changes of the third type.*

Proof Let M be a machine which converges on every function which is computable relative to K. Furthermore let $\sigma_0, \sigma_1, \sigma_2, \ldots = \lambda, 0, 1, \ldots$ be an enumeration of all strings. Now every string τ can be put into a standard form $\sigma_{a_0}\sigma_{a_1}\ldots\sigma_{a_n}\sigma_b$ such that the places $\sigma_{a_0}\sigma_{a_1}\ldots\sigma_{a_m}$ with $m = 0, 1, \ldots, n$ are just those where M makes a mind change. Now each such string τ is associated with a string $\eta(\tau) = 0a_0 0a_1 \ldots 0a_n 1b$: the additional numbers 0 indicate that the $a_0, a_1, \ldots, a_n$ are not the last values in the string and 1 indicates that b is the last. Let Q be the closure of $\{\eta(\tau) : \tau \in \mathbb{N}^*\}$ under prefixes. The ordering on Q is just the lexicographic order on these strings. Furthermore M starts with the counter value 1 0 and at every mind change of the form $M(\tau a) \neq M(\tau)$, M takes the value $\eta(\tau a)$. It can be verified that for two succeeding mind changes at τa and $\tau' a' \succ \tau a$ the corresponding values $\eta(\tau a)$ and $\eta(\tau' a')$ are of the form $0a_0 0a_1 \ldots 0a_n 1b$ and $0a_0 0a_1 \ldots 0a_n 0a_{n+1} 1b'$ so that the second value is below the first one. It remains now to show that Q has no infinite descending recursive sequence.

Assume now by the way of contradiction that Q has a descending computable sequence $\eta_0 > \eta_1 > \ldots$. Furthermore this sequence must contain strings of arbitrary length since every set $\mathbb{N}^k$ is well-ordered w.r.t. lexicographic order and does not have an infinite descending sequence. So the strings η_k approximate in the limit an infinite string g which can be computed with oracle K. Each finite string $g(0)g(1)\ldots g(2m+1)$ has an extension η_k in the sequence which belongs to some τ_k, it follows that $\eta(\tau_k) \succeq g(0)0g(2)0\ldots g(2m)0$ and so g defines an infinite function $f = \sigma_{g(0)}\sigma_{g(2)}\ldots$ which is computable relative to g and thus relative to K. M makes on f infinitely many mind changes in contradiction to the choice of f. ∎

This construction relatives in the following way: if a learner M for a class S converges on every function of Turing degree below A' then M can be learned using a set Q which has no infinite descending sequences which are computable relative to A. This set Q does not depend on A and so one obtains that Q does not have any infinite descending sequences if M converges on every function. Such a set Q is well-ordered. On the other side, if M uses a well-ordered set of counters then M converges on every function to a (possibly wrong) index. So one obtains the following result of Ambainis, Jain and Sharma [1].

Theorem 2.2 *A class S can be learned via a learner with bounded mind changes of the second type iff it can be learned via a learner which converges on every — recursive or nonrecursive — function to some guess.*

Somehow while Theorem 2.2 gives a characterization, Theorem 2.1 gives only a sufficient condition. It can be proven that the condition is not necessary. The next theorem fills the gap a bit by giving a weaker necessary condition. Since both theorems do not change the machine itself but only argue on how to construct Q in the first case and how Q has to look like in the second case, it is possiblt to obtain that convergence on functions computable relative to K implies convergence on all functions of hyperimmune-free Turing degree.

Theorem 2.3 *If a machine M learns a class with mind change bounds of the third type then M converges on every function of hyperimmune-free Turing degree. In particular, if M converges on every function computable relative to K then M also converges on every function of hyperimmune-free Turing degree.*

Proof Assume by way of contradiction that M is a machine respecting mind change bounds of the third type via set Q and that f is a function of hyperimmune-free Turing degree on which M diverges. Then the descending sequence $q_0, q_1, \ldots$ of values of Q which M outputs during its infinitely many mind changes. Furthermore Q has a computable one-one enumeration $r_0, r_1, \ldots$ and the function g implicitly defined by $q_n = r_{g(n)}$ has a computable majorant h. Now let $s_0 = r_0$ and $s_{n+1} = \max\{r_m : r_m < s_n \wedge m \leq h(n+1)\}$. It can be verified via induction that $s_n \geq q_n$ for all n using the following two facts: $q_{n+1} = r_{g(n+1)} < s_n$ and $g(n+1) \leq h(n+1)$. Thus it follows that this construction goes through for every n and one obtains an infinite descending computable sequence in Q which should not exist.

By combining this fact with Theorem 2.1 and by using that Theorem 2.1 does not change M but only adds to M the counter and Q, one obtains the second statement of this theorem. ∎

For the fourth type it is again possible to obtain a nice characterization. Indeed what the second type is in the world of all functions, that is the fourth type in the world of all computable functions. This analogy between these two types is so striking that many theorems will be formulated for the second and fourth type at the same time.

Theorem 2.4 *A class S can be learned via a learner* *with bounded mind changes of the fourth type iff it can be learned via a learner which converges on every computable function f to some guess.*

Proof Let M be a learner with bounded mind changes of the fourth type and f be a computable function. If M makes on f infinitely many mind changes then there is a descending sequence $q_0 > q_1 > \ldots$ of guesses in Q associated with these mind changes. This sequence is then computable in contradiction to the choice of Q, thus M makes only finitely many mind changes on any computable function.

For the other way round let M learn a class S of functions and converge on every computable function. Now let Q be the set of all strings σa such that $M(\sigma a) \neq M(\sigma)$ plus the empty string λ. Let the ordering on Q be the reverse string extension ordering, i.e., let $\sigma > \tau$ iff $\tau = \sigma\eta$ for some nonempty string $\eta \in \mathbb{N}^+$. Now M is equipped with a counter which is initialized as λ and which changes to σa whenever $M(\sigma a) \neq M(\sigma)$. So it remains to show that Q has no recursive infinite descending sequence, but if there would be one, say $\sigma_0 > \sigma_1 > \ldots$ then these σ_n would converge to a computable function f given by $f(x) = \sigma_{x+1}(x)$ and M would make infinitely many mind changes on f in contradiction to the choice of M. ∎

The next two theorems attack basic properties of this notions. Theorem 2.5 shows that the classes learnable with a bound on the number of mind changes are closed under union for each type. So mind change bounds always lead to a well-behaved but of course more restrictive notion than Ex. There is one great difference between Ex-learning without and with a mind change bound. The more general setting is not closed under union while a bound on the number of mind changes gives such a closure.

Theorem 2.5 *If two classes S_1 and S_2 are Ex-learnable respecting a mind change bound then $S_1 \cup S_2$ is also Ex-learnable respecting a mind change bound of the same type.*

Proof The proof makes use of two basic items which are presented now together with the main ideas to prove them.

- It can be assumed that M on no f converges to an index e with contradicts the data seen so far in the way that $\varphi_e(x)\downarrow \neq f(x)$ for some x. This can be obtained via a small update on the previously given machine M. The set Q of countervariables is replaced by $Q \times \{0,1\}$ and M either outputs the original guess e with counter value $(q,1)$ in place of the original value q or outputs an index for the everywhere undefined function with counter value $(q,0)$ if some x with $\varphi_e(x)\downarrow \neq f(x)$ had been found within a suitable number of simulation steps.
- For any two programs i and j there is a program e called the amalgamation of i and j such that $\varphi_e(x)$ takes the first value y to be found such that $\varphi_i(x)\downarrow = y$ or $\varphi_j(x)\downarrow = y$. This can be done via simulating the programs i and j in parallel. Note that φ_e is total if φ_i or φ_j is and that e can be effectively computed from i and j.

Let S_1 and S_2 be Ex-learnable with a bound on the number of mind changes via machines M_1 and M_2. Q_1 and Q_2 are the sets of counter values of M_1 and M_2. The combined learner M uses now amalgamated indices.

> $M(\sigma)$ outputs the amalgamation of $M_1(\sigma)\&M_2(\sigma)$ and has the countervalue $(q_1,q_2) \in Q_1 \times Q_2$ where the ordering on $Q = Q_1 \times Q_2$ is given by $(p_1,p_2) < (q_1,q_2)$ if either $p_1 < q_1$ or $p_1 = q_1 \wedge p_2 < q_2$.

The verification starts via showing that M converges on every $f \in S_1 \cup S_2$. By Theorem 2.4, M_1 and M_2 converge on any computable f. Each function in $S_1 \cup S_2$ is computable. So M_1 converges on f to some index i, M_2 to j and M to the amalgamation e of i and j. One of the programs i and j, say i, computes f. Thus φ_e is total. By the first item, $\varphi_j(x) = f(x)$ for all $x \in dom(\varphi_j)$. So it follows that the amalgamation always outputs the same value as f and so M converges on every $f \in S_1 \cup S_2$ to an index of f.

The next step is to show that the mind change bound is kept. M makes only a mind change if M_1 or M_2 do. Since one of these machines decreases its counter, M also decreases the resulting counter. So it remains to show that Q is in all four cases of the same type as Q_1 and Q_2 which is now verified for each of the four types.

First type: Q is a finite and linearly ordered set since Q_1 and Q_2 are. Such a set then can be represented as $\{0,1,\ldots,|Q|-1\}$.

Second type: Q is linearly ordered since Q_1 and Q_2 are. Assume now that $(p_1,q_1) > (p_2,q_2) > \ldots$ is a infinite descending chain. Now let $p_{i_1} = p_1$ and select for each p_{i_j} some $p_{i_{j+1}} < p_{i_j}$. Either this produces an infinite descending chain in Q_1 or stops at some p_{i_j}. Then $p_k = p_{i_j}$ for all $k > i_j$ and $q_{i_j} > q_{i_j+1} > \ldots$ forms an infinite descending chain in Q_2. Thus Q does not have an infinite descending sequence since Q_1 and Q_2 do not have them.

Third type and fourth type: In the case of the third type, Q is linear since Q_1 and Q_2 are. Furthermore replacing "infinite descending chain" by "computable infinite descending chain" in the argumentation for the second type gives a proof that neither the Q for the third nor that for the fourth type contains an infinite descending computable chain. ∎

Theorem 2.6 *The hierarchy given by these notions of bounded mind changes is proper.*

Diagonalization	T	Q for higher level algorithm
2nd vs. 1st level	$\{n\tau : \|\tau\| < n\}$	$\{0, \frac{1}{2}, \frac{2}{3}, \frac{3}{4}, \frac{4}{5}, \frac{5}{6}, \ldots, 1\}$
3rd vs. 2nd level	$\{q_0 q_1 \ldots q_n \in Q^* : q_0 > q_1 > \ldots > q_n\}$	contains infinite descending sequence, but no recursive one
4th vs. 3rd level	infinite recursive tree $T \subseteq \{1,2\}^*$ without infinite recursive branch	T with reverse ordering of string extension
5th vs. 4th level	$\{1, 2, \ldots\}^*$	none

Table 1: Separating the types of mind change bounds

Proof For each function f which is almost everywhere 0 let $\sigma(f)$ be the string of the non-zero values, e.g., if $f = 1000200001000200110^\infty$ then $\sigma(f) = 121211$. Now let $T \subseteq \{1, 2, \ldots\}^*$ be a set of strings. Then

$$S_T = \{f : (\forall^\infty x)\,[f(x) = 0] \wedge \sigma(f) \in T\}$$

denotes the set of functions with finite support derived from T. All levels of the hierarchy can be separated via a set of the form S_T.

Freivalds and Smith [6] already showed that the first two levels are different. A witness for this non-inclusion is the class of all non-increasing functions. A further, a bit more complicated, witness can also be constructed with the methods of this theorem: Let $T = \{n\tau : |\tau| < n\}$ and $Q = \{0, \frac{1}{2}, \frac{2}{3}, \frac{3}{4}, \frac{4}{5}, \frac{5}{6}, \ldots, 1\}$. The learner starts with $f = 0^\infty$ and $q = 1$. If then the input gets the form $0^k n$, the learner makes a mind change to $f = 0^k n 0^\infty$ and $q = \frac{n}{n+1}$. Now at most $n-1$ further non-zero values arrive each demanding for a mind change to $\eta 0^\infty$ where η is the input seen so far and decreases the counter at each mind change from $\frac{n}{n+1}$ through $\frac{n-1}{n}$, ..., $\frac{1}{2}$ to 0. The further verification is left to the reader.

The separation of the third from the second level is done via an enumerable set Q of rationals between 0 and 1 which contains an infinite descending sequence $1 > q_0 > q_1 > \ldots$ but no computable such sequence. Q is identified with the numbers $1, 2, \ldots$ in order to make the notation for T and S_T easier. Now let T contain all strings $r_0 r_1 \ldots r_n \in Q^*$ with $1 > r_0 > r_1 > \ldots > r_n$; where the ordering "$>$" is the one induced by the rationals and not the one of the natural numbers used to code the rationals. The set S_T is now learnable via mind changes of the third type: The learner initializes the counter to 1 and the guess to 0^∞. Whenever on the input appears a new non-zero element, say the input equals ηr_m (with $\sigma(\eta r_m 0^\infty) = r_0 r_1 \ldots r_m$) the counter is decreased to r_m and the learner changes its mind to $\eta r_m 0^\infty$. It is easy to see that the learner succeeds on all functions in S_T. On the other hand it is possible to construct for any learner M an infinite (and noncomputable) function $g = q_0 0^{a_0} q_1 0^{a_1} q_2 0^{a_2} \ldots$ such that $M(q_0 0^{a_0} q_1 0^{a_1} \ldots q_n 0^{a_n})$ guesses an index for $q_0 0^{a_0} q_1 0^{a_1} \ldots q_n 0^\infty$. So M makes infinitely often a mind change on the function g. By Theorem 2.2, S_T is not learnable with a mind change bound of the second type.

For the separation of the fourth from the third level of the hierarchy consider a recursive tree $T \subseteq \{1, 2\}^*$ which has some infinite branch but no recursive infinite branch. Now let the partially ordered set Q be just T equipped with an ordering given by $\eta > \tau$ iff τ is a proper extension of η for all elements $\eta, \tau \in Q$. Now an inference algorithm with bounded mind changes of the fourth type always guesses on input η the function $f = \eta 0^\infty$ and sets the counter to $\sigma(f)$ provided that $\sigma(f) \in T$. If $\sigma(f)$ is no longer in T then f is outside the class of functions to be learned; thus the learner abstains from any more guesses and does also not longer decrease the counter.

The proof that the class is not learnable via bounded mind changes of the third type is a bit involved. Assume by way of contradiction that M is such a computable machine and Q is the corresponding set of rationals. Let Q_s be the set of rationals in Q enumerated within s steps. Since T has an infinite path, there is such a path α having hyperimmune-free Turing degree, i.e., every total function computable relative to α has a computable majorant. To given α construct inductively a function $\alpha(0)0^{a_0}\alpha(1)0^{a_1}\ldots$ where $M(\alpha(0)0^{a_0}\alpha(1)0^{a_1}\ldots\alpha(n)0^{a_n})$ guesses $\alpha_0 0^{a_0}\alpha(1)0^{a_1}\ldots\alpha(n)0^\infty$ with counter $q_n \in Q$. Thus M makes on f infinitely many mind changes. f is computable relative to α and so also the function $g : n \to \min\{s : q_n \in Q_s\}$ is computable relative to α. g has a recursive majorant h. Now let $r_0 = q_0$ and r_{n+1} be the maximal element of $Q_{h(n+1)}$ below r_n. It can be verified inductively that each r_n is in Q and each $r_n \geq q_n$: $q_{n+1} < q_n \leq r_n$ and $q_{n+1} \in Q_{g(n+1)} \subseteq Q_{h(n+1)}$, thus q_{n+1} is an element of $Q_{h(n+1)}$ which is below r_n; since r_{n+1} is the maximal element of $Q_{h(n+1)}$ below r_n it follows that r_{n+1} exists and is greater or equal to q_{n+1}.

The sequence $r_0, r_1, \ldots$ is a recursive descending infinite sequence in Q which should not exist by the definition of Q, thus there is no such Q and M and the class can not be learned with bounded mind changes of the third type.

The last noninclusion, that there are sets which are only learnable without any bound on the mind changes is the class given by $T = \{1, 2, \ldots\}^*$, i.e., given by the whole class S_T of all functions with finite support. This class is learnable: on input η, the learner always guesses a canonical index for $\eta 0^\infty$ where canonical means that each input $\eta 0^k$ produces the same index which is possible since $\eta 0^\infty = \eta 0^k 0^\infty$. On the other hand it is well-known that $S_T \cup \{f : f = \varphi_{f(0)}\}$ is not in Ex while $\{f : f = \varphi_{f(0)}\}$ can be learned without any mind changes. Thus S_T can also not be learned with bounded mind changes since otherwise the union of these two classes would be in Ex by Theorem 2.5.

Table 1 summarizes the description of the examples witnessing the properness of the hierarchy. The second column gives the set T from which S_T is constructed and the third the set Q for the succeeding algorithm in the more general class. Here 5th level means just learning without any bound on the mind changes. ∎

3 Mind Changes and Hypothesis Spaces

Sometimes the learner does not only want to learn something, but also to produce good hypotheses, say only primitive recursive programs. But somehow to produce good programs might be more difficult than to produce any programs. Therefore it can be expected that learning w.r.t. a given enumeration is more difficult than w.r.t. an arbitrary enumeration. Lange and Zeugmann [17] therefore considered three settings:

(1) exact learning: here the learner has to use a given hypothesis space, that means the task is given as a set S plus an enumeration $\{f_0, f_1, \ldots\}$ of S. Learning f, the learner has to identify some index i such that $f = f_i$.

(2) class-preserving learning: here the learner can choose any space of hypotheses which contains exactly the functions in S: $S = \{f_0, f_1, \ldots\}$. This means the learner may rearrange the order of the functions but the learner does not have the right to add any new functions to the hypothesis space. This can be quite annoying. If for example the hypothesis space contains only those functions $0^n 1^\infty$ for which $n \in K$ then the learner might have to search for a long time to find such an n since the enumeration somehow reflects the enumeration process of K. A larger hypothesis space might be much easier to access.

(3) Class-comprising learning: here the learner has the right not only to reorder the functions but also to insert new ones into the space of hypotheses and so receive an enumeration which is much more flexible. The price paid is of course that certain beautiful properties like $S = \{f_0, f_1, \ldots\}$ might be lost. Also the new space may contain functions which are much more difficult to compute.

The following is shown in this section: There are classes which can be Ex-learned with a bound on the mind changes of the first type only class-preservingly but not exactly. Furthermore there are classes which can be Ex-learned with a constant bound on the number of mind changes class-comprisingly but not with any bound on the number of mind changes class-preservingly. If a class can be class-preservingly Ex-learned with a mind change bound of the second or fourth type then it can already be Ex-learned exactly with the same bound.

Theorem 3.1 *Let $S = \{f_0, f_1, \ldots\}$ be a uniformly recursive family of functions and let $\{g_0, g_1, \ldots\}$ be a further uniformly recursive enumeration for S. If S can be learned using indices for $\{f_0, f_1, \ldots\}$ with bounded mind changes of second or fourth type, then S can also be learned using indices for $\{g_0, g_1, \ldots\}$ and respecting the same type of bounded mind changes.*

Proof There is a limiting recursive function $h = \lim_n h_n$ which translates every index for the fs into one for the gs. This function is defined via $h_n(i) = j$ for the first j with $f_i(x) = g_j(x)$ for all $x \leq n$. Now given a learner M respecting mind change bound of the second or fourth type, i.e., a learner which converges on every / on every computable function f, then the corresponding learner N is given by $N(a_0 a_1 \ldots a_n) = h_n(M(a_0 a_1 \ldots a_n))$. If M converges on some function f to a value i then N converges to $h(i)$. So the conditions "M converges on every function" and "M converges on every computable function" are preserved and with them mind change bounds of second and fourth type. ∎

The next theorem gives an example of a class which can be learned with a constant bound of mind changes class-preservingly but not exactly, if it is given via a "bad" enumeration.

Theorem 3.2 *The class*

$$S = \{f : (\forall x)\,[2 \geq f(x) \geq f(x+1)]\}$$

can be learned with two mind changes from a good enumeration but not with any constant bound on the number of mind changes from a bad enumeration.

Proof The general algorithm is to conjecture $f = 2^\infty$ until a first x with $f(x) < 2$ is found. Then a mind change to $f = 2^x 1^\infty$ or $f = 2^x 0^\infty$ is made in dependence of $f(x)$. If $f(x) = 1$ then some $y > x$ with $f(y) = 0$

may require a second mind change to $f = 2^x 1^{y-x} 0^\infty$. So at most two mind changes are necessary provided that the enumeration makes it possible to compute the indices for the different conjectures easily. This is no problem if the enumeration is good, but on the other hand there are bad enumerations which hide the indices of the functions $2^x 1^\infty$ so well that constant bounds on the number of mind changes make it impossible to find them.

The triples $\langle x, y, z\rangle$ with $x, y \in \mathbb{N}$ and $z \in \{0, 1\}$ are identified with the positive natural numbers $\{1, 2, \ldots\}$. Furthermore let

$$a(x) = 1 + \max\{\varphi_y(x) : y \leq x \wedge \varphi_y(x)\downarrow\},$$

a is computable in the limit and has an approximation a_s, i.e., $a(x) = \lim_s a_s(x)$. Now let

$$\begin{array}{rcl} g_0 & = & 2^\infty \\ g_{\langle x,y,0\rangle} & = & 2^x 1^y 0^\infty \\ g_{\langle x,y,1\rangle} & = & \begin{cases} 2^x 1^z 0^\infty & \text{if } z \text{ is the first number beyond } y \text{ with } a_z(x) > a_y(x); \\ 2^x 1^\infty & \text{otherwise, that is, if there is no such } z. \end{cases} \end{array}$$

Assume now by the way of contradiction that some machine M learns S w.r.t. indices for $\{g_0, g_1, \ldots\}$ with at most k mind changes for some constant k. There are $k + 1$ functions $\varphi_{e_0}, \ldots, \varphi_{e_k}$ such that $\varphi_{e_i}(x)$ takes on input $2^x 1^\infty$ the value a_y iff the i-th guess of M has the form $\langle x, y, 1\rangle$; $\varphi_{e_i}(x)$ is undefined otherwise. By construction the last of these guesses has to have the form $\langle x, y, 1\rangle$ with $a_y(x)$ being an upper bound of $a(y)$. If $x > e_0 + e_1 + \ldots + e_k$ then one obtains the contradiction $\varphi_{e_j}(x) = a_y(x) \geq a(x) > \varphi_e(x)$ for $e = 0, 1, \ldots, x$, in particular for $e = e_j$. Thus no algorithm with a constant bound on the number of mind changes learns the class S using indices from the enumeration $\{g_0, g_1, \ldots\}$. ∎

Every uniformly recursive family of functions can be Ex-learned exactly w.r.t. any enumeration containing the family but no partial computable function. Somehow this result does not transfer to bounded mind changes and Theorem 3.1 can not be improved from class preserving enumerations to class comprising enumerations. That means the next theorem gives a family which can be learned class-comprisingly with only two mind changes but which can not be learned class-preservingly with any type of mind change bound.

Theorem 3.3 *The class*

$$S' = \{2^\infty\} \cup \{2^x 1^\infty : x \in K'\} \cup \{2^x 1^y 0^\infty : x, y \in \mathbb{N}\}$$

can be learned w.r.t. the standard numbering of all computable functions with the constant bound 2 on the number of mind changes. But if a machine M learns S' w.r.t. some enumeration of S' then it does not respect any mind change bounds.

Proof The learning algorithm w.r.t. the standard enumeration of all computable functions is the same as for the class S which contains S' in Theorem 3.2. It remains to show that S' has a uniformly recursive enumeration $g_0, g_1, \ldots$ and that any machine M learning S' w.r.t. some enumeration $\{f_0, f_1, \ldots\}$ of S' fails to satisfy a mind change bound of the fourth type.

There is a computable two-place function $h : \mathbb{N} \times \mathbb{N} \to \{0, 1\}$ such that x is in K' iff $h(x, y) = 1$ for almost all y. Similar to Theorem 3.2 the triples $\langle x, y, z\rangle$ with $x, y \in \mathbb{N}$ and $z \in \{0, 1\}$ are identified with the positive natural numbers $\{1, 2, \ldots\}$. Now the family $\{g_0, g_1, \ldots\}$ is defined such that the function $g_{\langle x,y,1\rangle}$ equals $2^x 1^\infty$ iff $h(x, z) = 1$ for all $z \geq y$. Given any x, such an y exists iff $x \in K'$.

$$\begin{array}{rcl} g_0 & = & 2^\infty \\ g_{\langle x,y,0\rangle} & = & 2^x 1^y 0^\infty \\ g_{\langle x,y,1\rangle} & = & \begin{cases} 2^x 1^z 0^\infty & \text{for the first } z \text{ beyond } y \text{ with } h(x, z) = 0; \\ 2^x 1^\infty & \text{if there is no such } z. \end{cases} \end{array}$$

Let $\{f_0, f_1, \ldots\}$ be a uniformly recursive enumeration for exactly S' and let M be a machine learning S' with indices from this enumeration. Assuming that M satisfies a mind change bound of the fourth type it is shown that then M is not computable; thus no computable machine can learn S' w.r.t. the given indices for $\{f_0, f_1, \ldots\}$ respecting a mind change bound of the fourth type.

By Theorem 2.4, M converges on every computable function, in particular M converges on every function $2^x 1^\infty$ to some index $e(x)$. $e(x)$ can be computed relative to the jump U of M; the jump $U = \{e : \varphi_e^M(e)\downarrow\}$ is the halting set for computations which use M as an oracle.

Now the correspondence $x \in K' \Leftrightarrow f_{e(x)} = 2^x 1^\infty$ holds and is due to the following observation: If $x \in K'$ then $2^x 1^\infty \in S'$ and M has to infer this function. So $f_{e(x)} = 2^x 1^\infty$. Otherwise $x \notin K'$ and $2^x 1^\infty \notin S'$. Since there is no index of $2^x 1^\infty$, M converges to an index of some other function: $f_{e(x)} \neq 2^x 1^\infty$.

Since the family $\{f_0, f_1, \ldots\}$ is uniformly recursive, the query $(\exists y)\,[f_{e(x)}(y) \neq 2^x 1^\infty(y)]$ can be answered with oracle K and in particular with oracle U. Thus $K' \leq_T U$ and the Turing degree of M must be high. M is not computable. ∎

4 Mind Change Complexity for BC

Case and Smith [5] gave the following definition for BC which is equivalent to the standard one: A class S is in BC iff there is a partial recursive machine M which predicts every $f \in S$ without any additional convergence requirements. That is if $f \in S$ then

$$M(f(0)f(1)\dots f(x))\downarrow = f(x+1)$$

for almost all x. In this context, mind changes are identified with prediction errors, e.g., a machine makes a mind change iff it finds out that it has to follow another function than it predicted. There are weak and hard mind changes: M has a weak mind change at $x+1$ iff $M(f(0)f(1)\dots f(x))$ does not converge and a hard one iff this expression converges to a value different from $f(x+1)$. [2]

Definition 4.1 A predictor M is equipped with a mind change counter if for every function f and for every hard mind change there is a future stage such that the counter is decreased inside the set Q of counter values; in particular iff M makes on f at x a hard mind change then the counter is decreased at some stage $y \geq x$. The four different types of mind change bounds correspond to the four restrictions on A defined above.

Similar to the case Ex one has the following characterization of BC-learning with mind change bounds of the second or fourth type.

Theorem 4.2 *A class S is BC-learnable with bounded mind changes of the second (fourth) type iff there is a BC-learner M for S which makes on no function (no computable function) infinitely many hard mind changes.*

The proof is essentially the same as in the Ex-case. The next Theorem shows that BC-learning with bounded mind changes is not a generalization of Ex-learning.

Theorem 4.3 *The concepts of* Ex*-learning and* BC*-learning with bounded mind changes are incomparable for all four types.*

Proof Merkle and Stephan [19] already showed that for every binary computable tree T the class S_T of its isolated and infinite branches is BC-learnable without any hard mind change; furthermore there are trees T such that this class S_T is not Ex-learnable. It follows immediately that the classes S_T of such trees are learnable under all bounds for hard mind changes and so witness one non-inclusion.

For the other non-inclusion consider the class $\mathbb{N}^* \cdot 0^\infty$ of all functions of "finite support". These functions can be learned under the criterion Ex. But they can not be BC-learned when the predictor respects any type of mind change bounds.

To see this, assume by the way of contradiction that M is a predictor having a mind change bound of the most general fourth type. Then a function f can be computed from an index of M by the following procedure: Having already a prefix σ of f, the algorithm to compute f searches the least values n, t, y such that $M(\sigma 0^n)$ converges in t computation steps to the value y. Since M predicts $\sigma 0^\infty$, such n, t, y must exist. Having found these n, t, y, the algorithm expands the current string σ to $\sigma' = \sigma 0^n (y+1)$. Used inductively, this procedure gives in the limit the infinite computable function f such that M has infinitely many hard mind changes on f and so — using Theorem 4.2 — M does not respect mind change bounds of the fourth type. ∎

The second part of the proof can easily be generalized such that it uses every class in the proof of Theorem 2.6 to separate the higher level in the bounded mind change hierarchy for Ex-learning from the lower level for BC-learning; thus the transition from Ex to BC does not enable to go down a level in the hierarchy of bounded mind changes.

5 Characterizing the Types of Bounded Mind Changes

Merkle and Stephan [19] introduced a notion to characterize when a class of functions is learnable according to the criteria Fin, Ex and BC via uniformly recursive families of trees. They already observed that the characterizations for learning $\{0,1\}$-valued functions are simpler than those for arbitrary functions since binary trees are much more well-behaved than general ones. On the other hand they showed that the results for binary trees can be extended to the case of arbitrary functions. So the authors state in this section only the results for the much more transparent case of $\{0,1\}$-valued functions. From now on until the end of this section, "function" means "$\{0,1\}$-valued function" and strings range over $\{0,1\}^*$.

Theorem 5.1 *A class S of functions is learnable via mind change bounds of the second (fourth) type iff there is a uniformly recursive family of binary trees such that every $f \in S$ is the only infinite branch of some tree and every function (every computable function) f is infinite branch of exactly one tree in this family.*

Proof First it is shown that for each class S of functions learnable with bounded mind changes of type two or

[2] The interested reader might want to know how this relates to the standard definition of BC-learning. Indeed in this original setting, one could also come up that there is a hard mind change between x and y iff $(\exists z)\,[\varphi_{M(f(0)f(1)\dots f(x))}(z)\downarrow \neq \varphi_{M(f(0)f(1)\dots f(y))}(z)]$. Somehow this definition has the disadvantage of needing both ends of the interval but this can not be avoided since M might output indices of the everywhere undefined function on the input between x and y. So one could also try to use a concept of mind change which is bit nearer at the concept of an "error" and so consider the following: M makes a mind change at x iff $(\exists z)\,[\varphi_{M(f(0)f(1)\dots f(x))}(z)\downarrow \neq f(z)]$. But the question whether a given class S of functions is BC-learnable with a mind change bound of one of the four types as defined in Definition 4.1 is the same for all three definitions of mind changes considered here. Therefore the present work follows the easiest definition in terms of a predictor.

four there is such a family of trees. As in Theorem 2.5 one may assume that S is learned via a machine M which on no function f converges to an index e with $\varphi_e(x)\downarrow \neq f(x)$ for some x. There is a recursive one-one enumeration $\tau_1, \tau_2, \ldots$ of all strings $\tau_k = \sigma a$ where M makes a mind change ($M(\sigma a) \neq M(\sigma)$); $\tau_0 = \lambda$ is added in order to avoid some anomalies. Using this sequence $\tau_0, \tau_1, \ldots$ the family of trees T_k is defined as follows:

> A string σ is in T_k if either $\sigma \preceq \tau_k$ or $\sigma \succ \tau_k \wedge (\forall \eta)\,[\tau_k \prec \eta \preceq \sigma \Rightarrow M(\eta) = M(\tau_k)]$. So T_k contains all prefixes of τ_k and all extensions which are reached from τ_k without any mind change.

For the verification consider first functions $f \in S$. On these M converges to an index e of f at some τ_k. So f is an infinite branch of T_k. By the choice of M, M converges on no function $g \neq f$ to the same index e since $g(x) \neq f(x) = \varphi_e(x)\downarrow$ for some x. So f is the only infinite branch of T_k. Furthermore M converges on every function f (in case of fourth type, every computable function f) and thus there is some τ_k such that M has no mind change on f after τ_k. It follows that f is on this tree T_k. Since furthermore the τ_k is given by the unique point of converges of M on f, each function is on at most one tree T_k. This finishes the verification.

For the other way round let T_k a family of binary trees which satisfies the requirements. There is a computable function s such that $\varphi_{s(e)}$ computes the unique infinite branch of the tree given by φ_e whenever φ_e specifies a binary tree with exactly one infinite branch [19, 20]. The learning algorithm now picks up always the index of the first tree where f is an infinite branch and transforms an index $t(k)$ of the tree T_k into an index for f.

> $M(\sigma) = s(t(k))$ for the first k with $\sigma \in T_k$.

This algorithm converges on every function (computable function) f since f is on some tree in this family and the learner converges to $s(t(k))$ for the first k such that f is an infinite branch of T_k. Furthermore if $f \in S$ then there is exactly one such tree and f is the only infinite branch of this tree; therefore $s(t(k))$ is an index for f and M converges to an index of this function. ∎

These characterizations can also be generalized to BC learning as follows.

Theorem 5.2 *A class S of functions is learnable via mind change bounds of the second (fourth) type iff there is a uniformly recursive family of binary trees such that every $f \in S$ is isolated infinite branch of some tree and every function f (every computable function f) is infinite branch of exactly one tree in this family.*

Merkle and Stephan [19] have already characterize the classes of languages identifiable by teams of finite learners. Teams of finite learners have the same power as single Ex-learners with a constant bound on the number of mind changes: each $[1,n]$Fin-team can be simulated by an Ex-learner with up to $2n$ mind changes and each single Ex-learner with up to m mind changes can be simulated by an $[1, m{+}1]$Fin-team. So the characterization for finite team learning is already a characterization for Ex-learning with a bound of the first type on the number of mind changes. The criterion of BC-learning with a bound on the number of mind changes has even a very natural characterization.

Theorem 5.3 *A class S is* BC*-learnable with a mind change bound of the first type iff there are finitely many binary computable trees such that each $f \in S$ is an isolated branch of some of these trees.*

Proof For the first direction assume that M BC-learns a class S with the counter values $Q = \{1, 2, \ldots, n\}$. Now for each $m = 1, 2, \ldots, n$ the tree T_m contains all nodes τ where there is no $\sigma a \preceq \tau$ with countervalue below m and prediction $M(\sigma)\downarrow \neq a$ within $|\tau|$ computation steps. For $f \in S$ the counter takes on f a least value m in the limit. Whenever M makes a wrong prediction on some $\sigma \preceq f$ then $M(\sigma) > m$ since M eventually reduces its countervalue after this wrong prediction. So $f \in T_m$. Furthermore there is a prefix τ such that $M(\tau) = m$ and $M(\sigma) = a$ for all σ and a with $\tau \preceq \sigma \preceq \sigma a \preceq f$; therefore M cuts of all branches through $\sigma(1-a)$ from T_m at discovering that $M(\sigma)$ predicts a and so f is isolated on T_m above τ.

For the other direction assume that $T_1, T_2, \ldots, T_n$ is a collection of binary trees such that each $f \in S$ is isolated infinite branch of some of these trees. Let $Q = \{0, 1, \ldots, n\}$ be the set of the counter values and use the following prediction algorithm:

> $N(\sigma)$ sets the counter to the number m of the trees T_k with $\sigma \in T_k$. If $m > 0$ then N predicts a iff there is some length $x > |\sigma|$ such that every string τ which is on m trees and extends σ also extends σa. Otherwise $m = 0$ and $N(\sigma)$ makes no prediction.

Let $f \in S$. f is isolated branch of some tree and furthermore in total on m trees. So f has a prefix $\tau \preceq f$ such that f is isolated above τ on one tree T_k and furthermore τ is only on the m trees. Then for each σ and a with $\tau \preceq \sigma \preceq \sigma a \preceq f$ the algorithm will predict a since there is no infinite branch on T_k above $\sigma(1-a)$ and so there is some $x > |\sigma|$ such that no extension of $\sigma(1-a)$ has length x, in particular no such extension is on m trees while $f(0)f(1)\ldots f(x)$ is an extension of σ which is on m trees. Thus the algorithm converges to the correct output a. Furthermore for every $\sigma a \preceq f$ with the counter value m, $N(\sigma)$ either converges to the value a or diverges: assume by way of contradiction that $N(\sigma)$ converges to $1-a$; this happens only if there is a length $x > |\sigma|$ such that no extension of σa of length x is on m trees in contradiction that the infinite extension

$f \succeq \sigma a$ is on m trees. Therefore the counter is decremented after the last hard mind change on f from some value above m to m and so N learns S with a mind change bound of type 1. ∎

6 Language Learning

A language is just an enumerable subset of $\mathbb{N}$. Language learning differs from function learning in two aspects: The languages may be not computable and therefore the learner has less ability to check the quality of an index; furthermore the most common model is learning from text so that the learner does not receive a characteristic function of the language L to be learned but only an infinite sequence containing all elements of L plus the symbol # in arbitrary order with arbitrary many repetitions. For convenience, L^* is written instead of $(L \cup \{\#\})^*$ and # is omitted in the range of a finite or infinite string: $range(\sigma) = \{y \in \mathbb{N} : (\exists x)\,[\sigma(x)\!\downarrow = y]\}$. Furthermore during the whole section: function learning is Ex-learning with input of the form $f(0)f(1)\ldots$; language learning is Ex-learning from text.

A class $S = \{L_0, L_1, \ldots\}$ is uniformly enumerable iff $\{(x, e) : x \in L_e\}$ is an enumerable set and S is uniformly recursive iff $\{(x, e) : x \in L_e\}$ is a computable set. Uniformly recursive families have been widely studied, in particular in context of monotonicity requirements [2, 8, 12, 13, 15, 16, 17, 24]. Uniformly enumerable families are just a natural generalization. Angluin [2] found a nice characterization which states when a uniformly recursive family can be learned from text; de Jongh and Kanazawa [11] generalized this criterion to uniformly enumerable families.

Beside looking at the fact how mind change bounds generalize to language learning from sets, it is an interesting question which kind of Angluin style characterizations has the learning of uniformly recursive or enumerable classes with mind change bounds.

The definition of the four levels of the hierarchy are very similar. Only in the fourth type one might think that it makes a difference whether convergence is required only on computable texts or on every — also noncomputable — text of any given language. Somehow the next theorem shows that this problem is not present.

Theorem 6.1 *Let S be a class of languages. Then the following is equivalent:*

(a) *S is learnable with bounded mind changes of the fourth type.*

(b) *Some M learns S and converges on every computable text.*

(c) *Some N learns S and converges on every text which belongs to some language.*

Proof The parts ($\mathtt{a} \Rightarrow \mathtt{b}$) and ($\mathtt{c} \Rightarrow \mathtt{a}$) are analogous to the proof of Theorem 2.2. So only the part ($\mathtt{b} \Rightarrow \mathtt{c}$) must be shown. Let M be a machine which converges on every computable text. N is now constructed according to the locking-sequence hunting construction [21]. Let $\sigma_0, \sigma_1, \ldots$ be an enumeration of all strings in $\mathbb{N}^*$ and let $a_0 a_1 \ldots$ be some text of some language L. Now N on input $a_0 a_1 \ldots a_n$ is defined as follows:

> $N(a_0 a_1 \ldots a_n) = M(\sigma_m)$ for the first m such that $range(\sigma_m) \subseteq \{a_0, a_1, \ldots, a_n\}$ and $M(\sigma_m \tau) = M(\sigma_m)$ for all $\tau \in \{a_0, a_1, \ldots, a_n\}^*$ with $|\sigma_m \tau| \le n$.

First this search terminates since it is satisfied for the string $\#^n$ which has some index m. Second since M converges on every computable text of L, L has a locking sequence σ_k in the sense that $M(\sigma_k \tau) = M(\sigma_k)$ for all $\tau \in L^*$. From the moment that $range(\sigma_k)$ is contained in $\{a_0, a_1, \ldots, a_n\}$, N outputs $M(\sigma_m)$ for some $m \le k$. On the other hand, N does not infinitely often output $M(\sigma_m)$ iff σ_m is not a locking sequence for L since there is some $\tau \in L^*$ with $M(\sigma_m \tau) \neq M(\sigma_m)$ and for all sufficiently large n, this witness against σ_m being a locking sequence for L is discovered. So N converges to $M(\sigma_k)$ for the least k such that σ_k is locking sequence for L.

Furthermore whenever M learns L, M outputs at σ_k an index for L and so also N converges to an index for L. Thus N learns the same languages as M (probably even some more) and so N learns S. ∎

The characterization of the mind change bounds of the second type via machines which converge on all functions transfers to the characterization a class of languages can be learned respecting mind change bounds of the second type iff it can be learned via a machine which converges on every infinite sequence of natural numbers and the symbol #, also if this sequence does not belong to any language (= enumerable set).

Theorem 6.2 *The hierarchy given by the four types of mind change bounds is proper.*

Proof For any function f, let the set $L_f = \{(x, f(x)) : x \in \mathbb{N}\}$ be the set associated to f and $S' = \{L_f : f \in S\}$ be the class of sets associated to a class of functions. The learning algorithms for both can be translated into each other; in particular these translations preserve mind change bounds such that for $n = 1, 2, 3, 4$ the following statement holds:

> S can be learned respecting mind change bounds of the n-th type iff S' can be learned respecting mind change bounds of the n-th type.

Formally an algorithm M for S is translated into a new algorithm N for S' as follows:

On input $(x_0, y_0)\,(x_1, y_1) \ldots (x_n, y_n)$, N first checks whether some pairs contradict each other, that is, N checks whether there are $i, j \in \{0, 1, \ldots, n\}$ with $x_i = x_j$ and $y_i \neq y_j$. If so N keeps the last guess and makes no mind change, otherwise M computes the longest string $b_0 b_1 \ldots b_m$ such that $b_i = y_j$ for some $j \leq n$ with $x_j = i$ and $i = 0, 1, \ldots, m$. Then N outputs an index $s(M(b_0 b_1 \ldots b_n))$ where the function s is given via $\varphi_{s(e)} = \{(x, \varphi_e(x)) : x \in dom(\varphi_e)\}$.

So N simulates on a given text T so long the algorithm M as a unique input for M can be retrieved. At that moment where the text T becomes contradictory, N just abstains from any further guess and mind change and so satisfies the same mind change bound as M. ▮

Theorem 6.3 *If a uniformly enumerable class is learnable then it is learnable relative any given enumeration of this class.*

Proof Let $S = \{L_0, L_1, \ldots\} \subseteq \{H_0, H_1, \ldots\}$ be two uniform enumerations and let S be learnable w.r.t. the second one via some machine M. Furthermore M converges on every text for the same language L_e to the same index; this can be enforced via the locking-sequence hunting construction from Theorem 6.1. Let $a_{e,0} a_{e,1} a_{e,2} \ldots$ be a text for L_e which is uniformly recursive. Given any text $a_0 a_1 a_2 \ldots$ for some language $L \in S$, the new learner N for indices from $\{L_0, L_1, L_2, \ldots\}$ works as follows.

$$N(a_0 a_1 \ldots a_n) = \begin{cases} e & \text{for the smallest } e \leq n \\ & \text{with } M(a_0 a_1 \ldots a_n) \\ & = M(a_{e,0} a_{e,1} \ldots a_{e,n}); \\ ? & \text{if there is no such } e. \end{cases}$$

L has a least index e. On $a_0 a_1 \ldots$ and on $a_{e,0} a_{e,1} \ldots$, M converges to e; but on $a_{e',0} a_{e',1} \ldots$ with $e' < e$, M converges to some other index. It follows that N outputs for almost all n exactly this index e and so converges to the least index of L. ▮

The positive result of Theorem 3.1 that for any uniformly enumerable families of functions it does not depend on the enumeration whether a mind change bound of second or fourth type can be obtained, does also hold for unformly enumerable families of languages. Indeed if in the proof above the learner M converges on every text / every computable text to an index i of some set H_i which equals some set L_j then the new learner N also converges to j. Therefore if $\{L_0, L_1, \ldots\} = \{H_0, H_1, \ldots\}$ then mind change bounds of the second and fourth type are preserved.

On the other hand also the negative results of Theorem 3.2 and Theorem 3.3 generalize since one can replace the families of functions f by the corresponding families of the sets $L_f = \{(x, f(x)) : x \in \mathbb{N}\}$. So the following theorem holds.

Theorem 6.4 *If a uniformly enumerable family can be learned with bounded mind changes of the second or fourth type from text w.r.t. a given enumeration then so can be done also w.r.t. indices of any other enumeration of the same class. But this does not hold for mind change bounds of the first type.*

Furthermore there is a uniformly enumerable family S which can be learned via a constant bound on the mind changes w.r.t. some acceptable numbering of all enumerable sets but which can not be learned with any bound on the mind changes w.r.t. an enumeration of just containing S.

Angluin [2] gave a characterization of those classes of uniformly recursive sets which are learnable from text. This characterization stated that for each language L_i there is a finite tell-tale $D_{f(i)}$ whose index $f(i)$ can be computed in the limit (or which alternatively can be enumerated via an index computable from i). This characterization can be adjusted to language learning with bounded mind changes of the second or fourth type.

Theorem 6.5 *A uniformly recursive class $S = \{L_0, L_1, \ldots\}$ of languages can be Ex-learned from text respecting bounded mind changes of the second (fourth) type iff there is an array of finite sets $D_{f(i,s)}$ such that*

- *For each i the $D_{f(i,s)}$ converge to a tell-tale $D_{f(i)}$ for L_i;*
- *If $i \leq j$ and $D_{f(i,s)} \subseteq L_j$ then $D_{f(i,s)} \subseteq D_{f(j)}$.*
- *There is no infinite chain (computable chain)*

  $$D_{f(i_0,s_0)} \subset D_{f(i_1,s_1)} \subset \ldots$$

  such that the stages $s_0 < s_1 < \ldots$ are also monotonically increasing.

7 Conclusion and Future Work

Constructive ordinals have been used successfully to count mind changes and so allow to model the convergence speed of learning rich concept classes like logic programs. This motivated to introduce two new general notions of mind change complexity. The naturalness of these notions was established by establishing a number of nice characterizations.

The four types of mind change notions described in this extended abstract have also been investigated for language learning from positive data and for classification of computable languages from piecemeal presentation of their characteristic functions. These results will be reported in the full version of the paper. Future work will explore Angluin [2] style characterization for learnability of uniformly enumerable collections of concepts in the context of mind change bounds of Type 2, 3 and

4 (see Lange and Zeugmann [18] for Type 1 characterization).

Refinements of results in this paper about the use of more expressive hypothesis spaces to reduce the mind change complexity are likely to yield insights into the design of empirical learning systems. This direction requires further investigation.

Although the present work focuses on properties and characterizations of these general bounds, future work will also look at concept classes whose learnability can be modeled using the general bounds of Type 3 and Type 4.

We would like the thank the referees for providing valuable suggestions, and for especially pointing us to the work of Parikh on quasi-well orderings and to the work of Kinber and Zeugmann [14] on a different approach to mind change complexity for BC identification.

References

[1] Andris Ambainis, Sanjay Jain and Arun Sharma: Ordinal mind change complexity of language identification. *Proceedings of the Third European Conference on Computational Learning Theory, Jerusalem* (1997), LNCS 1208 (1997) 301–316.

[2] Dana Angluin: Inductive Inference of Formal Languages from Positive Data, *Information and Control* 45 (1980) 117–135.

[3] Janis Barzdins and Rūsiņš Freivalds: Prediction and Limiting Synthesis of Recursively Enumerable Classes of Functions. In: *Theory of Algorithms and Programs, Vol. 1* (Latvian State University, edited by Janis Barzdins, Riga 1974) 112–128 (in Russian).

[4] Lenore Blum and Manuel Blum: Toward a mathematical Theory of Inductive Inference. *Information and Control*, 28 (1975) 125–155.

[5] John Case and Carl H. Smith: Comparison of Identification Criteria for Machine Inductive Inference. *Theoretical Computer Science* 25 (1983) 193–220.

[6] Rūsiņš Freivalds and Carl H. Smith: On the Role of Procrastination for Machine Learning, *Information and Computation* 107 (1993) 237–271.

[7] E. Mark Gold: Language Identification in the Limit. *Information and Control*, 10 (1967) 447–474.

[8] Klaus Peter Jantke: Monotonic and Non-Monotonic Inductive Inference. *New Generation Computing* 8 (1991) 349–360.

[9] Sanjay Jain and Arun Sharma: Elementary Formal Systems, Intrinsic Complexity and Procrastination. *Proceedings of the Ninth Annual Conference on Computational Learning Theory, Desenzano del Garda* (1996) 181–192, ACM Press.

[10] Sanjay Jain and Arun Sharma: Mind Change Complexity of Learning Logic Programs. Under preparation.

[11] Dick de Jongh and Makoto Kanazawa: Angluin's Theorem for Indexed Families of r.e. Sets and Applications. *Proceedings of the Ninth Annual Conference on Computational Learning Theory* (1996) 193–204.

[12] Shyam Kapur: Monotonic Language Learning. *Proceedings of the Third International Workshop on Algorithmic Learning Theory* (1992) 147–158.

[13] Shyam Kapur: Language learning under various types of constraint combinations. *Proceedings of the Fifth International Workshop on Algorithmic Learning Theory* (1994) 365–378.

[14] Efim Kinber and Thomas Zeugmann: Inductive inference of almost everywhere correct programs by reliably working strategies. *Elektronische Informationsverarbeitung und Kybernetik* 21 (1985) 91–100.

[15] Steffen Lange and Thomas Zeugmann: Types of Monotonic Language Learning and Their Characterization. *Proceedings of the Fifth Annual Conference on Computational Learning Theory* (1993) 377–390.

[16] Steffen Lange and Thomas Zeugmann: Monotonic versus Non-Monotonic Language Learning. *Proceedings of the Second Workshop on Nonmonotonic and Inductive Logic, Karlsruhe* (1993) 254–269.

[17] Steffen Lange and Thomas Zeugmann: Language Learning in Dependence on the Space of Hypotheses. *Proceedings of the Sixth Annual Conference on Computational Learning Theory* (1993) 127–136.

[18] Steffen Lange and Thomas Zeugmann: Language learning with bounded number of mind changes. *Proceedings of the Tenth Annual Symposium on Theoretical Aspects of Computer Science*, LNCS 665 (1993) 682–691, Springer-Verlag, Heidelberg.

[19] Wolfgang Merkle and Frank Stephan: Trees and Learning. *Proceedings of the Ninth Annual Conference on Computational Learning Theory* (1996) 270–279.

[20] Piergiorgio Odifreddi: *Classical recursion theory.* North-Holland, Amsterdam, 1989.

[21] Daniel N. Osherson, Michael Stob and Scott Weinstein: *Systems that learn.* Bradford / MIT Press, London, 1986.

[22] Karlis Podnieks: Comparing Various Concepts of Function Prediction, *Theory of Algorithms and Programs*, Latvian State University, Riga, 210:68–81,1974.

[23] Gerald E. Sacks: *Higher Recursion Theory*, Perspectives in Mathematical Logic, Springer-Verlag, Heidelberg, 1990.

[24] Rolf Wiehagen: A thesis in Inductive Inference. *Proceedings First Workshop on Nonmonotonic and Inductive Logic* (1990) 184–207.