Priced Learning

Sanjay Jain $^{\star 1}$, Junqi Ma[2], and Frank Stephan $^{\star\star 3}$

[1] School of Computing, National University of Singapore, Singapore 117417.
Email: sanjay@comp.nus.edu.sg
[2] School of Computing, National University of Singapore, Singapore 117417.
Email: ma.junqi@nus.edu.sg
[3] Department of Mathematics and Department of Computer Science, National University of Singapore,
Singapore 119076. Email: fstephan@comp.nus.edu.sg

Abstract. In iterative learning the memory of the learner can only be updated when the hypothesis changes; this results in only finitely many updates of memory during the overall learning history. Priced learning relaxes this constraint on the update of memory by imposing some price on the updates of the memory – depending on the current datum – and requiring that the overall sum of the costs incurred has to be finite. There are priced-learnable classes which are not iteratively learnable. The current work introduces the basic definitions and results for priced learning. This work also introduces various variants of priced learning.

1 Introduction

Learning from positive data in Gold's model [6] has the following basic scenario: Given a class of r.e. languages and an unknown language L from this class, the learner observes an infinite list containing all and only the elements of L. The order and multiplicity of the elements of L in the list may be arbitrary. As the learner is observing the members of the list, it outputs a sequence of hypotheses about what the input language might be. For learning the language, this sequence of hypotheses is required to converge to a grammar for L (for every input list of elements of L as above).

In general, in the above learning process, the learner can memorise all data observed so far and do comprehensive calculations without restrictions to the memory amount and usage. Several approaches have been formalised in order to restrict the amount of memory used. One of them is the strict separation between short-term and long-term memory where, whenever a datum is read, some computations are done in an unlimited short-term memory and then the data for the next round of learning is archived in a size-constrained long-term memory [4, 7]. The other approaches to limit the memory usage do not count bits and bytes, but rather allow only fixed number of elements of the input to be memorized. In such models, if the learner does not update its memory/hypothesis

* Supported in part by NUS grant numbers R146-000-181-112 and C252-000-087-001.
** Supported in part by NUS grant number R146-000-181-112.

on a receipt of a datum, then at later stages it is not able to know whether the corresponding datum was observed or not. In its most restrictive form, this type of learning is called incremental or iterative learning [3, 8, 13]. An iterative learner can memorise data only when it revises the hypothesis and not at any other point of time; thus it can, through the overall learning history, only finitely often revise its long-term memory. This restriction is quite severe and, for example, the class consisting of the set $\{1, 2, \ldots\}$ and all sets of the form $\{0, 1, \ldots, x\}$, where x is a natural number, is not iteratively learnable. To see this, note that if the learner first sees data from the infinite set $\{1, 2, \ldots\}$, then, after some time, the learner will need to converge to a grammar for $\{1, 2, \ldots\}$. Thereafter the learner cannot archive any data, in particular it cannot archive the maximum datum x seen so far. If later the datum 0 shows up, the learner knows that the input set must be of the form $\{0, 1, \ldots, x\}$; however, the learner can no longer recover the exact value of x from its memory.

The idea of priced learning is to relax the severe constraints placed on iterative learning. In priced learning, a price is charged for each update of the memory and it is required that during the learning process, the overall costs incurred is finite. In the case that this price charged is always the same constant c for each memory update, the corresponding notion would be exactly that of iterative learning. However, by allowing the price to depend on the datum x read when the memory is updated, one can give discounts for various types of data and permit, under certain circumstances, to update the memory infinitely often during the learning process. This concept of priced learning, however, still does not permit, in general, to do every possible update of the memory. Priced learning is therefore between the two extremes of iterative learning and the original unrestricted model of learning by Gold where the memory could be updated as often as desired. For the priced learning model $\mathbf{Priced}_f\mathbf{Ex}$ introduced in this paper, the costs incurred are guided by a price function f which is a parameter of the learning criterion. A $\mathbf{Priced}_f\mathbf{Ex}$-learner incurs, at every update of the memory on input datum x, the cost $1/(f(x) + 1)$ (where f is a recursive function). We refer the reader to Section 3 for the formal definitions. On reading a datum x, the learner has to decide whether it wants to update the memory so that on one hand, all important information are preserved in the long term memory and, on the other hand, the overall price of these updates does not go to infinity.

The present work investigates when, for different price functions f and g, $\mathbf{Priced}_f\mathbf{Ex}$-learnability implies $\mathbf{Priced}_g\mathbf{Ex}$-learnability. This depends on the existence of sets S such that $\sum_{x \in S} \frac{1}{f(x)+1}$ is finite but $\sum_{x \in S} \frac{1}{g(x)+1}$ is infinite (see Theorem 9 and Theorem 11). Theorem 15 gives a characterisation of classes which are $\mathbf{Priced}_f\mathbf{Ex}$-learnable for some recursive function f: these are exactly the classes which are learnable by a set-driven learner. Here a set-driven learner [9] is a learner whose output conjecture depends only on the set of elements it has observed in the input and not on their order or amount of repetitions.

Further results are based on whether a class can be learnt when the price functions are from a natural class $\mathcal{C}$ of price functions, irrespective of which price function from the class is actually chosen. Here, one may allow the learner to be chosen depending on the price function, uniformly or non-uniformly based on the program for the price function, or may even require it to be the same learner for all the price functions from the class $\mathcal{C}$. Section 6 onwards explore such questions. Let $\mathbf{MF} = \{f : f$ is recursive and unbounded and $\forall x\,[f(x) \leq f(x+1)]\}$ and $\mathbf{FF} = \{f : f$ is recursive and $\forall y\,[\mathrm{card}(f^{-1}(y))$ is finite$]\}$. Note that $\mathbf{MF}$ is a proper subset of $\mathbf{FF}$. It is shown in Theorem 17 that there exists a class of languages which can be $\mathbf{Priced}_f$-learnt for every $f \in \mathbf{FF}$, but which cannot be iteratively learnt. On the other hand, Theorem 18 shows advantages of price function being from $\mathbf{MF}$ compared to it being from $\mathbf{FF}$. Results in Section 7 deal with the question of when and what kind of uniformity constraints can be placed on learners which learn a class with respect to a price function from a class of price functions such as $\mathbf{MF}$ or $\mathbf{FF}$. Due to space constraints, some proofs are omitted.

2 Notations and Preliminaries

Any unexplained recursion theoretic notation is from Rogers' book [11]. The symbol $\mathbb{N}$ denotes the set of natural numbers $\{0, 1, 2, \ldots\}$. Subsets of natural numbers are referred to as languages. We let $\emptyset$, $\subseteq$, $\subset$, $\supseteq$ and $\supset$ respectively denote empty set, subset, proper subset, superset and proper superset. Cardinality of a set S is denoted by $\mathrm{card}(S)$. The maximum and minimum of a set S are denoted by $\max(S)$ and $\min(S)$, respectively, where $\max(\emptyset) = 0$ and $\min(\emptyset) = \infty$.

Let $\langle \cdot, \cdot \rangle$ denote a recursive pairing function which is a bijection from $\mathbb{N} \times \mathbb{N}$ to $\mathbb{N}$. Without loss of generality, we assume that pairing function is monotonic in both its arguments; in particular $\langle 0, 0 \rangle = 0$. The pairing function can be extended to pairing of n-tuples by using $\langle x_1, x_2, \ldots, x_n \rangle = \langle x_1, \langle x_2, \ldots, x_n \rangle \rangle$. We define projection functions π_1, π_2 as $\pi_1(\langle x_1, x_2 \rangle) = x_1$ and $\pi_2(\langle x_1, x_2 \rangle) = x_2$. Similarly, let $\pi_i^n(\langle x_1, x_2, \ldots, x_n \rangle) = x_i$. For a set L, let $\mathrm{cyl}(L) = \{\langle x, i \rangle : x \in L, i \in \mathbb{N}\}$.

By φ we denote a fixed *acceptable* programming system for the partial computable functions mapping $\mathbb{N}$ to $\mathbb{N}$ [11]. By φ_i we denote the partial function computed by the i-th program in the φ-system. Thus, i is also called a program/index for φ_i. Let $\mathcal{R}$ denote the class of all total recursive functions. For any partial function η, we let $\eta(x){\downarrow}$ denote that $\eta(x)$ is defined and $\eta(x){\uparrow}$ denote that $\eta(x)$ is undefined. By Φ we denote an arbitrary fixed Blum complexity measure [2] for the φ-system. Let

$$\varphi_{i,s}(x) = \begin{cases} \varphi_i(x), & \text{if } x < s \text{ and } \Phi_i(x) < s; \\ \uparrow, & \text{otherwise.} \end{cases}$$

Let $W_i = \mathrm{domain}(\varphi_i)$. W_i can be considered as the language accepted by the i-the φ-program φ_i. We also say that i is a grammar/index for W_i. Thus, $W_0, W_1, \ldots$, is an acceptable programming system for recursively enumerable languages. The symbol $\mathcal{E}$ denotes the set of all recursively enumerable (r.e.) languages. The symbol L ranges over $\mathcal{E}$; the symbol $\mathcal{L}$ ranges over subsets of $\mathcal{E}$. By $\overline{L}$, we denote the complement of L, that is $\overline{L} = \mathbb{N} - L$. By $L(x)$ we denote the value of the characteristic function of L at x; that is, $L(x) = 1$ if $x \in L$ and $L(x) = 0$ if $x \notin L$. By $W_{i,s}$ we denote the set $\{x < s : \Phi_i(x) < s\}$.

3 Models of Learning

We now present some concepts from language learning theory. A *text* is a mapping from $\mathbb{N}$ to $\mathbb{N} \cup \{\#\}$. The *content* of a text T, denoted $\mathrm{content}(T)$ is the set of natural numbers in the range of T, that is, $\{T(m) : m \in \mathbb{N}\} - \{\#\}$. We say that T is a *text for* L if $\mathrm{content}(T) = L$. Intuitively, $\#$ can be considered as pauses in the presentation of data to a learner. We let T with or without decorations range over texts. $T[n]$ denotes the initial segment of a text T with length n, that is, $T(0), T(1), \ldots, T(n-1)$.

Initial segments of texts are called finite *sequences* (or just sequences). We let σ and τ, with or without decorations, range over finite sequences. We let $\mathrm{content}(\sigma)$ denote the set of natural numbers in the range of σ. Let SEQ denote the set of all finite sequences. We assume some fixed computable ordering of members SEQ (so that we can talk about the least sequence, etc.). We let $|\sigma|$ denote the length of σ. We let Λ denote the empty sequence and $\sigma[n]$ denote the initial segment of σ of length n (where $\sigma[n] = \sigma$, if $n \geq |\sigma|$). Let $\sigma \diamond \tau$ denote the concatenation of two finite sequence σ and τ. Similarly, let $\sigma \diamond T$ denote the concatenation of finite sequence σ and text T. For $x \in \mathbb{N} \cup \{\#\}$, we let $\sigma \diamond x$ denote the concatenation of σ with the finite sequence containing just one element x. Let $\sigma \preceq T$ and $\sigma \preceq \tau$ denote that σ is an initial segment of T and τ respectively.

Definition 1 (Based on Gold [6]).

(a) A *learner* $\mathbf{M}$ is a (possibly partial) recursive mapping from $(\mathbb{N} \cup \{?\}) \times (\mathbb{N} \cup \{\#\})$ to $(\mathbb{N} \cup \{?\}) \times (\mathbb{N} \cup \{?\})$. A learner has initial memory mem_0 and initial hypothesis hyp_0.

(b) Suppose a learner $\mathbf{M}$ with initial memory mem_0 and initial hypothesis hyp_0 is given. Let $mem_0^{\mathbf{M},T} = mem_0$ and $hyp_0^{\mathbf{M},T} = hyp_0$. For $k \geq 0$, let $(mem_{k+1}^{\mathbf{M},T}, hyp_{k+1}^{\mathbf{M},T}) = \mathbf{M}(mem_k^{\mathbf{M},T}, T(k))$. Extend the definition of $\mathbf{M}$ to initial segments of texts by letting $\mathbf{M}(T[k]) = (mem_k^{\mathbf{M},T}, hyp_k^{\mathbf{M},T})$.
Intuitively, $mem_k^{\mathbf{M},T}$ is considered as the memory of the learner after having seen data $T[k]$ and $hyp_k^{\mathbf{M},T}$ is considered as the hypothesis of the learner after having seen data $T[k]$. Without loss of generality we assume that the memory is revised whenever the hypothesis is revised. Sometimes, we just say $\mathbf{M}(T[k]) = hyp_k^{\mathbf{M},T}$, where the memory of the learner is implicit.

(c) We say that $\mathbf{M}$ converges on a text T to a hypothesis hyp if the sequence $hyp_0^{\mathbf{M},T}, hyp_1^{\mathbf{M},T}, hyp_2^{\mathbf{M},T}, \ldots$ converges syntactically to hyp.

(d) We say that $\mathbf{M}$ **Ex**-learns a language L on text T if $\mathbf{M}$ is defined on all initial segments of the text T and $\mathbf{M}$ converges on T to a hypothesis hyp such that $W_{hyp} = L$.

(e) We say that $\mathbf{M}$ **Ex**-learns a language L (written: $L \in \mathbf{Ex}(\mathbf{M})$), if $\mathbf{M}$ **Ex**-learns L from each text T for L.

(f) We say that $\mathbf{M}$ $\mathbf{Ex}$-learns a class $\mathcal{L}$ of languages (written: $\mathcal{L} \subseteq \mathbf{Ex}(\mathbf{M})$) iff $\mathbf{M}$ $\mathbf{Ex}$-learns each $L \in \mathcal{L}$.
(g) $\mathbf{Ex} = \{\mathcal{L} : \exists \mathbf{M}\, [\mathcal{L} \subseteq \mathbf{Ex}(\mathbf{M})]\}$.

We say that $\mathbf{M}$ changes its mind (hypothesis, conjecture) at $T[n{+}1]$, if $hyp_n^{\mathbf{M},T} \neq hyp_{n+1}^{\mathbf{M},T}$. We say that $\mathbf{M}$ changes its memory at $T[n+1]$, if $mem_n^{\mathbf{M},T} \neq mem_{n+1}^{\mathbf{M},T}$ or $hyp_n^{\mathbf{M},T} \neq hyp_{n+1}^{\mathbf{M},T}$; that is, a learner cannot bring down the costs of mind changes by only changing the hypothesis and not doing adequate bookkeeping in the memory. Blum and Blum [1] gave a useful lemma for learnability of languages by learners based on topological considerations.

Definition 2 (Fulk [5]). Suppose $\mathbf{M}(\sigma) = (mem, hyp)$. Sequence σ is said to be a *stabilising sequence* for $\mathbf{M}$ on a language L if

(i) $\mathrm{content}(\sigma) \subseteq L$ and
(ii) for all τ such that $\sigma \subseteq \tau$ and $\mathrm{content}(\tau) \subseteq L$, $\mathbf{M}(\tau) = (\cdot, hyp)$. That is, $\mathbf{M}$ does not change its hypothesis beyond σ on any text T for L.

Definition 3 (Blum and Blum [1]). A sequence σ is said to be a *locking sequence* for $\mathbf{M}$ on a language L if

(i) σ is a stabilising sequence for $\mathbf{M}$ on L and
(ii) $W_{hyp} = L$, where $\mathbf{M}(\sigma) = (mem, hyp)$.

Lemma 4 (Blum and Blum [1]). *Suppose* $\mathbf{M}$ $\mathbf{Ex}$*-learns* L*. Then the following statements hold:*

(i) *There exists a locking sequence for* $\mathbf{M}$ *on* L*;*
(ii) *Every stabilising sequence for* $\mathbf{M}$ *on* L *is a locking sequence for* $\mathbf{M}$ *on* L*.*

A lemma similar to above can be shown for all of the criteria of learning considered in this paper.

For some of the results in this paper it is useful to consider set-driven learning, where the output of the learner depends just on the set of inputs seen.

Definition 5 (Osherson, Stob and Weinstein [9]).

(a) A learner $\mathbf{M}$ is said to be *set-driven* iff for all σ, τ, if $\mathrm{content}(\sigma) = \mathrm{content}(\tau)$, and $\mathbf{M}(\sigma) = (mem, hyp)$, then $\mathbf{M}(\tau) = (mem', hyp)$, for some memory mem'. That is, the hypothesis of the learner depends just on the content of the input and not on the exact order or the number of repetitions of the elements presented.
(b) $\mathbf{SD} = \{\mathcal{L} : \exists \mathbf{M}\, [\mathbf{M} \text{ is set-driven and } \mathcal{L} \subseteq \mathbf{Ex}(\mathbf{M})]\}$.

A related notion of rearrangement independence, where the learner is able to base its conjectures on the content and length of the input was considered by [12]. We now consider iterative learning.

Definition 6 (Based on Wexler and Culicover [13] and Wiehagen [14]).

(a) A learner $\mathbf{M}$ is said to be *iterative* if for any text T, $mem_k^{\mathbf{M},T} = hyp_k^{\mathbf{M},T}$ (where it is possible that both are undefined).
(b) $\mathbf{M}$ **ItEx**-learns a language L (class of languages $\mathcal{L}$) iff $\mathbf{M}$ is iterative and $\mathbf{M}$ **Ex**-learns L (class of languages $\mathcal{L}$).
(c) $\mathbf{ItEx} = \{\mathcal{L} : \exists \mathbf{M}\, [\mathbf{M}$ is iterative and $\mathcal{L} \subseteq \mathbf{Ex}(\mathbf{M})]\}$.

It can be shown that iterative learning is restrictive: $\mathbf{ItEx} \subset \mathbf{Ex}$ (see [14]).

There are several other variations of iterative learning considered in the literature where the memory is constrained in some way. For example, a learner $\mathbf{M}$ has k-bounded example memory [8, 10] iff for all texts T and n, $mem_n^{\mathbf{M},T}$ represents a set of size at most k, where $mem_0^{\mathbf{M},T} = \emptyset$ and $mem_{n+1}^{\mathbf{M},T} \subseteq mem_n^{\mathbf{M},T} \cup \{T(n)\}$.

As iterative learning (and most of its variations considered until now) put severe constraints on what can be memorised by a learner, we consider a variation where we allow the memory of grow arbitrarily large. However, there is a cost associated with each change of memory: if the learner changes its memory (or hypothesis) after seeing datum x, then it is charged some cost $f(x)$. Now the learner is said to identify the input language (for any given text) iff besides **Ex**-learning the language, the total cost charged to the learner on the input text is finite. Formally, this is defined as follows.

A *price function* is a recursive function mapping $\mathbb{N} \cup \{\#\}$ to $\mathbb{N}$.

Definition 7. Suppose f is a price function and $\mathcal{C}$ is a class of price functions.

(a) $\mathbf{M}$ $\mathbf{Priced}_f\mathbf{Ex}$-learns a language L (written: $L \in \mathbf{Priced}_f\mathbf{Ex}(\mathbf{M})$) iff $\mathbf{M}$ **Ex**-learns L and for all texts T for L, $\sum_{n:\mathbf{M}(T[n+1]) \neq \mathbf{M}(T[n])} \frac{1}{f(T(n))+1} < \infty$.
(b) $\mathbf{Priced}_f\mathbf{Ex} = \{\mathcal{L} : \exists \mathbf{M}\, [\mathcal{L} \subseteq \mathbf{Priced}_f\mathbf{Ex}(\mathbf{M})]\}$.
(c) $\mathbf{Priced}_{\mathcal{C}}\mathbf{Ex} = \{\mathcal{L} : \forall f \in \mathcal{C}\, [\mathcal{L} \in \mathbf{Priced}_f\mathbf{Ex}]\}$.

Note that in the above definition $\mathbf{M}(T[n])$ is taken as the pair $(mem_n^{\mathbf{M},T}, hyp_n^{\mathbf{M},T})$. For a learner $\mathbf{M}$ and price function g, $\mathbf{cost}_{\mathbf{M},g}(\sigma)$ is defined as follows. Let $S = \{m < |\sigma| : \mathbf{M}$ makes a memory change or a mind change at $\sigma[m+1]\}$. Let $\mathbf{cost}_{\mathbf{M},g}(\sigma) = \sum_{m \in S} 1/(g(\sigma(m)) + 1)$ and $\mathbf{cost}_{\mathbf{M},g}(T) = \sup_{\sigma \preceq T} \mathbf{cost}_{\mathbf{M},g}(\sigma)$. Here $\mathbf{cost}_{\mathbf{M},g}(\sigma)$ ($\mathbf{cost}_{\mathbf{M},g}(T)$) is called the cost of $\mathbf{M}$ on input σ (input T) with respect to the cost function g.

Let $price_f(S) = \sum_{x \in S} \frac{1}{f(x)+1}$. Intuitively, $price_f(S)$ denotes the cost with respect to the price function f if a mind change is made by the learner exactly once with respect to each member of S.

Let $\mathbf{M}_0, \mathbf{M}_1, \ldots$ denote a recursive enumeration of all the learning machines.

Definition 8. Suppose $\mathcal{C} \subseteq \mathcal{R}$. $\mathbf{EffPriced}_{\mathcal{C}}\mathbf{Ex} = \{\mathcal{L} : \exists f \in \mathcal{R}\, \forall i \in \mathbb{N}$ [if $\varphi_i \in \mathcal{C}$ then $\mathbf{M}_{f(i)}$ $\mathbf{Priced}_{\varphi_i}\mathbf{Ex}$-learns $\mathcal{L}$]$\}$.

4 General Cost Functions

The theorems in this section explore when $\mathbf{Priced}_f\mathbf{Ex} \subseteq \mathbf{Priced}_g\mathbf{Ex}$. Theorem 9 shows when $\mathcal{L} \in \mathbf{Priced}_f\mathbf{Ex}$ implies $\mathcal{L} \in \mathbf{Priced}_g\mathbf{Ex}$, and Theorem 11 shows when such an implication does not hold.

Theorem 9. *Suppose f and g are price functions for which no set S satisfies: $price_f(S)$ is finite and $price_g(S)$ is infinite. Then,* $\mathbf{Priced}_f\mathbf{Ex} \subseteq \mathbf{Priced}_g\mathbf{Ex}$.

Proposition 10. *Let S be an infinite r.e. set, g be a price function such that $price_g(S)$ is infinite. Then, $\mathcal{L} = \{S - \{x\} : x \in S - \min(S)\}$ is not* $\mathbf{Priced}_g\mathbf{Ex}$*-learnable.*

Proof. Suppose by way of contradiction that $\mathbf{M}$ $\mathbf{Priced}_g\mathbf{Ex}$-learns $\mathcal{L}$.

Suppose there are distinct $x, y \in S - \{\min(S)\}$ such that for some σ satisfying $\{x, y\} \subseteq \text{content}(\sigma) \subseteq S$, $\mathbf{M}$ did not change its memory and hypothesis whenever it received x, y in the input σ. Let T be a text for $S - \{x, y\}$. Let σ' be obtained from σ by deleting the occurrences of x and σ'' be obtained from σ by deleting the occurrences of y. Now, $\mathbf{M}$ converges to the same hypothesis on both $\sigma' \diamond T$ and $\sigma'' \diamond T$, however $\sigma' \diamond T$ and $\sigma'' \diamond T$ are respectively texts for two different languages $S - \{x\}$ and $S - \{y\}$ in $\mathcal{L}$. Thus, $\mathbf{M}$ does not $\mathbf{Priced}_g\mathbf{Ex}$-learn at least one of these languages.

On the other hand, if such distinct x, y and corresponding σ as above do not exist, then on any text T for a language $L \in \mathcal{L}$, the cost incurred by $\mathbf{M}$ on T is infinite. Thus, $\mathbf{M}$ does not $\mathbf{Priced}_g\mathbf{Ex}$-learn $\mathcal{L}$. $\square$

Theorem 11. *Suppose f and g are price functions such that there exists a set S satisfying that $price_f(S)$ is finite but $price_g(S)$ is infinite. Then* $\mathbf{Priced}_f\mathbf{Ex} \not\subseteq \mathbf{Priced}_g\mathbf{Ex}$.

Proof. Suppose there is a set S such that $price_f(S)$ is finite and $price_g(S)$ is infinite. Then, for each c, there exists a set S_c such that $price_f(S_c) < 1/(c+1)$ and $price_g(S_c) = \infty$. Such sets can be found by considering tails of the sum of the members in S. Hence one can, by effective search, find disjoint finite sets $\tilde{S}_0, \tilde{S}_1, \ldots$ such that $price_f(\tilde{S}_c) < 1/(c+1)$ and $price_g(\tilde{S}_c) \geq 1$. To see this, define $\tilde{S}_c$ by induction as the first finite set found (in some ordering of finite sets) such that $price_f(\tilde{S}_c) < 1/(w+1)$ and $price_g(\tilde{S}_c) \geq 1$, where $w > c + \max(\{f(x) : x \in \bigcup_{c'<c} \tilde{S}_{c'}\})$; note that the constraint on w implies that $\tilde{S}_c \cap (\bigcup_{c'<c} \tilde{S}_{c'}) = \emptyset$. Now let $\tilde{S}$ be the union of all sets $\tilde{S}_{2^k}$ for $k \in \mathbb{N}$. The set $\tilde{S}$ is an r.e. set.

One can let $\mathcal{L} = \{\tilde{S} - \{x\} : x \in \tilde{S}\}$. It is easy to see that $\mathcal{L} \in \mathbf{Priced}_f\mathbf{Ex}$, as on any input text T for $L \in \mathcal{L}$, the learner can memorise all data seen, and, in the limit, output a grammar for $\tilde{S} - \{x\}$, where x is the minimal element, if any, in $\tilde{S} - \text{content}(T)$. Note that such a learner makes a memory / mind change on any x at most once.

By Proposition 10, $\mathcal{L} \notin \mathbf{Priced}_g\mathbf{Ex}$. The theorem follows. $\square$

Theorem 9 and Theorem 11 imply that $\mathbf{Priced}_f\mathbf{Ex} = \mathbf{ItEx}$ iff $price_f(S)$ is infinite for every infinite subset S of $\mathbb{N}$. Theorem 11 can be generalised to find classes which have one learner working with all price functions which permit to learn this class. These classes can be chosen such that they are priced learnable for some but not all price functions and hence they do not coincide with iteratively learnable classes.

Theorem 12. *Given any price function f, there is a class $\mathcal{C}_f$ and a learner* $\mathbf{M}$ *such that for all price functions g, the following conditions are equivalent:*

- $\mathbf{Priced}_f\mathbf{Ex} \subseteq \mathbf{Priced}_g\mathbf{Ex}$*;*
- $\mathbf{M}$ $\mathbf{Priced}_g\mathbf{Ex}$*-learns $\mathcal{C}_f$;*
- *There is no set S such that $\mathrm{price}_f(S)$ is finite but $\mathrm{price}_g(S)$ is infinite.*

5 Priced Learning and Set-Drivenness

In this section we show that classes which are $\mathbf{Priced}_f\mathbf{Ex}$-learnable with respect to some price function f are learnable by a set-driven learner and vice versa, classes which are set-driven-learnable are $\mathbf{Priced}_g\mathbf{Ex}$-learnable with respect to some price function g. To this end we first give a modification of the definition of stabilising/locking sequences with respect to a cost function.

Definition 13. A sequence σ is a *g-cost-stabilising sequence* (*g-cost-locking sequence*) for $\mathbf{M}$ on L, if the following conditions hold:

(a) σ is a stabilising sequence (locking sequence) for $\mathbf{M}$ on L and
(b) for all τ such that $\sigma \preceq \tau$ and $\mathrm{content}(\tau) \subseteq L$, $\mathbf{cost}_{\mathbf{M},g}(\tau) \leq \mathbf{cost}_{\mathbf{M},g}(\sigma) + 1$.

The following can be proved in a way similar to locking sequence lemma in [1].

Lemma 14 (Based on Blum and Blum [1]). *If* $\mathbf{M}$ $\mathbf{Priced}_g\mathbf{Ex}$*-learns L then there exists a g-cost-locking sequence for* $\mathbf{M}$ *on L; Furthermore, all g-cost-stabilising sequences for* $\mathbf{M}$ *on L are g-cost-locking sequences for* $\mathbf{M}$ *on L.*

Furthermore, note that for any finite set S contained in some set learned by $\mathbf{M}$, and any price function g one can check if σ is a g-cost-stabilising sequence for $\mathbf{M}$ on S. This holds as the number of memory / mind changes of $\mathbf{M}$ on any text extending σ for S will be at most $\max(\{g(x)+1 : x \in S \cup \{\#\}\})$ if σ is indeed a g-cost-stabilising sequence for $\mathbf{M}$ on S. Thus, we can check if the tree of memory / mind changes for $\mathbf{M}$ on S is finite above σ.

Theorem 15. $\bigcup_{g\in\mathcal{R}} \mathbf{Priced}_g\mathbf{Ex} = \mathbf{SD}$.

Proof. Suppose $\mathcal{L} \in \mathbf{SD}$ as witnessed by $\mathbf{M}$. Let $g(x) = 1/2^x$. Let $\mathbf{N}(\sigma) = (\mathrm{content}(\sigma), hyp)$, where hyp is the hypothesis of $\mathbf{M}$ on input σ. Note that, as $\mathbf{M}$ is set-driven, $\mathbf{N}$ can easily obtain the hypothesis of $\mathbf{M}$ on input σ, using its memory $\mathrm{content}(\sigma)$, by running $\mathbf{M}$ on any τ such that $\mathrm{content}(\tau) = \mathrm{content}(\sigma)$. It is easy to verify that $\mathbf{N}$ $\mathbf{Priced}_g\mathbf{Ex}$-learns $\mathcal{L}$.

Now suppose $\mathcal{L} \in \mathbf{Priced}_g\mathbf{Ex}$ as witnessed by $\mathbf{M}$. Then, consider the following set-driven learner $\mathbf{N}$. The aim of the set-driven learner $\mathbf{N}$ is to find the least g-cost-stabilising sequence for $\mathbf{M}$ on the input language. $\mathbf{N}$ on input σ searches for the least g-cost-stabilising sequence τ for $\mathbf{M}$ on $\mathrm{content}(\sigma)$, if any, of length at most $|\mathrm{content}(\sigma)|$. Note that one can check if such a stabilising sequence exists. Now, $\mathbf{N}(\sigma) = (\mathrm{content}(\sigma), hyp)$, where hyp is the hypothesis of $\mathbf{M}$ on input τ if a τ as above exists; otherwise hyp is a canonical grammar for $\mathrm{content}(\sigma)$.

For a finite language $L \in \mathcal{L}$, whether there exists a g-cost-stabilising sequence for $\mathbf{M}$ on L of length at most $\mathrm{card}(L)$ or not, the learner $\mathbf{N}$ will output a grammar for L after it has received all the elements of L. Thus, $\mathbf{N}$ would $\mathbf{Ex}$-learn L.

For an infinite language $L \in \mathcal{L}$, there exists a g-cost-stabilising sequence for $\mathbf{M}$ on L of finite length, and thus for large enough initial segment of any text T for L, $\mathbf{N}$ will find the least g-cost-stabilising sequence τ. Then, $\mathbf{N}$ will output the hypothesis of $\mathbf{M}$ on τ as its hypothesis. Thus, $\mathbf{N}$ would $\mathbf{Ex}$-learn L.

As $\mathbf{N}$ defined above is set-driven, we have that $\mathcal{L} \in \mathbf{SD}$. $\square$

Theorem 9 and Theorem 11 along with $\mathbf{SD} = \mathbf{Priced}_g\mathbf{Ex}$, for the g used in the proof of Theorem 15, imply that $\mathbf{Priced}_f\mathbf{Ex} = \mathbf{SD}$ iff $\mathit{price}_f(\mathbb{N})$ is finite.

6 Classes of Cost Functions

In this section we will consider priced learning when the price function might be any of the functions in a class of price functions. In particular we look at the classes of monotonic functions ($\mathbf{MF}$) and finite-one functions ($\mathbf{FF}$) which map only finitely many inputs to the same output:

$$\begin{aligned}\mathbf{MF} &= \{f \in \mathcal{R} : \forall x\,[f(x) \leq f(x+1)] \text{ and } f \text{ is unbounded}\};\\ \mathbf{FF} &= \{f \in \mathcal{R} : \forall y\,[\mathrm{card}(f^{-1}(y)) \text{ is finite}]\}.\end{aligned}$$

The following lemma is useful for proving some results in this paper. It shows that if $\mathcal{L}$ is $\mathbf{Ex}$-learnable, then the cylindrification of $\mathcal{L}$ is $\mathbf{Priced}_{\mathbf{FF}}\mathbf{Ex}$-learnable.

Lemma 16. *If $\mathcal{L} \in \mathbf{Ex}$ and $\mathcal{L}' = \{\mathrm{cyl}(L) : L \in \mathcal{L}\}$ then $\mathcal{L}' \in \mathbf{EffPriced}_{\mathbf{FF}}\mathbf{Ex}$.*

The next theorem gives a class $\mathcal{L}$ that can be $\mathbf{Priced}_f\mathbf{Ex}$-learnt with respect to every price function f in $\mathbf{FF}$, even though $\mathcal{L}$ is not iteratively learnable. Moreover, the $\mathbf{Priced}_f\mathbf{Ex}$-learner can be obtained effectively from an index for f.

Theorem 17. $\mathbf{EffPriced}_{\mathbf{FF}}\mathbf{Ex} \not\subseteq \mathbf{ItEx}$.

Proof. The class $\mathcal{L}_1 \cup \mathcal{L}_2$ with

$$\begin{aligned}\mathcal{L}_1 &= \{W_e : e = \min(W_e) \text{ and } \forall x\,[\{2x, 2x+1\} \not\subseteq W_e]\} \text{ and}\\ \mathcal{L}_2 &= \{L : \mathrm{card}(L) < \infty \text{ and } \exists x\,[\{2x, 2x+1\} \subseteq L]\}\end{aligned}$$

is $\mathbf{Ex}$-learnable: The learner, on input σ, checks whether there is an x with $2x, 2x+1 \in \mathrm{content}(\sigma)$. If so then the learner conjectures a canonical index for $\mathrm{content}(\sigma)$ else the learner conjectures $\min(\mathrm{content}(\sigma))$. So, on a text T with both $2x, 2x+1 \in \mathrm{content}(T)$, the learner converges to a canonical index of $\mathrm{content}(T)$ whenever the latter is finite and thus the learner $\mathbf{Ex}$-learns $\mathcal{L}_2$; on a text T which contains at most one of $2x, 2x+1$ for any x, the learner converges to the minimum of $\mathrm{content}(T)$ and thus the learner $\mathbf{Ex}$-learns $\mathcal{L}_1$. Let

$$\mathcal{L} = \{\mathrm{cyl}(L) : L \in \mathcal{L}_1 \cup \mathcal{L}_2\}.$$

By Lemma 16, $\mathcal{L} \in$ **EffPriced**$_{\mathbf{FF}}$**Ex**. Now it remains to show that $\mathcal{L} \notin$ **ItEx**. Suppose by way of contradiction that **M ItEx**-learns $\mathcal{L}$. Then by Kleene's recursion theorem there exists an e such that W_e is the set of all x for which $\langle x, 0\rangle$ occurs in some σ_s which are defined below. Here, initially, $\sigma_0 = \langle e, 0\rangle$ and in stage s below, σ_{s+1} is constructed.

```
Stage $s$.
  1.   Search for an extension $\tau$ of $\sigma_s$ satisfying the following conditions:
         (i) $\mathbf{M}(\sigma_s) \neq \mathbf{M}(\tau)$;
        (ii) for all $x$ and $i$, if $\langle 2x, i\rangle \in \text{content}(\tau)$, then for all $j$, $\langle 2x+1, j\rangle \notin \text{content}(\tau)$;
       (iii) for all $i$, for all $x < e$, $\langle x, i\rangle \notin \text{content}(\tau)$.
  2.   If and when such a $\tau$ is found, let $\sigma_{s+1}$ be an extension of $\tau$ satisfying the following conditions:
         (i) $\forall x, i\,[[\langle x, i\rangle \in \text{content}(\tau)] \Rightarrow [\forall j \leq s\,[\langle x, j\rangle \in \text{content}(\sigma_{s+1})]]]$;
        (ii) $\forall x, i\,[[\langle x, i\rangle \in \text{content}(\sigma_{s+1})] \Rightarrow [\exists j\,[\langle x, j\rangle \in \text{content}(\tau)]]]$.
  3.   Go to stage $s+1$.
End Stage $s$.
```

Now if there are infinitely many stages in the above construction, then $T = \bigcup_{s\in\mathbb{N}} \sigma_s$ is a text for cyl(W_e), where $W_e \in \mathcal{L}_1$. However, **M** makes infinitely many mind changes on T. On the other hand if stage s starts but does not finish, then, for $y > \max(\{x : \langle x, 0\rangle \in \text{content}(\sigma_s)\})$, let $L_y = \{x : \langle x, 0\rangle \in \text{content}(\sigma_s)\} \cup \{2y, 2y+1\}$. Then for all texts T extending σ_s for any cyl(L_y) with $y > \max(\{x : \langle x, 0\rangle \in \text{content}(\sigma_s)\})$, $\mathbf{M}(T)$ converges to $\mathbf{M}(\sigma_s)$, and thus it fails to **ItEx**-learn $\mathcal{L}$, as all such L_y belong to $\mathcal{L}_2$. □

The next theorem shows that some class $\mathcal{L}$ can be **Priced**$_f$**Ex**-learnt with respect to every price function in **MF**, but not with respect to some price function in **FF**. Furthermore, the **Priced**$_f$**Ex** learner for $f \in$ **MF** can be obtained effectively from an index for f.

Theorem 18. **EffPriced**$_{\mathbf{MF}}$**Ex** $\not\subseteq$ **Priced**$_{\mathbf{FF}}$**Ex**.

Proof. Let $F_0, F_1, \ldots$ be an enumeration of disjoint finite sets such that for all i, if φ_i is total, then for all $x > i$, there exists a $y > \varphi_i(x)$ in F_x. Note that such an enumeration can be easily constructed.

Let $L_0 = \bigcup_{x>0} F_x$. For $e \in \mathbb{N}$, let $L_{e+1} = \bigcup_{x\in D_e\cup\{0\}} F_x$. Let $\mathcal{L} = \{L_e : e \in \mathbb{N}\}$.

Claim. $\mathcal{L} \in$ **EffPriced**$_{\mathbf{MF}}$**Ex**.

Let h be a recursive function such that for all i, $\varphi_{h(i)}(x)$ is the first z found in a search such that $\varphi_i(z) > 2^x$. Now, if $\varphi_i \in$ **MF**, then for all $x > h(i)$, there exists a $y \in F_x$ such that $y > \varphi_{h(i)}(x)$ and thus $\varphi_i(y) > 2^x$.

Given an index i for a cost function $g \in$ **MF**, consider the following learner **M** obtained effectively from i. Learner **M** will have a finite set as its memory.

Initially, $\mathbf{M}$'s memory is $\emptyset$. Whenever $\mathbf{M}$ receives $y \in F_x$ as input, it adds x to the memory if x is not already in the memory and $[x \leq h(i)$ or $g(y) > 2^x]$; otherwise the memory is unchanged. If the memory does not contain 0, then $\mathbf{M}$ outputs a canonical grammar for L_0. Otherwise it outputs a canonical grammar for L_{e+1}, where $D_e = S - \{0\}$, where S is the set in $\mathbf{M}$'s memory. It is easy to verify that for any text T, $\mathbf{cost}_{\mathbf{M},g}(T)$ is finite and $\mathbf{M}$ $\mathbf{Priced}_g\mathbf{Ex}$-learns $\mathcal{L}$.

Claim. $\mathcal{L} \notin \mathbf{Priced}_{\mathbf{FF}}\mathbf{Ex}$.

To see this, let g be such that $g(y) = x$, for $y \in F_x$. Suppose by way of contradiction that $\mathbf{M}$ $\mathbf{Priced}_g\mathbf{Ex}$-learns $\mathcal{L}$. Now, consider $\mathbf{M}$'s behaviour on text for L_0, where the elements are presented to it in order of elements from $F_1, F_2, \ldots$. If for every x, $\mathbf{M}$ makes mind change on some input from F_x, then clearly the cost is infinite. Otherwise, for some x, $\mathbf{M}$ did not make any mind change on the above text when presented with elements from F_x. Now, if after elements of F_x, $\mathbf{M}$ only gets elements from F_0, then $\mathbf{M}$ cannot distinguish between the input being L_e or $L_{e'}$, where $D_e = \{1, 2, \ldots, x\}$ and $D_{e'} = \{1, 2, \ldots, x-1\}$.

The above two claims prove the theorem. □

The following theorem shows that some class of languages $\mathcal{L}$ can be learnt with respect to every price function in $\mathbf{FF}$, but one cannot effectively obtain a learner for even monotonic price functions.

Theorem 19. $\mathbf{Priced}_{\mathbf{FF}}\mathbf{Ex} \not\subseteq \mathbf{EffPriced}_{\mathbf{MF}}\mathbf{Ex}$.

Proof. We will define the class $\mathcal{L}$ in dependence of a set $A \subseteq \{\langle x, r\rangle : x \geq 1, r \in \mathbb{N}\}$ to be constructed below by letting

$$L_m = \{0\} \cup A \cap \{\langle x, r\rangle : x \leq m, r \in \mathbb{N}\};$$
$$\mathcal{L} = \{A\} \cup \{L_m : m \in \mathbb{N}\}.$$

We will have the property that for all $g \in \mathbf{FF}$, for all but finitely many x, there exists an r such that $\langle x, r\rangle \in A$ and $g(\langle x, r\rangle) > 2^x$.

The above property allows for easy learning of $\mathcal{L}$ in the model $\mathbf{Priced}_{\mathbf{FF}}\mathbf{Ex}$. To see this, for $g \in \mathbf{FF}$, let R_g be the set of finitely many x such that for all r, $g(\langle x, r\rangle) \leq 2^x$. Then, the $\mathbf{Priced}_g\mathbf{Ex}$-learner $\mathbf{M}$ is defined as follows. The memory of $\mathbf{M}$ is a finite set. The memory mem_σ of $\mathbf{M}$ after having seen input σ consists of the following elements: (i) 0 if $0 \in \mathrm{content}(\sigma)$, (ii) all $x \in R_g$ such that for some $r \in \mathbb{N}$, $\langle x, r\rangle \in \mathrm{content}(\sigma)$, and (iii) all $x \geq 1$ such that for some $r \in \mathbb{N}$, $\langle x, r\rangle \in \mathrm{content}(\sigma)$ and $g(\langle x, r\rangle) > 2^x$. Furthermore, the hypothesis of the learner $\mathbf{M}$ after having seen input σ is: (a) a canonical index for A if $0 \notin mem_\sigma$, and (b) a canonical index for L_m, if $0 \in mem_\sigma$ and $m = \max(mem_\sigma)$. It is easy to verify that $\mathbf{M}$ is a $\mathbf{Priced}_g\mathbf{Ex}$-learner for $\mathcal{L}$.

We will now construct A. For this we will define functions $h_{i,j}$ along with sets $B_{i,j}$, for $j < 2^{i+1}$. The functions $h_{i,j}$, if total, will be in $\mathbf{MF}$. $B_{i,j}$ will be finite, but may change over time. Let $B^t_{i,j}$ denote its value at time t, where we will only start with $B_{i,j}$ at time $t \geq i$ (so it is assumed to be empty before that).

Intuitively, aim of $B_{i,j}$ and $h_{i,j}$, $j < 2^{i+1}$ is to make sure that φ_i does not witness that $\mathcal{L} \in \mathbf{EffPriced_{MF}Ex}$.

Let $X_{i,j}$, $j < 2^{i+1}$, denote the 2^{i+1} different subsets of $\{0, 1, 2, \ldots, i\}$.

In the definitions below, we assume some steps to be atomic so that there is no interference between different parts. For ease of notation we will write $\varphi_i(h_{i,j})$ below rather than φ_i(index for $h_{i,j}$).

```
Definition of $A$.
    Stage $s$
            Each stage $s$ is supposed to be atomic.
    1.      Enumerate $\langle s, 0\rangle$ in $A$.
    2.      For each $i < s$,
            2.1. Let $x < s$ be least, if any, such that
                        $x \notin \bigcup_{j<i,k<2^{j+1}} B^s_{j,k}$ and
                        for all $r$ such that $\langle x, r\rangle \in A$ enumerated up to now
                          $[\varphi_{i,s}(\langle x, r\rangle)\downarrow \leq 2^x$ or $\varphi_{i,s}(\langle x, r\rangle)\uparrow]$.
            2.2. If there is such an $x$ as above and $\exists r \leq s\, [\varphi_{i,s}(\langle x, r\rangle)\downarrow > 2^x]$
                   Then enumerate one such $\langle x, r\rangle$ in $A$.
            End For
            Go to stage $s + 1$
    End Stage $s$
```

Note that the construction of different $h_{i,j}$ is run in parallel (in dovetailing way, along with procedure for A above) for all $i \in \mathbb{N}$, $j < 2^{i+1}$.

```
Definition of $h_{i,j}$.
    Let $h_{i,j}(0) = 1$.
    Initially $\sigma_{i,j} = \Lambda$.
    Loop
    0.      Let $\sigma_{i,j}$ be extension of previous $\sigma_{i,j}$ so that it contains all of $A$
            enumerated up to now.
            (Above step is assumed to be atomic.)
    1.      Suppose $h_{i,j}$ has already been defined on inputs $\leq w$ up to now,
            and the maximum value in the range of $h_{i,j}$ up to now is $z$.
    2.      Pick $B_{i,j}$ of size $z + 2$ containing elements $> i + w$, such that none
            of the elements of $B_{i,j}$ have been used in any $B_{i',j'}$ at any earlier
            point in the construction up to now and all the elements of $B_{i,j}$
            are larger than any element enumerated by $A$ up to now.
            (We assume the above step to be atomic.)
    3.      Wait until for each $y \in B_{i,j}$,
             (i) $\langle y, 0\rangle$ is enumerated in $A$ and
            (ii) for each $i' \in X_{i,j}$, a pair $\langle y, r\rangle$ is enumerated into $A$ with
                 $\varphi_{i'}(\langle y, r\rangle) > 2^y$.
    4.      Extend $h_{i,j}$ monotonically (non-decreasing) so that $h_{i,j}(\langle y, r\rangle) =$
            $z+1$, for each $\langle y, r\rangle \in A$ enumerated up to now such that $y \in B_{i,j}$.
            (We assume above step to be atomic.)
```

5. Keep extending $h_{i,j}$ monotonically (non-decreasing), while searching for an extension τ of $\sigma_{i,j}$ such that
   (i) content(τ) is contained in A, and
   (ii) for each $y \in B_{i,j}$, there exists a $\langle y, r\rangle$ in content(τ) such that $\mathbf{M}_{\varphi_i(h_{i,j})}$ made a mind change/memory change when it received $\langle y, r\rangle$ as input.
6. If such an extension τ is found, update $\sigma_{i,j}$ to τ and go to the next iteration of the loop. If no extension as above is found, then this iteration of the loop never ends.

End Loop

For each j and $k < 2^{j+1}$, let $B_{i,j}$ denote the eventual value of $B_{i,j}$ (if this value does not exist, then let $B_{i,j}$ be $\emptyset$).

Now consider any i such that $\varphi_i \in \mathbf{FF}$. Let $C_i = \bigcup_{i'<i, j'<2^{i'+1}} B_{i',j'}$. Note that for $x \notin C_i$, eventually some element of the form $\langle x, r\rangle$, with $\varphi_i(\langle x, r\rangle) > 2^x$ will be enumerated in A. Thus, we have that $\mathcal{L} \in \mathbf{Priced_{FF}Ex}$.

Now we show that $\mathcal{L} \notin \mathbf{EffPriced_{MF}Ex}$. Suppose by way of contradiction that φ_i witnesses $\mathcal{L} \in \mathbf{EffPriced_{MF}Ex}$, that is, for all k, if $\varphi_k \in \mathbf{MF}$, then $\mathbf{M}_{\varphi_i(k)}$ is a $\mathbf{Priced}_{\varphi_k}\mathbf{Ex}$-learner for $\mathcal{L}$. Let j be such that $X_{i,j}$ is the set of $i' \leq i$ satisfying for all $y \notin \bigcup_{i''<i', j''<2^{i''+1}} B_{i'',j''}$, some $\langle y, r\rangle$ with $\varphi_{i'}(\langle y, r\rangle) > 2^y$ is enumerated in A. Now consider the construction of $h_{i,j}$. Now consider the following cases.

Case 1: Some iteration of the Loop for $h_{i,j}$ does not terminate.

In this case we have that $\mathbf{M}_{\varphi_i(h_{i,j})}$ fails to $\mathbf{Priced}_{h_{i,j}}\mathbf{Ex}$ learn $\mathcal{L}$. To see this let $\sigma_{i,j}$ be as in the iteration of the loop (for $h_{i,j}$) which starts but does not terminate. Consider τ which extends $\sigma_{i,j}$, and contains elements exactly from content$(\sigma_{i,j}) \cup (A \cap \{\langle y, r\rangle : y \in B_{i,j}\})$, where these $\langle y, r\rangle$ appear in τ in increasing order of y. Then by step 5 not succeeding in the loop, there exists a $y \in B_{i,j}$, for which $M_{\varphi_i(h_{i,j})}$ did not make a mind change/memory change when receiving $\langle y, r\rangle$ as input, for all $\langle y, r\rangle \in A$. Thus, we can fool the learner $\mathbf{M}_{\varphi_i(h_{i,j})}$ by giving 0 just after giving the elements $\langle y, r\rangle$ corresponding to the y above, and then extending it using all elements from L_{y-1}. Then, the learner $\mathbf{M}_{\varphi_i(h_{i,j})}$ is not able to distinguish between input being L_{y-1} or L_y.

Case 2: All iterations of the loop for $h_{i,j}$ terminate.

In this case consider the loop executions in which for each $y \in B_{i,j}$ none of the $i' \leq i$, $i' \notin X_{i,j}$ have a $\langle y, r\rangle \in A$ with $\varphi_{i'}(\langle y, r\rangle) > 2^y$. Note that all but finitely many loop executions have this property. In each of such executions, the construction would have found a mind change/memory change which costs in total at least 1 to the learner (as in each iteration, size of $B_{i,j}$ is $z + 2$, and the cost of memory / mind change on $\langle y, r\rangle$, $y \in B_{i,j}$ is $\frac{1}{z+2}$).

Thus, $\mathbf{M}_{\varphi_i(h_{i,j})}$ does not $\mathbf{Priced_{MF}Ex}$-learn $\mathcal{L}$. □

7 Other Uniformity Criteria

In this section we consider some uniformity criteria for priced learning.

Definition 20. Suppose $\mathcal{C} \subseteq \mathcal{R}$.

(a) $\mathbf{UniPriced}_{\mathcal{C}}\mathbf{Ex} = \{\mathcal{L} : \exists \mathbf{M}\, \forall f \in \mathcal{C}\, [\mathbf{M}\ \mathbf{Priced}_f\mathbf{Ex}\text{-learns } \mathcal{L}]\}$.
(b) $\mathbf{EffBPriced}_{\mathcal{C}}\mathbf{Ex} = \{\mathcal{L} : \exists g \in \mathcal{R}\, \forall i\, [\text{if } \varphi_i \in \mathcal{C} \text{ then } \forall b \geq i\, [\mathbf{M}_{g(b)}\ \mathbf{Priced}_{\varphi_i}\mathbf{Ex}\text{-learns } \mathcal{L}]]\}$.
(c) $\mathbf{UniFPriced}_{\mathcal{C}}\mathbf{Ex} = \{\mathcal{L} : \exists \mathbf{M}\, [\forall f \in \mathcal{C}\, [\text{card}(\mathcal{L} - \mathbf{Priced}_f\mathbf{Ex}(\mathbf{M})) \text{ is finite }] \text{ and } \mathbf{M}\ \mathbf{Ex}\text{-learns } \mathcal{L}\ ]\}$.

Intuitively, for $\mathbf{UniPriced}_{\mathcal{C}}\mathbf{Ex}$, the same learner $\mathbf{M}$ succeeds for all price functions in $\mathcal{C}$. For $\mathbf{EffBPriced}_{\mathcal{C}}\mathbf{Ex}$-learning the learner can be obtained effectively from a bound on the index for the price function.

For $\mathbf{UniFPriced}_{\mathcal{C}}\mathbf{Ex}$-learning, the same learner $\mathbf{M}$ $\mathbf{Ex}$-learns the class $\mathcal{L}$ and for all price functions f in $\mathcal{C}$, $\mathbf{M}$ $\mathbf{Priced}_f\mathbf{Ex}$-learns almost all the members of the class $\mathcal{L}$. Thus, the cost of learning can go infinite only for finitely many languages in the class for each price function $f \in \mathcal{C}$.

In addition we consider the case where the learner gets the cost function f as input rather than via index for it. In $\mathbf{ValPriced}_{\mathcal{C}}\mathbf{Ex}$ learning of $\mathcal{L}$, for all $f \in \mathcal{C}$, the same learner $\mathbf{M}$ $\mathbf{Priced}_f\mathbf{Ex}$-learn each member of $\mathcal{L}$ when, in each iteration, instead of a single datum x the pair $(x, f(x))$ is presented to the learner in the iteration, so that the learner knows how expensive it is to process a datum.

Theorem 21. (a) *Let $\mathcal{C}$ be a class of price functions.*
$\mathbf{ValPriced}_{\mathcal{C}}\mathbf{Ex} \subseteq \mathbf{EffPriced}_{\mathcal{C}}\mathbf{Ex} \subseteq \mathbf{Priced}_{\mathcal{C}}\mathbf{Ex}$.
(b) $\mathbf{EffPriced}_{\mathbf{FF}}\mathbf{Ex} \not\subseteq \mathbf{ValPriced}_{\mathbf{MF}}\mathbf{Ex}$.

Theorem 22. $\mathbf{ValPriced}_{\mathbf{FF}}\mathbf{Ex} \not\subseteq \mathbf{EffBPriced}_{\mathbf{MF}}\mathbf{Ex}$.

Theorem 23. $\mathbf{UniFPriced}_{\mathbf{FF}}\mathbf{Ex} \not\subseteq \mathbf{Priced}_{\mathbf{MF}}\mathbf{Ex}$.

Corollary 24. $\mathbf{UniFPriced}_{\mathbf{FF}}\mathbf{Ex} \not\subseteq \mathbf{EffPriced}_{\mathbf{MF}}\mathbf{Ex} \cup \mathbf{ValPriced}_{\mathbf{MF}}\mathbf{Ex}$.

Theorem 25. $\mathbf{UniFPriced}_{\mathbf{MF}}\mathbf{Ex} \not\subseteq \mathbf{Priced}_{\mathbf{FF}}\mathbf{Ex} \cup \mathbf{UniFPriced}_{\mathbf{FF}}\mathbf{Ex}$.

Theorem 26. $\mathbf{ValPriced}_{\mathbf{FF}}\mathbf{Ex} \not\subseteq \mathbf{UniFPriced}_{\mathbf{MF}}\mathbf{Ex}$.

It is open whether $\mathbf{UniPriced}_{\mathbf{MF}}\mathbf{Ex}$ or $\mathbf{UniPriced}_{\mathbf{FF}}\mathbf{Ex}$ contain a class which cannot be $\mathbf{ItEx}$-learnt. Similarly, it is open whether $\mathbf{EffBPriced}_{\mathbf{MF}}\mathbf{Ex}$ or $\mathbf{EffBPriced}_{\mathbf{FF}}\mathbf{Ex}$ contain a class which cannot be $\mathbf{ItEx}$-learnt.

8 Conclusions

In this paper we considered a generalization of memory limited learning, called priced learning, where the learner can update its memory based on any datum it receives, but it has a cost $\frac{1}{f(x)+1}$ associated with it, where f is the price function. The learning is said to be successful if the overall cost in the learning process is finite. We gave a characterization that the classes learnable as above with respect to some cost function are exactly the classes learnable by a set-driven learner. We also gave complete picture of when different price functions lead to different learnable classes. In addition we considered when priced learning is possible for all price functions from a class of functions, and when a learner can be effectively found from an index for a price function. It is open at present whether there exists a single learner which learns a non iteratively learnable class from all price functions which are monotonically non-decreasing and unbounded. Similar question is also open for the price functions having each number in the range only finitely often. These questions are open even for the case when the learner is given a bound on an index for the price function.

Acknowledgements: We thank the referees for several helpful comments.

References

1. Lenore Blum and Manuel Blum. Toward a mathematical theory of inductive inference. *Information and Control*, 28:125–155, 1975.
2. Manuel Blum. A machine independent theory of the complexity of recursive functions. *Journal of the Association of Computing Machinery*, 14:322–336, 1967.
3. John Case, Sanjay Jain, Steffen Lange and Thomas Zeugmann. Incremental concept learning for bounded data mining. *Information and Computation*, 152:74–110, 1999.
4. Rusins Freivalds, Efim Kinber and Carl H. Smith. On the impact of forgetting on learning machines. *Journal of the ACM*, 42:1146–1168, 1995.
5. Mark Fulk. Prudence and other conditions on formal language learning. *Information and Computation*, 85:1–11, 1990.
6. E. Mark Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.
7. Efim Kinber and Frank Stephan. Language learning from texts: mind changes, limited memory and monotonicity. *Information and Computation*, 123:224–241, 1995.
8. Steffen Lange and Thomas Zeugmann. Incremental learning from positive data. *Journal of Computer and System Sciences*, 53:88–103, 1996.
9. Daniel Osherson, Michael Stob and Scott Weinstein. Learning strategies. *Information and Control*, 53:32–51, 1982.
10. Daniel Osherson, Michael Stob and Scott Weinstein. *Systems That Learn, An Introduction to Learning Theory for Cognitive and Computer Scientists.* Bradford — The MIT Press, Cambridge, Massachusetts, 1986.
11. Hartley Rogers. *Theory of Recursive Functions and Effective Computability.* McGraw-Hill, 1967. Reprinted by MIT Press in 1987.
12. Gisela Schäfer-Richter. Uber Eingabeabhängigkeit und Komplexität von Inferenzstrategien. Ph.D. Thesis, RWTH Aachen, 1984.
13. Kenneth Wexler and Peter W. Culicover. *Formal Principles of Language Acquisition.* Cambridge, Massachusetts, The MIT Press, 1980.
14. Rolf Wiehagen. Limes-Erkennung rekursiver Funktionen durch spezielle Strategien. *Elektronische Informationsverarbeitung und Kybernetik* (EIK), 12:93–99, 1976.