The Structure of a Logarithmic Advice Class

Extended Abstract

Montserrat Hermo
Departamento de Lenguajes y Sistemas Informáticos
Universidad del Pais Vasco
Apdo. 649
E-20080 San Sebastian. Spain
e-mail: montse@si.ehu.es

Abstract

The complexity class Full-P/log, corresponding to a form of logarithmic advice for polynomial time, is studied.

In order to understand the inner structure of this class, we characterize Full-P/log in terms of Turing reducibility to a special family of sparse sets. Other characterizations of Full-P/log, relating it to sets with small information content, were already known. These used tally sets whose words follow a given regular pattern and tally sets that are regular in a resource-bounded Kolmogorov complexity sense.

We obtain here relationships between the equivalence classes of the mentioned tally and sparse sets under various reducibilities, which provide new knowledge about the logarithmic advice class.

Another way to measure the amount of information encoded in a language in a nonuniform class, is to study the relative complexity of computing advice functions for this language. We prove bounds on the complexity of advice functions for Full-P/log and for other subclasses of it. As a consequence, Full-P/log is located in the Extended Low Hierarchy.

1 Introduction

The study of nonuniform complexity classes appears in order to measure the size of algorithms accepting finite sets. In uniform models of computation, a program is valid for arbitrarily long inputs, while in nonuniform models, each program is valid only for inputs of a fixed length. There are many well known models for both. The traditional example of uniform models is the Turing machine and the most important representative of the nonuniform family is the Boolean circuit model.

There are several connections between them. Uniformity conditions can be set on the nonuniform model to obtain uniform complexity classes, or vice-versa advice functions can be used to extend uniform classes into nonuniform ones. Our framework is this second approach. More concretely: we study a particular nonuniform class defined by an advice function family.

The main characteristic of advice functions is that the value of the advice does not depend on the particular instance we want to decide, but just on its length. Hence, these advices provide external information to the algorithms.

Advice functions were introduced in [KL80], where two main classes were selected as bounds for the length of the advice words: polynomial bounds and logarithmic bounds. The most widely studied class are P/poly of the sets decidable in polynomial time, with the help of polynomially long advice strings (see [BDG88] and the references there), and P/ log corresponding also to polynomial time and logarithmic advices. [BS92].

We study here a modification of P/ log proposed in [Ko87] and also studied in [BHM92]. The reason for choosing this modified class is that P/ log is not closed under polynomial time Turing reducibility, so it is not possible to characterize P/ log as the reduction class of some family of sets, in the spirit of the existing characterizations of P/poly. Instead of studying P/ log, we work with the class denoted by Full-P/ log, in which the closure under polynomial time reductions is obtained via a more restrictive condition in the definition [BHM92]. Another argument to consider this class of our interest is the following. In [BGSS93] a model of neural nets whose processors can compute real functions is characterized with circuits. In fact, the class of languages accepted by this type of nets is exactly P/poly. But an interesting special case arises when considering natural bounds for the complexity of the net weights. For instance, if these bounds are in a particular low Kolmogorov complexity set, then the class of languages recognized by this nets is Full-P/ log.

This paper is organized as follows: Section 2 introduces notation and gives basic definitions and results. In Section 3, we present a new characterization of Full-P/ log as the class of languages that are Turing reducible to an especial family of sparse languages, namely "very low-Kolmogorov complexity sparse sets". There were already, other characterizations of Full-P/ log as the class of languages Turing reducibles to tally sets [BHM92]. On the one hand, tally sets whose words follow a regular pattern, and on the other hand, tally sets that are regular in terms of a resource-bounded Kolmogorov complexity. Following the ideas from Allender and Rubinstein [AR86] we show that the special family of sparse sets is exactly the class of languages that are isomorphic to tally sets, with regular pattern words. But curiously, we separate the family of sparse sets from the languages that are isomorphic to the second type of tally sets, namely, sets which are regular in a resource-bounded Kolmogorov complexity sense. To finish this section, we study, not only classes of isomorphic languages, but also classes that are Turing equivalent to the mentioned subsets of Full-P/log. In this case all of Turing equivalent classes coincide, and they are characterized as families having some kind of self-producible circuits. In this context, regarding the question of separating Turing reducibility from Turing equivalence, the answer seems to be very difficult. More concretely, it is as difficult as to separate P from NP.

In Section 4 the complexity of computing advice functions for sets in Full-P/log is studied. Since Full-P/log contains sets of arbitrarily high complexity, even nonrecursive ones, their advice functions must also have arbitrarily high complexity. For this reason we study the computation of advice functions for a set A in Full-P/log relative to A. In the same way, advices functions for sets in P/*poly* were studied in [Ga92]. Before this last work, Allender and Hemachandra [AH92], presented another one, where they established lower bounds on the location of classes in the Extended Low Hierarchy. In almost all cases, they were able to prove that their results were essentially optimal. It was not the case for the class P/*poly*. This class was located in the hierarchy in an optimal sense in [Kob93]. This work improved the results from the mentioned [Ga92], and as a consequence, it got the best possible location for P/*poly*. With the same motivation, we explain also where Full-P/log is inside this hierarchy.

To finish, some possible definitions of new classes inside the Extended Low Hierarchy are presented. The first nontrivial class inside it, is $EL_1^{p,\Sigma}$, so we ask whether it makes sense to define smaller classes in the hierarchy.

2 Preliminaries

Complexity classes are sets of formal languages. A formal language is a set of words over the alphabet $\{0,1\}$. The length of a word w is denoted $|w|$, and the cardinality of the finite set A by $\#A$.

For any alphabet Σ, Σ^* is the set of all words over Σ; $\Sigma^{\leq n}$ is the set of all words of length at most n, and $A^{\leq n} = A \cap \Sigma^{\leq n}$; similarly we have $\Sigma^{=n}$ and $A^{=n}$. Here we will use in particular the alphabet $\Sigma = \{0,1\}$ and $\Sigma = \{0\}$. A tally set is a set of words over the single letter alphabet $\{0\}$. A sparse set A is a set having at most polynomially many strings of each length: $|A^{\leq n}| \leq p(n)$ for some polynomial p. The strings of Σ^* are ordered by lengths and lexicographically within each length.

The symbol $\subset$ denotes proper inclusion.

We want to be able to encode several words into one, in such a way that both computing the encoding and recovering the coded words can be easily done. We have chosen a pairing function $\langle\rangle$ which given x and y, the word $\langle x, y\rangle$ is obtained by duplicating each bit of x, and appending y to that duplicate of x inserting a 01 in between.

If A and B are sets of words, $\chi_A^B \in \{0,1\}^\infty$ is the characteristic sequence of A relative to B. It is defined in the standard way: the i^{th} bit of the sequence is 1 if and only if the i^{th} word of B is in A. Similarly, $\chi_{A^{\leq n}}^B$ is the characteristic sequence of $A^{\leq n}$ relative to

B. The most studied sets in this work are tally sets. For any set $T \subseteq \{0\}^*$, we will write only χ_T denoting the characteristic sequence of T relative to $\{0\}^*$.

Many of our complexity classes can be defined in a completely standard way by time-bounded multitape Turing machines. We will focus on in the class P solvable by deterministic polynomial time algorithms, and PF the functional version of P. Occasionally we mention other classes such as NP.

When a machine M asks to an oracle A we will write M^A, and the language recognized by M^A will be $L(M, A)$. We denote by P^A the class of all sets that can be decided in polynomial time with oracle A, that is to say, the class of all sets that are Turing reducible to A. For a class of sets C, we denote by P^C, the class of languages that are Turing reducible to some set of C.

When we use $P^A[f(n)]$, we want to express the idea of sets recognized in polynomial time, asking to an oracle A, but only with a number of questions bounded by $f(n)$.

The notation $P_{||}^A$, means that the questions to the oracle are made at the same time, that is, in a parallel way.

Given an oracle B, we will need a characterization of $P^{NP^B}[O(\log)]$ from [Wa90]. He proved a more general result, but we will only use the following:

Theorem 1 $P^{NP^B}[O(\log)] = P_{||}^{NP^B}$.

We have to introduce a tool we will use in this paper: resource-bounded Kolmogorov complexity. Fix any universal Turing machine U. Define the sets of bounded Kolmogorov complexity strings $K[f(n), g(n)]$ as follows:

Definition 2 $x \in K[f(n), g(n)]$ if there exists y, $|y| \leq f(|x|)$, such that $U(y) = x$ in at most $g(|x|)$ steps.

We focus on the two classes we will use in this paper.

Definition 3 i/ $A \in K[\log, poly]$ if there exist c and k such that $A \subseteq K[c \log n, n^k]$.

ii/ $A \in K[\log\log, poly]$ if there exist c and k such that $A \subseteq KS[c \log(\log n), n^k]$.

We will also denote by $K[f(n), g(n)]$ the language $\bigcup_{A \in K[f(n),g(n)]} A$, that will be distinguished by context.

The notion of advice function was introduced in [KL80] to provide connections between uniform computation models such as resource-bounded Turing machines and nonuniform computation models.

Definition 4 Given a class of sets C and a class of bounding functions F, the class C/F is formed by the sets A such that

$$\forall n \; \exists w \; (|w| \leq h(n)) \; \forall x \; (|x| = n) \; x \in A \iff \langle x, w \rangle \in B$$

where $B \in C$ and $h \in F$.

The set C is usually a uniform complexity class, most frequently P, whereas the class $poly = \{n^{O(1)}\}$ of polynomials and the class $\log = O(\log n)$ of logarithms are the most frequently for the family of advice functions F.

We study here a variant of $P/\log$ closed under Turing reducibility proposed by [Ko87]:

Definition 5 A set A is in Full-P/log if

$$\forall n \; \exists w \; (|w| \leq c \log n) \; \forall x \; (|x| \leq n) \; x \in A \iff \langle x, w \rangle \in B$$

where $B \in P$ and c is a constant.

This class was studied in [BHM92], and some characterizations were given:

Theorem 6 [BHM92]
The following classes of languages are the same:

i/ P^{TALLY2} where $TALLY2 = \{T \text{ is a tally set} \mid T \subseteq \{0^{2^k} \mid k \in N\}\}$.

ii/ $P^{LOWTALLY}$ where $LOWTALLY = \{T$ is a tally set $\mid$ for all n, $\chi_{T^{\leq n}} \in K[c \log n, n^d]$ for some constant c and $d\}$.

iii/ Full-P/log.

If r is a reduction, then the set of languages that are r-equivalent to languages of the class C is denoted by $E_r(C)$. In the case of the m-reduction the class $E_{\equiv}(C)$ means languages p-isomorphic to a language in C. Remember the definition of p-isomorphism.

Definition 7 Two languages are p-isomorphic iff a bijection f exists such that:

i/ f is a m-reduction between both.

ii/ f and f^{-1} are computable in polynomial time.

We present now an important result from Allender and Rubinstein [AR86], that characterizes the class $K[\log, poly]$. (For more information see [BDG90]).

Theorem 8 [AR86] $K[\log, poly] = E_{\equiv}(\text{TALLY})$.

Balcázar, Book, and Schöning defined the Extended Low Hierarchy. Allender and Hemachandra [AH92], and Long and Sheu [LS91] refined the Extended Low Hierarchy by introducing intermediate levels based on the Δ and Θ-levels of the polynomial hierarchy, respectively.

Definition 9 [BBS86], [AH92], [LS91] The Σ, Δ, and Θ-levels of the Extended Low Hierarchy (denoted $EL_k^{p,\Sigma}$, $EL_k^{p,\Delta}$, and $EL_k^{p,\Theta}$) are defined as follows.

1. $EL_k^{p,\Sigma} = \{A \mid \Sigma_k^p(A) \subseteq \Sigma_{k-1}^p(A \oplus SAT)\}, 1 \leq k$
2. $EL_k^{p,\Delta} = \{A \mid \Delta_k^p(A) \subseteq \Delta_{k-1}^p(A \oplus SAT)\}, 2 \leq k$
3. $EL_k^{p,\Theta} = \{A \mid \Theta_k^p(A) \subseteq \Theta_{k-1}^p(A \oplus SAT)\}, 2 \leq k$

3 Very low-Kolmogorov complexity sparse sets and its relationships with other classes.

In this section, we identify the class Full-P/ log with the class of languages Turing reducible to sets of $K[\log\log, poly]$. The sets of this last class are very low-Kolmogorov complexity sparse sets. After that, we characterize this class $K[\log\log, poly]$ as the union of sets p-isomorphic to another known and easy class, namely the class of tally sets whose words follow a given regular pattern. To finish, not only p-isomorphism is studied but also Turing equivalence.

Theorem 10 The following classes are equivalent:

i/ P^{TALLY2}.

ii/ $P^{K[\log\log, poly]}$.

iii/ Full-P/ log.

Proof.
We use the fact that i/ $\Longleftrightarrow$ iii/. It was proven in [BHM92]

i/ $\Longrightarrow$ ii/

It is clear because $TALLY2 \subseteq K[\log\log, poly]$. Let T be a tally set $T \subseteq \{0^{2^k} \mid k \in N\}$. T is so easy that is in $K[\log\log, poly]$, because to produce an element $x = 0^{2^k}$ of T it is only necessary to know k. If k is used as seed then the length of each seed is $|k| = \log(\log 2^k) = \log(\log|x|)$. Thus L is in $K[\log\log, poly]$.

ii/ $\Longrightarrow$ iii/

Let us see that $P^{K[\log\log, poly]} \subseteq$ Full-P/log. Suppose that $L \in P^{K[\log\log, poly]}$, then there exists a set $S \in K[\log\log, poly]$ and a Turing machine M asking to S, such that $L = L(M, S)$. It is possible to keep the characteristic sequence of S relative to $K[\log\log, poly]$ in the following way: Define a $TALLY2$ set $T = \{0^{2^k} \mid$ the k^{th} element of $K[\log\log, poly]$ is in $S\}$. As the number of elements in $K[\log\log, poly]$ less or equal to n is bounded by $2^{\log(\log n)+1} - 1 \leq 2\log n$, then it is easy to check that instead of the set S we can use T as an oracle for a polynomial time machine M' that decides L.

The language $L = L(M', T)$ is then in P^{TALLY2}, and by Theorem 6, P^{TALLY2} and Full-P/ log coincide.

iii/ $\Longrightarrow$ i/

Again by Theorem 6, It is true that Full-P/ log $\subseteq P^{TALLY2}$. □

Although $TALLY2 \subseteq K[\log\log, poly]$, obviously there are sets in $K[\log\log, poly]$ that are not tally sets, so $K[\log\log, poly]$ is different from $TALLY2$, but we can prove that any language A is p-isomorphic to a tally set of $TALLY2$ iff A is in $K[\log\log, poly]$.

Theorem 11 $E_{\equiv}(TALLY2) = K[\log\log, poly]$.

The proof follows Allender and Rubinstein' s arguments in [AR86] and it is omitted from this extended abstract.

Looking at Full-P/log, not only this class coincides with P^{TALLY2}, but also with $P^{LOWTALLY}$, where $LOWTALLY$ is the class of tally sets that are regular in a resource-bounded Kolmogorov complexity sense.

$TALLY2 \subset LOWTALLY$, but $P^{TALLY2} = P^{LOWTALLY}$. The next question that we study is whether $E_{\equiv}(LOWTALLY)$ and $E_{\equiv}(TALLY2)$ coincide.

Proposition 12 There exists a set A p-isomorphic to a set $T \in LOWTALLY$, such that $A \notin K[\log\log, poly]$.

The proof consists in defining a LOWTALLY set with words having too high Kolmogorov complexity for belonging to a TALLY2 set.

Next step will be to characterize the class of Turing equivalent to $TALLY2$ and $LOWTALLY$. These are the classes $E_T(TALLY2)$ and $E_T(LOWTALLY)$. In order to do it, we define a concept that is similar to self-producible circuits, (for further details see [BDG88]).

Definition 13 A has **log-self-producible circuits** iff there exists B and f: $\{0\}^* \rightarrow \Sigma^*$ with $f \in PF^A$ such that:

i/ $\exists c$ a constant, $\forall n |f(0^n)| \leq c \log n$.

ii/ Given $x \in \Sigma^*$ $\forall y$ such that $|y| \leq |x|$ it holds that $y \in A \iff \langle y, f(0^{|x|})\rangle \in B$.

This concept —although a little artificial— will help us to characterize the classes $E_T(TALLY2)$ and $E_T(LOWTALLY)$, showing us that both of them coincide.

Theorem 14 The following classes are equivalent:

i/$E_T(LOWTALLY)$.

ii/ Languages with log-self-producible circuits.

iii/ $E_T(TALLY2)$.

The proof is given in the appendix. The arguments are based on the proof technique presented in [BHM92], and named "double exponential skip".

We have just characterized the class $E_T(TALLY2) = E_T(LOWTALLY)$, but it is interesting to note that $E_{\equiv}(TALLY2)$ and $E_{\equiv}(LOWTALLY)$ are different. It is also easy to see that the class of p-isomorphic sets to $LOWTALLY$ is smaller than the class of Turing equivalent sets to $LOWTALLY$. Consequently, until now, we are sure that $E_{\equiv}(TALLY2) = K[\log\log, poly] \subset E_{\equiv}(LOWTALLY) \subset E_T(TALLY2) = E_T(LOWTALLY) = \{A \mid A \text{ has log-self-producible circuits}\} \subseteq$ FULL-P/log. Later we will prove that checking whether the last inclusion is strict may be very difficult.

4 The complexity of computing advice functions

We study here the relative complexity of computing advice functions for sets in the classes $K[\log\log, poly]$, and Full-P/ log. From now on, when we use the concept of advice f, we want to say that for each length n, $f(0^n)$ helps not only the computation of the elements of length n, but also elements of all lengths less or equal to n.

For instance, if we look at the class $K[\log\log, poly]$, it is easy to see that every set A of it , has an advice function in $PF^A[O(\log)]$.

Proposition 15 if $A \in K[\log\log, poly]$, then $A \in$ Full-P$/PF^A[O(\log)]$.

The proof is very easy because we can use as an advice for length n, the characteristic sequence for $A^{\leq n}$ relative to $K[\log\log, poly]$.

Every set A in $K[\log\log, poly]$ has $\#A \leq c\log n$ for some constant c. But there exists sets in Full-P$/PF^A[O(\log)]$ with a bigger cardinality. Thus the inverse of the last proposition does not hold.

Regarding the class of languages with log-self-producible circuits, we know that it coincides exactly with Full-P$/PF^A$.

What happens with Full-P/ log is the following:

Theorem 16 $A \in$ Full-P/ log iff $A \in$ Full-P$/PF^{NP^A}[O(\log)]$.

Proof.

- If $A \in$ Full-P/ log, then does A have any advices in $PF^{NP^A}[O(\log)]$?

  There exists an advice function f with $|f(0^n)| \leq c\log n$ for some constant, and $B \in P$, such that, for all x it holds:

  $$\forall z\, |z| \leq |x|\, (\langle z, f(0^{|x|})\rangle \in B \iff z \in A)$$

  We will say that the word y, with $|y| \leq c\log n$ is a "good advice" iff $\forall u\, |u| \leq n\, (\langle u, y\rangle \in B \iff u \in A)$.

  This last predicate is in Co-NP^A.

  Let GA be the following oracle set:

  $$GA = \{\langle z, 0^n\rangle \mid |z| \leq c\log n \wedge \exists y\, z \sqsubseteq y\ |y| = c\log n \mid \text{"y" a "good advice"}\}$$

  The set $GA \in$ Co-NP^A

  The advice function f of A is in $PF^{NP^A}[O(\log)]$. To calculate $f(0^n)$ it has to ask to GA which are the consecutive prefixes.

  Thus, if $A \in$ Full-P/ log then $A \in$ Full-P$/PF^{NP^A}[O(\log)]$.

- The reverse is trivial. If $A \in$ Full-P$/PF^{NP^A}[O(\log)]$, then a logarithmic advice can be the answers of the questions to the oracle in NP^A. So $A \in$ Full-P/ log. □

Now, we are going to see that for all set A in Full-P/ log it holds that $NP^A \subseteq P^{A \oplus SAT}$. Several consequences are obtained from this. For instance, that for all sets A in Full-P/ log the classes $PF^{NP^A}[O(\log)]$ and $PF^{P^{A \oplus SAT}}[O(\log)]$ coincide.

Theorem 17 Full-P/ log $\subseteq EL_1^{\Sigma}$.

This result is also a consequence of another Theorem from [AKM92], but we prefer to give a direct proof in the appendix.

We have located all sets in Full-P/ log in the first level $EL_1^{p,\Sigma}$ of the Extended Low Hierarchy. A nontrivial class smaller than $EL_1^{p,\Sigma}$ is not known until now in this hierarchy. At the end of this section we propose some definitions for that.

A corollary from the two last Theorems is:

Corollary 18 The following are equivalent:

i/ $A \in$ Full-P/log.

ii/ $A \in$ Full-P/$PF^{NP^A}[O(\log)]$.

iii/ $A \in$ Full-P/$PF^{P^{A \oplus SAT}}[O(\log)]$.

Proof.
By the last Theorem, every A in Full-P/ log holds that $NP^A = P^{A \oplus SAT}$. Also by Theorem 16, it is well known that if $A \in$ Full-P/ log, then $A \in$ Full-P/$PF^{NP^A}[O(\log)]$. Then any set A in Full-P/ log is in Full-P/$PF^{P^{A \oplus SAT}}[O(\log)]$ too. The other way is trivial because there are only a logarithmic number of questions to the oracle. □

The next corollary is interesting because it proves that the question of separating P^{TALLY2} to $E_T(TALLY2)$ is very difficult:

Corollary 19 If Full-P/ log $\neq E_T(TALLY2)$ then $P \neq NP$.

Proof. By last corollary any $A \in$ Full-P/ log is in Full-P/$PF^{P^{A \oplus SAT}}[O(\log)]$ and vice-versa.

On the one hand, by Theorem 14, $A \in E_T(TALLY2)$ if and only if $A \in$ Full-P/PF^A. And on the other hand $A \in$ Full-P/ log if and only if $A \in$ Full-P/$PF^{P^{A \oplus SAT}}[\log]$. If $P = NP$ then it would happen that $A \in$ Full-P/ log if and only if $A \in$ Full-P/$PF^{P^A}[\log]$, and as Full-P/$PF^{P^A}[\log] \subseteq$ Full-P/PF^A necessarily Full-P/ log $\subseteq E_T(TALLY2)$, which would imply that $E_T(TALLY2) =$ Full-P/ log. □

Although separating Full-P/ log from $E_T(TALLY2)$ is very difficult, the question in the case of having $E_{\equiv}(TALLY2)$ it is very easy. Now we are going to separate the classes Full-P/ log and Full-P/$PF^A[O(\log)]$. Even more, the next Theorem proves that Full-P/$PF^A[O(\log)]$ and Full-P/PF^A do not coincide, and as a consequence Full-P/ log is different from Full-P/$PF^A[O(\log)]$.

Theorem 20 There is a set A in Full-P/PF^A such that A is not in Full-P/$PF^A[O(\log)]$.

The proof is given in the appendix.

5 further results

We would like to present here, some possible definitions for classes in the Extended Low Hierarchy, smaller than $EL_1^{p,\Sigma}$. In order to do this, it is interesting to note that:

- For all $A \in$ Full-P/log holds that $NP^A \subseteq P^{A \oplus SAT}$, because Full-P/log $\subseteq EL_1^{p,\Sigma}$.

  But also we have that:

- For all $A \in$ Full-P$/PF^A$ it holds that $NP^A \subseteq P_{||}^{A \oplus SAT}$

  Using Theorem 1 from [Wa90] we have the above result: suppose A in Full-P$/PF^A$. By Theorem 1 the class $P^{NP^A}[O(\log)] = P_{||}^{NP^A}$. But $A \in P^T$ and $T \in P^A$ with $T \in TALLY2$, so $P_{||}^{NP^A} = P_{||}^{NP^T}$. As T is a tally set then $P_{||}^{NP^T} = P_{||}^{T \oplus SAT} = P_{||}^{A \oplus SAT}$.

  If we have that $P_{||}^{NP^A} \subseteq P_{||}^{A \oplus SAT}$, then in particular $NP^A \subseteq P_{||}^{A \oplus SAT}$.

- For all $A \in$ Full-P$/PF^A[O(\log)]$ holds that $NP^A \subseteq P^{A \oplus SAT}[O(\log)]$.

  Let B be a set in NP^A with A having advices in $PF^A[O(\log)]$. Let M be the corresponding nondeterministic Turing machine, whose time is bounded by n^q, and such that $B = L(M, A)$. It means that each path of the computation of M on a word x can ask for the oracle at most words of length $|x|^q$. It is possible to find the advice for length $|x|^q$ in $PF^A[O(\log])$, and when we have this advice it is not necessary to consult to the oracle. So if A has advices in $PF^A[O(\log)]$ then $NP^A = NP^A[O(\log)]$.

  Also as A is in particular an element of Full-P/log, it holds that $NP^A \subseteq P^{A \oplus SAT}$, so it is proven that: $NP^A = NP^A[O(\log)] \subseteq P^{A \oplus SAT}[O(\log)]$.

The last two possibilities are nontrivial classes, smaller than $EL_1^{p,\Sigma}$. So they could be used as definition of other classes inside the Extended Low Hierarchy.

Acknowledgments

I am grateful to José Luis Balcázar, and Ricard Gavaldà, who motivated me to begin this work, and provided interesting ideas. I also thank Jorge Castro and Carlos Seara for their valuable comments. A lot of comments about English language were made by Elvira Mayordomo, but she is not guilty of any mistake.

References

[AH92] E. Allender and L.Hemachandra. Lower bounds for the low hierarchy. *Journal of the ACM*, 39(1), 234-250, 1992.

[AKM92] V. Arvind, J. Köbler, M. Mundhenk. Lowness and the Complexity of Sparse and Tally Descriptions. *Ulmer Informatik-Berichte*, Nr.92-04, 1992.

[AR86] E. Allender, E. W. Rubinstein. P-printable sets. *SIAM Journal on Computing*, 17, 1193-1202, 1988.

[BBS86] J.L. Balcázar, R. Book, U. Schöning. Sparse sets, lowness and highness. *SIAM J. Comput.*, 23, 679-688, 1986.

[BDG88] J.L. Balcázar, J. Díaz, J. Gabarró. Structural Complexity I. *EATCS Monographs on Theoretical Computer Science*, Vol. 11, Springer-Verlag 1988.

[BDG90] J.L. Balcázar, J. Díaz, J. Gabarró. Structural Complexity II. *EATCS Monographs on Theoretical Computer Science*, Vol. 22, Springer-Verlag 1990.

[BHM92] J.L. Balcázar, M. Hermo, E. Mayordomo. Characterizations of logarithmic advice complexity classes. *In Proceedings of the IFIP 12th World Computer Congress*, Vol. 1, 315-321, North-Holland, 1992.

[BGSS93] J.L. Balcázar, R. Gavaldà, H.T. Siegelmann, and E.D. Sontag. Some Structural Complexity Aspects of Neural Computation. *In Proceedings of Structural in Complexity Theory 8th annual conference*, 253-265, IEEE Computer Society Press, 1993.

[BS92] J.L. Balcázar, U. Schöning. Logarithmic advice classes. *Theoretical Computer Science*, 99, 279-29, 1992.

[Ga92] R. Gavaldà. Bounding the Complexity of Advice Functions. *In Proceedings of Structural in Complexity Theory 7th annual conference*, 249-254, IEEE Computer Society Press, 1992.

[HY84] J. Hartmanis, Y. Yesha. Computation times of NP sets of different densities. *Theoretical Computer Science*, 34, 17-32, 1984.

[KL80] R.M. Karp, R.J. Lipton. Some connections between nonuniform and uniform complexity classes. *In Proceedings 12th Annual Symposium on Theory of Computing*, 302-309, 1980.

[Ko87] K.-I Ko. On helping by robust oracle machines that take advice. *Theoretical Computer Science*, 52, 15-36, 1987.

[Kob93] J. Köbler. Locating P/*poly* optimally in the extended low hierarchy. *In Proceedings of STACS, 10th Annual Symposium on Theoretical Aspects of Computer Science*, 28-37, 1993.

[LS91] T.J. Long, M.-J. Sheu. A refinement of the low and high hierarchies. Technical Report OSU-CISRC-2/91-TR6, The Ohio State University, 1991.

[Wa90] K. W. Wagner. Bounded query classes. *SIAM J. Comput.*, Vol. 19, 5, 833-846, 1990.

6 Appendix

Proof of the Theorem 14.

- i/ $\Longrightarrow$ ii/

  Suppose that A is Turing equivalent to a tally set T such that $\chi_{T^{\leq n}} \in K[c \log n, n^d]$. It means that $A \in P^T$ and $T \in P^A$. How can we calculate $f(0^n)$?

  $A \in P^T$ implies that to decide whether x is in A, it is only necessary to know $\chi_{T^{\leq |x|^k}}$ for some constant k. $f(0^{|x|})$ can be the corresponding seed s for $\chi_{T^{\leq |x|^k}}$, which has length $|s| = c \log |x|$ with c another constant. This is because $\{\chi_{T^{\leq |x|^k}}\} \in K[\log, poly]$. If f is defined in this way, then the condition i/ that is to have log-self-producible circuits will hold.

  Regarding the condition ii/, for all y with $|y| \leq |x|$ it holds that $y \in A \Longleftrightarrow \langle y, f(0^{|x|})\rangle \in B$, with function f it is possible to obtain the characteristic function on T.

  We also need to use the fact that f is in PF^A. The algorithm testifying it, could be the following:

```
begin;
  input $0^n$;
    for $s = 0$ to $s = \underbrace{11\ldots1}_{c \log n}$, asking to the oracle $A$
      if $s$ is a seed to produce $\chi_{T^{\leq n^k}}$
      then keep $s$
    end for;
{ The kept seed $s$ is the last one }
  output $s$;
end.
```

- ii/ $\Longrightarrow$ iii/

  If A has log-self-producible circuits, by definition 13, there exists $f: \{0\}^* \rightarrow \Sigma^*$ with $f \in PF^A$ such that:

  - $\exists c$ a constant, $\forall n |f(0^n)| \leq c \log n$.
  - Given $x \in \Sigma^*$ $\forall y$ such that $|y| \leq |x|$ it holds that $y \in A \Longleftrightarrow \langle y, f(0^{|x|})\rangle \in B$.

  We will define a tally set T containing only words of length a power of 2, and will prove that $A \in P^T$.

  At this moment we use the mentioned technique "double exponential skip". The main idea is to keep only the information $f(0^n)$ corresponding to some selected n instead of storing all of them in the oracle $T \subseteq \{0^{2^n} \mid n \in N\}$. Of course, we have to select for the oracle infinitely many $f(0^n)$'s; but the fact that each of them is good for all the words with length smaller than n it is designed to allows us to select them arbitrarily far apart.

We must find a balance between two contradictory restrictions. If we select $f(0^n)$'s for the oracle too frequently then they will need too many bits, and some of them will be encoded too far away in the oracle; but if we select them too separated, then for some words the nearest valid advice would be too long to be extracted from the oracle by a polynomial time machine.

It turns out that there is a way of skipping $f(0^n)$'s for which the balance is satisfactory. We will encode in the oracle all the $f(0^n)$'s corresponding to lengths $n = 2^{2^m}$ for all $m \in N$ and skip all the intermediate ones. Each bit of each of the selected $f(0^n)$ will be encoded as the presence or absence of a word of the form 0^{2^m} in the set T.

The function corresponding to length 2^{2^0}, respectively $2^{2^1} \ldots 2^{2^m}$, has size c, respectively $2c \ldots 2^m c$. We use the first c powers of two, from 0^2 until 0^{2^c}, to encode the function for length $2^{2^0} = 2$. The second string to be stored has length $2c$, so this information needs $2c$ powers of two: use the next ones, from $0^{2^{c+1}}$ until $0^{2^{c+2c}}$. In general, the information of the function corresponding to the length 2^{2^m} is encoded in the tally set T by the elements $0^{2^{c+2c+2^2c+\cdots+2^{m-1}c+1}}$ until $0^{2^{c+2c+2^2c+\cdots+2^m c}}$.

So let T be

$$T = \{0^{2^{\left(\Sigma_{i \leq m-1} 2^i c\right)+p}} \mid 1 \leq p \leq 2^m c \text{ such that the } p^{th} \text{ bit of } f(0^{2^{2^m}}) \text{ is } 1\}$$

On the one hand we prove that $A \in P^T$. On input x, find an integer m such that $2^{2^{m-1}} < |x| \leq 2^{2^m}$. This can be done in linear time. Observe that this selection ensures that $\log\log|x| \leq m < \log\log(|x|^2)$. Now, for each value of p from 1 to $2^m c$, ask whether $0^{2^{\left(\Sigma_{i \leq m-1} 2^i c\right)+p}} \in T$ and, in this way, obtain all the bits of the function $f(0^{2^{2^m}})$, which can now be used to decide whether $x \in A$ in polynomial time. It remains to see that the queries can be asked in polynomial time; it suffices to see that they are polynomially long.

The number of queries is bounded by $c\log(|x|^2)$. A bound on the length of the oracle queries is $2^{c+2c+2^2c+\cdots+2^m c} = 2^{(1+2+2^2+\cdots+2^m)c} < 2^{2^{m+1}c}$. As $m < \log\log(|x|^2)$, the queries have length at most $2^{c2^{\log\log(|x|^d)}} = (2^{2^{\log\log(|x|^d)}})^c = |x|^{d'}$ for appropriate, small constants d and d'. So $A \in P^T$.

On the other hand, $T \in P^A$, because to know whether $0^{2^n} \in T$, we only have to look for $f(0^{2^n})$ and to check whether the bit corresponding to the element of T is 1. As $f(n) \in PF^A$, then $T \in P^A$. So A is Turing equivalent to a tally set $T \in TALLY2$. This implies that $A \in E_T(TALLY2)$

- iii/ $\Longrightarrow$ i/

Let A be a set in $E_T(TALLY2)$. Then there exists a tally $T \in TALLY2$ such that $A \in P^T$ and $T \in P^A$. As $TALLY2 \subset LOWTALLY$ then $A \in E_T(LOWTALLY)$. □

Proof of the Theorem 17.

Let A in Full-P/log $= P^{TALLY2}$. Is NP^A included in $P^{A\oplus SAT}$?

Suppose $L \in NP^A$ via a nondeterministic Turing machine M_j. As $L \in NP^{P^T}$ for a $T \in TALLY2$. then, there exists a deterministic Turing machine M, which works in time n^q, such that $L = L(M, L(M_j, T))$.

Let B_j be the following oracle set :

$$B_j = \{\langle x, t_1, t_2, 0^k\rangle \mid \exists z\, |z| \leq k \text{ and } x \sqsubseteq z \text{ and } M_j^{t_1}(z) \neq M_j^{t_2}(z)\}$$

Where each t_i is an encoding of a tally sets up to a determined length.

B_j is in NP.

Let us see that $L \in P^{A\oplus SAT}$. An algorithm that testifies it would be:

```
begin;
  input  $h$;
  Take every possible sequence $t_i$ for the tally set $T$;
{There are a polynomial number of t's for this}
  for all  pair $\langle t_i, t_j\rangle$ do
    $p := \lambda$;
    if $\langle p0, t_i, t_j, 0^{|h|^q}\rangle \in B_j$
      then $p := p0$
      else if  $\langle p1, t_i, t_j, 0^{|h|^q}\rangle \in B_j$
        then $p := p1$
      end if;
    end if;
  keep the word $p$;
  end do;
  for all  par $\langle t_i, t_j\rangle$ that is distinguished by some $p$ do
    reject $t_i$ or $t_j$ according to oracle $A$
  end do;
{ Now, the rest of $t_i$ can be used instead of $A$ }
  Ask to some $NP$ machine whether there exists an accepting path
  for $h$ using oracle $L(M_j, t_i)$;
end. □
```

Proof of the Theorem 20.

We will define a set $A \in$ Full-P$/PF^A$, such that:

$$P^A \neq P^A[O(\log)]$$

If we get it, then A can not have any advice in $PF^A[O(\log)]$, and we will have obtained the separation. This is because, if A has an advice in $PF^A[O(\log)]$, then it would be possible to recognize any set in P^A making only a logarithmic quantity of questions to the oracle.

Let B_A be in P^A such that $B_A \notin P^A[O(\log)]$:

$$B_A = \{0^{2^{2^m}} \mid m \in N \wedge \forall y \text{ (one of the first } m2^m \text{ words of length } 2^{2^m}) \ y \notin A\}$$

First of all, $B_A \in P^A$ since given $x = 0^{2^{2^m}}$ there is $m2^m = \log(\log|x|)\log|x|$ number of possible y's. Secondly, $B_A \notin P^A[O(\log)]$. we can construct the oracle A by diagonalization on an enumerating of Turing machines $\{M_i\}_{i \in N}$ making a logarithmic number of questions to the oracle, that is M_i can make at most, for all n, $c_i \log n$ questions to the oracle.

We assume that p_i is a polynomial bounding the running time of M_i^A. Without loss of generality, we can assume that all the p_i are nondecreasing, and that for every i and n, $p_i(n) \leq p_{i+1}(n)$. The construction will be performed in stages. At stage n, we shall possibly expand the set constructed so far, adding at most one word in such a way that we diagonalize against machine M_n of the enumeration. To do this, we choose a word long enough to ensure that the previously simulated computations are not disturbed by the potential addition of this word to the oracle, and such that the currently simulated computation is not disturbed either. Through the construction, $k(n)$ denotes the length of the word chosen at stage n. It is important to remark that every M_n on x can make at most $c_n \log|x|$ questions to the oracle, where c_n is a constant. We will also choose the number $k(n)$ in such a way that $k(n) > c_n$ for all n.

The construction is the following:

```
Stage 0
        $A_0 = \emptyset$;
        $k(0) = 0$;

Stage $n$
        Let $k(n)$ be the smallest number such that
          $p_n(k(n)) < 2^{k(n)}$ and $p_{n-1}(k(n-1)) < k(n)$ and $c_n < k(n)$;
        Take $M_n$ on $0^{2^{2^{k(n)}}}$;
        if $M_n(0^{2^{2^{k(n)}}})$ rejects
          then $A_n := A_{n-1}$
          else choose a word $y$ between the first $k(n)2^{k(n)}$ words of
            length $2^{2^{k(n)}}$, such that $y$ has not been asked for the oracle $A_{n-1}$;
            $A_n := A_{n-1} \cup \{y\}$;
end Stage $n$ .
```

1. In each stage n, there is a word y that has not been asked for the oracle A_{n-1}, because $M_n(0^{2^{2^{k(n)}}})$ can make at most $c_n 2^{k(n)}$ questions of length $2^{2^{k(n)}}$ to the oracle, and we chose the word y between the first $k(n)2^{k(n)} > c_n 2^{k(n)}$.

2. Also, the constructed set A is in Full-P$/PF^A$ because there is a tally set $T_A \in TALLY2$ such that $A \in P^{T_A}$ and $T_A \in P^A$. Namely, T_A is the following:

$$T_A = \{0^{2^{2^m}+|i|} \mid i^{th} \text{ word of length } 2^{2^m} \in A\}$$

As possible words y in A for length 2^{2^m} are the first $m2^m$, the position i has $|i| = m + \log m$ bits. Then, the set T_A is well defined because $2^{2^m+m+\log m} \leq 2^{2^{m+1}}$.

It has been possible to construct a set A in Full-P$/PF^A$ such that there exists $B_A \in P^A$ and $B_A \notin P^A[O(\log)]$. Then it is prove that A has not advices in $PF^A[O(\log)]$. □