# Modulo Classes and Advice

J. Ramachandran
Computer and Information Sciences
Ohio State University
Columbus, Ohio, 43210

**Abstract**

We prove an upper bound on computing advice for a language $A$ in the class of languages acceptable in deterministic polynomial time with logarithmic advice, $A \in$ P/log), relative to $A$ itself. We then provide conditions under which advice for a language $A \in$ P/log can be computed in PF($A$), and conditions for the existence of a set $A$ in P/log without advice in PF($A$). We show that paddable languages in MOD$_k$P/log that are also self-reducible are in MOD$_k$P, without advice. This is an extension of similar results for self-reducible paddable sets in P/log by Balcázar and Schöning [3].

**Keywords** Computational complexity, Kolmogorov Complexity, counting classes, advice.

## 1 Introduction

Karp and Lipton ([8]) defined *advice functions* as a notion of nonuniform complexity. A language $A$ is in the class $\mathcal{C}/\mathcal{F}$ if, for some choice of $B$ in the language class $\mathcal{C}$ and of function $f$ from the function class $\mathcal{F}$, for all $x$ in $\Sigma^*$, $x$ is in $A$ if and only if $\langle x, f(|x|)\rangle$ is in $B$. The *advice string* $f(|x|)$ depends only on the length of the input string $x$ and is presented at no cost. *Advice strings* are useful in encoding small sets of information such as polynomial sized circuits and the census of sparse sets.

The class P, of languages accepted by deterministic Turing machines in polynomial time, is commonly considered the class of feasible problems. Balcázar and Schöning, in [3], started the study of classes of the form $\mathcal{C}$/log. In this paper we first study the complexity of the class P/log. In contrast to P, P/log has undecidable problems due to the unbounded complexity of the advice function. We therefore focus on the complexity of computing the advice function of a language in P/log relative to the language itself. For example, a set $A$ is said to have self producible circuits if its advice function is computable in PF($A$), and it is unknown if all languages in P/log have self producible circuits. In the first half of this paper we attempt to resolve this question.

Valiant defined #P as the class of functions computable by counting the number of accepting paths of polynomial time bounded nondeterministic machines. Papadimitriou and Zachos first introduced the counting class $\oplus$P derived from #P ([11]), defined as the class of languages $L$ for which there exists a #P function $f$ such that, for all strings $x$ in $\Sigma^*$, $x$ is in $L$ if and only if $f(x)$ is odd. They showed $\oplus\mathrm{P}(\oplus\mathrm{P}) = \oplus\mathrm{P}$ and $\oplus\mathrm{P} \subseteq \mathrm{P}(\#\mathrm{P}[1])$. Since then, there has been considerable research on relationships between counting classes, the best known of which is Toda's theorem proving that the polynomial time hierarchy is contained in BP.$\oplus$P and in P(PP). Beigel and Gill in [6], generalizing on the notion of $\oplus$P, defined $\mathrm{MOD}_k\mathrm{P}$ as the class of languages $L$ for which there exists a #P function $f$, such that for all strings $x$ in $\Sigma^*$, $x$ is in $L$ if and only if $f(x)$ is 1 modulo $k$, for some choice of positive integer $k$. They studied closure properties and complete sets for $\mathrm{MOD}_k\mathrm{P}$, and proved relativized separations of $\mathrm{MOD}_k\mathrm{P}$ from other regular classes such as NP, PP, and $\mathrm{MOD}_j\mathrm{P}$ for $j \neq k$ ([5]).

In the second half of this paper we combine the two notions of advice and modulo counting classes. We study the nonuniform complexity of languages in $\mathrm{MOD}_k\mathrm{P}$ given logarithmic length bounded advice. This study is an extension of similar results for self-reducible paddable sets in P/log by Balcázar and Schöning who, in [3], started the study of classes of the form $\mathcal{C}$/log. The proof illustrates an interesting connection between modulo counting classes and advice.

Our results are organized as follows:

1. We prove an upper bound for computing advice for a language in P/ log, relative to the language itself.

2. We provide conditions under which languages in P/ log have self producible circuits.

3. We use Kolmogorov complexity to construct a language in P/log without self producible circuits, under the hypothesis that there exist languages in P that at infinitely many lengths consist of only Kolmogorov random strings.

4. We show that paddable languages in $\mathrm{MOD}_k\mathrm{P}$/log that are also self-reducible are actually in $\mathrm{MOD}_k\mathrm{P}$, without advice. $\mathrm{MOD}_k\mathrm{P}$/log in general has undecidable sets, therefore the assumption of self reducibility is crucial in computing advice.

## 2 Definitions and Notations

We use the definitions of standard complexity concepts as found in [1] and [2]. A function $f$ is in the function class log if there exists a constant $c$ such that, for all $n \in \mathcal{N}$, $|f(n)| \leq c \log n$. Recall the definition of resource bounded Kolmogorov complexity. Let $\phi$ be a universal partial recursive function and let $T_\phi$, a Turing machine that computes $\phi$, be a universal transducer. Let $x, y$, and $p$ be strings in $\Sigma^*$, $|x| = n$, and let $t : \mathcal{N} \rightarrow \mathcal{N}$ and $s : \mathcal{N} \rightarrow \mathcal{N}$ be a time and a space bound respectively. If $T_\phi(p, y) = x$ then $\phi(p, y) = x$, and $\langle p, y\rangle$ is a *description* of $x$.

**Definition 2.1** *[10] The resource bounded Kolmogorov complexity of $x$ conditional to $\phi$ and $y$, denoted $C_\phi^{t,s}(x|y)$, is the length of the shortest program $p$ such that $\phi(p, y)$ is $x$. The simulation of $\phi$ by $T_\phi$ runs in time $t(|x|)$ and space $s(|x|)$.*

Note that in the definition of $C_\phi^{t,s}(x|y)$ the information after the "|" is given for free. $y$ may consist of several strings paired together. The Invariance Theorem in [10] states that

**Theorem 2.2** *There exists a universal partial function $\phi_0$ such that for all other partial functions $\phi$, and all $x, y \in \Sigma^*$, there exists a constant $c$ dependent only on $\phi$ such that $C_{\phi_0}^{ct \log t, cs}(x|y) = C_\phi^{t,s}(x|y) + c$.*

The subscript $\phi$ may be dropped from $C_\phi^{t,s}(x|y)$ when the time, space, and length bounds are not sensitive to these changes. The parameters $y, t$ or $s$ can be dropped in the following circumstances: $y$ is the empty string; $t$ is infinity; or $s$ is infinity, respectively. For example $C_\phi^{t,s}(x)$ is the *unconditional resource bounded Kolmogorov complexity* of $x$.

**Definition 2.3** $\mathrm{K}[l(n), t(n)] = \{x \in \Sigma^{=n} \mid C_\phi^{t(n),\infty}(x) = l(n)\}$.

The class $\mathrm{K}[l(n), t(n)]$ denotes the set of strings $x \in \Sigma^{=n}$ that can be generated by a fixed universal transducer running some $l(n)$ bit program in $t(n)$ time.

## 3 Constructing advice for P/log

Gavaldá and Köbler presented results on constructing advice for P/poly in [7, 9]. In this section we answer the same question for the class P/log.

**Definition 3.1** *A set $A$ belongs to* P/log *if there exists a $B$ in* P*and a function $h : \mathcal{N} \rightarrow \Sigma^*$ in* log *such that, for all $x \in \Sigma^*$, $x \in A$ if and only if $\langle x, h(|x|)\rangle \in B$.*

Let $A$ be a set in P/log, let $B \in \mathrm{P}$ and $h \in \log$ be as above, and let $c$ be a constant such that $|h(|x|)| \leq c \log |x|$.

**Definition 3.2** *A function $f : \Sigma^* \rightarrow \Sigma^*$ is said to compute good advice for $A$ with respect to $B$ if $f$ is in the class:*

$$\textit{good-advice}(A|B) = \{f \in \log \mid \forall x\ [\ x \in A \Longleftrightarrow \langle x, f(0^{|x|})\rangle \in B\ ]\}$$

As there are $2n^c$ candidates for good advice for $A$ at length $n$ it is easy to show that:

**Proposition 3.3** *If $A \in \mathrm{P}/\log$, then there exists an $f$ in $\mathrm{PF}(\mathrm{NP}(A))$ such that $f$ computes good advice for $A$ with respect to $B$ ($\in$ good-advice$(A|B)$).*

**Proof.** On input $0^n$, search for a $w \in \Sigma^{\leq c.\log n}$ that satisfies the co-NP$^A$ predicate:

$$\forall x \in \Sigma^{=n}\ [x \in A \Longleftrightarrow \langle x, w\rangle \in B]$$

□

We proceed to tighten the upper bound of Proposition 3.3 by constructing a polynomial-sized set of strings of length $n$ that can be used to separate good advice from bad advice.

**Definition 3.4** .

1. *Let $B \in \mathrm{P}$ and let $c$ be a constant. Then the set of discriminating strings is*

$$\textit{discr}(B, c) = \{x \in \Sigma^* \mid \exists w_1, w_2 \in \Sigma^{\leq c \log |x|}\ [\langle x, w_1\rangle \in B \Longleftrightarrow \langle x, w_2\rangle \notin B]\}$$

2. *$S$ is an adequate discriminator for $B$ at length $n$ for advice of length at most $c \log n$ if*

   (a) *$S \subseteq \textit{discr}(B, c) \cap \Sigma^{=n}$.*

   (b) *$|S| \leq p(n)$ for some polynomial.*

   (c) *For all $x \in \Sigma^{=n}$ and for all $w_1, w_2 \in \Sigma^{\leq c \log n}$*

$$\exists y\ [\langle y, w_1\rangle \in B \Longleftrightarrow \langle x, w_2\rangle \notin B] \Longrightarrow \exists x \in S\ [\langle x, w_1\rangle \in B \Longleftrightarrow \langle x, w_2\rangle \notin B]$$

3. *good-discr$(B, c) = \{f : 0^* \rightarrow 2^{\Sigma^*} \mid \forall n\ [f(0^n)$ is an adequate discriminator for $B$ at length $n$ for advice of length at most $c \log n]\}$.*

**Theorem 3.5** *If $A \in \mathrm{P}/\log$ witnessed by $B$ and some $h \in \log$, then there exists an $f$ in $\mathrm{PF}(\mathrm{NP} \oplus A)$ such that $f \in$ good-advice$(A|B)$.*

**Proof.** Let $A \in \mathrm{P}/\log$ witnessed by the set $B \in \mathrm{P}$ and advice function $h$, and let $h(n)$, $|h(n)| \leq c \log n$, be advice for strings of length $n$.

**Claim:** There is a $g$ in PF(NP) such that $g \in$ *good-discr*$(B, c)$.

**Proof.** Consider the following P set:

$$W = \{\langle x, w_1, w_2\rangle \mid |x| = n, |w_1| \leq |w_2| \leq c \log n, \text{ and } [\langle x, w_1\rangle \in B \Longleftrightarrow \langle x, w_1\rangle \notin B]\}$$

Then the following set is in NP:

$$PW = \{\langle 0^n, u, w_1, w_2\rangle \mid |u| \leq n \text{ and } \exists v, |uv| = n, \langle uv, w_1, w_2\rangle \in W\}$$

Define $g(0^n)$ as the set $S_n$ produced as follows: For each $\langle w_1, w_2\rangle$, $|w_1| \leq |w_2| \leq c \log n$, construct a string $\langle x, w_1, w_2\rangle \in W$ by prefix search over $PW$. As there are at most $4n^{2c}$ pairs $\langle w_1, w_2\rangle$, the set $S_n$ of strings so constructed is in *good-discr*$(B, c)$. □

**Claim:** There is a $f$ in $\mathrm{PF}(g \oplus A)$ such that $f \in$ *good-advice*$(A|B)$.

**Proof.** Let $g(0^n) = \{x_1, \ldots, x_k\}$. Query $A$ on each $x_i \in g(0^n)$. Output the first $w \in \Sigma^{\leq c.\log n}$ such that, for all $i \leq k$, $\langle x_i, w\rangle \in B \Longleftrightarrow x_i \in A$. This $w$ is good advice for $A$ because $g(n)$ is an adequate discriminator for $B$. □

The theorem follows from the above claims. □

## 4 Lower bounds on advice for P/log

Theorem 3.5 shows the existence of an $f \in$ *good-advice*$(A|B)$ in $\mathrm{PF}(\mathrm{NP} \oplus A)$. Thus, the results of the last section show that $\mathrm{P}/\log$ is in $\mathrm{EL}_1^{P,\Sigma}$ (Extended Low 1, defined in [13] as the class of sets $A$ such that $\mathrm{NP}(A) \subseteq \mathrm{P}(A \oplus \mathrm{NP})$). We now ask if this upper bound can be improved, and examine the conditions under which such an improvement is possible.

**Definition 4.1** *A set $A$ in* P/log *has* self producible circuits *if there exists an $f \in \mathrm{PF}(A)$ such that $f \in$ good-advice$(A|B)$.*

**Proposition 4.2** *If* $\mathrm{P} = \mathrm{NP}$*, then* P/log *has self producible circuits.*

**Proof.** If $\mathrm{P} = \mathrm{NP}$ then, as a corollary to Theorem 3.5, advice for $A \in \mathrm{P}/\log$ is computable in $\mathrm{PF}(A)$. □

### 4.1 Conditions for existence of self producible circuits

Recall, from Definitions 3.2 and 3.4, the definitions of *good-advice*$(A|B)$ and *good-discr*$(B,c)$. In this section we present some necessary and sufficient conditions for a language $A \in \mathrm{P}/\log$ to have self producible circuits.

Note that Theorem 3.5 shows the existence of an $f \in$ *good-advice*$(A|B)$ in $\mathrm{PF}(g \oplus A)$, where $g$ is any function in *good-discr*$(B,c)$. Thus,

$$\exists g \in \textit{good-discr}(B,c) \ \cap \ \mathrm{PF} \implies \exists f \in \textit{good-advice}(A|B) \ \cap \ \mathrm{PF}(A)$$

We now provide conditions under which the hypothesis of the above statement holds.

**Definition 4.3** *1. A set $L$ is P-printable if there exists an $f \in \mathrm{PF}$ such that $f(0^n) = L^{\leq n}$.*

*2. The ranking function of $L$ is an $r_L \in \mathrm{PF}$ such that $r_L(x) = \|\{y < x \mid y \in L\}\|$.*

*3. $L$ is P-isomorphic to a language $L'$ if and only there exists a polynomial-time invertible polynomial-time bijection between $L$ and $L'$.*

Rubinstein in [12] showed the following:

**Theorem 4.4** *[12] If $S$ is a sparse set then the following four statements are equivalent.*

*1. $S$ is P-printable.*

*2. The ranking function of $S$ is computable in polynomial time.*

*3. $S$ is P-isomorphic to a tally set in* P.

*4. $S^{=n} \in \mathrm{K}[c.\log n, n^c]$ for some constant $c$ depending on $S$ only.*

Let $A$ be in $\mathrm{P}/\log$, witnessed by $B \in \mathrm{P}$ and $h \in \log$, and let $h(n) = w, |w| \leq c \log n$ be the advice string. Let $B$ be recognized by TM $M$ in time $p(n)$, for some polynomial $p$. Using the above results of Rubinstein [12] we present sufficient conditions for the existence of a $g \in$ *good-discr*$(B,c) \cap \mathrm{PF}$. The following theorem is straightforward:

**Theorem 4.5** *If there exists a $c' > 0$ such that for each $n$ there exists an $S_n \subseteq \mathrm{K}[c' \log n, n^{c'}]$ such that $S_n$ is an adequate discriminator for $B$ at length $n$ for advice of length at most $c \log n$, then there exists a $g \in$ good-discr$(B,c) \cap \mathrm{PF}$.*

**Proof.** Define a function $g$ that, on input $0^n$, generates the set $S = \mathrm{K}[c' \log n, n^{c'}] \cap \Sigma^{=n}$ in polynomial time (by running every program of length $c' \log n$ for $n^{c'}$ steps and discarding the outputs of length other than $n$). The set $S$ must be trimmed to $S' \subseteq \mathit{discr}(B, c) \cap \Sigma^{=n}$, which can be done in polynomial time. Let $g(0^n) = S'$. By hypothesis $g$ is in $\mathit{good\text{-}discr}(B, c)$. $\square$

From Theorem 4.5 and Theorem 4.5, we have:

**Corollary 4.6** *If there exists an $S$ such that either $S$ is P-printable, or the ranking function of $S$ is computable in polynomial time, or $S$ is P-isomorphic to a tally set in* P*and, for all $n$, $S^{=n}$ is an adequate discriminator for $B$ at length $n$ for advice of length at most $c \log n$, then there exists a $g \in \mathit{good\text{-}discr}(B, c) \cap \mathrm{PF}$.*

Let $\langle g(0^n) \rangle$ denote the string encoding the set $g(0^n)$. Note the following converse:

**Proposition 4.7** *If there is a $g$ in $\mathit{good\text{-}discr}(B, c) \cap \mathrm{PF}$, then there exists a constant $c' > 0$ such that if $S = \cup_n \, g(0^n)$, then for all $n$, $\langle g(0^n) \rangle \in \mathrm{K}[c' \log n, n^{c'}]$, $S$ is P-printable, the ranking function of $S$ is computable in polynomial time, and $S$ is P-isomorphic to a tally set in* P.

Theorem 4.5 and Proposition 4.7 give necessary and sufficient conditions for some function $g$ to be in $\mathit{good\text{-}discr}(B, c) \cap \mathrm{PF}$. This would imply that for any set $A$ in P/log as witnessed by $B \in \mathrm{P}$ and some $h \in \mathit{log}$, $|h(x)| \leq c \log n$, $A$ has self producible circuits. In general we do not know if the converse holds; i.e.

$$\exists f \in \mathit{good\text{-}advice}(A|B) \ \cap \ \mathrm{PF}(A) \implies \exists g \in \mathit{good\text{-}discr}(B, c) \ \cap \ \mathrm{PF}.$$

We now examine conditions under which a set in P/log would not have self producible circuits.

### 4.2 Conditions for nonexistence of self producible circuits

We prove a partial converse to Theorem 4.5 below. We show that the assumption that there exists a P set that almost everywhere consists of only hard strings leads to the existence of a set in P/log without self producible circuits. (Actually $A \in \mathrm{P}/\mathcal{O}(1)$ where $\mathcal{O}(1)$ represent the class of constant functions.) We begin with a technical lemma.

Let $M$ be a deterministic polynomial time bounded oracle Turing transducer running in time $n^k$. Define the *first positive query* of $M$ on some input $x \in \Sigma^{=n}$ as the first query string $y$

made to the oracle that is in the oracle set, and as undefined if no such query is made. We say an oracle set $A$ is a *diagonalization oracle* if $A^{\leq n^k} = T \cup A'$ where $T \subset \Sigma^{\leq n-1}$ and $A' \subset \Sigma^{=n}$.

**Lemma 4.8** *Let $A$ be a diagonalization oracle. The first positive query $y$ of length $n$, made by $M(0^n)$ to $A$ is in* $\mathrm{K}[c \log n, n^c]$*, for some constant $c$ depending only on $M$ and $k$.*

**Proof.** For all $n$, the first positive query string $y \in \Sigma^{=n}$ queried by $M(0^n)$, if one exists, can be produced by the following program that encodes all the following:

1. a constant number of bits for $M$,
2. $\log n$ bits to describe $n$,
3. $k \log n$ bits to describe the position $p$ of the first query $y$ in the list of queries made by $M$ to oracle $A$ that is answered YES, and
4. this description : Run $M$ on $0^n$, answering each of the first $p-1$ queries by NO or by using $T$. Output the $p^{\text{th}}$ query.

Thus, for all $n$ the first positive string $y$ in $A$ queried by $M$ is in $K[c \log n, n^c]$, for large enough constant $c$. □

**Theorem 4.9** *Let $C$ be a language in* P *such that, for each constant $c$, there exists infinitely often an $n$ such that $C^{=n}$ is not empty and $C^{=n} \cap \mathrm{K}[c. \log n, n^c] = \emptyset$. Then there exists an $A$ in* $\mathrm{P}/\mathcal{O}(1)$*, witnessed by a $B \in \mathrm{P}$ (dependent on $C$) and an $h \in \mathcal{O}(1)$, that does not have self producible circuits.*

**Proof.** Given a number $c$, we call $n$ *good for c* if $C^{=n}$ is not empty and $C^{=n} \cap \mathrm{K}[c. \log n, n^c] = \emptyset$. Let $M_1, M_2, \ldots$ be an enumeration of deterministic oracle Turing machines, $M_i$ running in time $n^i + i$.

Let $B = \{\langle x, 1\rangle \mid x \in C\}$ and note that $C \in \mathrm{P} \Longrightarrow B \in \mathrm{P}$. We will define $h : 0^* \rightarrow \{0,1\}$, thereby implicitly defining $A = \{x \mid \langle x, h(|x|)\rangle \in B\}$. Note that $A \in \mathrm{P}/\mathcal{O}(1)$.

We construct $A \in \mathrm{P}/\mathcal{O}(1)$ by diagonalization. Let $A_0 = \emptyset$, and $n_0 = 1$. In stage $r = \langle k, i\rangle$ we eliminate the machine $M_i$ as a candidate for computing advice for $A$, in polynomial time using $A$ as an oracle set.

**Stage** $r = \langle k, i \rangle$**:**

Choose $n = n_r > {n_{r-1}}^{r-1} + r - 1$ so that

1. all queries made in previous stages are shorter than $n_r$, and
2. $n_r$ is good for $k$.

Let $h(m) = 0$ for $n_{r-1} + 1 \leq m \leq n_r - 1$. Run $M_i$ on input $0^{n_r}$ using $A_{r-1}$ as an oracle. Let $h(n) = 1 \Longleftrightarrow M_i(0^{n+r})$ outputs 0.

If $A$ has self-producible circuits, then there exists an oracle transducer $N$, that runs in polynomial time, such that with oracle $A$, for all $n$, $N$ on input $0^n$ computes $h(n)$. Note that for every $r = \langle k, i \rangle$, $M_i(0^{n_r})$ with oracle $A_{r-1}$ produces bad advice, and for any query $M_i$ makes to strings of length other than $n_r$, oracle set $A_{r-1}$ and $A$ are in agreement. Thus, as $N$ produces good advice, $N$ must query some string in $A$ of length $n_r$ and receive the answer YES. Let the first such positive query for length $n_r$ be $y_{n_r}$.

By Lemma 4.8, we have that for some fixed constant $c$ and for each $n$ every first positive query $y_{n_r}$ of length $n_r$ made by $N$ on input $0^{n_r}$ is in $\mathrm{K}[c \log |y_{n_r}|, |y_{n_r}|^c]$. But

$$y_{n_r} \in A \Longleftrightarrow \langle y_{n_r}, h(n_r) \rangle \in B \Longleftrightarrow (h(n_r) = 1 \wedge y_{n_r} \in C)$$

This implies that $\forall r\ [y_{n_r} \in C^{=n_r} \cap K[c. \log n_r, n_r^c]]$.

However, consider any stage $r$ of th form $r = \langle c, i \rangle$. $n_r$ was chosen so that $n_r$ is good for $c$; i.e., $1C^{=n_r} \cap K[c. \log n_r, n_r^c] = \emptyset$.

Hence a contradiction. Therefore $A$ does not have self producible circuits. $\square$

Theorem 4.9 essentially says that if a language $C$ in P is composed almost everywhere of exclusively hard strings, then given an arbitrary length it is impossible to detect if $C$ is empty at this length. However, the advice in the above proof answers the question for free. Therefore computing advice even with reference to $A$ is impossible.

## 5 MOD classes, self-reducibility and log advice

We now consider self-reducible languages in $\mathrm{MOD}_k\mathrm{P}/\log$, and their relationship to $\mathrm{MOD}_k\mathrm{P}$. We use standard definitions of structural complexity concepts as found in [1] and [2].

**Definition 5.1** *[4] A language $L$ is in* $\mathrm{MOD}_k\mathrm{P}$ *if there exists a* #P *function $f$ such that for all $x \in \Sigma^*$ $[x \in L \Longleftrightarrow f(x) \equiv 1 \bmod k]$.*

Only prime $k$ will be considered in the remainder of this paper.

**Definition 5.2** *A set $A$ is self-reducible if there exists a deterministic polynomial time oracle machine $M$ such that $A = L(M, A)$, and for each $x$, $M$ on input $x$ queries the oracle only on strings $y, |y| < |x|$, regardless of the choice of oracle set.*

A function $f$ is in the function class log if there exists a constant $c$ such that, for all $n \in \mathcal{N}$, $|f(n)| \leq c \log n$.

**Definition 5.3** *A set $A$ belongs to* $\mathrm{MOD}_k\mathrm{P}/\log$ *if there exists a $B$ in* $\mathrm{MOD}_k\mathrm{P}$ *and a function $f : \mathcal{N} \rightarrow \Sigma^*$ in* log, *such that, for all $x \in \Sigma^*$, $x \in A$ if and only if $\langle x, h(|x|)\rangle \in B$.*

We now consider self-reducible languages in $\mathrm{MOD}_k\mathrm{P}/\log$, and their relationship to $\mathrm{MOD}_k\mathrm{P}$.

**Definition 5.4** *A set $A$ belongs to strong-*$\mathrm{MOD}_k\mathrm{P}/\log$ *if there exists a* $\mathrm{MOD}_k\mathrm{P}$ *set $B$ and a function $f : \mathcal{N} \rightarrow \Sigma^*$ in* log, *such that, for all $x \in \Sigma^*$, and for all $m \geq |x|$, $x \in A$ if and only if $\langle x, f(m)\rangle \in B$.*

Definition 5.4 essentially means that a strong advice string for $x$ is also good advice for all strings of length less than $x$. Let $B(w) = \{x \mid \langle x, w\rangle \in B\}$.

Balcázar and Schöning proved that self-reducible sets in strong-P/log are in P. In Theorem 5.7 we show that self-reducible sets in strong-$\mathrm{MOD}_k\mathrm{P}/\log$ are in $\mathrm{MOD}_k\mathrm{P}$. The proof of Theorem 5.7 involves rewriting the deterministic polynomial time algorithm from Balcázar and Schöning [3] as a nondeterministic polynomial time algorithm, where the number of accepts is congruent to 1 mod $k$ if and only if the former algorithm accepts.

We start with a definition and a technical lemma. Recall that an NPMV function is computed by an nondeterministic polynomial time transducer that can produce multiple values in it's computation tree. Given $f$ in NPMV, define $set_f(x)$ as the set of values produced over all paths.

**Definition 5.5** *A function $f : \Sigma^* \rightarrow \Sigma^*$ is in* $\mathrm{MOD}_k\mathrm{PSV}$ *if there exists an function $g$ in* NPMV *such that for all $x$ $f(x) \in set_g(x)$ and $f(x)$ is the only value in $set_g(x)$ appearing on* 1 mod $k$ *paths, all other values appearing on* 0 mod $k$ *paths each.*

We first prove the following useful lemma:

**Lemma 5.6**

1. $\mathrm{MOD}_k\mathrm{P}(\mathrm{MOD}_k\mathrm{P}) \subseteq \mathrm{MOD}_k\mathrm{P}$.

2. $\mathrm{P}(\mathrm{MOD}_k\mathrm{PSV}) \subseteq \mathrm{MOD}_k\mathrm{P}$.

3. $\mathrm{PF}(\mathrm{MOD}_k\mathrm{P}) \subseteq \mathrm{MOD}_k\mathrm{PSV}$.

**Proof.**

1. It is well know that $\mathrm{MOD}_k\mathrm{P}$ is low for $\mathrm{MOD}_k\mathrm{P}$, ([14],[11]).

2. Assume $L \in \mathrm{P}(\mathrm{MOD}_k\mathrm{PSV})$. Then there exists a DOTM $M$ that queries a function oracle $f \in \mathrm{MOD}_k\mathrm{PSV}$. If we replace each oracle query $y$ for the value $f(y)$, with the computation tree of the function $f$, by definition the number of paths that will have the correct value $f(y)$ is congruent to $1 \bmod k$. Thus the number of paths with the same oracle queries and answers as $M$ makes and receives, using $f$ as an oracle, is also congruent to $1 \bmod k$. Any path which chooses an incorrect answer to a query to $f$ appears (along with its entire computation subtree) $0 \bmod k$ times. Therefore the number of computation paths in this new machine is congruent to $1 \bmod k$, if and only if $M^f(x)$ accepts.

3. This proof is similar to 2. Assume $f \in \mathrm{PF}(\mathrm{MOD}_k\mathrm{P})$. Then there exists a deterministic oracle transducer $M$ that queries a language oracle $L \in \mathrm{MOD}_k\mathrm{P}$. If we replace each oracle query $y$ for the value $\chi_L(y)$ with the computation tree of the NDTM $M_L$ (that witnesses $L \in \mathrm{MOD}_k\mathrm{P}$), by definition the number of paths with all the same queries and answers as $M$ appears $1 \bmod k$ times. The one difference is that now each computation path instead of accepting or rejecting the input, now produces a value for the function, using $M$. The number of paths with incorrect choices for $\chi_L(y)$ on query $y$, is congruent to $0 \bmod k$. The statement now follows from the definition of $\mathrm{MOD}_k\mathrm{PSV}$.

□

We will use the above facts to show:

**Theorem 5.7** *If $A$ is self reducible and $A \in$ strong-$\mathrm{MOD}_k\mathrm{P}/\log$, then $A \in \mathrm{MOD}_k\mathrm{P}$.*

**Proof.** Let $M$ witness the self-reduction of $A$. Since $A \in$ strong-MOD$_k$P$/\log$, there is a set $B \in$ MOD$_k$P, and a sequence $\{w_n\}_{n \in \mathcal{N}}$ of strings such that $|w_n| \leq c.\log n$ for some constant $c$ and for all $n$, and $A^{=n} = B(w_m)^{=n}$ for all $n$, and for all $m \geq n$.

The following algorithm will be shown to accept $A$, and from the claims that follow, $A \in$ MOD$_k$P.

---

Main:

Input $x, |x| = n$.

Find a $u, |u| \leq c \log n$, such that First-Good-Advice$(x, u)$ holds.

Run $M^{B(u)}(x)$

---

The above is in $\mathrm{P}^{\mathrm{First-Good-Advice} \oplus B}$.

---

Procedure First-Good-Advice$(x, u)$ : **boolean**

$b \leftarrow$ Good-Advice$(x, u)$

**for** $v, |v| \leq c \log n$, and $v < u$ **do**

$b \leftarrow b \wedge \neg$ Good-Advice$(x, v)$

**end**($*$ **for** $*$)

**return** $b$

---

The above is in $\mathrm{P}^{\mathrm{Good-Advice}}$.

---

Procedure Good-Advice$(x, u)$ : **boolean**

$b \leftarrow$ TRUE

**for** $v, |v| \leq c \log n$, **do**

$b \leftarrow b \wedge$ Test$(x, u, v)$

**end**($*$ **for** $*$)

**return** b

The above is in $\mathrm{P}^{\mathrm{Test}}$.

Let $\mathrm{CON}(x, w)$ be the predicate $[x \in B(w) \iff x \in L(M, B(w))]$. Let $E$ be the predicate: $[x \in L(M, B(u)) \iff x \in L(M, B(v))]$.

---

Procedure $\mathrm{Test}(x, u, v)$ : **boolean**

**if** $(x \in L(M, B(u)) \iff x \in L(M, B(v)))$ **then**

  **return** TRUE

**else**

  $y \leftarrow$ First-Contra-Query$(x, u, v)$

  **if** $\neg\mathrm{CON}(y, u)$ **then return** FALSE

  **elsif** $\neg\mathrm{CON}(y, v)$ **then return** TRUE

  **else return** $\mathrm{Test}(y, u, v)$ **end**$(*$ **if** $*)$

**end**$(*$ **if** $*)$

---

The above is in $\mathrm{P}^{\mathrm{CON} \oplus \mathrm{E} \oplus \mathrm{First-Contra-Query}}$.

---

Procedure First-Contra-Query$(x, u, v)$ : STRING

Let $p$ bound the number of queries made by $M(x)$

Guess an $i, 1 \leq i \leq p(|x|)$

Guess a sequence of answers $a_1, \ldots, a_{i-1}$ for the first $i-1$ queries

Use these answers to arrive at the $i^{th}$ query $y_i$, made by $M$ using answers $a_1, \ldots, a_{i-1}$

Evaluate the $\mathrm{MOD}_k\mathrm{P}$-predicate that is TRUE if and only if

1. $\forall j < i, B(u)$ and $B(v)$ agree that query $y_j$ is answered $a_j$
2. $B(u)$ and $B(v)$ disagree on query $y_i = y$

**if** the predicate is TRUE **then return** $y$ **end**$(*$ **if** $*)$

---

The above is in $\mathrm{PF}^{B}$.

The correctness and complexity of the algorithm just presented follows from a sequence of claims about the properties of the procedures of the algorithm.

**Claim 1.** CON and $E$ are in $\mathrm{MOD}_k\mathrm{P}$.

This follows from Lemma 5.6, and $L(M, B(u))$ and $L(M, B(v))$ being in $\mathrm{MOD}_k\mathrm{P}$.

**Claim 2.** The tail recursion of procedure Test terminates in polynomial (in $|x| + |u| + |v|$) number of steps. This is because each call is length-decreasing in its first parameter, hence when $|y| = 1$, the **if** condition is always TRUE.

**Claim 3.** The following predicate, informally described, is in $\mathrm{P}^B$: Given $i$ strings $y_1, \ldots, y_i$ and $i - 1$ answers $a_1, \ldots, a_{i-1}$, verify that

$$(\forall j < i)[M^{B(u)}(y_j) = M^{B(v)}(y_j) = a_j] \text{ and } [M^{B(u)}(y_i) \neq M^{B(v)}(y_i)]$$

Again, by Lemma 5.6, the predicate is in $\mathrm{MOD}_k\mathrm{P}$.

**Claim 4.** First-Contra-Query$(x, u, v)$ is in $\mathrm{MOD}_k\mathrm{PSV}$.

Firstly note that First-Contra-Query$(x, u, v)$ is in NPMV. We need to show that the number of output paths that do produce $y$ such that the predicate of Claim 3. holds, is congruent to 1 mod $k$.

If there is a query $y_i$, such that $\neg(y_i \in B(u) \Longleftrightarrow y_i \in B(v))$, made by $M^{B(u)}(x)$ and $M^{B(v)}(x)$, then there is a smallest $i = i_0$ where this happens. It is easy to see in this case that there is a unique correct guess for $i$ and unique guesses for $a_1, \ldots, a_{i_0-1}$ such that the predicate is TRUE. All other output paths of $f$ either incorrectly guessed $i$ too low ($L(M, B(u))$ actually agrees with $L(M, B(v))$ with regard to the membership of $y_i$), or too high ($L(M, B(u))$ disagrees with $L(M, B(v))$ with regard to the membership of some $y_j, j < i$.

The number of paths that output the correct $y_i$ is congruent to 1 mod $k$, whereas all other outputs appear 0 mod $k$ times, from the fact that the predicate of Claim 3 is in $\mathrm{MOD}_k\mathrm{P}$.

**Claim 5.** Since First-Contra-Query$(x, u, v)$ is in $\mathrm{MOD}_k\mathrm{PSV}$, Test$(x, u, v)$ is a predicate in $\mathrm{MOD}_k\mathrm{P}$. This implies Good-Advice$(x, u)$ is in $\mathrm{MOD}_k\mathrm{P}$, which in turn implies First-Good-Advice$(x, u)$ is in $\mathrm{MOD}_k\mathrm{P}$.

This all follows from Lemma 5.6

**Claim 6.** For every string $x, |x| = n$, every $m \geq n$, and every $v$ such that $|v| \leq c \log n$, Test$(x, w_m, v)$ is TRUE. Suppose to the contrary. Choose a minimal $x$ and some $v$ such that Test$(x, w_m, v)$ is FALSE. This $x$ cannot be a leaf on the self-reducibility structure of $M$ as we would return TRUE otherwise. So $M(x)$ must have at least one oracle query and the ELSE clause is entered. Let the oracles $B(u)$ and $B(v)$ disagree on $y$. Then CON$(y, w_m)$ is TRUE by the definition of $w_m$. Also the next test for consistency CON$(y, v)$ is TRUE as Test$(x, w_m, v)$ is assumed to return FALSE. Therefore it must be that the recursive call to Test$(y, w_m, v)$ returns FALSE contradicting the minimality of $x$.

**Claim 7.** For every string $x, |x| = n$, every $u, |u| \leq c \log n$, and every $m \geq n$, if Test$(x, u, w_m)$ is TRUE then $x \in L(M, B(u)) \Longleftrightarrow x \in L(M, B(w_m))$

Assume to the contrary. Fix a $u$ for which the predicate $x \in L(M, B(u)) \Longleftrightarrow x \in L(M, B(w_m))$ fails. Choose a minimal $x$ such that (1) Test$(x, u, w_m)$ holds, and (2) $x \in L(M, B(u)) \Longleftrightarrow x \notin L(M, B(w_m))$. Since $x \in L(M, B(u)) \Longleftrightarrow x \in L(M, B(w_m))$ ($u$ is bad advice) we enter the else clause of Test, get a contradicting query $y$ such that CON$(y, u)$ is TRUE (otherwise we would have returned FALSE), and as CON$(y, w_m)$ is always TRUE, we go to the recursive call which, by assumption, must also return TRUE. That means $y \in L(M, B(u)) \Longleftrightarrow y \in L(M, B(w_m))$. But $y < x$, which is contradicts $x$'s minimality.

**Claim 8.** To complete the proof note that some $u = w_m$ of length at most $c \log n$ is good advice. First-Good-Advice$(x, u)$ is TRUE for only one string $u, |u| \leq c \log n$, where $u$ is good advice. ($u$ need not be $w_m$ itself, we only need $\forall x, [x \in L(M, B(u)) \Longleftrightarrow x \in L(M, B(w_n))]$ by Claim 7.)

□

As in [3], the above proof is given with respect to self reducibility defined on the length order. The proof works under the assumptions of a polynomial well founded ordering as well. Furthermore, if the self-reduction tree can, by padding properties of the considered language $A$, be considered to consist only of strings of the same length as the input, then the assumption "$A \in$ strong-MOD$_k$P/log" can be weakened to "$A \in$ MOD$_k$P/log". Since all natural complete sets for PSPACE, NP, PP and MOD$_k$P have such padding properties and are self-reducible, we have the following corollaries:

**Corollary 5.8** *Let $\mathcal{C}$ be a class from the set* $\{\mathrm{NP}, \mathrm{PP}, \mathrm{MOD}_k\mathrm{P}$ (all prime $k$) $\}$. *If* $\mathcal{C} \subseteq \mathrm{MOD}_k\mathrm{P}/\log$ *then* $\mathcal{C} \subseteq \mathrm{MOD}_k\mathrm{P}$.

**Corollary 5.9** *If* $\mathrm{PSPACE} \subseteq \mathrm{MOD}_k\mathrm{P}/\log$ *then* $\mathrm{PSPACE} = \mathrm{MOD}_k\mathrm{P}$.

Since there exist relativized negations of all the above conclusions (by Beigel [5]) the hypothesis for each corollary is unlikely.

## Acknowledgments

I thank Tim Long for his many helpful comments and corrections.

## References

[1] Balcázar, J., Diaz, J., and Gabarró, J. *Structural Complexity I.* Springer-Verlag, Berlin, 1988.

[2] Balcázar, J., Diaz, J., and Gabarró, J. *Structural Complexity II.* Springer-Verlag, Berlin, 1990.

[3] Balcázar, J., and Schöning, U. Logarithmic advice classes. Technical Report LSI-88-12, Univ. Politecnica Catalunyu, Spain, to appear in Theoretical Computer Science.

[4] Beigel, R. NP-hard sets are P-superterse unless R = NP. Technical Report 88-04, Department of Computer Science, The Johns Hopkins University.

[5] Beigel, R. Relativized counting classes: Relations among thresholds, parity, and mods. Tech. rep., The Johns Hopkins University, Department of Computer Science, The Johns Hopkins University, Baltimore, MD 21218, 1988.

[6] Beigel, R., and Gill, J. Counting classes: Thresholds, parity, mods and fewness. Tech. rep., The Johns Hopkins University, Department of Computer Science, The Johns Hopkins University, Baltimore, MD 21218, 1988.

[7] Gavaldá, R. *Kolmogorov Randomness and Its applications to Structural Complexity Theory.* PhD thesis, Universitat Politecnica de Catalunya, April 1992.

[8] Karp, R., and Lipton, R. Some connections between nonuniform and uniform complexity classes. In *Proc. 12th ACM Symp. on Theory of Computing* (1980), pp. 302–309.

[9] Köbler, J. Locating P/poly optimally in the extended low hierarchy. Tech. Rep. Nr. 92-05, Ulmer Informatik-Berichte, 1992.

[10] Li, M., and Vitanyi, P. *An Introduction to Kolmogorov Complexity and its applications.* Springer-Verlag, 1993.

[11] Papadimitriou, C., and Zachos, S. Two remarks on the power of counting. In *Proc. sixth GI conf. on Theoretical Comp. Sc.* (1982), pp. 269–276. Springer-Verlag,LCNS, #145.

[12] Rubinstein, R. *Structural Complexity Classes of Sparse Sets.* PhD thesis, Northeastern University, August 1988.

[13] Schöning, U. *Complexity and Structure.* Lecture Notes in Computer Science Vol. 211. Springer-Verlag, 1986.

[14] Toda, S., and Ogiwara, M. Counting classes are at least as hard as the the polynomial-time hierarchy. Tech. Rep. CSIM 90-09, University of Electro-Comm., Department of Computer Science and Information Mathematics, Chofu-shi, Tokyo 182, Japan, 1990.