Average-Case Intractability vs. Worst-Case Intractability [1]

Johannes Köbler

Institut für Informatik, Humboldt-Universität zu Berlin, D-10099 Berlin

Rainer Schuler

Abt. Theoretische Informatik, Universität Ulm, Oberer Eselsberg, D-89069 Ulm

Abstract

We show that not all sets in NP (or other levels of the polynomial-time hierarchy) have efficient average-case algorithms unless the Arthur-Merlin classes MA and AM can be derandomized to NP and various subclasses of P/poly collapse to P. Furthermore, other complexity classes like P(PP) and PSPACE are shown to be intractable on average unless they are easy in the worst case.

1 Introduction

In recent literature, it has been shown that several NP-complete problems are efficiently solvable on average with respect to certain natural distributions on the instances. Prominent examples are the graph colorability problem (see [Wil84]) and the Hamiltonian path problem (see [GS87]). However, this is probably not true for all NP-complete problems. Consider for example problems that are complete for the class DistNP which consists of all pairs (L, μ) such that L is in NP and μ is a polynomial-time computable distribution. Well known DistNP-complete problems are the bounded halting problem, the tiling problem, Post's correspondence problem, the word problem for Thue systems and groups, or LR(k) testing for context-free grammars [Lev86,Gur91,WB95,Wan95,Kar97]. As shown in [Lev86,Gur91], DistNP-complete problems are intractable on average unless every NP problem is easy

[1] Results included in this paper have appeared in the proceedings of the 23rd International Symposium on the Mathematical Foundations of Computer Science (MFCS), 1998.

on average. To be more precise, let $\mathsf{AP_{FP}}$ denote the class of sets that are decidable in time polynomial on average with respect to every polynomial-time computable distribution. Then no DistNP-complete problem is efficiently solvable on average unless $\mathsf{NP} \subseteq \mathsf{AP_{FP}}$.

Ben-David, Chor, Goldreich and Luby [BCGL92] showed the following interesting connection between average-case complexity and worst-case complexity. If every NP problem is easy on average, then all sets in nondeterministic exponential time can be decided in (worst-case) exponential time (in symbols: $\mathsf{NE} = \mathsf{E}$). The reason for this connection is that the average-case complexity of a tally set coincides with its worst-case complexity, i.e., any tally set in $\mathsf{AP_{FP}}$ belongs to P. Hence, the average case assumption $\mathsf{NP} \subseteq \mathsf{AP_{FP}}$ implies that any tally set in NP belongs to P (which is equivalent to $\mathsf{NE} = \mathsf{E}$).

A different structural property that can be used to relate the average case and worst case complexities of a computational problem is random self-reducibility. As noted in [AL83], a random self-reducible problem is intractable on average (under the distribution induced by the random self-reduction) unless it is easy in the worst case (see also, e.g., [AFK89,Fei93,FF93]). In fact, Lipton [Lip91] used an idea of Beaver and Feigenbaum [BF90] to show that multivariate polynomials of low degree are (functionally) random self-reducible. It follows from Lipton's result that if the permanent is efficiently computable on all but a sufficiently small (polynomial) fraction of all $n \times n$ matrices (over GF(p) where $p > n+1$ is prime), then the permanent of any $n \times n$ matrix can be computed in expected polynomial time.[1] Using this property of the permanent it is not hard to see that $\mathsf{P(PP)} \not\subseteq \mathsf{AP_{FP}}$ unless $\mathsf{PP} = \mathsf{ZPP}$. As shown in this paper, $\mathsf{P(PP)} \subseteq \mathsf{AP_{FP}}$ even implies $\mathsf{PP} = \mathsf{P}$. This means that P(PP) contains sets that are intractable on the average unless all sets in P(PP) are easy in the worst case. The same relationship is shown for the class PSPACE, as well as for the middle bit class MP and the generalized Mod class ModP that have been introduced and studied in [GKR$^+$95] and [KT96], respectively. It remains open however whether PH contains sets that are intractable on average unless PH collapses.

The just mentioned collapse of P(PP) downto P is obtained as a corollary to the result that PH is intractable on average unless $\mathsf{BPP} = \mathsf{P}$. More generally, we show that NP (or other levels of PH) contains sets that are intractable on average unless various subclasses of P/poly (and of NP/poly) collapse to P (to NP, respectively). To derive these results, we use the assumption that a certain problem A is easy on average to bound the worst-case complexity of some (other) set B. Typically, B belongs to a nonuniform complexity class, A captures the complexity of computing or checking the correctness of an advice

[1] In [GLR$^+$91,GS92,FL92,CPS99] it has been subsequently shown that much weaker assumptions are sufficient.

string for B, and the average case assumption on A is used to efficiently eliminate the need for advice. More precisely, we show the following:

- If computing advice strings is easy on average then we can do without advice.
- If checking the correctness of an advice string is efficient on average with respect to a distribution that focuses on particular advice strings, then the advice can be replaced by nondeterminism.
- If the correctness of advice strings can be efficiently checked on average and correct advice strings abound, then the advice can be replaced by randomness or nondeterminism.

We obtain these consequences by exploiting the following special properties of any set $A \in \mathsf{AP}_{\mathsf{FP}}$. Firstly, for any P-printable domain D there is an algorithm that decides A efficiently on all inputs in the domain D. Secondly, since A is efficiently decidable on average with respect to the standard distribution μ_{st} (which is uniform on Σ^n), there is an algorithm for A that is polynomial in the worst case for all but a polynomial fraction of the strings of each length.

The rest of the paper is organized as follows. In Section 2 we recall some notation from (average-case) complexity theory. In Section 3 we investigate some basic properties of sets in the class $\mathsf{AP}_{\mathsf{FP}}$ and derive some closure properties of this class. In the remaining sections we derive different collapse consequences for nonuniform classes under different assumptions about the average-case complexity of computing (Section 4) and verifying (Section 5) the advice. In Section 6 we consider the special case that correct advice strings abound and finally, in Section 7, we investigate classes where the advice can be obtained by a zero-error randomized computation.

2 Preliminaries

All languages we consider are over the binary alphabet $\Sigma = \{0, 1\}$. We use $\chi_L(x)$ to denote the characteristic function of a language L. The cardinality of a finite set L is denoted by $\|L\|$. The *length* of a string $x \in \Sigma^*$ is denoted by $|x|$ and the empty string (of zero length) is denoted by λ. The *join* of two sets A and B is $A \oplus B = \{0x \mid x \in A\} \cup \{1x \mid x \in B\}$. The join of two language classes is defined as the class that consists of all joins of sets from each of the two classes.

Strings in 0^* are called *tally* and a set T is *tally* if $T \subseteq 0^*$. A set S is called *sparse* if the cardinality of $S \cap \Sigma^n$ is bounded above by a polynomial in n. TALLY denotes the class of all tally sets, and SPARSE denotes the class of all sparse sets.

$\mathbb{N}$ denotes the set of non-negative integers and by log we denote the function $\log n = \max\{1, \lceil \log_2 n \rceil\}$.

We assume that the reader is familiar with fundamental complexity theoretic concepts such as (oracle) Turing machines and the polynomial-time hierarchy, denoted by PH (see, for example, [BDG95,Sch86]). As usual, FP denotes the set of functions $f : \Sigma^* \to \Sigma^*$ that are computable in polynomial time. To encode pairs (or tuples) of strings we use a standard polynomial-time computable pairing function denoted by $\langle \cdot, \cdot \rangle$ whose inverses are also computable in polynomial time. We assume that this function encodes pairs of tally strings again as a tally string.

Next we review the notion of advice functions introduced by Karp and Lipton [KL80] to characterize non-uniform complexity classes. A function $h : 0^* \to \Sigma^*$ is called a *polynomial-length function* if for some polynomial p and for all $n \geq 0$, $|h(0^n)| = p(n)$. For a class $\mathcal{C}$ of sets, let $\mathcal{C}/\mathrm{poly}$ be the class of sets L such that there is a set $I \in \mathcal{C}$ and a polynomial-length function h such that for all n and for all $x \in \Sigma^n$,

$$x \in L \;\Leftrightarrow\; \langle x, h(0^n) \rangle \in I.$$

The function h is called an *advice function* for L, whereas I is the corresponding *interpreter set*.

The notion of instance complexity and the class $\mathsf{IC}[\log, \mathrm{poly}]$ of sets of strings with low instance complexity were introduced in [OKSW94].

Definition 2.1 *We say that a set A is in $\mathsf{IC}[\log, \mathrm{poly}]$ if there exist a constant $c > 0$, a set $H \subseteq \Sigma^*$ of programs and an interpreter set $I \in \mathsf{P}$ such that for every $x \in \Sigma^*$,*

(1) there exists a $p \in H$ of length at most $c \log(|x|) + c$ such that either $\langle x, 0p \rangle \in I$ or $\langle x, 1p \rangle \in I$, and
(2) for every $p \in H$, $\langle x, 1p \rangle \in I$ implies $x \in A$ and $\langle x, 0p \rangle \in I$ implies $x \notin A$.

It is known that $\mathsf{P}/\log \subsetneq \mathsf{IC}[\log, \mathrm{poly}] \subsetneq \mathsf{P}/\mathrm{poly}$ [OKSW94] and that a set A belongs to $\mathsf{IC}[\log, \mathrm{poly}]$ if and only if A and its complement are both conjunctively reducible to a tally set [AHH+93].

Next we consider functions computed by nondeterministic (or randomized) transducers. A nondeterministic transducer is a nondeterministic Turing machine T with a write-only output tape. On input x, machine T outputs $y \in \Sigma^*$ if there is an accepting path on input x along which y is output. We use $output_T(x)$ to denote the set of all output strings produced by T on input x. Notice that the function f computed by T can be partial and multivalued. Following Selman's notation [Sel94], we use $set\text{-}f(x)$ to denote the (possibly empty) set of function values for input x. Using this notation, the function class NPMV is defined as follows.

Definition 2.2 (cf. [BLS84])

- NPMV *is the class of multivalued, partial functions f for which there is a polynomial-time nondeterministic transducer T such that for all $x \in \Sigma^*$, set-$f(x) = output_T(x)$.*
- *Let f, g be multivalued, partial functions. Then g is called a* refinement *of f, if for all $x \in \Sigma^*$ it holds that set-$g(x) \subseteq$ set-$f(x)$ and $g(x)$ is defined whenever $f(x)$ is defined (i.e., set-$f(x) \neq \emptyset$ implies set-$g(x) \neq \emptyset$.)*

In the recent literature on average-case complexity basically two ways have been considered to formalize the intuitive notion of feasible (or natural) distributions on the instance space. One way is to consider a distribution as feasible if there is a randomized algorithm that *efficiently generates* each instance with probability $\mu'(x)$ where μ' denotes the density function of the distribution. Such distributions are called efficiently samplable [BCGL92]. The more restrictive way is to require that the distribution function $\mu(x) = \sum_{y \leq x} \mu'(y)$ is *efficiently computable* [Lev86,Gur91,BCGL92]. As shown in [BCGL92], every efficiently computable distribution is also efficiently samplable.

Definition 2.3 [BCGL92] *A distribution μ is called* P-computable *(in symbols: $\mu \in$ FP) if on input x, the binary expansion of $\mu(x)$ is computable by a function in* FP*. If μ is efficiently computable relative to some oracle in a class $\mathcal{C}$, then we say that the distribution is* P($\mathcal{C}$)-computable *(in symbols: $\mu \in$ FP($\mathcal{C}$)).*

It is easy to see that not all distributions with a polynomial-time computable density function μ' are P-computable unless P = PP.

Definition 2.4 [Lev86] *A function $f : \Sigma^* \to \mathbb{N}$ is* polynomial on μ-average*, if there exists a constant $\epsilon > 0$ such that*

$$\sum_{x \neq \lambda} \frac{f^\epsilon(x)}{|x|} \mu'(x) < \infty.$$

The class of functions polynomial on μ-average has similar closure properties as the class of polynomials [Lev86,Gur91]. A further important property is robustness under the polynomial domination of distributions: If f is polynomial on μ-average and μ *dominates* ν, meaning that there exists a polynomial p such that for all x, $\mu'(x) \cdot p(|x|) \geq \nu'(x)$, then f also is polynomial on ν-average [Lev86,Gur91].

3 Average-case complexity classes

Following [Lev86,Gur91] we assume that all natural distributions are either P-computable or dominated by a P-computable distribution. In this sense, a set A is efficiently decidable on average (under natural distributions) if for any $\mu \in \mathsf{FP}$, A is decidable in time polynomial on μ-average. Next we define the type of average case complexity classes we are interested in in this paper.

Definition 3.1 [SY96] *Let $\mathcal{F}$ be a set of distributions. A set A is* decidable in average polynomial time under distributions in $\mathcal{F}$ *(in symbols, $A \in \mathsf{AP}_{\mathcal{F}}$) if for every distribution $\mu \in \mathcal{F}$ there exists a deterministic Turing machine M such that $A = L(M)$ and the running time of M is polynomial on μ-average.*

It is known that the class $\mathsf{AP}_{\mathsf{FP}}$ is properly contained in E and that it contains problems that are Turing complete for E [Sch95,Sch96]. Note that in contrast to worst-case complexity, where $\mathsf{NP} \subseteq \mathsf{P}$ implies that $\mathsf{PH} \subseteq \mathsf{P}$, it is not known whether $\mathsf{NP} \subseteq \mathsf{AP}_{\mathsf{FP}}$ implies that all sets in $\Delta_2^p = \mathsf{P}(\mathsf{NP})$ are contained in $\mathsf{AP}_{\mathsf{FP}}$ (see [Imp95,SW95] for an exposition).

To see the difficulty consider the computation of a deterministic Turing machine M which decides a set A with the help of an oracle $B \in \mathsf{NP}$, and let us assume that the distribution on the inputs of M is P-computable.

In case the oracle queries are adaptive, it depends on the oracle set B which queries are actually made, and therefore it might be hard to compute the distribution induced on the oracle queries. But also in the nonadaptive setting, where the queries of M only depend on the input and not on the oracle, the induced distribution need not be P-computable [SY95].

If, however, B has a certain paddability property, then A can be reduced to B in such a way that the induced distribution on the oracle queries is again P-computable, implying that the queries (and therefore the input) can be decided efficiently on average. Using the stronger assumption that B belongs to $\mathsf{AP}_{\mathsf{FP}(B)}$ it even follows that all sets in $\mathsf{P}(B)$ belong to $\mathsf{AP}_{\mathsf{FP}}$ (see Theorem 3.4 below).

In order to prove the just mentioned closure properties of $\mathsf{AP}_{\mathsf{FP}}$, we make use of the notion of Turing reducibility between distributional problems (A, μ) where A is a set and μ is a distribution.

Definition 3.2 [BCGL92] *A distributional problem (A, μ)* Turing reduces *to a distributional problem (B, ν) via some polynomial-time oracle machine M if*

(1) M with oracle B accepts A, and
(2) there exists a polynomial p such that $\nu'(y)p(|y|) \geq \sum \mu'(x)$ where the sum

is taken over all strings x such that M on input x asks the query y to the oracle B (in symbols: $y \in Q(x, M, B)$).

As stated in [BCGL92] the class of efficiently decidable distributional problems is closed under Turing reducibility.

Theorem 3.3 [BCGL92] *If (A, μ) is* Turing reducible *to (B, ν) and if B is decidable in time polynomial on ν-average, then A is decidable in time polynomial on μ-average.*

Now we are ready to prove the closure properties of $\mathsf{AP}_{\mathsf{FP}}$ claimed above. We use the subscript tt (for "truth-table") to indicate that the computation can be performed by a non-adaptive oracle Turing machine. Further, $pad(A)$ denotes the padded version $\{0^{|x|}1xy10^i \mid x \in \Sigma^*, y \in A, i \geq 0\}$ of A.

Theorem 3.4 *Let $\mathcal{C}$ be any language class. Then $pad(A) \in \mathsf{AP}_{\mathsf{FP}(\mathcal{C})}$ implies $\mathsf{P}_{tt}(A) \subseteq \mathsf{AP}_{\mathsf{FP}(\mathcal{C})}$. In case $A \in \mathcal{C}$ the assumption $pad(A) \in \mathsf{AP}_{\mathsf{FP}(\mathcal{C})}$ even implies $\mathsf{P}(A) \subseteq \mathsf{AP}_{\mathsf{FP}(\mathcal{C})}$.*

Proof. Let L be a set accepted by some oracle Turing machine M with oracle A whose running time is bounded by some polynomial p. We can assume that on any input x, M asks exactly 2^k different queries for some $k \geq 0$. Consider the oracle Turing machine M' which on input x simulates M but replaces each oracle query y by the query $0^{|x|}1xy10^i$ where $i = p(|x|) - |y|$. Then M' accepts L with oracle $pad(A)$ which by assumption is in $\mathsf{AP}_{\mathsf{FP}(\mathcal{C})}$. We notice that M' is a monotone oracle machine in the sense that for all strings $x < x'$, any oracle query on input x is lexicographically smaller than any oracle query on input x'. Furthermore, it is easy to determine for any string z the lexicographically smallest input string x (denoted by x_z) such that $z \leq 0^{|x|}1x1^{p(|x|)+1}$. Note that if there is any input x such that $M'(x)$ asks query z then $x = x_z$.

Let μ be an arbitrary distribution in $\mathsf{FP}(\mathcal{C})$ and consider the distribution ν induced by μ on the queries of M' defined as

$$\nu'(z) = \begin{cases} \frac{\mu'(x_z)}{m(x_z)}, & \text{if } z \in Q(x_z, M', pad(A)), \\ 0, & \text{otherwise.} \end{cases}$$

where $m(x_z) = \|Q(x_z, M', pad(A))\|$. It is not difficult to see that (L, μ) Turing reduces to $(pad(A), \nu)$ via M'. Thus it only remains to show that ν is $\mathsf{P}(\mathcal{C})$-computable. Let x^- denote the predecessor of x in lexicographic order. Then it follows by the monotonicity of M' that

$$\begin{aligned} \nu(z) = \sum_{w \leq z} \nu'(w) \;&=\; \sum_{w < x_z} \mu'(w) + l \cdot \mu'(x_z)/m \\ &= \begin{cases} l(x_z) \cdot \mu'(\lambda)/m(x_z), & \text{if } x_z = \lambda, \\ \mu(x_z^-) + l(x_z) \cdot \mu'(x_z)/m(x_z), & \text{otherwise} \end{cases} \end{aligned}$$

where $l(x_z)$ is the number of strings in $Q(x_z, M', pad(A))$ less than or equal to z. Now it is easy to see that ν is $\mathsf{P}(\mathcal{C})$-computable, provided that either M is nonadaptive or A is contained in $\mathcal{C}$. $\square$

Since the classes Σ_k^p have complete paddable sets, we immediately get the following two corollaries.

Corollary 3.5 *For any oracle class $\mathcal{C}$, $\Sigma_k^p \subseteq \mathsf{AP}_{\mathsf{FP}(\mathcal{C})}$ implies $\Theta_{k+1}^p \subseteq \mathsf{AP}_{\mathsf{FP}(\mathcal{C})}$.*

Corollary 3.6 *For any integers $l \geq k \geq 1$, $\Sigma_k^p \subseteq \mathsf{AP}_{\mathsf{FP}(\Sigma_l^p)}$ implies $\Delta_{k+1}^p \subseteq \mathsf{AP}_{\mathsf{FP}(\Sigma_l^p)}$.*

As noted by Ben-David et al. [BCGL92], all sets in $\mathsf{AP}_{\mathsf{FP}}$ are decidable in polynomial time on tally inputs. This follows from the fact that there exists a distribution which gives high probability to every tally string. More generally, we can use the assumption that a problem is efficiently decidable on average with respect to a particular distribution to show that it is efficiently decidable (in the worst case) on a certain domain.

Definition 3.7 *Let D be a subset of Σ^*. We say that a set A is* efficiently decidable on domain D*, if there exists a Turing machine M which decides A and the running time of M is polynomially bounded on all strings in D.*

The standard distribution on Σ^*, μ_{st}, gives uniform probability to all strings of the same length. More precisely, for all $n \geq 0$ and all $x \in \Sigma^n$ let

$$\mu'_{\mathrm{st}}(x) = 2^{-2\log(n+2)+1} \cdot 2^{-n}.$$

We notice that $\mu'_{\mathrm{st}}(x) \geq 2^{-n}/8n^2$ for $n \geq 1$ and that the distribution μ_{st} is P-computable.

Theorem 3.8 *For any polynomial p, every set $A \in \mathsf{AP}_{\mathsf{FP}}$ is efficiently decidable on some domain D of density $\|D \cap \Sigma^n\| \geq (1 - 1/p(n)) \cdot 2^n$.*

Proof. Let M be a Turing machine witnessing the fact that A is decidable in time polynomial on μ_{st}-average, i.e., for some constant c,

$$\sum_{x \neq \lambda} \frac{f(x)^{1/c}}{|x|} \mu'_{\mathrm{st}}(x) < \infty,$$

where f denotes the running time of M. For any input length n, let X_n denote the set of strings $x \in \Sigma^n$ such that $f(x) \geq (8n^3 p(n))^c$. Then, since $\mu'_{\mathrm{st}}(x) \geq 2^{-n}/8n^2$ for $n \geq 1$, it follows that

$$\infty > \sum_{n \geq 1} \sum_{x \in \Sigma^n} \frac{f(x)^{1/c}}{n} \mu'_{\mathrm{st}}(x) \geq \sum_{n \geq 1} \sum_{x \in X_n} \frac{8n^3 p(n)}{n} \mu'_{\mathrm{st}}(x) \geq \sum_{n \geq 1} \|X_n\| 2^{-n} p(n),$$

implying that $\|X_n\| \leq 2^n/p(n)$ for almost all n. $\square$

Let D be a (infinite) set and let $rank_D$ denote the ranking function of D, i.e., $rank_D(x) = \|\{y \in D \mid y < x\}\|$ is the number of strings in D that are lexicographically smaller than x. The *natural distribution on* D, denoted by μ_D, gives positive probability only to strings in D. More precisely, for all $x \in \Sigma^*$, let

$$\mu'_D(x) = \begin{cases} 2^{-2\log(rank_D(x)+2)+1}, & \text{if } x \in D \\ 0, & \text{otherwise.} \end{cases}$$

Then it is clear that μ_D is $\mathsf{P}(\mathcal{C})$-computable, if $rank_D$ is computable in $\mathsf{FP}(\mathcal{C})$. Furthermore, in case D is sparse, $\mu'_D(x) \geq 1/p(|x|)$ holds for some polynomial p and all strings in D.

A set A is called $\mathsf{P}(\mathcal{C})$-printable (cf. [HY84]), if there exists a set $C \in \mathcal{C}$ and a polynomial-time bounded oracle transducer T such that the output of T with oracle C and input 0^n is an enumeration of all strings in A of length n.

Theorem 3.9 *Let $A \in \mathsf{AP}_{\mathsf{FP}(\mathcal{C})}$ for some language class $\mathcal{C}$. Then A is efficiently decidable on any $\mathsf{P}(\mathcal{C})$-printable domain.*

Proof. Let D be a $P(\mathcal{C})$-printable set and let μ_D be the natural distribution on D. Since the ranking function of a $P(\mathcal{C})$-printable set is in $\mathsf{FP}(\mathcal{C})$, it follows that μ_D is $\mathsf{P}(\mathcal{C})$-computable. Moreover, since D is sparse, $\mu'_D(x) \geq 1/p(|x|)$ for some polynomial p and every string $x \in D$. On the other hand, the assumption $A \in \mathsf{AP}_{\mathsf{FP}(\mathcal{C})}$ implies that there is a Turing machine M that decides A in time polynomial on μ_D-average. That is, for some constant c,

$$\sum_{x \neq \lambda} \frac{f(x)^{1/c}}{|x|} \mu'_D(x) < c,$$

where f denotes the running time of M. Hence, for any string $x \in D$, $f(x)$ is bounded by $(c|x|/\mu'_D(x))^c \leq (c|x|p(|x|))^c$, implying that the running time of M is polynomially bounded on inputs from D. $\square$

Corollary 3.10 [BCGL92] *Any set in $\mathsf{AP}_{\mathsf{FP}}$ is efficiently decidable on the tally domain 0^*, implying that $\mathsf{AP}_{\mathsf{FP}} \cap \mathrm{TALLY} \subseteq \mathsf{P}$.*

4 Computing the advice efficiently on average

The unlikely collapse of $\mathsf{NE} = \mathrm{NTIME}(2^{O(n)})$ to $\mathsf{E} = \mathrm{DTIME}(2^{O(n)})$ was the first consequence that has been derived from the assumption that all NP problems are decidable in time polynomial on μ-average for any distribution $\mu \in \mathsf{FP}$.

Theorem 4.1 [BCGL92] *If* $\mathsf{NP} \subseteq \mathsf{AP}_{\mathsf{FP}}$, *then* $\mathsf{E} = \mathsf{NE}$ *(or, equivalently,* $\mathsf{NP} \cap \mathrm{TALLY} \subseteq \mathsf{P}$*).*

Proof. Recall that $\mathsf{E} = \mathsf{NE}$ if and only if every tally set in NP is in P [Boo74]. Since, by Corollary 3.10, every tally set in $\mathsf{AP}_{\mathsf{FP}}$ is already in P it follows that $\mathsf{NP} \subseteq \mathsf{AP}_{\mathsf{FP}}$ implies $\mathsf{E} = \mathsf{NE}$. □

Thus, if NP problems have efficient average-case decision algorithms, then $\mathsf{NP} \cap \mathrm{TALLY} \subseteq \mathsf{P}$, and therefore, any set in P/poly for which the advice is computable in $\mathsf{FP}(\mathsf{NP} \cap \mathrm{TALLY})$ belongs to P. We observe that similar collapse consequences can be derived for other subclasses of P/poly (see Corollary 4.4 below). Roughly speaking, we use the assumption that the advice is efficiently computable with an oracle that is easy on average to actually eliminate the need for advice. To be more precise, call an oracle machine M *honest*, if there is a constant $c > 0$ such that on inputs of length n, M only asks queries of length at least n^c. Then we show that any set in P/poly is efficiently decidable without advice, provided that the advice function h is efficiently computable by a non-adaptive and honest transducer relative to some oracle A (in symbols: $h \in \mathsf{FP}_{tt,honest}(A)$) which is easy on average.

Theorem 4.2 *If* $A \in \mathsf{AP}_{\mathsf{FP}}$, *then any advice function in* $\mathsf{FP}_{tt,honest}(A)$ *is computable in* FP.

Proof. Observe that for a nonadaptive and honest oracle transducer T, the set

$$D = \{y \in \Sigma^* \mid \text{for some } n \geq 0, T \text{ on input } 0^n \text{ asks query } y\}$$

is P-printable. Using the assumption $A \in \mathsf{AP}_{\mathsf{FP}}$, it follows by Theorem 3.9 that A is efficiently decidable on D. But this implies that the advice function computed by T^A is in FP. □

Note that under the stronger assumption that all sets that Turing reduce to A are easy on average it follows from Corollary 3.10 that all advice functions in $\mathsf{FP}(A)$ belong to FP.

Corollary 4.3 *If* $\mathsf{P}(A) \subseteq \mathsf{AP}_{\mathsf{FP}}$ *then any advice function in* $\mathsf{FP}(A)$ *is in* FP.

Proof. Observe that for any advice function $f \in \mathsf{FP}(A)$ there is a tally oracle $A' \in \mathsf{P}(A)$ such that $f \in \mathsf{FP}(A')$ (define, e.g., $A' = \{\langle 0^n, 0^i \rangle \mid \text{the } i\text{-th query in the computation of } f(0^n) \text{ is in } A\}$). □

We notice that the conclusion of Corollary 4.3 is equivalent to the statement $\mathsf{FE}(A)$ is in FE, where FE is the class of functions computable in time $2^{O(n)}$.

Now, using results from [BS92,AKM96,Köb94,Gav95], we can state similar collapse consequences as in Theorem 4.1 for several subclasses of P/poly.

Corollary 4.4

(1) If $\mathsf{NP} \subseteq \mathsf{AP_{FP}}$ *then* $\mathsf{NP} \cap \mathsf{P}/log = \mathsf{P}$.
(2) If $\Delta_2^p \subseteq \mathsf{AP_{FP}}$ *then* $\Delta_2^p \cap \mathsf{IC}[\log, \mathrm{poly}] = \mathsf{P}$.
(3) If $\Sigma_2^p \subseteq \mathsf{AP_{FP}}$ *then all sets in* $\Sigma_2^p \cap \Pi_2^p$ *that conjunctively, disjunctively, or bounded truth-table reduce to some sparse set are in* P.
(4) If $\Delta_3^p \subseteq \mathsf{AP_{FP}}$ *then* $\Sigma_2^p \cap \Pi_2^p \cap \mathsf{P}/poly = \mathsf{P}$ *and hence* $\mathsf{BPP} = \mathsf{P}$.
(5) For $k \geq 3$, *if* $\Sigma_k^p \subseteq \mathsf{AP_{FP}}$ *then* $\Sigma_k^p \cap \Pi_k^p \cap \mathsf{P}/poly = \mathsf{P}$.

Proof.

(1) As shown in [BS92], every set L in $\mathsf{NP} \cap \mathsf{P}/\log$ is contained in $\mathsf{P}(S)$ for some sparse set $S \in \mathsf{NP}$. Furthermore, as shown in [Har83], $\mathsf{P}(\mathsf{NP} \cap \mathrm{SPARSE}) = \mathsf{P}(\mathsf{NP} \cap \mathrm{TALLY})$, implying that $\mathsf{NP} \cap \mathsf{P}/\log \subseteq \mathsf{P}(\mathsf{NP} \cap \mathrm{TALLY})$. Assuming $\mathsf{NP} \subseteq \mathsf{AP_{FP}}$, the collapse now follows by Corollary 3.10.
(2) As shown in [AKM96], every set L in $\mathsf{IC}[\log, \mathrm{poly}]$ is in $\mathsf{P}(T)$ for some tally set $T \in \mathsf{P}(\mathsf{NP} \oplus L)$. Therefore, if additionally $L \in \Delta_2^p$, then $L \in \mathsf{P}(T)$ for some tally set $T \in \Delta_2^p$. Since by Corollary 3.10 every tally set in $\mathsf{AP_{FP}}$ is in P it follows by the assumption $\Delta_2^p \subseteq \mathsf{AP_{FP}}$ that $L \in \mathsf{P}$.
(3) In [AKM96] it is also shown that every set L that conjunctively, disjunctively, or bounded truth-table reduces to some sparse set is in $\mathsf{P}(S)$ for some sparse oracle S that can be decided in $\mathsf{NP}(L \oplus \mathsf{NP})$ with the help of an advice function h in $\mathsf{FP}(\mathsf{NP}(L) \cap \mathrm{TALLY})$. Therefore, if additionally $L \in \Sigma_2^p \cap \Pi_2^p$, then h is computable in $\mathsf{FP}(\Sigma_2^p \cap \mathrm{TALLY})$ and thus, using the assumption $\Sigma_2^p \subseteq \mathsf{AP_{FP}}$, in FP. This implies $S \in \Sigma_2^p$ and an analogous reasoning as in the proof of part one gives us $L \in \mathsf{P}$.
(4) As shown in [Köb94], every set $L \in \mathsf{P}/\mathrm{poly}$ has an advice function h computable in $\mathsf{FP}(\mathsf{NP}(L) \oplus \Sigma_2^p)$. Therefore, if additionally $L \in \Sigma_2^p \cap \Pi_2^p$, then $h \in \mathsf{FP}(\Sigma_2^p)$ and thus, using the assumption $\Delta_3^p \subseteq \mathsf{AP_{FP}}$ and Corollary 4.3, in FP. The consequence that $\mathsf{BPP} \subseteq \mathsf{P}$ is immediate since $\mathsf{BPP} \subseteq \Sigma_2^p \cap \Pi_2^p$ [Sip83,Lau83] and $\mathsf{BPP} \subseteq \mathsf{P}/\mathrm{poly}$ [BG81].
(5) As shown in [Gav95], every set $L \in \mathsf{P}/\mathrm{poly}$ is in $\mathsf{P}(T)$ for some tally set T in $\mathsf{NP}(L \oplus \Sigma_2^p)$. Therefore, if additionally $L \in \Sigma_k^p \cap \Pi_k^p$, then $L \in \mathsf{P}(T)$ for some tally set $T \in \mathsf{NP}(\Sigma_k^p \cap \Pi_k^p) = \Sigma_k^p$. Since by Corollary 3.10 every tally set in $\mathsf{AP_{FP}}$ is in P it follows by the assumption $\Sigma_k^p \subseteq \mathsf{AP_{FP}}$ that $L \in \mathsf{P}$.

□

We notice that the conclusions of the implications stated in Corollary 4.4 are actually derived from the (possibly weaker) assumption that all tally sets in the respective levels of PH are in P. Hence, part 4 subsumes the result shown in [BFNW93] that $\mathsf{BPP} = \mathsf{P}$ follows from the assumption $\Sigma_4^p \cap \mathrm{TALLY} \subseteq \mathsf{P}$. Very recently, Buhrman, Fortnow, and Pavan [BFP03] have shown that

$\mathsf{BPP} = \mathsf{P}$ can also be derived from the assumption $\mathsf{NP} \subseteq \mathsf{AP}_{\mathsf{FP}}$ (which seems incomparable to our assumption $\Delta_3^p \cap \mathrm{TALLY} \subseteq \mathsf{P}$).

As a further application we use Corollary 4.4 to show that complexity classes like $\mathsf{P}(\mathsf{PP})$ and PSPACE are intractable in the average case unless they collapse downto P. In the proof, we exploit the fact that these classes contain complete random self-reducible problems. We use the following definition of random self-reducibility (see also [FKN90,Fei93,FF93]).

Definition 4.5 *Let $f : \Sigma^* \to \Sigma^*$ be a function.*

- *We say that f is* random self-reducible *if there are a polynomial p, a set B of density $\|B \cap \Sigma^n\| \geq (3/4)2^n$, and polynomial-time computable functions g and h such that for all n and $x \in \Sigma^n$,*
  - *for every $r \in B \cap \Sigma^{p(n)}$, $f(x) = g(x, r, b_1, \ldots, b_{p(n)})$, where $b_i = f(h(x, r, i))$ for $i = 1, \ldots, p(n)$, and*
  - *if r is chosen uniformly from $\Sigma^{p(n)}$, then $h(x, r, i)$ is uniform over Σ^n, for $i = 1, \ldots, p(n)$.*
- *A set A is called* random self-reducible *if its characteristic function χ_A is random self-reducible.*
- *We say that f is* bounded-error computable in polynomial time *if there are a randomized transducer T and a constant $\varepsilon > 0$ such that*
  - *T runs in polynomial time and*
  - *on any input x, T outputs $f(x)$ with probability at least $\frac{1}{2} + \varepsilon$.*
- *Let q be a polynomial such that $|f(x)| \leq q(|x|)$ for all $x \in \Sigma^*$. Then $C_q(f)$ denotes the set*

$$C_q(f) \;=\; \{\langle x, j, b\rangle \mid j \in \{1, \ldots, q(|x|)\} \text{ and the } j\text{th bit of } f(x) \text{ is } b\}.$$

By applying ideas from [Lip91] we easily get the following lemma.

Lemma 4.6 *Any random self-reducible function f with $C_q(f) \in \mathsf{AP}_{\mathsf{FP}}$ is bounded-error computable in polynomial time.*

Proof. Let f be a random self-reducible function and let q be a polynomial such that $C_q(f)$ is in $\mathsf{AP}_{\mathsf{FP}}$ and $|f(x)| \leq q(|x|)$ holds for all $x \in \Sigma^*$. Further, let p be a polynomial, B be a set, and g and h be polynomial-time computable functions according to Definition 4.5. Since we can assume that all strings $\langle y, j, b\rangle$ with $y \in \{0,1\}^n$ and $(j, b) \in \{1, \ldots, q(n)\} \times \{0, 1\}$ have (uniform) length $l(n) = n + O(\log n)$, the function $s(n) = 8p(n)2^{l(n)-n}$ is polynomially bounded. Hence, it follows from Theorem 3.8 that $C_q(f)$ is efficiently decidable on some domain D of density $\|D \cap \Sigma^{l(n)}\| \geq (1 - 1/s(n)) \cdot 2^{l(n)}$. Now consider the following randomized transducer T:

> On input x randomly guess a string $r \in \Sigma^{p(n)}$. If for $i = 1, \ldots, p(n)$ and $j = 1, \ldots, q(n)$, both strings $\langle h(x, r, i), j, 0\rangle$ and $\langle h(x, r, i), j, 1\rangle$ belong to

D, then determine the function values $b_i = f(h(x, r, i))$, for $i = 1, \ldots, p(n)$, and output $g(x, r, b_1, \ldots, b_{p(n)})$. Otherwise reject.

We call a string $y \in \{0,1\}^n$ bad, if there exists a pair $(j, b) \in \{1, \ldots, q(n)\} \times \{0,1\}$ such that $\langle y, j, b\rangle \notin D$. Since at most $2^{l(n)}/s(n) = 2^n/8p(n)$ many strings of length $l(n)$ do not belong to D, it follows that a randomly chosen string y of length n is bad with probability at most $1/8p(n)$. Clearly, T on input x only fails to output $f(x)$ if r does not belong to B or at least one of the strings $h(x, r, 1), \ldots, h(x, r, p(n))$ is bad, and so the probability for this is bounded by $1/4 + p(n)(1/8p(n)) = 3/8$. □

Lemma 4.6 immediately gives us the following implications.

Theorem 4.7 *If* $\mathsf{P}_{tt}(A) \subseteq \mathsf{AP}_{\mathsf{FP}}$ *(*$\mathsf{P}(A) \subseteq \mathsf{AP}_{\mathsf{FP}}$*), then any random self-reducible function in* $\mathsf{FP}_{tt}(A)$ *(respectively,* $\mathsf{FP}(A)$*) is bounded-error computable in polynomial time. Further, any random self-reducible set in* $\mathsf{AP}_{\mathsf{FP}}$ *belongs to* BPP*.*

By using results from [Lip91,FF93,GKR$^+$95,KT96] we now get the following corollary.

Corollary 4.8 *For* $\mathcal{K} \in \{\mathsf{PSPACE}, \mathsf{P}(\mathsf{PP}), \mathsf{MP}, \mathsf{ModP}\}$*,* $\mathcal{K}$ *is not contained in* $\mathsf{AP}_{\mathsf{FP}}$ *unless* $\mathcal{K} = \mathsf{P}$*.*

Proof. We first consider the case that $\mathsf{PSPACE} \subseteq \mathsf{AP}_{\mathsf{FP}}$. Since any set $A \in \mathsf{PSPACE}$ is reducible to a random self-reducible function $f \in \mathsf{FP}(\mathsf{PSPACE})$ (for example, the multilinear extension of A has these properties [Lip91,BFL91]), Theorem 4.7 implies that $\mathsf{PSPACE} = \mathsf{BPP}$. Since $\Delta_3^p \subseteq \mathsf{PSPACE}$, it follows by Corollary 4.4 that $\mathsf{PSPACE} = \mathsf{P}$.

Next assume that $\mathcal{K} \in \{\mathsf{MP}, \mathsf{ModP}\}$ is contained in $\mathsf{AP}_{\mathsf{FP}}$. Since MP and ModP have complete paddable sets and since $\mathsf{FP}_{tt}(\#\mathsf{P}) = \mathsf{FP}_{tt}(\mathcal{K})$ [GKR$^+$95,KT96], it follows by Theorems 3.4 and 4.7 that any random self-reducible function in $\mathsf{FP}_{tt}(\#\mathsf{P})$ is bounded-error computable in polynomial time. Since the permanent of $n \times n$ matrices (over GF(p) where the prime $p > n + 1$ is given as part of the input) is a random self-reducible function in $\#\mathsf{P}$ [Lip91] and since computing the permanent is Turing-hard for the class PP [Val79], we get $\mathsf{PP} = \mathsf{BPP}$. But since $\Delta_3^p \subseteq \mathsf{P}_{tt}(\#\mathsf{P})$ [Tod91], $\mathsf{PP} = \mathsf{P}$ follows by Corollary 4.4, implying that $\mathcal{K} = \mathsf{P}$.

Finally, assume that $\mathsf{P}(\mathsf{PP}) \subseteq \mathsf{AP}_{\mathsf{FP}}$. Since $\mathsf{P}(\mathsf{MP}) = \mathsf{P}(\mathsf{PP})$ we get $\mathsf{MP} = \mathsf{P}$, implying that $\mathsf{P}(\mathsf{PP}) = \mathsf{P}$. □

5 Verifying the advice efficiently on average

In this section we use average-case assumptions to derive collapse consequences for several subclasses of $\mathsf{NP}/\mathrm{poly}$. Let L be a set in $\mathsf{NP}/\mathrm{poly}$. In order to derive $L \in \mathsf{NP}$ it suffices to show that the correctness of a given advice string can be checked by an NP computation. To formalize the complexity of verifying the advice we use the notion of multivalued advice functions.

Definition 5.1 *Let I be an interpreter set that uses length $p(n)$ advice strings to decide length n instances of a set L.*

- *We call a string w of length $p(n)$* correct for L *(with respect to I and p) if for all $x \in \Sigma^n$,*
$$x \in L \;\Leftrightarrow\; \langle x, w\rangle \in I.$$
- *A multivalued function h is called* advice function for L *(with respect to I and p) if for all n,*
  - *$h(0^n)$ is defined,*
  - *set-$h(0^n)$ only contains strings of length $p(n)$, and*
  - *all strings in set-$h(0^n)$ are correct.*
- *A set H is called* advice set for L *if H is the range of some multivalued advice function h for L, i.e., $H = \bigcup_{n\geq 0}$ set-$h(0^n)$. In case H' is the range of some refinement h' of h, we call H' a* refinement *of H.*

Now we are ready to show that computing the advice can be replaced by nondeterminism, provided that checking the correctness of an advice string is efficient on average.

Theorem 5.2 *If A belongs to $\mathsf{AP}_{\mathsf{FP}(\mathsf{NP}(A))}$, then any advice function $h \in \mathsf{NPMV}_{honest}(A)$ has a refinement in NPMV.*

Proof. Let h be an advice function in $\mathsf{NPMV}_{honest}(A)$ with respect to some interpreter set I and polynomial p and let T be some honest nondeterministic oracle transducer computing h under oracle A. Let q be a polynomial bounding the running time of T and let $c > 0$ be a constant such that $|y| > n^c$ for all oracle queries y of T on input 0^n. Consider the set D consisting of all queries y asked by T^A on input 0^n on its leftmost accepting computation:

$$D = \{y \mid \exists n : T^A(0^n) \text{ asks query } y \text{ on its leftmost accepting computation}\}.$$

Since T is honest it follows that D is $\mathsf{P}(\mathsf{NP}(A))$-printable. Moreover, since A belongs to $\mathsf{AP}_{\mathsf{FP}(\mathsf{NP}(A))}$, it follows from Theorem 3.9 that A is efficiently decidable on D. But this implies that h has a refinement in NPMV. □

As an application we state two corollaries.

Corollary 5.3 *If* $\mathsf{NP} \subseteq \mathsf{AP}_{\mathsf{FP}(\Sigma_2^p)}$ *then any advice function in* $\mathsf{NPMV}(\mathsf{NP})$ *has a refinement in* NPMV*, implying that any set in* NP*/poly with an advice function in* $\mathsf{NPMV}(\mathsf{NP})$ *belongs to* NP*.*

Proof. Let $L \in \mathsf{NP}/\mathrm{poly}$ via an advice function $h \in \mathsf{NPMV}(\mathsf{NP}) = \mathsf{NPMV}_{honest}(\mathsf{NP})$. Since by assumption $\mathsf{NP} \subseteq \mathsf{AP}_{\mathsf{FP}(\Sigma_2^p)}$, it follows from Theorem 5.2 that h has a refinement in NPMV. □

Notice that a set A is in $\mathsf{NP}/\mathrm{poly} \cap \text{co-}\mathsf{NP}/\mathrm{poly}$ if and only if there are a polynomial p and interpreter sets $I \in \mathsf{NP}$ and $I' \in \text{co-}\mathsf{NP}$ such that for all n there is a string w of length $p(n)$ such that for all strings x of length n,

$$x \in A \Leftrightarrow \langle x, w\rangle \in I \Leftrightarrow \langle x, w\rangle \in I'.$$

A belongs to the class $\mathsf{NPMV}_t/\mathrm{poly}$ (see [AK02,CHW99]) if I and I' additionally fulfill the property that for all n, all strings x of length n, and all strings w of length $p(n)$, either $\langle x, w\rangle \in I$ or $\langle x, w\rangle \in I'$. Clearly, $(\mathsf{NP} \cap \text{co-}\mathsf{NP})/\mathrm{poly} \subseteq \mathsf{NPMV}_t/\mathrm{poly} \subseteq \mathsf{NP}/\mathrm{poly} \cap \text{co-}\mathsf{NP}/\mathrm{poly}$. In [AK02] it is shown that any self-reducible set $A \in \mathsf{NPMV}_t/\mathrm{poly}$ is low for Σ_2^p. In fact, the proof shows that A (as well as its complement) is in $\mathsf{NP}/\mathrm{poly}$ via an advice function in $\mathsf{NPMV}(\mathsf{NP})$, implying the following corollary.

Corollary 5.4 *If* $\mathsf{NP} \subseteq \mathsf{AP}_{\mathsf{FP}(\Sigma_2^p)}$ *then any self-reducible set in* NPMV_t*/poly belongs to* $\mathsf{NP} \cap$ *co-*NP*.*

6 Abundant advice

An advice set $H \subseteq \bigcup_{n\geq 0} \Sigma^{p(n)}$ is called *abundant* if for some polynomial q and all n, H contains at least $2^{p(n)}/q(n)$ strings of length $p(n)$.

Theorem 6.1 *Any abundant advice set* $H \in \mathsf{AP}_{\mathsf{FP}}$ *has an abundant refinement* $H' \in \mathsf{P}$*.*

Proof. Let H be an abundant advice set in $\mathsf{AP}_{\mathsf{FP}}$, i.e., $\|H \cap \Sigma^{p(n)}\| \geq 2^{p(n)}/q(n)$ for some polynomial q. By Theorem 3.8, H is efficiently decidable on some domain $D \in \mathsf{P}$ of density $\|D \cap \Sigma^{p(n)}\| \geq (1 - 1/2q(n))2^{p(n)}$. Letting $H' = H \cap D$, it follows that $\|H' \cap \Sigma^{p(n)}\| \geq 2^{p(n)}/2q(n)$. □

The concept of abundance for advice sets has been (implicitly) used by [Imp95] to show that $\mathsf{NP} \subseteq \mathsf{AP}_{\mathsf{FP}}$ implies $\mathsf{BPP} = \mathsf{ZPP}$. To get this result, Impagliazzo exploited the fact that any set in BPP belongs to P/poly via an abundant advice set $H \in \text{co-}\mathsf{NP}$ (implicit in [NW94]). More recently, it has been shown in [AK01,GZ97] that the Arthur-Merlin class MA is contained in $\mathsf{ZPP}(\mathsf{NP})$. In fact, the proof given in [AK01] actually shows that any set in MA (AM)

belongs to NP/poly via an abundant advice set in co-NP (respectively, Π_2^p). By using these results we get as an extension of Impagliazzo's result that under the assumption $\mathsf{NP} \subseteq \mathsf{AP_{FP}}$ the class MA can be derandomized, i.e., $\mathsf{MA} = \mathsf{NP}$, whereas under the stronger assumption $\Sigma_2^p \subseteq \mathsf{AP_{FP}}$ even AM can be derandomized, i.e., $\mathsf{AM} = \mathsf{NP}$. Note that $\mathsf{AM} = \mathsf{NP}$ has some immediate strong implications as, for example, that Graph Isomorphism is in $\mathsf{NP} \cap$co-NP.

Corollary 6.2

(1) If $\mathsf{NP} \subseteq \mathsf{AP_{FP}}$ *then* $\mathsf{MA} = \mathsf{NP}$.
(2) If $\Sigma_2^p \subseteq \mathsf{AP_{FP}}$ *then* $\mathsf{AM} = \mathsf{NP}$.

Proof. As shown in [AK01], any set $A \in \mathsf{MA}$ has an abundant advice set $H \in$ co-NP with respect to some interpreter set $I \in \mathsf{NP}$. Since by assumption NP and therefore also co-NP is contained in $\mathsf{AP_{FP}}$, it follows by Theorem 6.1 that H has an (abundant) refinement $H' \in \mathsf{P}$. Now it is easy to decide A in NP: On input x, guess an advice string w of suitable length and accept if and only if $w \in H'$ and $\langle x, w \rangle \in I$. The proof that $\Sigma_2^p \subseteq \mathsf{AP_{FP}}$ implies $\mathsf{AM} = \mathsf{NP}$ proceeds along exactly the same lines. □

Note that the above proof shows that in order to derive $\mathsf{MA} = \mathsf{NP}$ ($\mathsf{AM} = \mathsf{NP}$) it suffices to assume that for any set L in co-NP (respectively, Π_2^p) there is some nondeterministic Turing machine for L whose running time is polynomial on average with respect to the standard distribution.

7 Computing advice with zero-error

In this section we consider nonuniform classes where the advice can be computed by a randomized procedure with zero error probability. Let h be a (multivalued) advice function with respect to some interpreter set I and some polynomial p. h is called $\mathsf{FZPP}(A)$-computable if there is a polynomial-time randomized oracle transducer T such that for all n,

- T^A on input 0^n produces some output with probability at least $1/2$ and
- $output_T^A(0^n) = set\text{-}h(0^n)$.

If the oracle A is the empty set then we just say that h is FZPP-computable. Now we are ready to show that if the advice is computable by a zero-error randomized transducer asking parallel queries to some oracle A for which $\mathsf{P}_{tt}(A)$ is easy on average, then we can get rid of the oracle.

Theorem 7.1 *If* $\mathsf{P}_{tt}(A) \subseteq \mathsf{AP_{FP}}$ *(*$\mathsf{P}(A) \subseteq \mathsf{AP_{FP}}$*), then any advice function* h *in* $\mathsf{FZPP}_{tt}(A)$ *(respectively,* $\mathsf{FZPP}(A)$*) has an* FZPP*-computable refinement* h'.

Proof. Let T be a randomized oracle transducer computing h by asking (parallel) queries to A and let q be a polynomial bounding the running time of T. Then the set

$$B = \{\langle r, n, i, b\rangle \mid T^A \text{ on input } 0^n \text{ and random sequence } r \in \Sigma^{q(n)} \text{ outputs an advice string } w \in \Sigma^{p(n)} \text{ whose } i\text{-th bit is } b\}$$

is easily seen to belong to $\mathsf{P}_{tt}(A)$ (respectively, $\mathsf{P}(A)$) and hence in $\mathsf{AP}_{\mathsf{FP}}$. Since we can assume that all strings $\langle r, n, i, b\rangle$ in B have length $l(n) = q(n) + O(\log n)$, the function $2^{l(n)-q(n)+2}$ is polynomially bounded. Hence, it follows from Theorem 3.8 that B is efficiently decidable on some domain D of density $\|D \cap \Sigma^{l(n)}\| \geq (1 - 1/2^{l(n)-q(n)+2}) \cdot 2^{l(n)}$. Now consider the following randomized transducer T':

> On input 0^n randomly guess a string $r \in \Sigma^{q(n)}$ and check whether for $i = 1, \ldots, p(n)$, at least one of the two strings $\langle r, n, i, 0\rangle$ and $\langle r, n, i, 1\rangle$ belongs to $B \cap D$. If so then output the corresponding advice string w, otherwise reject.

Since for at least $2^{q(n)-1}$ strings $r \in \Sigma^{q(n)}$, B contains for every $i \in \{1, \ldots, p(n)\}$ exactly one of the two strings $\langle r, n, i, 0\rangle$ and $\langle r, n, i, 1\rangle$, and since at most $2^{l(n)}/2^{l(n)-q(n)+2} = 2^{q(n)-2}$ of them do not belong to D, it follows that T' on input 0^n outputs with probability at least $1/4$ some advice string $w \in H$. This (together with a standard probability amplification argument) completes the proof. □

By using results from [BCG+96,KW98] it is now easy to derive the following corollary.

Corollary 7.2

(1) [Wat96] If $\Delta_2^p \subseteq \mathsf{AP}_{\mathsf{FP}}$ *then every self-reducible set in* P*/poly is in* ZPP*.*
(2) If $\mathsf{NP} \subseteq \mathsf{AP}_{\mathsf{FP}(\mathsf{NP})}$ *then every self-reducible set in* P*/poly is in* ZPP*.*

It is interesting to note that Corollary 7.2 implies stronger collapse consequences for the polynomial hierarchy. For example, if $\Delta_2^p \subseteq \mathsf{AP}_{\mathsf{FP}}$ then $\mathsf{NP} \subseteq \mathsf{P}/\mathrm{poly}$ implies $\mathsf{PH} = \mathsf{ZPP}$.

Acknowledgments. We are very grateful to Osamu Watanabe for permitting us to include part one of Corollary 7.2 in the paper. Also, we thank an anonymous referee for several comments that very much improved the readability of the paper.

References

[AFK89] M. Abadi, J. Feigenbaum, and J. Kilian. On hiding information from an oracle. *Journal of Computer and System Sciences*, **39**:21–30, 1989.

[AHH+93] V. Arvind, Y. Han, L. A. Hemachandra, J. Köbler, A. Lozano, M. Mundhenk, M. Ogiwara, U. Schöning, R. Silvestri, and T. Thierauf. Reductions to sets of low information content. In K. Ambos-Spies, S. Homer, and U. Schöning, editors, *Complexity Theory, Current Research*, 1–45. Cambridge University Press, 1993.

[AK01] V. Arvind and J. Köbler. On pseudorandomness and resource-bounded measure. *Theoretical Computer Science*, **255**(1–2):205–221, 2001.

[AK02] V. Arvind and J. Köbler. New lowness results for ZPP(NP) and other complexity classes. *Journal of Computer and System Sciences*, **65**(2):257–277, 2002.

[AKM96] V. Arvind, J. Köbler, and M. Mundhenk. Upper bounds for the complexity of sparse and tally descriptions. *Mathematical Systems Theory*, **29**(1):63–94, 1996.

[AL83] D. Angluin and D. Lichtenstein. Provable security of cryptosystems: a survey. Technical Report YALEU/DCS/TR-288, Yale University, New Haven, CT, 1983.

[BCG+96] N. Bshouty, R. Cleve, R. Gavaldà, S. Kannan, and C. Tamon. Oracles and queries that are sufficient for exact learning. *Journal of Computer and System Sciences*, **52**:421–433, 1996.

[BCGL92] S. Ben-David, B. Chor, O. Goldreich, and M. Luby. On the theory of average case complexity. *Journal of Computer and System Sciences*, **44**:193–219, 1992.

[BDG95] J. L. Balcázar, J. Díaz, and J. Gabarró. *Structural Complexity I.* EATCS Monographs on Theoretical Computer Science. Springer-Verlag, Berlin Heidelberg New York, second edition, 1995.

[BF90] D. Beaver and J. Feigenbaum. Hiding instances in multioracle queries. In *Proc. 7th Symposium on Theoretical Aspects of Computer Science*, Lecture Notes in Computer Science #415, 37–48. Springer-Verlag, Berlin Heidelberg New York, 1990.

[BFL91] L. Babai, L. Fortnow, and C. Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, **1**:1–40, 1991.

[BFNW93] L. Babai, L. Fortnow, N. Nisan, and A. Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, **3**:307–318, 1993.

[BFP03] H. Buhrman, L. Fortnow, and A. Pavan. Some results on derandomization. In *Proc. 20th Symposium on Theoretical Aspects of Computer Science*, Lecture Notes in Computer Science #2607, 212–222. Springer-Verlag, Berlin Heidelberg New York, 2003.

[BG81] C.H. Bennet and J. Gill. Relative to a random oracle A, P(A) $\neq$ NP(A) $\neq$ co-NP(A) with probability 1. *SIAM Journal on Computing*, **10**:69–113, 1981.

[BLS84] R. Book, T. Long, and A. L. Selman. Quanitative relativizations of complexity classes. *SIAM Journal on Computing*, **13**:461–487, 1984.

[Boo74] R. Book. Tally languages and complexity classes. *Information and Control*, **26**:186–193, 1974.

[BS92] J.L Balcázar and U. Schöning. Logarithmic advice classes. *Theoretical Computer Science*, **99**:279–290, 1992.

[CHW99] J. Cai, L. A. Hemaspaandra, and G. Wechsung. Robust reductions. *Theory of Computing Systems*, **32**(6):625–647, 1999.

[CPS99] J.-Y. Cai, A. Pavan, and D. Sivakumar. On the hardness of permanent. In *Proc. 16th Symposium on Theoretical Aspects of Computer Science*, Lecture Notes in Computer Science #1563, 90–99. Springer-Verlag, Berlin Heidelberg New York, 1999.

[Fei93] J. Feigenbaum. Locally random reductions in interactive complexity theory. In *Advances in Computational Complexity Theory*, DIMACS Series on Discrete Mathematics and Theoretical Computer Science #13, 73–98. American Mathematical Society, 1993.

[FF93] J. Feigenbaum and L. Fortnow. Random-self-reducibility of complete sets. *SIAM Journal on Computing*, **22**:994–1005, 1993.

[FKN90] J. Feigenbaum, S. Kannan, and N. Nisan. Lower bounds on random-self-reducibility. In *Proc. 5th Structure in Complexity Theory Conference*, 100–109. IEEE Computer Society Press, 1990.

[FL92] U. Feige and C. Lund. On the hardness of computing permanent of random matrices. In *Proc. 24th ACM Symposium on Theory of Computing*, 643–654. ACM Press, 1992.

[Gav95] R. Gavaldà. Bounding the complexity of advice functions. *Journal of Computer and System Sciences*, **50**(3):468–475, 1995.

[GKR+95] F. Green, J. Köbler, K. Regan, T. Schwentick, and J. Torán. The power of the middle bit of a #P function. *Journal of Computer and System Sciences*, **50**(3):456–467, 1995.

[GLR+91] P. Gemmell, R. J. Lipton, R. Rubinfeld, M. Sudan, and A. Wigderson. Self-testing/correcting for polynomials and for approximate functions. In *Proc. 23rd ACM Symposium on Theory of Computing*, 32–42. ACM Press, 1991.

[GS87] Y. Gurevich and S. Shelah. Expected computation time for Hamiltonian path problem. *SIAM Journal on Computing*, **16**:486–502, 1987.

[GS92] P. Gemmell and M. Sudan. Highly resilient correctors for polynomials. *Information Processing Letters*, **43**:169–174, 1992.

[Gur91] Y. Gurevich. Average case completeness. *Journal of Computer and System Sciences*, **42**(3):346–398, 1991.

[GZ97] O. Goldreich and D. Zuckerman. Another proof that BPP⊆PH (and more). Technical Report TR97-045, Electronic Colloquium on Computational Complexity, October 1997.

[Har83] J. Hartmanis. On sparse sets in NP–P. *Information Processing Letters*, **16**:55–60, 1983.

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

[Imp95] R. Impagliazzo. A personal view of average-case complexity. In *Proc. 10th Structure in Complexity Theory Conference*, 134–147. IEEE Computer Society Press, 1995.

[Kar97] C. Karg. LR(k) testing is average-case complete. In *Proc. 12th Annual IEEE Conference on Computational Complexity*, 74–80. IEEE Computer Society Press, 1997.

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

[Köb94] J. Köbler. Locating P/poly optimally in the extended low hierarchy. *Theoretical Computer Science*, **134**(2):263–285, 1994.

[KT96] J. Köbler and S. Toda. On the power of generalized MOD-classes. *Mathematical Systems Theory*, **29**(1):33–46, 1996.

[KW98] J. Köbler and O. Watanabe. New collapse consequences of NP having small circuits. *SIAM Journal on Computing*, **28**(1):311–324, 1998.

[Lau83] C. Lautemann. BPP and the polynomial hierarchy. *Information Processing Letters*, **17**:215–217, 1983.

[Lev86] L. Levin. Average case complete problems. *SIAM Journal on Computing*, **15**:285–286, 1986.

[Lip91] R. J. Lipton. New directions in testing. In J. Feigenbaum and M. Merritt, editors, *Distributed Computing and Cryptography*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science #2. American Mathematical Society, 1991.

[NW94] N. Nisan and A. Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, **49**:149–167, 1994.

[OKSW94] P. Orponen, K. Ko, U. Schöning, and O. Watanabe. Instance complexity. *Journal of the ACM*, **41**(1):96–121, 1994.

[Sch86] U. Schöning. *Complexity and Structure*, Lecture Notes in Computer Science #211. Springer-Verlag, Berlin Heidelberg New York, 1986.

[Sch95] R. Schuler. Some properties of sets tractable under every polynomial-time computable distribution. *Information Processing Letters*, **55**:179–184, 1995.

[Sch96] R. Schuler. Truth-table closure and Turing closure of average polynomial time have different measures in EXP. In *Proc. 11th Annual IEEE Conference on Computational Complexity*, 190–197. IEEE Computer Society Press, 1996.

[Sel94] A. L. Selman. A taxonomy of complexity classes of functions. *Journal of Computer and System Sciences*, 1994.

[Sip83] M. Sipser. A complexity theoretic approach to randomness. In *Proc. 15th ACM Symposium on Theory of Computing*, 330–335. ACM Press, 1983.

[SW95] R. Schuler and O. Watanabe. Towards average-case complexity analysis of NP optimization problems. In *Proc. 10th Structure in Complexity Theory Conference*, 148–159. IEEE Computer Society Press, 1995.

[SY95] R. Schuler and T. Yamakami. Sets computable in polynomial time on average. In *Proc. 1st International Computing and Combinatorics Conference*, Lecture Notes in Computer Science #959, 400–409. Springer-Verlag, Berlin Heidelberg New York, 1995.

[SY96] R. Schuler and T. Yamakami. Structural average case complexity. *Journal of Computer and System Sciences*, **52**:308–327, 1996.

[Tod91] S. Toda. PP is as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, **20**:865–877, 1991.

[Val79] L. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, **8**:189–201, 1979.

[Wan95] J. Wang. Average-case completeness of a word problem for groups. In *Proc. 27th ACM Symposium on Theory of Computing*, 325–334. ACM Press, 1995.

[Wat96] O. Watanabe, 1996. Personal communication.

[WB95] J. Wang and J. Belanger. On the NP-isomorphism problem with respect to random instances. *Journal of Computer and System Sciences*, **50**:151–164, 1995.

[Wil84] H. Wilf. Backtracking: An o(1) expected time algorithm for the graph coloring problem. *Information Processing Letters*, **18**:119–122, 1984.