Electronic Colloquium on Computational Complexity, Report No. 42 (2010)

Relativized Worlds Without Worst-Case to Average-Case Reductions for NP

Thomas Watson*

March 11, 2010

Abstract

We prove that relative to an oracle, there is no worst-case to errorless-average-case reduction for NP. This result is the first progress on an open problem posed by Impagliazzo in 1995, namely to construct an oracle relative to which NP is worst-case hard but errorless-average-case easy. We also handle classes that are somewhat larger than NP. In fact, we prove that relative to an oracle, there is no worst-case to errorless-average-case reduction from NP to $\mathrm{BPP}_{\mathrm{path}}$. The latter class contains $\mathrm{P}_{\|}^{\mathrm{NP}}$ and captures the power of randomized computations conditioned on efficiently testable events. We also handle reductions from NP to the polynomial-time hierarchy and beyond, under restrictions on the number of queries the reductions can make.

1 Introduction

The study of average-case complexity concerns the power of algorithms that are allowed to make mistakes on a small fraction of inputs. Of particular importance is the relationship between worst-case complexity and average-case complexity. For example, cryptographic applications require average-case hard problems, and it would be desirable to base the existence of such problems on minimal, worst-case complexity assumptions.

For the class PSPACE, it is known that worst-case hardness and average-case hardness are equivalent [3]. That is, if PSPACE is worst-case hard then it is also average-case hard. For the class NP, the situation is not well-understood. A central open problem in average-case complexity is to prove that if NP is worst-case hard then it is also average-case hard. There are some known barriers to proving this proposition. Bogdanov and Trevisan [6] considered the possibility of a *proof by reduction*. They showed that the proposition cannot be proven by a nonadaptive reduction unless the polynomial-time hierarchy collapses; it remains open to provide evidence against the existence of adaptive reductions. Impagliazzo and Rudich [13] considered the possibility of a *relativizing proof*. They showed that the proposition cannot be proven by relativizing techniques, by constructing a relativized heuristica, which is a world in which NP is worst-case hard but average-case easy.

The Bogdanov-Trevisan and Impagliazzo-Rudich results both concern average-case algorithms that may output the wrong answer on a small fraction of inputs. In light of these barriers, it is natural to consider the following proposition, which is potentially easier to prove: If NP is worst-case hard then it is also hard for *errorless* average-case algorithms, which may output "don't know"

*Computer Science Division, University of California, Berkeley. Supported by a National Science Foundation Graduate Research Fellowship.

ISSN 1433-8092

on a small fraction of inputs but must never output the wrong answer.[1] There are no previously known natural barriers to proving this proposition. In fact, strengthening the Impagliazzo-Rudich relativized heuristica to errorless average-case algorithms was explicitly posed as an open problem in 1995 by Impagliazzo [10]. We provide the first barrier, by ruling out the possibility of a *relativizing proof by reduction*. More precisely, we construct an oracle relative to which there is no reduction (even an adaptive one) that shows that if NP $\not\subseteq$ BPP then $\big(\mathrm{NP}, \mathrm{PS\textsc{amp}}\big) \not\subseteq \mathrm{AvgZPP}$, where $\big(\mathrm{NP}, \mathrm{PS\textsc{amp}}\big)$ is the class of distributional NP problems under polynomial-time samplable distributions, and AvgZPP is the class of distributional problems with polynomial-time errorless average-case randomized algorithms. We even rule out relativizing proofs by reduction that if NP is worst-case hard then certain classes larger than NP are errorless-average-case hard.

1.1 Results

Our first result concerns the class $\mathrm{BPP_{path}}$, which was introduced by Han et al. [9], who also showed that relative to every oracle, $\mathrm{P}^{\mathrm{NP}}_{\parallel} \subseteq \mathrm{BPP_{path}} \subseteq \mathrm{BPP}^{\mathrm{NP}}$.

Theorem 1. *There exists an oracle relative to which there is no reduction of type*

$$\big(\mathrm{BPP_{path}}, \mathrm{PS\textsc{amp}}\big) \subseteq \mathrm{AvgZPP} \;\Rightarrow\; \mathrm{UP} \subseteq \mathrm{BPP}.$$

Note that the type of reduction considered in Theorem 1 is weaker than a worst-case to errorless-average-case reduction for NP, because $\mathrm{BPP_{path}}$ is larger than NP, and UP is smaller than NP. Ruling out weaker reductions yields a stronger result.

We also prove a similar result for $\mathrm{BPP}^{\mathrm{NP}}_{\parallel,\, o(n/\log n)}$, which denotes the class $\mathrm{BPP}^{\mathrm{NP}}$ restricted to have $o(n/\log n)$ rounds of adaptivity in the NP oracle access but any number of queries within each round. In the current state of knowledge, $\mathrm{BPP}^{\mathrm{NP}}_{\parallel,\, o(n/\log n)}$ is incomparable to $\mathrm{BPP_{path}}$.

Theorem 2. *There exists an oracle relative to which there is no reduction of type*

$$\big(\mathrm{BPP}^{\mathrm{NP}}_{\parallel,\, o(n/\log n)}, \mathrm{PS\textsc{amp}}\big) \subseteq \mathrm{AvgZPP} \;\Rightarrow\; \mathrm{UP} \subseteq \mathrm{BPP}.$$

If we restrict our attention to reductions that use a limited number of queries, then we can handle classes even larger than $\mathrm{BPP_{path}}$ and $\mathrm{BPP}^{\mathrm{NP}}_{\parallel,\, o(n/\log n)}$.

Theorem 3. *For every polynomial q there exists an oracle relative to which there is no q-query reduction of type*

$$\big(\mathrm{PH}, \mathrm{PS\textsc{amp}}\big) \subseteq \mathrm{AvgZPP} \;\Rightarrow\; \mathrm{UP} \subseteq \mathrm{BPP}.$$

Since $\mathrm{BPP_{path}} \subseteq \mathrm{PH}$ and $\mathrm{BPP}^{\mathrm{NP}}_{\parallel,\, o(n/\log n)} \subseteq \mathrm{PH}$ relative to every oracle, it may appear at first glance that Theorem 3 subsumes Theorem 1 and Theorem 2. The reason it does not is because of the order of the quantifiers. In Theorem 3, the reduction may not make as many queries as it likes; it may only make a fixed polynomial q number of queries even though its running time may be an arbitrarily high degree polynomial.

If we are willing to sacrifice all but two queries, then we can go quite a bit further than PH.

[1] An equivalent notion of an errorless average-case algorithm is one that always outputs the correct answer but whose running time is only "polynomial-on-average" [15].

Theorem 4. *For every uniform complexity class of languages $\mathcal{C}$ there exists an oracle relative to which there is no 2-query reduction of type*

$$\big(\mathcal{C}, \text{PS}\textsc{amp}\big) \subseteq \text{AvgZPP} \;\Rightarrow\; \text{UP} \subseteq \text{BPP}.$$

The term "uniform complexity class of languages" has a somewhat technical meaning, which is explained in Section 2, but it encompasses all "ordinary" complexity classes such as PSPACE and EXP^{EXP}.

All of our results apply only to reductions that have oracle access to the input/output relationship of a hypothesized average-case algorithm, but do not have access to the code for such an algorithm. (Thus the reductions have access to two oracles: the reduction oracle and the relativization oracle.)

Our theorems can be generalized in various ways. For example, Theorem 1, Theorem 2, and Theorem 3 all hold with AvgZPP replaced by the deterministic version AvgP, by essentially the same proofs.[2] We have chosen to state the results using AvgZPP because we feel it is more natural to allow randomized algorithms in average-case complexity. As another example, Theorem 1 and Theorem 2 both hold with BPP replaced by BQP, by inserting a quantum query lower bound for the OR function [4] at the appropriate point in the arguments, instead of a randomized lower bound. We have chosen the particular statements of our four theorems so as to highlight the interesting aspects, avoid getting carried away with generalizations, and make the relationships among them clear.

In Section 2 we provide preliminaries, which clarify the precise meanings of our theorems. In Section 3 we give the intuition for our four theorems. In Section 4 we describe the basic setup that is common to the formal proofs of all four theorems. Section 5 contains the formal proof of Theorem 1. Section 6 contains the formal proof of Theorem 2. Section 7 contains the formal proof of Theorem 3. Section 8 contains the formal proof of Theorem 4. In Section 9 we conclude the paper with a list of open problems regarding oracles in average-case complexity.

1.2 Regarding Black-Box Reductions

Our notion of a reduction is more general than the notion of a "black-box reduction" that has been considered in the literature (for example, in [17, 18]). We now clarify this issue. Very roughly, a worst-case to average-case reduction from language L_1 to language L_2 is like an error-correcting code,[3] where the truth table of L_1 is the *information word*, the truth table of L_2 is the *code word*, the algorithm used to define L_2 is the *encoder*, and the reduction itself is the *decoder*. The standard notion of a black-box reduction requires that both the encoder is black-box, meaning that L_2 is defined as a function of L_1, and the decoder is black-box, meaning that it makes queries to the corrupted code word rather than having access to a succinct description of the corrupted code word (namely, the code for a hypothesized average-case algorithm for L_2). In the reductions we consider, the decoder is required to be black-box, but the encoder is not. Instead, our results are about *relativization*, which is more general than black-box encoding. The difference is somewhat subtle, but it boils down to the fact that in our relativized worlds, the encoder has access not to the truth table of L_1, but rather to the deterministic polynomial-time relation underlying the UP algorithm that solves L_1. The decoder also has access to this relation.[4]

[2] For Theorem 1 and Theorem 2, exactly the same proofs work. For Theorem 3, a minor tweak is needed.

[3] We exploit this connection in the proofs of Theorem 3 and Theorem 4.

[4] This is the culprit for the restriction on the number of queries in Theorem 3.

A natural goal is to extend our results to handle fully-non-black-box reductions, which are only guaranteed to work when given the code for an appropriate average-case algorithm. However, ruling out "proofs by fully-non-black-box reduction" turns out to be as hard as ruling out arbitrary proofs. For example, we claim that relative to every oracle, the following are equivalent.

(1) Strong relativized heuristica holds: $\big(\mathrm{NP}, \mathrm{PS}\textsc{amp}\big) \subseteq \mathrm{AvgZPP}$ and $\mathrm{NP} \not\subseteq \mathrm{BPP}$.

(2) There is no fully-non-black-box reduction of type

$$\big(\mathrm{NP}, \mathrm{PS}\textsc{amp}\big) \subseteq \mathrm{AvgZPP} \;\Rightarrow\; \mathrm{NP} \subseteq \mathrm{BPP}.$$

Clearly (1) implies (2). To see that (2) implies (1), consider two cases. If $\mathrm{NP} \subseteq \mathrm{BPP}$, then there is a trivial reduction that ignores the hypothesized AvgZPP algorithm for $\big(\mathrm{NP}, \mathrm{PS}\textsc{amp}\big)$. If $\big(\mathrm{NP}, \mathrm{PS}\textsc{amp}\big) \not\subseteq \mathrm{AvgZPP}$, then every algorithm is vacuously an appropriate fully-non-black-box reduction, because the universal quantification over AvgZPP algorithms for $\big(\mathrm{NP}, \mathrm{PS}\textsc{amp}\big)$ is over an empty set.

2 Preliminaries

We refer the reader to the textbooks [2, 8] for background on complexity theory and definitions of standard complexity classes. We refer the reader to the survey paper [5] for background on average-case complexity. In this section we provide preliminaries that are not completely standard.

2.1 Complexity Classes

For any randomized algorithm M, we let M_r denote M using internal randomness r.

Definition 1. $\mathrm{BPP}_{\mathrm{path}}$ *denotes the class of languages L such that for some polynomial-time randomized algorithm M that outputs two bits, and for all x,*

- $\Pr_r\big[M_r(x)_2 = 1\big] > 0$ *and*
- $\Pr_r\big[M_r(x)_1 = L(x) \;\big|\; M_r(x)_2 = 1\big] \geq 2/3$.

The above definition of $\mathrm{BPP}_{\mathrm{path}}$ is not the same as the original one given by Han et al. [9], but it is equivalent relative to every oracle, and it is more convenient for our purposes. Intuitively, $\mathrm{BPP}_{\mathrm{path}}$ captures the power of polynomial-time randomized computations after conditioning on efficiently testable events. The class could also be called PostBPP by analogy with the corresponding quantum class PostBQP [1].

Definition 2. $\mathrm{BPP}^{\mathrm{NP}}_{\|,\, o(n/\log n)}$ *denotes the class* $\mathrm{BPP}^{\mathrm{NP}}$ *restricted to have $o(n/\log n)$ rounds of adaptivity in the* NP *oracle access but any number of queries within each round.*

We now define the average-case complexity classes we need. Recall that in average-case complexity, we study distributional problems (L, D) where L is a language and $D = (D_1, D_2, \ldots)$ is an ensemble of probability distributions, where D_n is distributed over $\{0,1\}^n$. Recall that $\mathrm{PS}\textsc{amp}$ denotes the class of polynomial-time samplable ensembles, and $\mathcal{U}$ denotes the class consisting of only the uniform ensemble U. If $\mathcal{C}$ is a class of languages and $\mathcal{D}$ is a class of ensembles then $(\mathcal{C}, \mathcal{D}) = \big\{(L, D) \;:\; L \in \mathcal{C} \text{ and } D \in \mathcal{D}\big\}$.

Definition 3. HeurBPP *denotes the class of distributional problems* (L, D) *that have a polynomial-time heuristic scheme, that is, a randomized algorithm* M *that takes as input* x *and* $\delta > 0$*, runs in time polynomial in* $|x|$ *and* $1/\delta$*, and for all* n *and all* $\delta > 0$ *satisfies*

$$\Pr_{x \sim D_n, r} \big[M_r(x, \delta) \neq L(x)\big] \;\leq\; \delta.$$

Definition 4. AvgZPP *denotes the class of distributional problems* (L, D) *that have a polynomial-time errorless heuristic scheme, that is, a randomized algorithm* M *that takes as input* x *and* $\delta > 0$*, runs in time polynomial in* $|x|$ *and* $1/\delta$*, always outputs* $L(x)$ *or* $\perp$*, and for all* n *and all* $\delta > 0$ *satisfies*

$$\Pr_{x \sim D_n, r} \big[M_r(x, \delta) = \perp\big] \;\leq\; \delta.$$

2.2 Reductions

We consider reductions that have access to a *reduction oracle*. Suppose $\mathcal{C}_1$ is a class of computational problems and P_1 is a computational problem of the appropriate type (for us, language or distributional problem), and similarly suppose $\mathcal{C}_2$ is a class and P_2 is a problem of the appropriate type.

Definition 5. *We say a reduction* R *is of type*

$$P_2 \in \mathcal{C}_2 \;\Rightarrow\; P_1 \in \mathcal{C}_1$$

if for every reduction oracle that behaves like a $\mathcal{C}_2$ *algorithm for* P_2 *(minus the requirement that the behavior be efficiently computable),* R *behaves like a* $\mathcal{C}_1$ *algorithm for* P_1 *(including the efficiency constraint, when every call to the reduction oracle is charged the appropriate resources according to* $\mathcal{C}_2$*).*

Let us clarify Definition 5. The definition is satisfied by any reduction whose existence would prove that if $P_2 \in \mathcal{C}_2$ then $P_1 \in \mathcal{C}_1$. It must be the case that if any hypothesized $\mathcal{C}_2$ algorithm for P_2 is plugged into the reduction oracle, the reduction becomes a $\mathcal{C}_1$ algorithm for P_1. Importantly, the reduction oracle *does not* simply return answers for P_2; it behaves like a $\mathcal{C}_2$ algorithm. The definition of any class $\mathcal{C}_2$ has two aspects: the efficiency aspect and the behavior aspect. We take both of these into account. For example, if $\mathcal{C}_2 = \mathrm{BPTIME}(2^{n^\epsilon})$, then each query to the reduction oracle is charged time $O(2^{n^\epsilon})$ where n is the length of the query, and the reduction oracle only returns the correct answer with probability $\geq 2/3$.

Definition 6. *We say there exists a reduction of type*

$$\mathcal{C}_2' \subseteq \mathcal{C}_2 \;\Rightarrow\; \mathcal{C}_1' \subseteq \mathcal{C}_1$$

if for every $P_1 \in \mathcal{C}_1'$ *there exists a* $P_2 \in \mathcal{C}_2'$ *and a reduction of type*

$$P_2 \in \mathcal{C}_2 \;\Rightarrow\; P_1 \in \mathcal{C}_1.$$

We make a few remarks about Definition 6.

- When $\mathcal{C}_1'$ has an appropriately complete problem P_1, this is equivalent to saying there exists a $P_2 \in \mathcal{C}_2'$ and a reduction of the above type, for the fixed problem P_1.

- Note that we do not require that the reduction is uniform in the sense of there being a fixed algorithm R that computes the reduction for every $P_1 \in \mathcal{C}'_1$ given the code for a $\mathcal{C}'_1$-type algorithm for P_1.

- Note that when we say there is a reduction of the above type, this assertion gets weaker as $\mathcal{C}'_2$ and $\mathcal{C}_1$ get larger and $\mathcal{C}_2$ and $\mathcal{C}'_1$ get smaller.

2.3 Relativization

When we relativize to an oracle language A, every computation gets unrestricted oracle access to A. This includes samplers and reductions. Thus reductions have access to two oracles: the reduction oracle and the relativization oracle. When we write $R^{B,A}$ we mean B is the reduction oracle and A is the relativization oracle for reduction R.

To illustrate the formal framework set up so far, we give the precise statement of Theorem 2. There exists a language A and a language $L_1 \in \mathrm{UP}^A$ such that for all languages $L_2 \in \big(\mathrm{BPP}^{\mathrm{NP}}_{\parallel,\, o(n/\log n)}\big)^A$, all ensembles $D \in \mathrm{PS}\textsc{amp}^A$, and all polynomial-time randomized reductions $R^{\circ,\circ}$, $R^{\circ,A}$ is not of type

$$(L_2, D) \in \mathrm{AvgZPP}^A \;\Rightarrow\; L_1 \in \mathrm{BPP}^A.$$

The latter means that there exists an $x \in \{0,1\}^*$ and a randomized function $B : \{0,1\}^* \times \mathbb{R}_{>0} \to \{0,1,\bot\}$ which is a valid AvgZPP oracle for (L_2, D), such that

$$\Pr_{r,B}\Big[R_r^{B,A}(x) = L_1(x)\Big] \;<\; 2/3$$

where the probability is over both the internal randomness of R and the randomness of B (each query is answered with fresh independent randomness). When we say B is a valid AvgZPP oracle for (L_2, D) we mean that $B(w,\delta)$ always returns $L_2(w)$ or $\bot$, and for all n and all $\delta > 0$,

$$\Pr_{w\sim D_n, B}\big[B(w,\delta) = \bot\big] \;\le\; \delta.$$

When we say $R^{\circ,\circ}$ runs in polynomial time, this includes the fact that each query $B(w,\delta)$ to the reduction oracle is charged time polynomial in $|w|$ and $1/\delta$. In other words, δ must always be at least inverse polynomial. Throughout the paper we tacitly assume that "polynomial-time reductions" have this restriction, since $\mathcal{C}_2$ is always AvgZPP. We clarify that $D \in \mathrm{PS}\textsc{amp}^A$ means that for some randomized algorithm S°, $S^A(n)$ runs in time polynomial in n and outputs a sample distributed according to D_n. Finally, we clarify that $\big(\mathrm{BPP}^{\mathrm{NP}}_{\parallel,\, o(n/\log n)}\big)^A$ is the class of languages L_2 for which there exists a language $L_3 \in \mathrm{NP}^A$ and a polynomial-time randomized algorithm $M^{\circ,\circ}$ that only uses $o(n/\log n)$ rounds of adaptivity in its access to the first oracle, such that for all $x \in \{0,1\}^*$,

$$\Pr_r\Big[M_r^{L_3,A}(x) = L_2(x)\Big] \;\ge\; 2/3.$$

Regarding Theorem 3 and Theorem 4, there is one further issue to consider. For reductions that are allowed an unlimited number of queries (like in Theorem 1 and Theorem 2), the error probability of $1/3$ in the definition of BPP is unimportant since it can be amplified from $1/2 - 1/\mathrm{poly}(n)$ to $1/2^{\mathrm{poly}(n)}$. However, amplification increases the number of queries, so the error probability is not arbitrary for Theorem 3 and Theorem 4. For example, the existence of a q-query $\big(1/2-1/\operatorname{poly}(n)\big)$-error reduction of type

$$\big(\mathrm{PH}, \mathrm{PS{\small AMP}}\big) \subseteq \mathrm{AvgZPP} \;\Rightarrow\; \mathrm{UP} \subseteq \mathrm{BPP}$$

does not seem to imply the existence of a q-query 1/3-error reduction of the same type, but it still does imply that if $\big(\mathrm{PH}, \mathrm{PS{\small AMP}}\big) \subseteq \mathrm{AvgZPP}$ then $\mathrm{UP} \subseteq \mathrm{BPP}$. For this reason, we allow an error probability of $1/2-1/\operatorname{poly}(n)$ (for arbitrarily high degree polynomials) in Theorem 3 and Theorem 4.

2.4 Clean Reductions

We now precisely define the restriction on $\mathcal{C}$ in Theorem 4.

Definition 7. *We say that $\mathcal{C}$ is a* uniform complexity class of languages *if there is a countable collection of functions $\{M_1, M_2, \ldots\}$ mapping oracle languages A to languages M_i^A, such that the following conditions all hold.*

- *For every i and every x, $M_i^A(x)$ only depends on a finite number of bits of A.*
- *For every i and every x there exists a property $P_{i,x}(A)$ that only depends on the bits of A that $M_i^A(x)$ depends on, such that $\mathcal{C}^A = \big\{M_i^A \;:\; \forall x\; P_{i,x}(A)\big\}$.*
- *For every i and every linear-time computable function $f : \{0,1\}^* \to \{0,1\}^*$ there exists a j such that for all A the following two conditions hold: $M_j^A = M_i^A \circ f$, and if $M_i^A \in \mathcal{C}^A$ then $M_j^A \in \mathcal{C}^A$.*

The second condition says the class is defined by a property of the computation (for example, bounded error) holding for all inputs. The third condition says the class is closed under linear-time deterministic mapping reductions. Observe that $\mathrm{BPP_{path}}$, $\mathrm{BPP}^{\mathrm{NP}}_{\|,\, o(n/\log n)}$, PH, PSPACE, and $\mathrm{EXP^{EXP}}$ are all examples of uniform complexity classes under this definition.

The following complicated-looking lemma just says that in all four of our theorems, we can assume without loss of generality that on inputs of length n, any candidate reduction only queries the reduction oracle on inputs of length n^d and only with $\delta = 1/n^d$ for some positive integer d.

Lemma 1. *For every polynomial-time randomized reduction $R^{\circ,\circ}$ (where the reduction oracle is of the form $\{0,1\}^* \times \mathbb{R}_{>0} \to \{0,1,\bot\}$) there exists a polynomial-time randomized reduction $R^{\circ,\circ}_{clean}$ and a positive integer d such that the following holds. For every polynomial-time sampler S° there exists a polynomial-time sampler S°_{clean}, and for every uniform complexity class of languages $\mathcal{C}$ and every i there exists an i_{clean}, such that for every relativization oracle A, the following properties all hold.*

- *If $R^{\circ,A}$ is of type*

  $$\big(M_i^A, D^A\big) \in \mathrm{AvgZPP}^A \;\Rightarrow\; L \in \mathrm{BPP}^A$$

  for some language L, where D^A is the ensemble sampled by S^A, then $R^{\circ,A}_{clean}$ is of type

  $$\big(M^A_{i_{clean}}, D^A_{clean}\big) \in \mathrm{AvgZPP}^A \;\Rightarrow\; L \in \mathrm{BPP}^A$$

  where D^A_{clean} is the ensemble sampled by S^A_{clean}.

- *On inputs of length n, R_{clean} only queries the reduction oracle on inputs of length n^d and only with $\delta = 1/n^d$.*
- *R_{clean} always makes the same number of queries to the reduction oracle as R does.*
- *If $M_i^A \in \mathcal{C}^A$ then $M_{i_{clean}}^A \in \mathcal{C}^A$.*

Proof sketch. The basic idea is to take the answers to all the inputs to M_i^A up to the longest length R on inputs of length n could possibly query the reduction oracle, and put them in some larger input length n^d. Here d needs to be large enough that $1/n^d$ times the longest length R could query is less than the smallest value of δ that R could possibly query (which is at least inverse polynomial). The reason for multiplying by the longest length is that an error of $1/n^d$ in the AvgZPP oracle could get amplified by this amount when restricted to any particular input length that is stored "within" n^d. The index i_{clean} is just the j guaranteed by Definition 7 for index i and the mapping reduction we just informally described. ☐

3 Intuition

In Section 3.1 we describe the intuition behind the proofs of Theorem 1 and Theorem 2. Then in Section 3.2 we describe the intuition behind the proofs of Theorem 3 and Theorem 4.

3.1 Intuition for Theorem 1 and Theorem 2

We start by informally describing how to construct an oracle relative to which there is no reduction of type

$$\big(\text{NP}, \mathcal{U}\big) \subseteq \text{HeurBPP} \;\Rightarrow\; \text{UP} \subseteq \text{BPP}.$$

To obtain Theorem 1 and Theorem 2, we must strengthen HeurBPP to AvgZPP,[5] strengthen $\mathcal{U}$ to PSAMP, and strengthen NP to BPP_{path} and $\text{BPP}^{\text{NP}}_{\|,\, o(n/\log n)}$. We describe how to do this below. Handling larger classes than NP is the most technically interesting strengthening.

Fix an arbitrary NP-type algorithm M and an arbitrary polynomial-time randomized reduction R, and fix a sufficiently large n. We explain how to diagonalize against the pair M, R. For simplicity we assume that on inputs of length n, R only queries the reduction oracle on inputs of length n^d and only with $\delta = 1/n^d$ for some positive integer d; thus we can omit the δ. We consider relativization oracles of the form $A : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, which we think of as $2^n \times 2^n$ tables. Let $L_1^A : \{0,1\}^n \to \{0,1\}$ be defined by $L_1^A(x) = \bigvee_y A(xy)$. That is, L_1^A is the language of strings x such that there exists a 1 in the xth row of A. Let $L_2^A : \{0,1\}^{n^d} \to \{0,1\}$ denote the language computed by M^A. We only consider A, L_1^A, L_2^A at these input lengths since all other input lengths are irrelevant.

We wish to construct an A such that for some $x \in \{0,1\}^n$ and some deterministic[6] reduction oracle $B : \{0,1\}^{n^d} \to \{0,1\}$, B agrees with L_2^A on at least a $1 - 1/n^d$ fraction of inputs and $R^{B,A}(x)$ outputs $L_1^A(x)$ with probability $< 2/3$. This will show that R fails to be a reduction of type

$$\big(L_2^A, U\big) \in \text{HeurBPP}^A \;\Rightarrow\; L_1^A \in \text{BPP}^A.$$

[5]Usually AvgZPP is thought of as being a weaker class than HeurBPP (since AvgZPP $\subseteq$ HeurBPP), but it is stronger in our situation.

[6]B will be deterministic here even though randomness is allowed; this makes the result stronger.

We also need to ensure that there is at most one 1 in each row of A so that $L_1^A \in \mathrm{UP}^A$, but this will fall right out of the construction. We construct A through an iterative process, and we use a potential function argument to show that this process makes steady progress toward our goal. The process iteratively modifies the relativization oracle, and we use A to denote the relativization oracle throughout the whole process.[7] Thus the table denoted by A changes many times throughout our argument, and the languages L_1^A and L_2^A change accordingly. Initially A is all 0's.

Let us consider the computation of R on some input x. It is trying to figure out whether there is a 1 in the xth row of A, in other words, compute $L_1^A(x)$. It has two sources of information about $L_1^A(x)$: the relativization oracle A itself, and the reduction oracle B. If R did not have access to B, then we could diagonalize in a standard way: Observe how R behaves given that the xth row of A is all 0's. If R outputs 1 with high probability, then we are done. If R outputs 1 with low probability, then we find a bit in the xth row that R queries with only tiny probability and flip that bit (such a bit must exist because R does not have enough time to keep an eye on the entire row); then R still outputs 1 with low probability, but now $x \in L_1^A$. Thus R must rely on the reduction oracle B for help.

Our construction has two stages. The goal of stage 1 is to gain the upper hand by rendering B useless to R. Then in stage 2 we deliver the coup de grâce with the standard diagonalization argument. We cannot guarantee that B is useless for every x, but we only need it to be useless for some x. Specifically, suppose we could set up A in such a way that there exists an x such that

(1) the xth row of A is all 0's, and

(2) for all y, flipping $A(xy)$ would cause $L_2^A(w)$ to change for at most a $1/n^d$ fraction of w's.

Then declaring B to be L_2^A for the particular A we have set up, we know that we can leave A alone or we can flip any bit in the xth row, and for all these possibilities B is a valid HeurBPP oracle for the new L_2^A. Then we can observe the behavior of R on input x, using this fixed B for the reduction oracle, and diagonalize against R in the standard way with the assurance that whatever happens to A during this second stage, B will remain valid.

How do we set up A so that such an x exists? We do this iteratively. In each iteration, we find a certain x whose row is currently all 0's, which is our "best guess" for the good x. If condition (2) is satisfied for this x, then we are done. Otherwise, there is some column y that violates condition (2). Then we flip the bit $A(xy)$ to 1 and continue with the next iteration. We just need to show that there are $< 2^n$ iterations before we succeed. For this, we define a potential function Φ^A that assigns an energy value to A. The key is to show that if y violates condition (2) for our best guess x, then flipping $A(xy)$ must cause a significant decrease in potential. Since Φ^A must remain bounded, there cannot be too many iterations before M is beaten into submission and our best guess x works.

Let us hold off on the definition of Φ^A and focus on finding a best guess x. Our ultimate goal is to ensure that if we flip any bit in the xth row, most of the inputs to L_2^A "don't notice". There is an asymmetry between inputs that are accepted by M^A and those that are rejected. If $w \in \{0,1\}^{n^d}$ is such that $M^A(w)$ rejects, then if *any* of the exponentially many computation paths "notices" a change in A, the whole computation could become accepting. However, if $M^A(w)$ accepts, then

[7] More formally, we could say we define a sequence of relativization oracles $A_0, A_1, A_2, \dots$ that leads to some final version $A_k = A$. We omit the subscripts throughout the argument and simply refer to A with the understanding that this means the "current" version.

we can pick an arbitrary accepting computation path of $M^A(w)$ to be the "designated" one. Only polynomially many bits of A are queried by M on this path, and as long as none of these bits is flipped, w "won't notice" any change to A because $M^A(w)$ will still accept. In particular, there are only polynomially many x's such that $M^A(w)$ queries some bit in the xth row on the designated path. Thus for every w with $L_2^A(w) = 1$, the vast majority of x have the property that flipping any bit in the xth row does not cause $L_2^A(w)$ to change to 0. By an averaging argument, most x have the property that for most $w \in \{0,1\}^{n^d}$, flipping any bit in the xth row does not cause $L_2^A(w)$ to change from 1 to 0. For the current A, there must exist an x with the latter property and such that the xth row is all 0's, since (by induction) we know there are not very many x's with a 1 in their row currently. This is our best guess x.

We know that flipping any bit in the xth row causes only a small fraction of all $w \in \{0,1\}^{n^d}$ to change from 1 to 0 under L_2^A. This is good, but it is only half the story. We would also like that flipping any bit in the xth row causes only a small fraction of w's to change from 0 to 1. Suppose we budget a $1/2n^d$ fraction of w's to change from 1 to 0, and a $1/2n^d$ fraction to change from 0 to 1. Now if some y violates condition (2), then it must be the case that flipping $A(xy)$ causes at least a $1/2n^d$ fraction of w's to change from 0 to 1. We want to define the potential function so that having w's change from 0 to 1 under L_2^A causes a decrease in potential. A natural choice is

$$\Phi^A = \Pr_{w \sim U_{n^d}} \left[L_2^A(w) = 0 \right].$$

Flipping $A(xy)$ causes at least a $1/2n^d$ probability mass to leave the event $L_2^A(w) = 0$. However, as much as a $1/2n^d$ probability mass could enter the event due to w's that change from 1 to 0, which could essentially cancel out the drop in potential from the w's that changed from 0 to 1! The solution is to change our budgeting. If we budget a $1/3n^d$ fraction of w's to change from 1 to 0 and a $2/3n^d$ fraction to change from 0 to 1, then flipping $A(xy)$, where y violates condition (2), causes at least a $2/3n^d$ probability mass to leave the event, while at most a $1/3n^d$ probability mass enters the event. Thus Φ^A goes down by at least $1/3n^d$, and there are at most $3n^d < 2^n$ iterations before our best guess x works. This concludes the argument.

Very roughly, the big picture is as follows. For an input that is accepted by M^A, it is easy to ensure that the answer under L_2^A does not change when we make modifications to A. For an input that is rejected by M^A, we cannot ensure that the answer does not change, but the point is that if it does change, then *we can ensure that it does not change again*, since the input is now accepted.

3.1.1 Intuition for Strengthening HeurBPP to AvgZPP

Let x denote our best guess at the end of stage 1. Suppose we knew that there exists a set $W \subseteq \{0,1\}^{n^d}$ of density at most $1/n^d$ such that for all $w \notin W$ and all y, flipping $A(xy)$ does not change $L_2^A(w)$. Then setting

$$B(w) = \begin{cases} L_2^A(w) & \text{if } w \notin W \\ \perp & \text{if } w \in W \end{cases} \tag{1}$$

where A is the relativization oracle at the end of stage 1, we would have that B is a valid AvgZPP oracle for L_2^A no matter whether we leave A alone or flip any bit in the xth row. Then we could diagonalize in the standard way, by observing how R behaves on input x using this fixed B and the current A, and either leaving A alone or flipping some bit in the xth row to make R output the wrong answer with high probability.

The existence of such a W is too much to ask for. However, this is only because we were trying to find a B that would remain a valid AvgZPP oracle for *all* of the $2^n + 1$ diagonalization options. We do not really need all these options. Let Y be an arbitrary fixed set of columns of size $|Y| = 4t$, where t is the running time of R on inputs of length n. Then running R on input x with any fixed B and the current A, there must be a $y \in Y$ such that $A(xy)$ gets queried with probability $\leq 1/4$. If R outputs 1 with probability $\leq 1/3$ then after flipping this $A(xy)$, R outputs 1 with probability $< 2/3$ and hence errs. Thus it suffices to have $4t + 1$ diagonalization options, namely leaving A alone or flipping some $A(xy)$ with $y \in Y$. Suppose we knew that there exists a set $W \subseteq \{0,1\}^{n^d}$ of density at most $1/n^d$ such that for all $w \notin W$ and all $y \in Y$, flipping $A(xy)$ does not change $L_2^A(w)$. Then defining B as in Equation (1), we could diagonalize by either leaving A alone or flipping $A(xy)$ for some $y \in Y$ with the assurance that whatever happens, B will remain valid.

Now the existence of such a W is *not* too much to ask for. Using the argument for the HeurBPP case with a small adjustment of parameters, we can ensure that flipping any bit in the xth row causes $L_2^A(w)$ to change for at most a $1/4tn^d$ fraction of w's. Then we can take W to be the set of all w such that there exists a $y \in Y$ such that flipping $A(xy)$ changes $L_2^A(w)$.

3.1.2 Intuition for Strengthening $\mathcal{U}$ to PSAMP

There are two approaches: one that is direct, and one that uses a result of Impagliazzo and Levin [11]. Neither is difficult. We first describe the direct approach.

First, observe that if U_{n^d} were replaced by some other distribution on $\{0,1\}^{n^d}$ that is independent of A, then the whole argument above would carry through, just by replacing "fraction of w's" with "probability mass of w's" under this distribution. Now in addition to M and R, we need to worry about an arbitrary polynomial-time sampler S, and we need to ensure that B is a valid AvgZPP oracle for $\left(L_2^A, D^A\right)$, where D^A denotes the distribution sampled by $S^A(n^d)$. If S did not query A at all, then D^A would be independent of A and thus we could use the same argument, by the above observation. Two issues arise because S is allowed to query A. First, when we flip a bit during stage 1, this affects

$$\Phi^A = \Pr_{w \sim D^A}\left[L_2^A(w) = 0\right]$$

in terms of not only the event but also the distribution. Second, when we flip a bit during stage 2, this affects the distributional problem $\left(L_2^A, D^A\right)$ for which B needs to be a valid AvgZPP oracle, in terms of not only the language but also the distribution.

Handling these issues is just a matter of tweaking the argument to ensure that our modifications to A cause only small statistical deviations in D^A. Specifically, consider the beginning of an iteration of stage 1, and let D denote D^A for the current A (thus D is fixed and will not react to changes in A). Now suppose we choose our best guess x as before, but based on this distribution D. Then by the above argument we know that for every y, flipping $A(xy)$ would either cause

$$\Pr_{w \sim D}\left[L_2^A(w) = 0\right]$$

to go down by a significant amount, or cause $L_2^A(w)$ to change with only small probability over $w \sim D$. It can be shown that this is good enough for our purpose provided that for all y, flipping $A(xy)$ results in a D^A that is statistically very close to D. To ensure the latter, we choose our best guess x not only so that the xth row is all 0's and flipping any bit in the xth row only causes a small probability mass of $w \sim D$ to change from 1 to 0 under L_2^A, but also so that the probability

$S^A(n^d)$ queries any bit in the xth row is small. This is possible because the vast majority of x's satisfy the latter condition since S runs in polynomial time.

An alternative approach to handling PSAMP uses a result due to Impagliazzo and Levin [11]. They proved that if $\mathcal{C}$ is a class of languages containing NP and satisfying certain simple closure properties, then relative to every oracle, there exists a reduction of type

$$\big(\mathcal{C}, \text{PSAMP}\big) \subseteq \text{AvgZPP} \;\Rightarrow\; \big(\mathcal{C}, \mathcal{U}\big) \subseteq \text{AvgZPP}.$$

The proof of this result appears in Section 5.2 of [5] and is based on a result of Impagliazzo and Luby on distributionally inverting one-way functions [12]. By composing this reduction with the hypothesized reduction, we can assume without loss of generality that the distributional problem we are reducing to uses the uniform ensemble. In the formal proofs of Theorem 1 and Theorem 2, rather than use the Impagliazzo-Levin result we opt to directly handle the samplable ensembles because doing so makes the arguments self-contained at only a slight cost in complicatedness.

3.1.3 Intuition for Strengthening NP to BPP_{path}

Let us revert from PSAMP to $\mathcal{U}$. For both Theorem 1 and Theorem 2, the differences from the above proof are in the definition of the potential function Φ^A, the choice of our best guess x, and the argument that if some y violates condition (2) for our best guess x, then flipping $A(xy)$ causes a significant decrease in potential.

For Theorem 1, instead of an NP-type algorithm we have a BPP_{path}-type algorithm M. Let us hold off on how to define Φ^A and how to choose our best guess x. Consider an arbitrary iteration of stage 1, let A denote the current relativization oracle, and suppose we have somehow picked a certain x such that the xth row of A is all 0's. Suppose there is a y such that flipping $A(xy)$ causes $L_2^A(w)$ to change for a significant fraction of w's. We want it to be the case that flipping $A(xy)$ also causes a significant decrease in potential. Let A' denote A with $A(xy)$ flipped to 1.

Consider a w such that $L_2^{A'}(w) \neq L_2^A(w)$. Let us make the bold assumption that for all choices of M's internal randomness r such that $M_r^A(w)_2 = 1$, we have $M_r^{A'}(w) = M_r^A(w)$ (that is, both output bits match). Then by the definition of BPP_{path} we have

$$\begin{aligned}
\Pr_r\Big[M_r^{A'}(w)_2 = 1\Big] &\geq 3 \cdot \Pr_r\Big[M_r^{A'}(w)_1 = L_2^A(w) \text{ and } M_r^{A'}(w)_2 = 1\Big] \\
&\geq 3 \cdot \Pr_r\Big[M_r^{A}(w)_1 = L_2^A(w) \text{ and } M_r^{A}(w)_2 = 1\Big] \\
&\geq 3 \cdot \left(\Pr_r\Big[M_r^A(w)_2 = 1\Big] \cdot 2/3\right) \\
&= 2 \cdot \Pr_r\Big[M_r^A(w)_2 = 1\Big]
\end{aligned}$$

where the second line follows because the event in the second line is a subset of the event on the right side of the first line. In other words, switching from A to A' forces the conditioning event to at least double in size, in order to reduce the probability of outputting $L_2^A(w)$ in the first bit (conditioned on that event) from $\geq 2/3$ to $\leq 1/3$. Thus

$$-\log_2 \Pr_r\Big[M_r^{A'}(w)_2 = 1\Big] \;\leq\; -\log_2 \Pr_r\Big[M_r^{A}(w)_2 = 1\Big] - 1.$$

This suggests using

$$\Phi^A = \mathop{\mathrm{E}}_{w \in \{0,1\}^{n^d}} \left[-\log_2 \Pr_r \left[M_r^A(w)_2 = 1 \right] \right]$$

where w is chosen uniformly at random, because then when we flip $A(xy)$, a significant fraction of w's each contribute a significant negative amount to the potential difference $\Phi^{A'} - \Phi^A$. There are three issues.

(1) We need to make sure the potential is not too large to begin with.

(2) We made an unjustified assumption about the behavior of M.

(3) We also need to make sure that the contribution of bad w's to the potential difference does not cancel out the negative contribution of good w's.

Issue (1) is not problematic: Since we may assume r is chosen uniformly from $\{0,1\}^{\mathrm{poly}(n)}$, for every w and every A we must have

$$\Pr_r \left[M_r^A(w)_2 = 1 \right] \geq 2^{-\mathrm{poly}(n)}$$

since otherwise the conditioning event would be empty and M^A would fail to define a language in $\mathrm{BPP}_{\mathrm{path}}^A$ (for the violating A), which would suffice to diagonalize against the pair M, R.

For issue (2), first note that if we relax our assumption to be that for *almost all* r such that $M_r^A(w)_2 = 1$, we have $M_r^{A'}(w) = M_r^A(w)$, then flipping $A(xy)$ still causes the probability of the conditioning event to go up by at least a constant factor (say 3/2) assuming $L_2^{A'}(w) \neq L_2^A(w)$. Now we use our ability to choose x. Since M runs in polynomial time, it can be shown that most x are *useful*, in the sense that for the vast majority of w's it is the case that for almost all r such that $M_r^A(w)_2 = 1$, $M_r^A(w)$ does not query any bit in the xth row. Thus we can pick our best guess x so that x is useful and the xth row of A is all 0's. Then for our fixed x and y, we know that the vast majority of w's have the property that for almost all r such that $M_r^A(w)_2 = 1$, we have $M_r^{A'}(w) = M_r^A(w)$. Call the remaining w's *horrible*. Call w *good* if $L_2^{A'}(w) \neq L_2^A(w)$ and w is not horrible. Call w *bad* if it is not good. By a union bound we know that a significant fraction of w's are good, and each good w contributes a significant negative amount to the potential difference $\Phi^{A'} - \Phi^A$.

Finally we consider issue (3). We consider the horrible w's and the bad-but-not-horrible w's separately. The contribution of each horrible w to $\Phi^{A'} - \Phi^A$ could be as large as $\mathrm{poly}(n)$ (inside the expectation), but only a tiny fraction of w's are horrible so this only puts a small dent in the negative contribution from the good w's. Almost all of the w's could be bad-but-not-horrible, but the contribution of each such w to $\Phi^{A'} - \Phi^A$ can be at most a tiny positive amount, since of the r's with $M_r^A(w)_2 = 1$, almost all of them are such that $M_r^{A'}(w)_2 = M_r^A(w)_2 = 1$. Thus the bad-but-not-horrible w's only put a small dent in the negative contribution from the good w's.

3.1.4 Intuition for Strengthening NP to $\mathrm{BPP}_{\|,\, o(n/\log n)}^{\mathrm{NP}}$

Again, we consider $\mathcal{U}$ instead of PSAMP. Now instead of a single algorithm we have a pair M, N where N is an NP-type algorithm and M is a polynomial-time randomized algorithm that uses $o(n/\log n)$ rounds of adaptivity in its access to the first oracle. We let L_3^A denote the language computed by N^A, and we let L_2^A denote the language computed by $M^{L_3^A, A}$ (assuming bounded error is satisfied for every input).[8] Again, suppose we have somehow picked our best guess x, such that the xth row of the current A is all 0's, and suppose there is a y such that flipping $A(xy)$ causes $L_2^A(w)$ to change for a significant fraction of w's. We want it to be the case that flipping $A(xy)$ also causes a significant decrease in potential. Let A' denote A with $A(xy)$ flipped to 1.

We make the simplifying assumption that M has oracle access only to L_3^A and not to A. Extending the argument to the general case is not difficult; it just involves taking an extra precaution when picking our best guess x to ensure that hardly any w's "notice" the change from A to A' via the second oracle.

For each w such that $L_2^{A'}(w) \neq L_2^A(w)$, it must be the case that

$$M_r^{L_3^{A'}}(w) \;\neq\; M_r^{L_3^A}(w) \tag{2}$$

for at least 1/3 of the r's. Thus we know that Inequality (2) holds for a significant fraction of pairs w, r. Let $M_r^{L_3^A}(w)_{i,j} \in \{0,1\}^*$ denote the jth query within the ith round of adaptivity of $M_r^{L_3^A}(w)$. We wish to define Φ^A in terms of the bits

$$L_3^A\Big(M_r^{L_3^A}(w)_{i,j}\Big)$$

over the choice of w, r, i, j. We compare these bits with the corresponding bits when A is replaced by A'. Very roughly, the intuition is similar to the NP case described at the beginning of Section 3.1: We would like that hardly any of the bits go from 1 to 0 (since the bits that are 1 under A should be "stable" if we choose x appropriately) while a significant fraction go from 0 to 1 (due to Inequality (2) holding for a significant fraction of pairs w, r). Thus it is tempting to define Φ^A to be the fraction of w, r, i, j whose bit is 0. The problem with this intuition is the adaptivity: If the w, r, i^*, j^* bit is different under A and A', then for all $i > i^*$ and all j we could have $M_r^{L_3^{A'}}(w)_{i,j} \neq M_r^{L_3^A}(w)_{i,j}$ in which case the values of the w, r, i, j bit under A and A' have nothing to do with each other. In particular, if the w, r, i^*, j^* bit changes then for all $i > i^*$ and all j, the w, r, i, j bit could go from 1 to 0, thus undoing all the "stability" we thought we had accrued. The solution is that in the potential function, we weight the bits inverse exponentially in i, so that even if this bad scenario happens, the absolute value of the contribution of w, r, i^*, j^* to the potential difference $\Phi^{A'} - \Phi^A$ swamps the absolute value of the total contribution of w, r, i, j over all $i > i^*$ and all j.

Let us be a bit more precise with this intuition. For an arbitrary pair w, r, let i^* be the smallest value (if it exists) such that for some j, the w, r, i^*, j bit changes when we switch from A to A' (note that i^* depends on w, r). Then the bits w, r, i, j for all $i < i^*$ and all j have 0 contribution to the potential difference, and the bits w, r, i, j for all $i > i^*$ and all j have negligible total contribution compared to the contribution of w, r, i^*, j for any j. Thus we just need to consider the bits of the form w, r, i^*, j. Analogously to the intuition for Theorem 1, we consider three types of pairs w, r.

Call w, r *horrible* if for some j, the w, r, i^*, j bit changes from 1 to 0. The contribution of each horrible pair to the potential difference may be a large positive amount (the worst case is when $i^* = 1$), but the overall contribution of horrible pairs will be tiny provided only a tiny fraction of pairs are horrible. We ensure the latter by picking our best guess x appropriately, using the "stability" of accepting nondeterministic computations, and using the fact that the computations

[8] We again only deal with L_2^A on inputs of length n^d, but we consider L_3^A on all input lengths. We could assume all queries M makes to its first oracle have the same length, but it turns out this would not make the proof any simpler.

$M_r^{L_3^{A'}}(w)$ and $M_r^{L_3^A}(w)$ proceed identically up through the i^*th round (which allows us to just look at the strings $M_r^{L_3^A}(w)_{i,j}$ and ensure that most of them are not in $L_3^A \backslash L_3^{A'}$).

Call w, r *good* if w, r is not horrible but i^* does exist (and thus for some j, the w, r, i^*, j bit changes from 0 to 1). Call w, r *bad* if it is not good. Whenever w, r is not horrible and Inequality (2) holds, w, r must be good since there must be *some* bit w, r, i, j that changes when we switch from A to A'. By a union bound we know that a significant fraction of pairs are good. Thus the contribution of a good pair w, r to the potential difference is negative, and the weight of the contribution is inverse exponential in i^*, which is significant since $i^* \leq o(n/\log n)$.[9] Thus the overall contribution of good pairs is a significant negative amount.

Finally, consider the *bad-but-not-horrible* pairs w, r. For these, i^* must not exist, and thus there is 0 contribution to the potential difference.

Overall we get a significant drop in potential, as desired.

3.2 Intuition for Theorem 3 and Theorem 4

It is well-known that error-correcting codes can be used to construct worst-case to average-case reductions, at least for large complexity classes such as PSPACE [3, 16]. To be applicable, the codes must have very efficient encoders (since this dictates the complexity of the language being reduced to) and very efficient decoders (since this dictates the complexity of the reduction itself). Our strategy for proving Theorem 3 and Theorem 4 is to set up the relativization oracle in such a way that error-correcting codes are in some sense the *only* way to construct worst-case to average-case reductions of the appropriate types, and then argue that the efficiency of the resulting encoders and decoders is too good to be true. That is, we would like to be able to extract a good error-correcting code from any purported reduction and then apply known lower bounds on the efficiency of encoders and decoders for such codes. For Theorem 3, we use a result due to Viola [18] which states that good error-correcting codes[10] cannot be encoded by small constant-depth circuits. For Theorem 4, we use a lower bound due to Kerenidis and de Wolf [14] on the length of 2-query locally decodable codes.

Our approach for Theorem 3 and Theorem 4 is in some sense a *dual* approach to the one we used for Theorem 1 and Theorem 2. As before, we have a reduction R that is trying to solve a problem with the aid of a relativization oracle A and a reduction oracle B. Before, our goal was to *render B useless to R* so we could focus on how R interacted with A. Now, our goal is to *render A useless to R* so we can focus on how R interacts with B. Before, we found a good *row* of A and filled in that row adversarially. Now, we find a good *column* of A and fill in that column adversarially.

Unlike in the proofs of Theorem 1 and Theorem 2, we cannot use the Impagliazzo-Levin result to reduce PSamp to $\mathcal{U}$ since it uses too many queries. But again, directly handling the samplable ensembles presents no major difficulties. Thus, for the rest of this section we assume PSamp is replaced by $\mathcal{U}$.

The basic setup is the same as before. We have an algorithm M (PH-type for Theorem 3 or arbitrary complexity for Theorem 4). We have a polynomial-time randomized reduction R that uses a limited number of queries to the reduction oracle. For simplicity we assume that on inputs of length n, R only queries the reduction oracle on inputs of length n^d and only with $\delta = 1/n^d$ for some

[9]The $\log n$ comes from the polynomially many queries in each round. Theorem 2 also holds if we allow $o(n)$ *queries* rather than $o(n/\log n)$ *rounds of adaptivity*.

[10]His result even applies to list-decodable codes, but we do not need this stronger result.

positive integer d. We construct a sequence of relativization oracles $A : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, and we define $L_1^A : \{0,1\}^n \to \{0,1\}$ by $L_1^A(x) = \bigvee_y A(xy)$, and we let $L_2^A : \{0,1\}^{n^d} \to \{0,1\}$ denote the language computed by M^A. For the final version of A, we want $R^{B,A}(x)$ to output $L_1^A(x)$ with probability $< 1/2 + 1/n^{\log n}$ for some $x \in \{0,1\}^n$ and some $B : \{0,1\}^{n^d} \to \{0,1,\perp\}$ that agrees with L_2^A on at least a $1 - 1/n^d$ fraction of inputs and returns $\perp$ on the rest. We have $1/2 + 1/n^{\log n}$ instead of 2/3 for the reason discussed at the end of Section 2.3.

Let us start by pretending that R never queries A. Then it is completely straightforward to extract a good binary error-correcting code from M, R: Pick an arbitrary column y and define

$$C : \{0,1\}^{2^n} \to \{0,1\}^{2^{n^d}}$$

by viewing the input as a function $Z : \{0,1\}^n \to \{0,1\}$ and the output as a function $C(Z) : \{0,1\}^{n^d} \to \{0,1\}$ given by $C(Z) = L_2^{A_Z}$ where A_Z denotes the relativization oracle with Z as the yth column and 0's everywhere else. If R really is of the hypothesized type no matter which Z we use, then it immediately follows that R is a decoder that recovers any bit $Z(x) = L_1^{A_Z}(x)$ of the information word from any corrupted code word B that has at most a $1/n^d$ fraction of erasures (and no flipped bits).

For Theorem 3, note that C has relative minimum distance $> 1/n^d$ and each bit of C is encodable by a small constant-depth circuit (since M is a PH-type algorithm with oracle access to Z). This contradicts a result of Viola [18] which says that such a code cannot exist. Thus there must be some Z for which R is not of the hypothesized type.

For Theorem 4, note that C is a 2-query locally decodable code in the sense that each bit of the information word can be recovered with probability at least $1/2 + 1/n^{\log n}$ assuming there are at most a $1/n^d$ fraction of erasures.[11] Since the code word length is only *quasipolynomial* in the information word length, this contradicts a result of Kerenidis and de Wolf [14] which says that the length of such a code must be *nearly exponential*.[12] Thus there must be some Z for which R is not of the hypothesized type. Since the lower bound holds regardless of the complexity of encoding, we can handle *any* uniform complexity class of languages.

Now we return to the "real world" where R may query A. Then the above argument, with an arbitrary fixed y, does not work because R might know y in which case R can easily go look up the answers to L_1^A in the yth column. We must choose y so as to "hide" the answers from R. Restricting the number of queries R can make to B is essential for this: If R can make n queries then M can easily let R know what y is by explicitly writing y over and over again in the truth table L_2^A, and R would have no trouble retrieving this information from any B that has sufficient agreement with L_2^A. (Of course in Theorem 3, R *can* use n, or any fixed polynomial, number of queries. But this is easily remedied by just adding $2^{\text{poly}(n)}$ columns to the table A, with a high enough degree polynomial, so that we *can* hide the answers from R. Henceforth we assume R only uses $n^{o(1)}$ queries, so that we can stick with 2^n columns.)

Suppose we could choose y so that for every x and every $B : \{0,1,\}^{n^d} \to \{0,1,\perp\}$, the probability that $R^{B,0}(x)$ (where 0 denotes the all 0's relativization oracle) queries a bit in the yth column is at most $1/2n^{\log n}$. Then we would know that for every Z, every x, and every B that is valid

[11] Usually, locally decodable codes are defined in terms of flipped bits rather than erasures, but they are equivalent up to small differences in parameters.

[12] The lower bound is only *nearly* exponential since the relative minimum distance and the advantage over 1/2 in correct decoding probability are subconstant in our case.

for L_2^{Az}, the probability $R^{B,0}(x)$ outputs $L_1^{Az}(x)$ is within $1/2n^{\log n}$ of the probability $R^{B,Az}(x)$ outputs $L_1^{Az}(x)$ and is hence at least $1/2+1/2n^{\log n}$. This would suffice for a contradiction, because we could use $R^{B,0}$ for the decoder. Actually this property of y is more than we really need. If we replace "every x" with "most x" then we could just remove the bad x's from consideration, at a small loss in the information word length, and we would still get a contradiction. Now to find such a y, we use the fact that quantifying over all B is the same as quantifying over all paths of adaptivity in R's access to B, and there are a limited number of such paths. Specifically, for every x and every r there are only a small number of columns of the relativization oracle that get queried by $R_r^{\circ,0}(x)$ over all possible reduction oracles (namely, at most the running time of R times 3 to the number of reduction oracle queries). By an averaging argument, there is some y such that for most x's, all but a $1/2n^{\log n}$ fraction of r's are such that $R_r^{B,0}(x)$ does not query any bit in the yth column, for any B. This is good enough for our purpose.

The bottom line is that there are basically only two ways M could help R solve L_1^A: by telling R the answers, or by telling R where to find the answers in A. The former is impossible because then we would have an error-correcting code that is too good to be true, and the latter is impossible because R cannot make enough queries to B to retrieve the identity of y.

4 Generic Setup for the Formal Proofs

We now describe the basic setup that is common to the proofs of all four theorems. However, this setup will need to be customized a bit for each of the four proofs.

We have a uniform complexity class of languages $\mathcal{C}$ with enumeration $\{M_1, M_2, \ldots\}$. Consider an arbitrary triple i, S, R where $i \in \mathbb{N}$, S is a polynomial-time sampler, and R is a polynomial-time randomized reduction. Using Lemma 1 we can assume without loss of generality that on inputs of length n, R only queries the reduction oracle on inputs of length n^d and only with $\delta = 1/n^d$ for some positive integer d. For an arbitrary relativization oracle $A \subseteq \{0,1\}^*$ we make the following definitions. Let L_1^A denote the NP^A language defined by

$$L_1^A \;=\; \big\{x \;:\; \exists y \text{ such that } |y| = |x| \text{ and } xy \in A\big\}.$$

If M_i^A defines a language in $\mathcal{C}^A$ then let L_2^A denote this language.[13] Let D^A denote the $\mathrm{PS{\small AMP}}^A$ ensemble defined by S^A.

We wish to construct a relativization oracle A^* so that $L_1^{A^*} \in \mathrm{UP}^{A^*}$ (by ensuring that in the definition of $L_1^{A^*}$, y is always unique if it exists) and so that for all i, S, R, either $M_i^{A^*}$ fails to define a language in $\mathcal{C}^{A^*}$, or otherwise

$$\Pr_{r_R, B}\Big[R_{r_R}^{B,A^*}(x) = L_1^{A^*}(x)\Big] \;<\; 2/3$$

for some $x \in \{0,1\}^*$ and some randomized function $B : \{0,1\}^* \times \mathbb{R}_{>0} \to \{0,1,\bot\}$ which is a valid AvgZPP oracle for $\big(L_2^{A^*}, D^{A^*}\big)$, thereby ensuring that the reduction R°,A^*} fails to be of type

$$\big(L_2^{A^*}, D^{A^*}\big) \subseteq \mathrm{AvgZPP}^{A^*} \;\Rightarrow\; L_1^{A^*} \subseteq \mathrm{BPP}^{A^*}.$$

We construct a sequence of relativization oracles by starting with $\emptyset$ and adding strings and never taking them back out. We take A^* to be the limit of this sequence. Throughout the proofs,

[13] Technically M_i^A *equals* the language L_2^A according to Definition 7, but the notation L_2^A is more convenient for the proofs.

we simply refer to the "current" A with the understanding that this is the set of strings that have been included so far. We diagonalize against each triple i, S, R in sequence. After each round of diagonalization, we have the requirement that A^* matches the current A up through a certain input length, and we know that the current A contains no strings longer than that length. Now consider an arbitrary round, and suppose i, S, R is the triple to diagonalize against.

If there exists an A' consistent with the requirements of previous rounds and such that $M_i^{A'}$ fails to define a language in $\mathcal{C}^{A'}$, say with x as the violating input, then we update A to match A' up through the largest input length $M_i^{A'}(x)$ can query, and we require that A^* matches the new A up through this input length. This ensures that $M_i^{A^*}$ fails to define a language in $\mathcal{C}^{A^*}$, and we can move on to the next round.

Otherwise, we know that whatever we do to A, L_2^A will always be defined. Choose n large enough so that the following three things hold.

- The relativization oracle is fresh for all input lengths $\geq n$.
- The asymptotic constraints throughout the arguments are satisfied.
- The "relevant computations" all run in time $n^{\log n}$ without a big O.

The "relevant computations" include S on input n^d, R on inputs of length n, and (depending on the theorem) possibly the underlying computations of M_i on inputs of length n^d. We construct A at input length $2n$ to ensure that at the end of this round,

$$\Pr_{r_R, B}\left[R_{r_R}^{B,A}(x) = L_1^A(x)\right] \;<\; 2/3$$

for some $x \in \{0,1\}^n$ and some randomized function $B : \{0,1\}^{n^d} \to \{0,1,\bot\}$ which is a valid AvgZPP oracle for $\left(L_2^A, D^A\right)$ at input length n^d with respect to $\delta = 1/n^d$. Note that it makes sense to run $R^{B,A}(x)$ since this computation only queries B on inputs of length n^d and only with $\delta = 1/n^d$ (so we are justified in omitting the δ). This suffices to diagonalize against i, S, R because we can require that A^* matches the new A up through input length $n^{\log n}$ and up through the longest input length M_i can query on inputs of length n^d, thus ensuring the following three things.

- $L_1^{A^*}(x) = L_1^A(x)$.
- $R^{B,A^*}(x)$ behaves the same as $R^{B,A}(x)$.
- $L_2^{A^*}|_{n^d} = L_2^A|_{n^d}$ and $D_{n^d}^{A^*} = D_{n^d}^A$, which implies that B is a valid AvgZPP oracle for $\left(L_2^{A^*}, D^{A^*}\right)$ at input length n^d with respect to $\delta = 1/n^d$ and can thus be extended to a full valid AvgZPP oracle for $\left(L_2^{A^*}, D^{A^*}\right)$ without changing the behavior of $R^{B,A^*}(x)$.

5 Proof of Theorem 1

We use the setup from Section 4, customized as follows. We have $\mathcal{C} = \mathrm{BPP_{path}}$, and M_i corresponds to a $\mathrm{BPP_{path}}$-type algorithm M. Also, M on inputs of length n^d counts as "relevant computations" and thus runs in time $n^{\log n}$ without a big O.

5.1 Main Construction

Recall that M, S, R, n are fixed. For all relativization oracles A (not just the one we have constructed so far) we define the potential

$$\Phi^A \;=\; \mathop{\mathrm{E}}_{r_S}\Big[-\log_2 \Pr_{r_M}\Big[M^A_{r_M}\big(S^A_{r_S}(n^d)\big)_2 = 1 \Big] \Big].$$

The construction has two stages.

Stage 1. This stage proceeds in iterations. For a given iteration, let A denote the current relativization oracle after the previous iteration. If there exist $x \in \{0,1\}^n$ and $y \in \{0,1\}^n$ such that $x \notin L_1^A$ and $\Phi^{A\cup\{xy\}} \le \Phi^A - 1/n^{3\log n}$ then update $A := A \cup \{xy\}$ and continue with the next iteration. Otherwise, halt stage 1 and proceed to stage 2.

The following lemma is the technical heart of the proof of Theorem 1. We first finish the proof of Theorem 1 assuming the lemma, and then we prove the lemma in Section 5.2.

Lemma 2. *At the end of stage 1, there exists an $x \in \{0,1\}^n$ such that $x \notin L_1^A$ and for all $y \in \{0,1\}^n$,*

$$\Pr_{r_S}\Big[L_2^{A\cup\{xy\}}\big(S^A_{r_S}(n^d)\big) \neq L_2^A\big(S^A_{r_S}(n^d)\big) \Big] \;\le\; 1/8n^{d+\log n} \tag{3}$$

and

$$\Pr_{r_S}\Big[S^{A\cup\{xy\}}_{r_S}(n^d) \neq S^A_{r_S}(n^d) \Big] \;\le\; 1/2n^d. \tag{4}$$

Stage 2. Let A denote the current relativization oracle at the end of stage 1, and let x be as guaranteed by Lemma 2. Let $Y \subseteq \{0,1\}^n$ be an arbitrary set of size $4n^{\log n}$. Define a deterministic reduction oracle $B : \{0,1\}^{n^d} \to \{0,1,\bot\}$ by

$$B(w) \;=\; \begin{cases} L_2^A(w) & \text{if } L_2^{A\cup\{xy\}}(w) = L_2^A(w) \text{ for all } y \in Y \\ \bot & \text{otherwise} \end{cases}.$$

There are two cases.

Case 1. If

$$\Pr_{r_R}\Big[R^{B,A}_{r_R}(x) = 1 \Big] \;>\; 1/3$$

then we will use A for the relativization oracle at the beginning of the next round of diagonalization, without changing it. Since $x \notin L_1^A$, we have

$$\Pr_{r_R}\Big[R^{B,A}_{r_R}(x) = L_1^A(x) \Big] \;<\; 2/3.$$

We just need to verify that B is a valid AvgZPP oracle for $\big(L_2^A, D^A\big)$ at input length n^d with respect to $\delta = 1/n^d$. Obviously, $B(w)$ always returns $L_2^A(w)$ or $\bot$, by our definition of B. We have

$$\Pr_{w\sim D^A_{n^d}}\big[B(w) = \bot \big] \;=\; \Pr_{r_S}\Big[B\big(S^A_{r_S}(n^d)\big) = \bot \Big]$$

$$
\begin{aligned}
&= \Pr_{r_S}\Big[\exists y \in Y \text{ such that } L_2^{A\cup\{xy\}}\big(S_{r_S}^A(n^d)\big) \neq L_2^A\big(S_{r_S}^A(n^d)\big)\Big] \\
&\le \sum_{y\in Y} \Pr_{r_S}\Big[L_2^{A\cup\{xy\}}\big(S_{r_S}^A(n^d)\big) \neq L_2^A\big(S_{r_S}^A(n^d)\big)\Big] \\
&\le \sum_{y\in Y} 1/8n^{d+\log n} \\
&= |Y| \cdot 1/8n^{d+\log n} \\
&= 1/2n^d \\
&\le 1/n^d = \delta
\end{aligned}
$$

where the fourth line follows by Lemma 2. Thus we have succeeded in diagonalizing against M, S, R as described at the end of Section 4.

Case 2. If

$$
\Pr_{r_R}\Big[R_{r_R}^{B,A}(x) = 1\Big] \;\le\; 1/3
$$

then for each $y \in Y$ we define

$$
\pi_y \;=\; \Pr_{r_R}\Big[R_{r_R}^{B,A}(x) \text{ queries } A(xy)\Big].
$$

Since $R^{B,A}(x)$ runs in time $n^{\log n}$, we have $\sum_{y\in Y} \pi_y \le n^{\log n}$. Thus there exists a $y \in Y$ such that $\pi_y \le n^{\log n}/|Y| = 1/4$. Fix this y. We will update the relativization oracle to be $A \cup \{xy\}$ for the end of this round of diagonalization. Since $x \in L_1^{A\cup\{xy\}}$, we have

$$
\begin{aligned}
\Pr_{r_R}\Big[R_{r_R}^{B,A\cup\{xy\}}(x) = L_1^{A\cup\{xy\}}(x)\Big] &\le \Pr_{r_R}\big[R_{r_R}^{B,A}(x) = 1 \text{ or } R_{r_R}^{B,A\cup\{xy\}}(x) \neq R_{r_R}^{B,A}(x)\big] \\
&\le \Pr_{r_R}\big[R_{r_R}^{B,A}(x) = 1 \text{ or } R_{r_R}^{B,A}(x) \text{ queries } A(xy)\big] \\
&\le \Pr_{r_R}\Big[R_{r_R}^{B,A}(x) = 1\Big] + \pi_y \\
&\le 1/3 + 1/4 \\
&< 2/3.
\end{aligned}
$$

We just need to verify that B is a valid AvgZPP oracle for $\big(L_2^{A\cup\{xy\}}, D^{A\cup\{xy\}}\big)$ at input length n^d with respect to $\delta = 1/n^d$. Since $y \in Y$, we have that for all w, if $B(w) \neq \bot$ then $B(w) = L_2^A(w) = L_2^{A\cup\{xy\}}(w)$, by our definition of B. We also have

$$
\begin{aligned}
\Pr_{w\sim D_{n^d}^{A\cup\{xy\}}}\big[B(w) = \bot\big] &= \Pr_{r_S}\Big[B\big(S_{r_S}^{A\cup\{xy\}}(n^d)\big) = \bot\Big] \\
&\le \Pr_{r_S}\Big[B\big(S_{r_S}^A(n^d)\big) = \bot \text{ or } S_{r_S}^{A\cup\{xy\}}(n^d) \neq S_{r_S}^A(n^d)\Big] \\
&\le \Pr_{r_S}\Big[B\big(S_{r_S}^A(n^d)\big) = \bot\Big] + \Pr_{r_S}\Big[S_{r_S}^{A\cup\{xy\}}(n^d) \neq S_{r_S}^A(n^d)\Big] \\
&\le 1/2n^d + 1/2n^d \\
&= 1/n^d = \delta
\end{aligned}
$$

where the fourth line follows by the calculation from case 1 and by Lemma 2. Thus we have succeeded in diagonalizing against M, S, R as described at the end of Section 4.

5.2 Proof of Lemma 2

For all A (not just the one we have constructed so far) and all r_S, let us define

$$\Phi^A_{r_S} \;=\; -\log_2 \Pr_{r_M}\left[M^A_{r_M}\left(S^A_{r_S}(n^d)\right)_2 = 1\right]$$

so that $\Phi^A = \mathrm{E}_{r_S}\left[\Phi^A_{r_S}\right]$. For all A consistent with the requirements of previous rounds, the following holds. For all $w \in \{0,1\}^{n^d}$, since we are assured that

$$\Pr_{r_M}\left[M^A_{r_M}(w)_2 = 1\right] \;>\; 0$$

and since $M^A(w)$ runs in time $n^{\log n}$, we have

$$\Pr_{r_M}\left[M^A_{r_M}(w)_2 = 1\right] \;\geq\; 2^{-n^{\log n}}.$$

Therefore $0 \leq \Phi^A_{r_S} \leq n^{\log n}$ for all r_S, and hence $0 \leq \Phi^A \leq n^{\log n}$.

From here on out, A denotes the current relativization oracle at the end of stage 1. Since there are at most $n^{4\log n}$ iterations before stage 1 terminates, we have

$$\Pr_{x\in\{0,1\}^n}\left[x \in L^A_1\right] \;\leq\; n^{4\log n}/2^n$$

where x is chosen uniformly at random. For $x \in \{0,1\}^n$ define

$$p_x \;=\; \mathop{\mathrm{E}}_{r_S}\left[\Pr_{r_M}\left[\exists y \in \{0,1\}^n \text{ such that } M^A_{r_M}\left(S^A_{r_S}(n^d)\right) \text{ queries } A(xy) \;\middle|\; M^A_{r_M}\left(S^A_{r_S}(n^d)\right)_2 = 1\right]\right].$$

Recall that the conditioning is valid since we are assured that

$$\Pr_{r_M}\left[M^A_{r_M}(w)_2 = 1\right] \;>\; 0$$

for all $w \in \{0,1\}^{n^d}$. Since $M^A(w)$ runs in time $n^{\log n}$, we have $\sum_x p_x \leq n^{\log n}$ and thus

$$\Pr_{x\in\{0,1\}^n}\left[p_x > 1/n^{7\log n}\right] \;<\; n^{8\log n}/2^n.$$

For $x \in \{0,1\}^n$ define

$$s_x \;=\; \Pr_{r_S}\left[\exists y \in \{0,1\}^n \text{ such that } S^A_{r_S}(n^d) \text{ queries } A(xy)\right].$$

Since $S^A(n^d)$ runs in time $n^{\log n}$, we have $\sum_x s_x \leq n^{\log n}$ and thus

$$\Pr_{x\in\{0,1\}^n}\left[s_x > 1/n^{4\log n}\right] \;<\; n^{5\log n}/2^n.$$

By a union bound we find that

$$\Pr_{x\in\{0,1\}^n}\left[x \notin L^A_1 \text{ and } p_x \leq 1/n^{7\log n} \text{ and } s_x \leq 1/n^{4\log n}\right]$$

$$
\begin{aligned}
&> 1 - \left(n^{4\log n}/2^n\right) - \left(n^{8\log n}/2^n\right) - \left(n^{5\log n}/2^n\right) \\
&> 0.
\end{aligned}
$$

Thus there exists an $x \in \{0,1\}^n$ such that $x \notin L_1^A$ and $p_x \le 1/n^{7\log n}$ and $s_x \le 1/n^{4\log n}$. Fix this x. We claim that this x satisfies the condition of Lemma 2. Suppose for contradiction that there exists a $y \in \{0,1\}^n$ such that either Inequality (3) does not hold or Inequality (4) does not hold. Fix this y. We claim that $\Phi^{A\cup\{xy\}} \le \Phi^A - 1/n^{3\log n}$, thus contradicting the fact that stage 1 halted. Henceforth we let A' denote $A \cup \{xy\}$. We partition the sample space of S's internal randomness into four events.

$$
\begin{aligned}
E_1 &= \left\{ r_S \;:\; S_{r_S}^{A'}(n^d) \neq S_{r_S}^{A}(n^d) \right\} \\
E_2 &= \Big\{ r_S \;:\; r_S \notin E_1 \text{ and} \\
&\qquad \Pr_{r_M}\left[M_{r_M}^{A'}\left(S_{r_S}^A(n^d)\right) \neq M_{r_M}^{A}\left(S_{r_S}^A(n^d)\right) \;\middle|\; M_{r_M}^{A}\left(S_{r_S}^A(n^d)\right)_2 = 1 \right] > 1/n^{3\log n} \Big\} \\
E_3 &= \left\{ r_S \;:\; r_S \notin E_1 \cup E_2 \text{ and } L_2^{A'}\left(S_{r_S}^A(n^d)\right) \neq L_2^{A}\left(S_{r_S}^A(n^d)\right) \right\} \\
E_4 &= \left\{ r_S \;:\; r_S \notin E_1 \cup E_2 \cup E_3 \right\}
\end{aligned}
$$

For E_2, note that $M_{r_M}^{A'}\left(S_{r_S}^A(n^d)\right) \neq M_{r_M}^{A}\left(S_{r_S}^A(n^d)\right)$ means that at least one of the two output bits is different.

Proposition 1. $\Pr_{r_S}\left[r_S \in E_1\right] \le 1/n^{4\log n}$ *and for all* $r_S \in E_1$, $\Phi_{r_S}^{A'} - \Phi_{r_S}^{A} \le n^{\log n}$.

Proposition 2. $\Pr_{r_S}\left[r_S \in E_2\right] \le 1/n^{4\log n}$ *and for all* $r_S \in E_2$, $\Phi_{r_S}^{A'} - \Phi_{r_S}^{A} \le n^{\log n}$.

Proposition 3. $\Pr_{r_S}\left[r_S \in E_3\right] \ge 1/n^{2\log n}$ *and for all* $r_S \in E_3$, $\Phi_{r_S}^{A'} - \Phi_{r_S}^{A} \le -1/2$.

Proposition 4. $\Pr_{r_S}\left[r_S \in E_4\right] \le 1$ *and for all* $r_S \in E_4$, $\Phi_{r_S}^{A'} - \Phi_{r_S}^{A} \le 2/n^{3\log n}$.

From these four propositions it follows that

$$
\begin{aligned}
\Phi^{A'} - \Phi^A &= \mathop{\mathrm{E}}_{r_S}\left[\Phi_{r_S}^{A'} - \Phi_{r_S}^{A}\right] \\
&= \mathop{\mathrm{E}}_{r_S}\left[\Phi_{r_S}^{A'} - \Phi_{r_S}^{A} \;\middle|\; r_S \in E_1\right] \cdot \Pr_{r_S}\left[r_S \in E_1\right] + \\
&\quad \mathop{\mathrm{E}}_{r_S}\left[\Phi_{r_S}^{A'} - \Phi_{r_S}^{A} \;\middle|\; r_S \in E_2\right] \cdot \Pr_{r_S}\left[r_S \in E_2\right] + \\
&\quad \mathop{\mathrm{E}}_{r_S}\left[\Phi_{r_S}^{A'} - \Phi_{r_S}^{A} \;\middle|\; r_S \in E_3\right] \cdot \Pr_{r_S}\left[r_S \in E_3\right] + \\
&\quad \mathop{\mathrm{E}}_{r_S}\left[\Phi_{r_S}^{A'} - \Phi_{r_S}^{A} \;\middle|\; r_S \in E_4\right] \cdot \Pr_{r_S}\left[r_S \in E_4\right] \\
&\le 1/n^{3\log n} + 1/n^{3\log n} - 1/2n^{2\log n} + 2/n^{3\log n} \\
&\le -1/n^{3\log n}
\end{aligned}
$$

which is what we wanted to show.

Proof of Proposition 1. The first assertion follows because

$$\begin{aligned}\Pr_{r_S}\left[r_S \in E_1\right] &\leq \Pr_{r_S}\left[S^A_{r_S}(n^d) \text{ queries } A(xy)\right] \\ &\leq s_x \\ &\leq 1/n^{4\log n}.\end{aligned}$$

The second assertion follows trivially from the fact that $\Phi^{A'}_{r_S} \leq n^{\log n}$ and $\Phi^A_{r_S} \geq 0$. □

Proof of Proposition 2. The first assertion follows because

$$\begin{aligned}\Pr_{r_S}\left[r_S \in E_2\right] &\leq \Pr_{r_S}\left[\Pr_{r_M}\left[M^{A'}_{r_M}\left(S^A_{r_S}(n^d)\right) \neq M^A_{r_M}\left(S^A_{r_S}(n^d)\right) \,\middle|\, M^A_{r_M}\left(S^A_{r_S}(n^d)\right)_2 = 1\right] > 1/n^{3\log n}\right] \\ &\leq \Pr_{r_S}\left[\Pr_{r_M}\left[M^A_{r_M}\left(S^A_{r_S}(n^d)\right) \text{ queries } A(xy) \,\middle|\, M^A_{r_M}\left(S^A_{r_S}(n^d)\right)_2 = 1\right] > 1/n^{3\log n}\right] \\ &\leq \mathop{\mathrm{E}}_{r_S}\left[\Pr_{r_M}\left[M^A_{r_M}\left(S^A_{r_S}(n^d)\right) \text{ queries } A(xy) \,\middle|\, M^A_{r_M}\left(S^A_{r_S}(n^d)\right)_2 = 1\right]\right] \cdot n^{3\log n} \\ &\leq p_x \cdot n^{3\log n} \\ &\leq 1/n^{4\log n}.\end{aligned}$$

The second assertion follows trivially from the fact that $\Phi^{A'}_{r_S} \leq n^{\log n}$ and $\Phi^A_{r_S} \geq 0$. □

Proof of Proposition 3. This proposition is in some sense the crux of the whole proof. Since $1/n^{4\log n} \leq 1/2n^d$, Proposition 1 implies that Inequality (4) holds and therefore Inequality (3) does not hold. The first assertion follows because

$$\begin{aligned}\Pr_{r_S}\left[r_S \in E_3\right] &\geq \Pr_{r_S}\left[L^{A'}_2\left(S^A_{r_S}(n^d)\right) \neq L^A_2\left(S^A_{r_S}(n^d)\right)\right] - \Pr_{r_S}\left[r_S \in E_1\right] - \Pr_{r_S}\left[r_S \in E_2\right] \\ &> 1/8n^{d+\log n} - 1/n^{4\log n} - 1/n^{4\log n} \\ &\geq 1/n^{2\log n}\end{aligned}$$

where the first line follows by a union bound and the second line follows by the negation of Inequality (3) and by Proposition 1 and Proposition 2.

We now argue the second assertion. Since $r_S \notin E_1$, we have $S^{A'}_{r_S}(n^d) = S^A_{r_S}(n^d)$. Let w denote this string. Then we have

$$\begin{aligned}&\Pr_{r_M}\left[M^{A'}_{r_M}\left(S^{A'}_{r_S}(n^d)\right)_2 = 1\right]/3 \\ =\ &\Pr_{r_M}\left[M^{A'}_{r_M}(w)_2 = 1\right]/3 \\ \geq\ &\Pr_{r_M}\left[M^{A'}_{r_M}(w)_1 \neq L^{A'}_2(w) \text{ and } M^{A'}_{r_M}(w)_2 = 1\right] \\ =\ &\Pr_{r_M}\left[M^{A'}_{r_M}(w)_1 = L^A_2(w) \text{ and } M^{A'}_{r_M}(w)_2 = 1\right] \\ \geq\ &\Pr_{r_M}\left[M^A_{r_M}(w)_1 = L^A_2(w) \text{ and } M^A_{r_M}(w)_2 = 1 \text{ and } M^{A'}_{r_M}(w) = M^A_{r_M}(w)\right] \\ \geq\ &\Pr_{r_M}\left[M^A_{r_M}(w)_1 = L^A_2(w) \text{ and } M^A_{r_M}(w)_2 = 1\right] - \Pr_{r_M}\left[M^{A'}_{r_M}(w) \neq M^A_{r_M}(w) \text{ and } M^A_{r_M}(w)_2 = 1\right]\end{aligned}$$

$$
\begin{aligned}
&= \left(\Pr_{r_M}\left[M^A_{r_M}(w)_1 = L^A_2(w) \;\middle|\; M^A_{r_M}(w)_2 = 1\right] - \Pr_{r_M}\left[M^{A'}_{r_M}(w) \neq M^A_{r_M}(w) \;\middle|\; M^A_{r_M}(w)_2 = 1\right]\right)\cdot \\
&\quad \Pr_{r_M}\left[M^A_{r_M}(w)_2 = 1\right] \\
&\geq \left(2/3 - 1/n^{3\log n}\right)\cdot \Pr_{r_M}\left[M^A_{r_M}(w)_2 = 1\right] \\
&\geq \Pr_{r_M}\left[M^A_{r_M}(w)_2 = 1\right]/2 \\
&= \Pr_{r_M}\left[M^A_{r_M}\big(S^A_{r_S}(n^d)\big)_2 = 1\right]/2
\end{aligned}
$$

where the third line follows by the fact that

$$\Pr_{r_M}\left[M^{A'}_{r_M}(w)_1 \neq L^{A'}_2(w) \;\middle|\; M^{A'}_{r_M}(w)_2 = 1\right] \;\leq\; 1/3$$

by Definition 1, the fourth line follows by the fact that $L^{A'}_2(w) \neq L^A_2(w)$, and the third-from-last line follows by Definition 1 and because $r_S \notin E_1 \cup E_2$. The second assertion now follows because $\log_2(3/2) \geq 1/2$. □

Proof of Proposition 4. The first assertion is trivial. We now argue the second assertion. Since $r_S \notin E_1$, we have $S^{A'}_{r_S}(n^d) = S^A_{r_S}(n^d)$. Let w denote this string. Then we have

$$
\begin{aligned}
\Pr_{r_M}\left[M^{A'}_{r_M}\big(S^{A'}_{r_S}(n^d)\big)_2 = 1\right] &= \Pr_{r_M}\left[M^{A'}_{r_M}(w)_2 = 1\right] \\
&\geq \Pr_{r_M}\left[M^A_{r_M}(w)_2 = 1 \text{ and } M^{A'}_{r_M}(w) = M^A_{r_M}(w)\right] \\
&= \left(1 - \Pr_{r_M}\left[M^{A'}_{r_M}(w) \neq M^A_{r_M}(w) \;\middle|\; M^A_{r_M}(w)_2 = 1\right]\right)\cdot \Pr_{r_M}\left[M^A_{r_M}(w)_2 = 1\right] \\
&\geq \left(1 - 1/n^{3\log n}\right)\cdot \Pr_{r_M}\left[M^A_{r_M}(w)_2 = 1\right] \\
&\geq 2^{-2/n^{3\log n}}\cdot \Pr_{r_M}\left[M^A_{r_M}(w)_2 = 1\right] \\
&= 2^{-2/n^{3\log n}}\cdot \Pr_{r_M}\left[M^A_{r_M}\big(S^A_{r_S}(n^d)\big)_2 = 1\right]
\end{aligned}
$$

where the fourth line follows because $r_S \notin E_1 \cup E_2$. The second assertion follows. □

6 Proof of Theorem 2

We use the setup from Section 4, customized as follows. We have $\mathcal{C} = \mathrm{BPP}^{\mathrm{NP}}_{\|,\, o(n/\log n)}$, and M_i corresponds to a pair M, N where M is a $\mathrm{BPP}^{\circ}_{\|,\, o(n/\log n)}$-type algorithm and N is an NP-type algorithm. Thus L^A_2 is the $\big(\mathrm{BPP}^{\mathrm{NP}}_{\|,\, o(n/\log n)}\big)^A$ language computed by $M^{L^A_3, A}$ where L^A_3 denotes the NP^A language computed by N^A. Also, M on inputs of length n^d, as well as N on all inputs that could be queried by M on inputs of length n^d, count as "relevant computations" and thus all run in time $n^{\log n}$ without a big O.

Assume without loss of generality that for some nonnegative integer e, M on inputs of length n^d always makes exactly n^e queries to its first oracle within each round of adaptivity and always has the same number of rounds of adaptivity. Let $M^{L^A_3, A}_{r_M}(w)_{i,j} \in \{0,1\}^*$ denote the jth query made within the ith round of adaptivity.

6.1 Main Construction

Recall that M, N, S, R, n are fixed. For all relativization oracles A (not just the one we have constructed so far) we define the potential

$$\Phi^A \;=\; \mathop{\mathrm{E}}_{r_S, r_M}\left[\sum_{i,j}(3n^e)^{-i}\Bigl(1 - L_3^A\Bigl(M_{r_M}^{L_3^A,A}\bigl(S_{r_S}^A(n^d)\bigr)_{i,j}\Bigr)\Bigr)\right].$$

The construction is identical to the construction from the proof of Theorem 1 except that we require the potential to go down by at least $1/2^{n/2}$ in each iteration of stage 1.

The following lemma is the technical heart of the proof of Theorem 2. The statement is identical to the statement of Lemma 2 but it refers to the new construction.

Lemma 3. *At the end of stage 1, there exists an $x \in \{0,1\}^n$ such that $x \notin L_1^A$ and for all $y \in \{0,1\}^n$,*

$$\Pr_{r_S}\left[L_2^{A\cup\{xy\}}\bigl(S_{r_S}^A(n^d)\bigr) \neq L_2^A\bigl(S_{r_S}^A(n^d)\bigr)\right] \;\leq\; 1/8n^{d+\log n} \tag{5}$$

and

$$\Pr_{r_S}\left[S_{r_S}^{A\cup\{xy\}}(n^d) \neq S_{r_S}^A(n^d)\right] \;\leq\; 1/2n^d. \tag{6}$$

6.2 Proof of Lemma 3

For all A (not just the one we have constructed so far) and all r_S, r_M, i, j, let us define

$$\Phi^A_{r_S,r_M,i,j} \;=\; (3n^e)^{-i}\Bigl(1 - L_3^A\Bigl(M_{r_M}^{L_3^A,A}\bigl(S_{r_S}^A(n^d)\bigr)_{i,j}\Bigr)\Bigr)$$

and

$$\Phi^A_{r_S,r_M} \;=\; \sum_{i,j}\Phi^A_{r_S,r_M,i,j}$$

so that $\Phi^A = \mathrm{E}_{r_S,r_M}\left[\Phi^A_{r_S,r_M}\right]$. Since $n^e\sum_{i=1}^{\infty}(3n^e)^{-i} \leq 1$, we have $0 \leq \Phi^A_{r_S,r_M} \leq 1$ for all r_S, r_M, and hence $0 \leq \Phi^A \leq 1$.

From here on out, A denotes the current relativization oracle at the end of stage 1. Since there are at most $2^{n/2}$ iterations before stage 1 terminates, we have

$$\Pr_{x\in\{0,1\}^n}\left[x \in L_1^A\right] \;\leq\; 1/2^{n/2}$$

where x is chosen uniformly at random. For $x \in \{0,1\}^n$ define

$$p_x \;=\; \Pr_{r_S,r_M}\left[\exists y \in \{0,1\}^n \text{ such that } M_{r_M}^{L_3^A,A}\bigl(S_{r_S}^A(n^d)\bigr) \text{ queries } A(xy)\right].$$

Since $M^{L_3^A,A}(w)$ runs in time $n^{\log n}$ for all $w \in \{0,1\}^{n^d}$, we have $\sum_x p_x \leq n^{\log n}$ and thus

$$\Pr_{x\in\{0,1\}^n}\left[p_x > 1/2^{n/2}\right] \;<\; n^{\log n}/2^{n/2}.$$

For every $v \in L_3^A$ pick an arbitrary accepting computation path of $N^A(v)$ to be the "designated" path. For $x \in \{0,1\}^n$ define

$$q_x = \Pr_{r_S, r_M, i, j}\Big[M_{r_M}^{L_3^A, A}\big(S_{r_S}^A(n^d)\big)_{i,j} \in L_3^A \text{ and } \exists y \in \{0,1\}^n \text{ such that } N^A\Big(M_{r_M}^{L_3^A, A}\big(S_{r_S}^A(n^d)\big)_{i,j}\Big) \text{ queries } A(xy) \text{ on the designated path}\Big]$$

where i, j are chosen uniformly at random. Since $N^A(v)$ runs in time $n^{\log n}$ for every v of interest, we have $\sum_x q_x \le n^{\log n}$ and thus

$$\Pr_{x \in \{0,1\}^n}\big[q_x > 1/2^{n/2}\big] < n^{\log n}/2^{n/2}.$$

For $x \in \{0,1\}^n$ define

$$s_x = \Pr_{r_S}\Big[\exists y \in \{0,1\}^n \text{ such that } S_{r_S}^A(n^d) \text{ queries } A(xy)\Big].$$

Since $S^A(n^d)$ runs in time $n^{\log n}$, we have $\sum_x s_x \le n^{\log n}$ and thus

$$\Pr_{x \in \{0,1\}^n}\big[s_x > 1/2^{n/2}\big] < n^{\log n}/2^{n/2}.$$

By a union bound we find that

$$\begin{aligned}
&\Pr_{x \in \{0,1\}^n}\Big[x \notin L_1^A \text{ and } p_x \le 1/2^{n/2} \text{ and } q_x \le 1/2^{n/2} \text{ and } s_x \le 1/2^{n/2}\Big] \\
&> 1 - \big(1/2^{n/2}\big) - \big(n^{\log n}/2^{n/2}\big) - \big(n^{\log n}/2^{n/2}\big) - \big(n^{\log n}/2^{n/2}\big) \\
&> 0.
\end{aligned}$$

Thus there exists an $x \in \{0,1\}^n$ such that $x \notin L_1^A$ and $p_x \le 1/2^{n/2}$ and $q_x \le 1/2^{n/2}$ and $s_x \le 1/2^{n/2}$. Fix this x. We claim that this x satisfies the condition of Lemma 3. Suppose for contradiction that there exists a $y \in \{0,1\}^n$ such that either Inequality (5) does not hold or Inequality (6) does not hold. Fix this y. We claim that $\Phi^{A \cup \{xy\}} \le \Phi^A - 1/2^{n/2}$, thus contradicting the fact that stage 1 halted. Henceforth we let A' denote $A \cup \{xy\}$. We partition the joint sample space of S's internal randomness and M's internal randomness into five events.

$$\begin{aligned}
E_1 &= \Big\{(r_S, r_M) \;:\; S_{r_S}^{A'}(n^d) \ne S_{r_S}^A(n^d)\Big\} \\
E_2 &= \Big\{(r_S, r_M) \;:\; (r_S, r_M) \notin E_1 \text{ and } M_{r_M}^{L_3^A, A}\big(S_{r_S}^A(n^d)\big) \text{ queries } A(xy)\Big\} \\
E_3 &= \Big\{(r_S, r_M) \;:\; (r_S, r_M) \notin E_1 \cup E_2 \text{ and } \exists i, j \text{ such that } M_{r_M}^{L_3^A, A}\big(S_{r_S}^A(n^d)\big)_{i,j} \in L_3^A \backslash L_3^{A'}\Big\} \\
E_4 &= \Big\{(r_S, r_M) \;:\; (r_S, r_M) \notin E_1 \cup E_2 \cup E_3 \text{ and } \exists i, j \text{ such that } M_{r_M}^{L_3^A, A}\big(S_{r_S}^A(n^d)\big)_{i,j} \in L_3^{A'} \backslash L_3^A\Big\} \\
E_5 &= \Big\{(r_S, r_M) \;:\; (r_S, r_M) \notin E_1 \cup E_2 \cup E_3 \cup E_4\Big\}
\end{aligned}$$

Proposition 5. $\Pr_{r_S, r_M}\big[(r_S, r_M) \in E_1\big] \le 1/2^{n/2}$ *and for all* $(r_S, r_M) \in E_1$, $\Phi_{r_S, r_M}^{A'} - \Phi_{r_S, r_M}^A \le 1$.

Proposition 6. $\Pr_{r_S, r_M}\big[(r_S, r_M) \in E_2\big] \le 1/2^{n/2}$ *and for all* $(r_S, r_M) \in E_2$, $\Phi_{r_S, r_M}^{A'} - \Phi_{r_S, r_M}^A \le 1$.

Proposition 7. $\Pr_{r_S,r_M}\left[(r_S,r_M)\in E_3\right]\leq 1/2^{n/3}$ *and for all* $(r_S,r_M)\in E_3$, $\Phi^{A'}_{r_S,r_M}-\Phi^{A}_{r_S,r_M}\leq 1$.

Proposition 8. $\Pr_{r_S,r_M}\left[(r_S,r_M)\in E_4\right]\geq 1/n^{2\log n}$ *and for all* $(r_S,r_M)\in E_4$, $\Phi^{A'}_{r_S,r_M}-\Phi^{A}_{r_S,r_M}\leq -1/2^{n/4}$.

Proposition 9. $\Pr_{r_S,r_M}\left[(r_S,r_M)\in E_5\right]\leq 1$ *and for all* $(r_S,r_M)\in E_5$, $\Phi^{A'}_{r_S,r_M}-\Phi^{A}_{r_S,r_M}\leq 0$.

From these five propositions it follows that

$$
\begin{aligned}
\Phi^{A'}-\Phi^{A} &= \mathop{\mathrm{E}}_{r_S,r_M}\left[\Phi^{A'}_{r_S,r_M}-\Phi^{A}_{r_S,r_M}\right]\\
&= \mathop{\mathrm{E}}_{r_S,r_M}\left[\Phi^{A'}_{r_S,r_M}-\Phi^{A}_{r_S,r_M}\;\middle|\;(r_S,r_M)\in E_1\right]\cdot\Pr_{r_S,r_M}\left[(r_S,r_M)\in E_1\right]+\\
&\quad \mathop{\mathrm{E}}_{r_S,r_M}\left[\Phi^{A'}_{r_S,r_M}-\Phi^{A}_{r_S,r_M}\;\middle|\;(r_S,r_M)\in E_2\right]\cdot\Pr_{r_S,r_M}\left[(r_S,r_M)\in E_2\right]+\\
&\quad \mathop{\mathrm{E}}_{r_S,r_M}\left[\Phi^{A'}_{r_S,r_M}-\Phi^{A}_{r_S,r_M}\;\middle|\;(r_S,r_M)\in E_3\right]\cdot\Pr_{r_S,r_M}\left[(r_S,r_M)\in E_3\right]+\\
&\quad \mathop{\mathrm{E}}_{r_S,r_M}\left[\Phi^{A'}_{r_S,r_M}-\Phi^{A}_{r_S,r_M}\;\middle|\;(r_S,r_M)\in E_4\right]\cdot\Pr_{r_S,r_M}\left[(r_S,r_M)\in E_4\right]+\\
&\quad \mathop{\mathrm{E}}_{r_S,r_M}\left[\Phi^{A'}_{r_S,r_M}-\Phi^{A}_{r_S,r_M}\;\middle|\;(r_S,r_M)\in E_5\right]\cdot\Pr_{r_S,r_M}\left[(r_S,r_M)\in E_5\right]\\
&\leq 1/2^{n/2}+1/2^{n/2}+1/2^{n/3}-1/n^{2\log n}2^{n/4}\\
&\leq -1/2^{n/2}
\end{aligned}
$$

which is what we wanted to show.

Proof of Proposition 5. The first assertion follows because

$$
\begin{aligned}
\Pr_{r_S,r_M}\left[(r_S,r_M)\in E_1\right] &\leq \Pr_{r_S}\left[S^{A}_{r_S}(n^d)\text{ queries }A(xy)\right]\\
&\leq s_x\\
&\leq 1/2^{n/2}.
\end{aligned}
$$

The second assertion follows trivially from the fact that $\Phi^{A'}_{r_S,r_M}\leq 1$ and $\Phi^{A}_{r_S,r_M}\geq 0$. ☐

Proof of Proposition 6. The first assertion follows because

$$
\begin{aligned}
\Pr_{r_S,r_M}\left[(r_S,r_M)\in E_2\right] &\leq \Pr_{r_S,r_M}\left[M^{L^A_3,A}_{r_M}\big(S^{A}_{r_S}(n^d)\big)\text{ queries }A(xy)\right]\\
&\leq p_x\\
&\leq 1/2^{n/2}.
\end{aligned}
$$

The second assertion follows trivially from the fact that $\Phi^{A'}_{r_S,r_M}\leq 1$ and $\Phi^{A}_{r_S,r_M}\geq 0$. ☐

Proof of Proposition 7. The first assertion follows because

$$
\Pr_{r_S,r_M}\left[(r_S,r_M)\in E_3\right] \leq \Pr_{r_S,r_M}\left[\exists i,j\text{ such that }M^{L^A_3,A}_{r_M}\big(S^{A}_{r_S}(n^d)\big)_{i,j}\in L^A_3\backslash L^{A'}_3\right]
$$

$$\begin{aligned}
&\leq \Pr_{r_S,r_M}\Big[\exists i,j \text{ such that } M_{r_M}^{L_3^A,A}\big(S_{r_S}^A(n^d)\big)_{i,j} \in L_3^A \text{ and} \\
&\qquad N^A\Big(M_{r_M}^{L_3^A,A}\big(S_{r_S}^A(n^d)\big)_{i,j}\Big) \text{ queries } A(xy) \text{ on the designated path}\Big] \\
&\leq q_x \cdot n^{e+1} \\
&\leq 1/2^{n/3}
\end{aligned}$$

where the second-to-last line follows because there are only $n^e \cdot o(n/\log n)$ pairs i,j and the last line follows because $q_x \leq 1/2^{n/2}$. The second assertion follows trivially from the fact that $\Phi_{r_S,r_M}^{A'} \leq 1$ and $\Phi_{r_S,r_M}^{A} \geq 0$. □

Proof of Proposition 8. This proposition is in some sense the crux of the whole proof. Since $1/2^{n/2} \leq 1/2n^d$, Proposition 5 implies that Inequality (6) holds and therefore Inequality (5) does not hold. We claim that if $(r_S, r_M) \notin E_1 \cup E_2 \cup E_3 \cup E_4$ then

$$M_{r_M}^{L_3^{A'},A'}\big(S_{r_S}^A(n^d)\big) \;=\; M_{r_M}^{L_3^A,A}\big(S_{r_S}^A(n^d)\big).$$

This is because every query $M_{r_M}^{L_3^A,A}\big(S_{r_S}^A(n^d)\big)$ makes to its second oracle has the same answer under A' and A, and every query it makes to its first oracle has the same answer under $L_3^{A'}$ and L_3^A. Thus the computations $M_{r_M}^{L_3^{A'},A'}\big(S_{r_S}^A(n^d)\big)$ and $M_{r_M}^{L_3^A,A}\big(S_{r_S}^A(n^d)\big)$ proceed identically, making the same queries and receiving the same answers, and hence they produce the same output. The first assertion now follows because

$$\begin{aligned}
&\Pr_{r_S,r_M}\big[(r_S,r_M) \in E_4\big] \\
\geq\; &\Pr_{r_S,r_M}\Big[(r_S,r_M) \notin E_1 \cup E_2 \cup E_3 \text{ and } M_{r_M}^{L_3^{A'},A'}\big(S_{r_S}^A(n^d)\big) \neq M_{r_M}^{L_3^A,A}\big(S_{r_S}^A(n^d)\big)\Big] \\
\geq\; &\Pr_{r_S,r_M}\Big[M_{r_M}^{L_3^{A'},A'}\big(S_{r_S}^A(n^d)\big) \neq M_{r_M}^{L_3^A,A}\big(S_{r_S}^A(n^d)\big)\Big] \\
&- \Pr_{r_S,r_M}\big[(r_S,r_M) \in E_1\big] - \Pr_{r_S,r_M}\big[(r_S,r_M) \in E_2\big] - \Pr_{r_S,r_M}\big[(r_S,r_M) \in E_3\big] \\
\geq\; &\Pr_{r_S}\Big[L_2^{A'}\big(S_{r_S}^A(n^d)\big) \neq L_2^A\big(S_{r_S}^A(n^d)\big)\Big]/3 \\
&- \Pr_{r_S,r_M}\big[(r_S,r_M) \in E_1\big] - \Pr_{r_S,r_M}\big[(r_S,r_M) \in E_2\big] - \Pr_{r_S,r_M}\big[(r_S,r_M) \in E_3\big] \\
>\; &1/24n^{d+\log n} - 1/2^{n/2} - 1/2^{n/2} - 1/2^{n/3} \\
\geq\; &1/n^{2\log n}
\end{aligned}$$

where the third line follows by a union bound and the second-to-last line follows by the negation of Inequality (5) and by Proposition 5, Proposition 6, and Proposition 7.

We now argue the second assertion. Since $(r_S, r_M) \notin E_1$, we have $S_{r_S}^{A'}(n^d) = S_{r_S}^A(n^d)$. Let w denote this string. Let i^* be the smallest value such that for some j^*,

$$M_{r_M}^{L_3^A,A}(w)_{i^*,j^*} \;\in\; L_3^{A'} \backslash L_3^A.$$

We claim the following three things.

(1) For all $i > i^*$ and all j, $\Phi^{A'}_{r_S,r_M,i,j} - \Phi^{A}_{r_S,r_M,i,j} \leq (3n^e)^{-i}$.

(2) For all $i \leq i^*$ and all j, $\Phi^{A'}_{r_S,r_M,i,j} - \Phi^{A}_{r_S,r_M,i,j} \leq 0$.

(3) $\Phi^{A'}_{r_S,r_M,i^*,j^*} - \Phi^{A}_{r_S,r_M,i^*,j^*} = -(3n^e)^{-i^*}$.

Combining the three claims, we have

$$\begin{aligned}\Phi^{A'}_{r_S,r_M} - \Phi^{A}_{r_S,r_M} &= \sum_{i,j} \Phi^{A'}_{r_S,r_M,i,j} - \Phi^{A}_{r_S,r_M,i,j} \\ &\leq -(3n^e)^{-i^*} + n^e \sum_{i>i^*} (3n^e)^{-i} \\ &\leq -(3n^e)^{-i^*}/2 \\ &\leq -1/2^{O(i^* e \log n)} \\ &\leq -1/2^{n/4}\end{aligned}$$

where the last line follows because $i^* \leq o(n/\log n)$. Note that (1) is trivial. To verify (2) and (3), note that every query $M^{L_3^A,A}_{r_M}(w)$ makes to its second oracle has the same answer under A' and A (since $(r_S, r_M) \notin E_1 \cup E_2$), and every query it makes to its first oracle up through round $i^* - 1$ is neither in $L_3^A \backslash L_3^{A'}$ (since $(r_S, r_M) \notin E_1 \cup E_2 \cup E_3$) nor in $L_3^{A'} \backslash L_3^A$ (by minimality of i^*) and thus has the same answer under $L_3^{A'}$ and L_3^A. Thus the computations $M^{L_3^{A'},A'}_{r_M}(w)$ and $M^{L_3^A,A}_{r_M}(w)$ proceed identically up to round i^*, making the same queries and receiving the same answers before round i^* and making the same queries in round i^*. Hence for all $i \leq i^*$ and all j, we have

$$M^{L_3^{A'},A'}_{r_M}(w)_{i,j} \;=\; M^{L_3^A,A}_{r_M}(w)_{i,j}.$$

For all $i \leq i^*$ and all j, since $M^{L_3^A,A}_{r_M}(w)_{i,j} \notin L_3^A \backslash L_3^{A'}$ we have

$$L_3^{A'}\Big(M^{L_3^{A'},A'}_{r_M}\big(S^{A'}_{r_S}(n^d)\big)_{i,j}\Big) \;\geq\; L_3^{A}\Big(M^{L_3^{A},A}_{r_M}\big(S^{A}_{r_S}(n^d)\big)_{i,j}\Big)$$

which proves (2). By the definition of i^*, j^* we have

$$L_3^{A'}\Big(M^{L_3^{A'},A'}_{r_M}\big(S^{A'}_{r_S}(n^d)\big)_{i^*,j^*}\Big) \;=\; 1$$

and

$$L_3^{A}\Big(M^{L_3^{A},A}_{r_M}\big(S^{A}_{r_S}(n^d)\big)_{i^*,j^*}\Big) \;=\; 0$$

which proves (3). □

Proof of Proposition 9. The first assertion is trivial. We now argue the second assertion. In the proof of Proposition 8 we argued that if $(r_S, r_M) \notin E_1 \cup E_2 \cup E_3 \cup E_4$ then the computations $M^{L_3^{A'},A'}_{r_M}\big(S^{A'}_{r_S}(n^d)\big)$ and $M^{L_3^{A},A}_{r_M}\big(S^{A}_{r_S}(n^d)\big)$ proceed identically, making the same queries and receiving the same answers. In particular,

$$L_3^{A'}\Big(M^{L_3^{A'},A'}_{r_M}\big(S^{A'}_{r_S}(n^d)\big)_{i,j}\Big) \;=\; L_3^{A}\Big(M^{L_3^{A},A}_{r_M}\big(S^{A}_{r_S}(n^d)\big)_{i,j}\Big)$$

for all i, j, which implies that $\Phi^{A'}_{r_S,r_M} = \Phi^{A}_{r_S,r_M}$. □

7 Proof of Theorem 3

Fix a polynomial q. We use the setup from Section 4, customized as follows. We have $\mathcal{C} = \text{PH}$, and M_i corresponds to a PH-type algorithm M. We redefine

$$L_1^A \;=\; \big\{x \;:\; \exists y \text{ such that } |y| = |x| + 2q(|x|) \text{ and } xy \in A\big\}$$

using $|y| = |x| + 2q(|x|)$ instead of $|y| = |x|$, and thus we need to construct A at input length $2n + 2q(n)$ rather than $2n$. We only diagonalize against reductions R that use at most q queries to the reduction oracle. Also, M on inputs of length n^d counts as "relevant computations" and thus runs in time $n^{\log n}$ without a big O. For the reason discussed at the end of Section 2.3, we have the stronger requirement that at the end of this round,

$$\Pr_{r_R,B}\Big[R_{r_R}^{B,A}(x) = L_1^A(x)\Big] \;<\; 1/2 + 1/n^{\log n}$$

with $1/2 + 1/n^{\log n}$ instead of $2/3$. Finally, note that it can never be the case that M^A fails to define a language in PH^A, since PH is a syntactically defined class.

We generalize the notion of a reduction oracle: If $B : \{0,1\}^{n^d} \to \{0,1,\bot\}^{\mathbb{N}}$ is a deterministic function then running $R_{r_R}^{B,A}(x)$ means that for each w, the ith time the computation queries $B(w)$ it gets $B(w)(i)$ as a response. Thus a randomized function $B : \{0,1\}^{n^d} \to \{0,1,\bot\}$ is a distribution over such deterministic functions, where each $B(w)(i)$ is independent and the distribution of $B(w)(i)$ depends only on w and not on i.

7.1 Main Construction

Recall that M, S, R, n are fixed. Let A denote the current relativization oracle at the beginning of this round. For $x \in \{0,1\}^n$ and $y \in \{0,1\}^{n+2q(n)}$ define

$$p_{x,y} \;=\; \Pr_{r_R}\Big[\exists B : \{0,1\}^{n^d} \to \{0,1,\bot\}^{\mathbb{N}} \text{ and } \exists x' \in \{0,1\}^n \text{ such that } R_{r_R}^{B,A}(x) \text{ queries } A(x'y)\Big]$$

and

$$p_y \;=\; \mathop{\mathrm{E}}_{x\in\{0,1\}^n}\big[p_{x,y}\big]$$

where x is chosen uniformly at random. For each $x \in \{0,1\}^n$ and r_R, the computation $R_{r_R}^{B,A}(x)$ has at most $3^{q(n)}$ computation paths over the possible responses it could get from B (recall that A is fixed). On each of these computation paths, $R_{r_R}^{B,A}(x)$ can query at most $n^{\log n}$ bits of A since it runs in time $n^{\log n}$. Thus there are at most $n^{\log n}3^{q(n)}$ pairs $(x', y) \in \{0,1\}^n \times \{0,1\}^{n+2q(n)}$ for which there exists a $B : \{0,1\}^{n^d} \to \{0,1,\bot\}^{\mathbb{N}}$ such that $R_{r_R}^{B,A}(x)$ queries $A(x'y)$. It follows that $\sum_y p_y \le n^{\log n}3^{q(n)}$ and thus

$$\Pr_{y\in\{0,1\}^{n+2q(n)}}\big[p_y > 1/2n^{\log n}\big] \;<\; 2n^{2\log n}3^{q(n)}/2^{n+2q(n)}$$

where y is chosen uniformly at random. For $y \in \{0,1\}^{n+2q(n)}$ define

$$s_y \;=\; \Pr_{r_S}\Big[\exists x' \in \{0,1\}^n \text{ such that } S_{r_S}^A(n^d) \text{ queries } A(x'y)\Big].$$

Since $S^A(n^d)$ runs in time $n^{\log n}$, we have $\sum_y s_y \leq n^{\log n}$ and thus

$$\Pr_{y \in \{0,1\}^{n+2q(n)}} \left[s_y > 1/2n^d \right] \; < \; 2n^{d+\log n}/2^{n+2q(n)}.$$

By a union bound we find that

$$\begin{aligned} & \Pr_{y \in \{0,1\}^{n+2q(n)}} \left[p_y \leq 1/2n^{\log n} \text{ and } s_y \leq 1/2n^d \right] \\ > \; & 1 - \left(2n^{2\log n} 3^{q(n)}/2^{n+2q(n)}\right) - \left(2n^{d+\log n}/2^{n+2q(n)}\right) \\ > \; & 0. \end{aligned}$$

Thus there exists a $y \in \{0,1\}^{n+2q(n)}$ such that $p_y \leq 1/2n^{\log n}$ and $s_y \leq 1/2n^d$. Fix this y. Now

$$\Pr_{x \in \{0,1\}^n} \left[p_{x,y} \geq 1/n^{\log n} \right] \; \leq \; 1/2$$

and thus there exists a set $X \subseteq \{0,1\}^n$ of size $|X| = 2^{n-1}$ such that for all $x \in X$, $p_{x,y} < 1/n^{\log n}$. To prove the theorem, it suffices to show that there exists a $Z \subseteq \{xy \; : \; x \in X\}$, an $x \in X$, and a randomized function $B : \{0,1\}^{n^d} \to \{0,1,\bot\}$ which is a valid AvgZPP oracle for $\left(L_2^{A\cup Z}, D^{A\cup Z}\right)$ at input length n^d with respect to $\delta = 1/n^d$, such that

$$\Pr_{r_R, B} \left[R_{r_R}^{B, A\cup Z}(x) = L_1^{A\cup Z}(x) \right] \; < \; 1/2 + 1/n^{\log n}$$

because we can then update the relativization oracle to be $A \cup Z$ for the end of this round.

Suppose for contradiction that this does not hold. We can assume that r_S is sampled uniformly at random from $\{0,1\}^{n^{\log n}}$ when S is run on input n^d. Define an error-correcting code

$$C : \{0,1\}^{2^{n-1}} \to \{0,1\}^{2^{n^{\log n}}}$$

as follows, where the information word is viewed as a subset $Z \subseteq \{xy \; : \; x \in X\}$ and the code word is viewed as a function $C(Z) : \{0,1\}^{n^{\log n}} \to \{0,1\}$.

$$C(Z)(r_S) \; = \; L_2^{A\cup Z}\left(S_{r_S}^A(n^d)\right)$$

Claim 1. *The relative minimum distance of C is $> 1/2n^d$.*

We prove claim 1 shortly. Let k denote the number of quantifiers M uses, and recall that M runs in time $n^{\log n}$ on inputs of length n^d. Since each bit of $C(Z)$ corresponds to running $M^{A\cup Z}$ on a fixed input of length n^d, each bit of $C(Z)$ is computable by a circuit of depth k and size $2^{n^{\log n}}$ where each input to the circuit is the output of a deterministic computation running in time $n^{\log n}$ with oracle access to $A \cup Z$. Since A is fixed, each of the inputs to this circuit is computable by a DNF with top fan-in $2^{n^{\log n}}$ and bottom fan-in $n^{\log n}$ whose inputs correspond to strings in $\{xy \; : \; x \in X\}$, that is, coordinates of the information word.

The bottom line is that there exists a binary error-correcting code with information word length 2^{n-1} and relative minimum distance $> 1/2n^d$ such that each bit of the code word is computable by a circuit of depth $k+2$ and size $2^{2n^{\log n}} n^{\log n}$. This contradicts the following result.

Theorem 5 (Viola [18]). *If there exists a binary error-correcting code with information word length ν and relative minimum distance γ such that each bit of the code word is computable by a circuit of depth κ and size σ, then $\nu\gamma \leq O(\log^{\kappa-1}\sigma)$.*

Theorem 5 holds regardless of the rate of the code.

Proof of Claim 1. We exhibit a decoder that can handle up to a $1/2n^d$ fraction of erasures.

- **Input:** $C' : \{0,1\}^{n^{\log n}} \to \{0,1,\bot\}$
- **Output:** $Z' \subseteq \big\{xy \ : \ x \in X\big\}$ given by

$$Z' \ = \ \Big\{xy \ : \ \Pr_{r_R,B}\Big[R^{B,A}_{r_R}(x) = 1\Big] > 1/2\Big\}$$

where the randomized function $B : \{0,1\}^{n^d} \to \{0,1,\bot\}$ is defined by

$$\Pr_B\big[B(w) = b\big] \ = \ \Pr_{r_S}\Big[C'(r_S) = b \ \Big| \ S^A_{r_S}(n^d) = w\Big]$$

if

$$\Pr_{r_S}\Big[S^A_{r_S}(n^d) = w\Big] \ > \ 0$$

and otherwise

$$\Pr_B\big[B(w) = \bot\big] \ = \ 1$$

For an arbitrary $Z \subseteq \big\{xy \ : \ x \in X\big\}$, assume that C' agrees with $C(Z)$ on at least a $1 - 1/2n^d$ fraction of r_S's and outputs $\bot$ on the rest. Then we just need to show that $Z' = Z$. We do this by showing that for an arbitrary $x \in X$,

$$\Pr_{r_R,B}\Big[R^{B,A}_{r_R}(x) = L^{A\cup Z}_1(x)\Big] \ > \ 1/2$$

which implies that $xy \in Z'$ if and only if $x \in L^{A\cup Z}_1$ if and only if $xy \in Z$.

We start by showing that B is a valid AvgZPP oracle for $\big(L^{A\cup Z}_2, D^{A\cup Z}\big)$ at input length n^d with respect to $\delta = 1/n^d$. We have that $B(w)$ always equals $L^{A\cup Z}_2(w)$ or $\bot$, since if r_S is such that $S^A_{r_S}(n^d) = w$ and $C'(r_S) \neq \bot$ then

$$C'(r_S) \ = \ C(Z)(r_S) \ = \ L^{A\cup Z}_2\big(S^A_{r_S}(n^d)\big) \ = \ L^{A\cup Z}_2(w).$$

We have

$$\begin{aligned}\Pr_{r_S,B}\Big[B\big(S^A_{r_S}(n^d)\big) = \bot\Big] \ &= \sum_{w\in\{0,1\}^{n^d}} \Pr_{r_S,B}\Big[B\big(S^A_{r_S}(n^d)\big) = \bot \ \Big| \ S^A_{r_S}(n^d) = w\Big] \cdot \Pr_{r_S,B}\Big[S^A_{r_S}(n^d) = w\Big] \\ &= \sum_{w\in\{0,1\}^{n^d}} \Pr_B\big[B(w) = \bot\big] \cdot \Pr_{r_S}\Big[S^A_{r_S}(n^d) = w\Big]\end{aligned}$$

$$
\begin{aligned}
&= \sum_{w\in\{0,1\}^{n^d}} \Pr_{r_S}\left[C'(r_S)=\bot \;\middle|\; S^A_{r_S}(n^d)=w\right]\cdot \Pr_{r_S}\left[S^A_{r_S}(n^d)=w\right] \\
&= \Pr_{r_S}\left[C'(r_S)=\bot\right] \\
&\le 1/2n^d
\end{aligned}
$$

and

$$
\begin{aligned}
\Pr_{r_S}\left[S^{A\cup Z}_{r_S}(n^d)\ne S^A_{r_S}(n^d)\right] &\le \Pr_{r_S}\left[\exists z\in Z \text{ such that } S^A_{r_S}(n^d) \text{ queries } A(z)\right] \\
&\le s_y \\
&\le 1/2n^d
\end{aligned}
$$

and thus

$$
\begin{aligned}
\Pr_{w\sim D^{A\cup Z},B}\left[B(w)=\bot\right] &= \Pr_{r_S,B}\left[B\big(S^{A\cup Z}_{r_S}(n^d)\big)=\bot\right] \\
&\le \Pr_{r_S,B}\left[B\big(S^{A}_{r_S}(n^d)\big)=\bot \text{ or } S^{A\cup Z}_{r_S}(n^d)\ne S^A_{r_S}(n^d)\right] \\
&\le \Pr_{r_S,B}\left[B\big(S^{A}_{r_S}(n^d)\big)=\bot\right] + \Pr_{r_S}\left[S^{A\cup Z}_{r_S}(n^d)\ne S^A_{r_S}(n^d)\right] \\
&\le 1/2n^d + 1/2n^d \\
&= 1/n^d = \delta.
\end{aligned}
$$

Now we have

$$
\begin{aligned}
\Pr_{r_R,B}\left[R^{B,A\cup Z}_{r_R}(x)\ne R^{B,A}_{r_R}(x)\right] &\le \mathop{\mathrm{E}}_{B}\left[\Pr_{r_R}\left[\exists z\in Z \text{ such that } R^{B,A}_{r_R}(x) \text{ queries } A(z)\right]\right] \\
&\le \mathop{\mathrm{E}}_{B}\left[p_{x,y}\right] \\
&= p_{x,y} \\
&< 1/n^{\log n}
\end{aligned}
$$

and thus

$$
\begin{aligned}
\Pr_{r_R,B}\left[R^{B,A}_{r_R}(x)=L_1^{A\cup Z}(x)\right] &\ge \Pr_{r_R,B}\left[R^{B,A\cup Z}_{r_R}(x)=L_1^{A\cup Z}(x) \text{ and } R^{B,A\cup Z}_{r_R}(x)=R^{B,A}_{r_R}(x)\right] \\
&\ge \Pr_{r_R,B}\left[R^{B,A\cup Z}_{r_R}(x)=L_1^{A\cup Z}(x)\right] - \Pr_{r_R,B}\left[R^{B,A\cup Z}_{r_R}(x)\ne R^{B,A}_{r_R}(x)\right] \\
&> \big(1/2+1/n^{\log n}\big) - 1/n^{\log n} \\
&= 1/2
\end{aligned}
$$

where the third line follows by our contradiction assumption. □

8 Proof of Theorem 4

We use the setup from Section 4, customized as follows. We only diagonalize against reductions R that use at most 2 queries to the reduction oracle. For the reason discussed at the end of Section

2.3, we have the stronger requirement that at the end of this round,

$$\Pr_{r_R,B}\left[R^{B,A}_{r_R}(x) = L^A_1(x)\right] \;<\; 1/2 + 1/n^{\log n}$$

with $1/2 + 1/n^{\log n}$ instead of $2/3$. The proof is so similar to the proof of Theorem 3 that we just sketch how it plays out. We can work with $|y| = n$ (rather than $|y| = n + 2q(n)$ as in the proof of Theorem 3).

8.1 Main Construction

Recall that M_i, S, R, n are fixed. Let A denote the current relativization oracle at the beginning of this round. There exists a $y \in \{0,1\}^n$ such that $p_y \le 1/4n^{\log n}$ and $s_y \le 1/2n^d$, and there exists a set $X \subseteq \{0,1\}^n$ of size $|X| = 2^{n-1}$ such that for all $x \in X$, $p_{x,y} \le 1/2n^{\log n}$. Then there exists a $Z \subseteq \{xy \;:\; x \in X\}$, an $x \in X$, and a randomized function $B : \{0,1\}^{n^d} \to \{0,1,\bot\}$ which is a valid AvgZPP oracle for $\left(L^{A\cup Z}_2, D^{A\cup Z}\right)$ at input length n^d with respect to $\delta = 1/n^d$, such that

$$\Pr_{r_R,B}\left[R^{B,A\cup Z}_{r_R}(x) = L^{A\cup Z}_1(x)\right] \;<\; 1/2 + 1/n^{\log n}$$

since otherwise we can extract an error-correcting code

$$C : \{0,1\}^{2^{n-1}} \to \{0,1\}^{2^{n^{\log n}}}$$

with the following properties. There is a randomized decoder that can handle up to a $1/2n^d$ fraction of erasures, and it recovers any bit of the information word with probability at least

$$\left(1/2 + 1/n^{\log n}\right) - 1/2n^{\log n} \;=\; 1/2 + 1/2n^{\log n}.$$

To recover any bit, the decoder runs $R^{B,A}(x)$ for some $x \in \{0,1\}^n$ and some randomized function B. Since R makes at most 2 queries to B, and since each query to B can be answered with at most 1 query to the corrupted code word C', the decoder makes at most 2 queries to C'.

The bottom line is that there exists a binary error-correcting code with information word length 2^{n-1} and code word length $2^{n^{\log n}}$ and a decoder that uses 2 queries to recover any bit of the information word with probability at least $1/2 + 1/2n^{\log n}$ when at most a $1/2n^d$ fraction of the code word bits are erased. This contradicts the following result.

Theorem 6 (Kerenidis and de Wolf [14]). *If there exists a binary error-correcting code with information word length ν and code word length μ and a decoder that uses 2 queries to recover any bit of the information word with probability at least $1/2 + \epsilon$ when at most a γ fraction of the code word bits are erased, then $\mu \ge 2^{\Omega(\gamma\epsilon^3\nu)}$.*

Remarkably, the proof of Theorem 6 is based on quantum information theory. Kerenidis and de Wolf proved the stronger bound $\mu \ge 2^{\Omega(\gamma\epsilon^2\nu)}$ assuming that the decoder is guaranteed to work even if a γ fraction of the code word bits are flipped rather than just erased. The extra ϵ in the exponent in Theorem 6 grossly accounts for the generalization from flips to erasures. It may be possible to prove the stronger bound for erasure decoders, but Theorem 6 as stated is already good enough for our purpose.

The complexity of M_i is immaterial because Theorem 6 holds without any constraints on the efficiency of the encoder.

9 Open Problems

The problem of constructing a strong relativized heuristica, that is, an oracle relative to which $\big(\mathrm{NP}, \mathrm{PS}\textsc{amp}\big) \subseteq \mathrm{AvgZPP}$ but $\mathrm{NP} \not\subseteq \mathrm{BPP}$, remains open.

Can Theorem 3 be strengthened to handle reductions with an unlimited number of queries? This would subsume Theorem 1, Theorem 2, and Theorem 3. It is even open to prove that there exists an oracle relative to which there is no nonadaptive reduction of type

$$\big(\mathrm{P}^{\mathrm{NP}}, \mathrm{PS}\textsc{amp}\big) \subseteq \mathrm{HeurBPP} \;\Rightarrow\; \mathrm{P}^{\mathrm{NP}} \subseteq \mathrm{BPP}.$$

It seems that proving the full result may require new insight about the structure of the adaptivity in worst-case to average-case reductions. In our proofs we carefully sidestepped the issue of adaptivity. In Theorem 1 and Theorem 2 we did this by maneuvering into a situation where we could fix a valid AvgZPP reduction oracle ahead of time, in which case the pattern of queries to the reduction oracle was immaterial. In Theorem 3 and Theorem 4 we did this by restricting the number of queries, so that all potential paths of adaptivity could be considered simultaneously. Even for nonadaptive reductions, this restriction on the number of queries is essential to our proof of Theorem 3.

Impagliazzo and Levin [11] proved that relative to every oracle, there exists a nonadaptive reduction of type

$$\big(\mathrm{NP}, \mathrm{PS}\textsc{amp}\big) \subseteq \mathrm{HeurBPP} \;\Rightarrow\; \big(\mathrm{NP}, \mathcal{U}\big) \subseteq \mathrm{HeurBPP}.$$

This reduction uses polynomially many queries. It is open to construct such a reduction using a smaller number of queries, ideally a mapping reduction. It would be interesting to prove that there exists an oracle relative to which no such mapping reduction exists.

In the worst-case setting, it is well-known that relative to every oracle, $\mathrm{NP} \subseteq \mathrm{BPP}$ implies $\Sigma_2\mathrm{P} \subseteq \mathrm{BPP}$. It is open to prove the average-case analogue $\big(\mathrm{NP}, \mathrm{PS}\textsc{amp}\big) \subseteq \mathrm{HeurBPP}$ implies $\big(\Sigma_2\mathrm{P}, \mathrm{PS}\textsc{amp}\big) \subseteq \mathrm{HeurBPP}$. It would be interesting to prove that there exists an oracle relative to which this is not true. Note that every such oracle gives a relativized heuristica, since relative to every oracle, $(\Sigma_2\mathrm{P}, \mathrm{PS}\textsc{amp}) \not\subseteq \mathrm{HeurBPP}$ implies $\Sigma_2\mathrm{P} \not\subseteq \mathrm{BPP}$ implies $\mathrm{NP} \not\subseteq \mathrm{BPP}$.

Bogdanov and Trevisan [6] proved that relative to every oracle, if there exists a nonadaptive reduction of type

$$\big(\mathrm{NP}, \mathrm{PS}\textsc{amp}\big) \subseteq \mathrm{HeurBPP} \;\Rightarrow\; \mathrm{NP} \subseteq \mathrm{BPP}$$

then the polynomial-time hierarchy collapses to the third level. It is open to extend this result to adaptive reductions. It would be interesting to prove that there exists an oracle relative to which such an adaptive reduction exists and yet the polynomial-time hierarchy is infinite. Can the "Book trick" [7] be used? Less generally, it would be interesting to prove that there exists an oracle relative to which an adaptive reduction of the above type exists but no nonadaptive reduction of the above type exists.

Acknowledgments

First and foremost, I thank Luca Trevisan for suggesting the research topic. I thank Luca Trevisan and Dieter van Melkebeek for helpful comments.

References

[1] S. Aaronson. Quantum Computing, Postselection, and Probabilistic Polynomial-Time. *Proceedings of the Royal Society A*, 461(2063): 3473-3482, 2005.

[2] S. Arora and B. Barak. *Computational Complexity: A Modern Approach.* Cambridge University Press, 2009.

[3] 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.

[4] C. Bennett, E. Bernstein, G. Brassard, and U. Vazirani. Strengths and Weaknesses of Quantum Computing. *SIAM Journal on Computing*, 26(5): 1510-1523, 1997.

[5] A. Bogdanov and L. Trevisan. Average-Case Complexity. *Foundations and Trends in Theoretical Computer Science*, 2(1), 2006.

[6] A. Bogdanov and L. Trevisan. On Worst-Case to Average-Case Reductions for NP Problems. *SIAM Journal on Computing*, 36(4): 1119-1159, 2006.

[7] L. Fortnow. Relativized Worlds with an Infinite Hierarchy. *Information Processing Letters*, 69(6): 309-313, 1999.

[8] O. Goldreich. *Computational Complexity: A Conceptual Perspective.* Cambridge University Press, 2008.

[9] Y. Han, L. Hemaspaandra, and T. Thierauf. Threshold Computation and Cryptographic Security. *SIAM Journal on Computing*, 26(1): 59-78, 1997.

[10] R. Impagliazzo. A Personal View of Average-Case Complexity. In *Proceedings of the 10th IEEE Conference on Structure in Complexity Theory*, pages 134-147, 1995.

[11] R. Impagliazzo and L. Levin. No Better Ways to Generate Hard NP Instances than Picking Uniformly at Random. In *Proceedings of the 31st IEEE Symposium on Foundations of Computer Science*, pages 812-821, 1990.

[12] R. Impagliazzo and M. Luby. One-Way Functions are Essential for Complexity Based Cryptography. In *Proceedings of the 30th IEEE Symposium on Foundations of Computer Science*, pages 230-235, 1989.

[13] R. Impagliazzo and S. Rudich. Unpublished manuscript.

[14] I. Kerenidis and R. de Wolf. Exponential Lower Bound for 2-Query Locally Decodable Codes via a Quantum Argument. *Journal of Computer and System Sciences*, 69(3): 395-420, 2004.

[15] L. Levin. Average Case Complete Problems. *SIAM Journal on Computing*, 15(1): 285-286, 1986.

[16] M. Sudan, L. Trevisan, and S. Vadhan. Pseudorandom Generators Without the XOR Lemma. *Journal of Computer and System Sciences*, 62(2): 236-266, 2001.

[17] L. Trevisan and S. Vadhan. Pseudorandomness and Average-Case Complexity via Uniform Reductions. *Computational Complexity*, 16(4): 331-364, 2007.

[18] E. Viola. The Complexity of Constructing Pseudorandom Generators from Hard Functions. *Computational Complexity*, 13(3-4): 147-188, 2004.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de