Random oracles, the Polynomial-Time Hierarchy, and Constant-depth Circuits: A survey

Andrew He, Lisa Yang

May 7, 2016

Abstract

It has long been known that, relative to a random oracle, PH^A and PSPACE^A are separate with probability 1 [Hås86]. We present the recent result proved in [RST15] that, in fact, the polynomial hierarchy PH^A is infinite with probability 1.

This result follows from certain average-case circuit lower bounds for constant-depth circuits. These lower bounds rely on a generalization of the standard "random restriction" method for proving AC_0 bounds, namely random projections. This paper seeks to motivate and explain these generalizations from the previously known worst-case lower bounds.

Contents

1 Introduction

The study of Turing machines generally treats Turing machines as black boxes which take strings as input and perform some computation which can be simulated by a Universal Turing Machine. Results proven about Turing machines viewed this way *relativize* with respect to oracles O: if the Turing machines are allowed to make queries to O, then the result still holds in this "relativized world". Unfortunately, Baker, Gill and Solovay showed that there exists some oracle A such that $\mathsf{P}^A = \mathsf{NP}^A$, as well as some oracle B such that $\mathsf{P}^B \neq \mathsf{NP}^B$, so the question of $\mathsf{P} = \mathsf{NP}$ cannot be decided relative to all oracles.

Instead, we can consider this question relative to a random oracle. By the Kolmogorov zero-one law, two complexity classes are almost always or almost never separated relative a randomly chosen oracle (with probability 1 or 0). Initially, it was conjectured that this probability was 1 if and only if the two classes are separate without any oracles (the *Random Oracle Hypothesis*); this was proven false, most notably with the counterexample that $\mathsf{IP} = \mathsf{PSPACE}$, but not relative to random oracles.

It has been known since around the 1980s that there exists some oracle such that $\mathsf{PH}^A \neq \mathsf{PSPACE}^A$, and for all k, $\Sigma_k^{\mathsf{P},A} \neq \Sigma_{k+1}^{\mathsf{P},A}$ [Hås86] In 1986, Cai proved that $\mathsf{PH}^A \neq \mathsf{PSPACE}^A$ for *random* oracles A [Cai86]. Recently, Rossman, Servedio and Tan proved the last piece of this result that $\Sigma_k^{\mathsf{P},A} \neq \Sigma_{k+1}^{\mathsf{P},A}$ for random oracles A.

The central results of these random oracles are actually bounds on constant-depth circuits, a very well-studied area. At a high level, we can think of Turing machines with oracles as analogous to circuits: we convert a Turing machine M^A run on a particular input x ($|x| = n$) to a circuit $\mathcal{C}$ with $N \triangleq 2^{\mathsf{poly}(n)}$ input wires corresponding to the values of A. (Phrased as non-uniform Turing machines, the input string becomes the advice string, and the oracle becomes the input.) Then, Turing machines in P correspond to $\mathsf{poly}(n) = \mathsf{polylog}(N)$-depth decision trees (which can be written as either DNFs or CNFs). Then, adding a quantifier corresponds to adding an alternating layer on top of the tree, so Σ_k^{P} Turing machines correspond to depth-$(k+1)$ (alternating) circuits with an OR on top. Figure 1 summarizes this correspondence. Note that this correspondence is not a true one-to-one correspondence: proving random oracle bounds from circuit bounds still requires a diagonalization argument, as we need to create mismatches over *original inputs of the Turing machines.*

We state the main circuit bounds referenced in this paper:

Theorem 1. *[Yao85, Hås86]. There exists a universal constant c_0 such that any depth d circuit C of size at most $2^{n^{c_0(\frac{1}{d})}}$ does not agree with* PARITY *on all inputs.*

Theorem 2. *[Cai86, Hås86]. There exists a universal constant c_0 such that for any $\epsilon > 0$, any depth d circuit C of size at most $2^{n^{c_0(\frac{1}{d})}}$ does not agree with* PARITY *on at least $(\frac{1}{2} - \epsilon) \cdot 2^n$ inputs.*

Theorem 3. *[Hås86]. There exists a universal constant c_0 such that any depth d circuit C of size at most $2^{n^{c_0(\frac{1}{d})}}$ and bottom fan-in $c_0(\frac{\log n}{d})$ does not agree with* Sipser_d *on all inputs.*

Theorem 4. *[RST15]. There exists a universal constant c_0 such that for any $\epsilon > 0$, any*

Oracle world	Circuit world
Turing machine on size-n input	Circuit C with $N = 2^{\mathsf{polylog}(n)}$ inputs
Oracle O	Input wires $x \in \{0,1\}^N$
Turing machine in P	$\mathsf{polylog}(N)$-depth decision tree
Turing machine in NP	$\mathsf{polylog}(N)$-width DNF
Turing machine in coNP	$\mathsf{polylog}(N)$-width CNF
Turing machine in Σ_d^{P} or Π_d^{P}	Depth-$d+1$ circuit with $\mathsf{polylog}(N)$ bottom fan-in; alternatively, a depth d circuit
Random oracle	Random input
Separation relative to some oracle	Worst-case lower bounds
Separation relative to random oracles	Average-case lower bounds

Figure 1: Oracle $\longleftrightarrow$ circuit mapping (expanded from [Bar15])

depth d circuit C of size at most $2^{n^{c_0(\frac{1}{d})}}$ and bottom fan-in $c_0(\frac{\log n}{d})$ does not agree with Sipser_d on at least $(\frac{1}{2} - \epsilon) \cdot 2^n$ inputs.

The remainder of the paper will work towards proving these bounds. Section 2 will give our general framework of proving circuit lower bounds using random restrictions, and Section 3 will prove Theorem 2 using this method. We will then extend our restrictions to prove Theorems 3 and 4 in Sections 4 and 5, respectively. All of these proofs use the powerful switching lemma, whose proof we defer to Section 6. Finally, Section 7 will use these main theorems to prove bounds on PH and PSPACE relative to random oracles.

The primary goal of this paper is not to provide a rigorous proof, but instead is to motivate the powerful generalizations of random restrictions which have been developed. As such, we will occasionally gloss over or even omit details from these proofs, including some of the exact constants used. These details can be generally be filled in by the reader, or can be found in the source papers themselves.

2 Circuit lower bounds via random restrictions

Our main tool to prove the circuit complexity bounds is the method of *random restrictions*. These restrictions allow us to reduce the depth of a circuit by randomly fixing some small subset of the inputs. We first introduce the general framework and desired properties for proving bounds with these random restrictions, and then explore the restrictions used to prove each of the depth-hierarchy theorems.

Remark. Throughout this survey, we assume our circuits (or functions computable by circuits) are arranged so that successive layers alternate between OR and AND gates; otherwise, we could merge adjacent layers and shrink the circuit. We also 0-index the layers of circuits, so that a depth-d circuit has gates at levels $0 \dots d-1$.

2.1 Desired properties of restrictions

We use the notation introduced by [RST15]. We'd like to show that a particular *target function* $f : \{0,1\}^n \to \{0,1\}$ cannot be approximated by any depth-d circuit C. (Our target function is usually some structured but difficult function like PARITY.) We then specify some series of (distributions of) random restrictions $\{\mathcal{R}_k\}_{k=2}^d$ so that:

- **Property 1: Circuit C simplifies.** We would like our random restrictions to actually reduce the depth of C. In particular, we successively apply restrictions $\boldsymbol{\rho}_k \leftarrow \mathcal{R}_k$ for each k from d to 2, and would like the kth restriction to, with high probability, decrease the depth of C from k to $k-1$. We usually prove this with the help of a *switching lemma*, which states that a random restriction of a CNF or DNF can be written as a small-depth decision tree. At the end of our process, our circuit is thus a relatively simple small-depth decision tree.
- **Property 2: Target f remains structured.** We would like for the target function to, unlike the circuit, remain structured and difficult (with high probability). We prove this by explicitly using the structure of our target function and showing that it is (in some sense) preserved.
- **Property 3: The reduced f cannot computed by simple circuits.** We would like to qualitatively show that f in fact remains hard. Thus, we show that the structured functions we get from f cannot be computed by the simple circuits we get from C. This step reflects the main balancing act: we need to chose our probability of restricting each variable carefully so as to reduce the circuit with high probability, without making the target function too simple.

These two properties already can prove worst-case bounds, by taking union bounds over the first two properties: if with some probability, the restricted circuit is "too simple" to compute the "well-structured" restricted target function, then the circuit cannot exactly compute the target function, as restrictions only fix inputs.

However, these are insufficient to prove the average-case bounds that f and C are uncorrelated over all inputs, not just the restriction. To do this, we need the following additional property:

- **Property 4: Restrictions complete to $\mathcal{U}$.** We would like our restrictions to mimic the overall distribution of inputs. More formally, we would like choosing a random input $\boldsymbol{X} \leftarrow \mathcal{U}(\{0,1\}^n)$ to be equivalent to first choosing restrictions $\boldsymbol{\rho}_k \leftarrow \mathcal{R}_k$ and then drawing remaining inputs from some distribution $\mathcal{D}$.

We also need the stronger version of Property 3 that the reduced f cannot be approximated by simple circuits (over $\mathcal{D}$). Then, with high probability, the restricted circuit and restricted target are uncorrelated over $\mathcal{D}$, so the original circuit and original target are uncorrelated over the uniform distribution.

2.2 Our random restrictions

We now present a brief overview of the restrictions and target functions used to prove our bounds.

First, in Section 3, we separate constant-depth circuits from general circuits. We use PARITY as our target function, which computes the parity of the number of 1s in the input. This is fairly simple to analyze, and was first considered by Yao and Hastad in 1985. We chose our random restrictions so that, with some fixed probability p, we leave the variable unfixed; otherwise, we fix it at 0 or 1 with equal probability. This retains the structure of PARITY, as PARITY restricted to some subset of the variables is just PARITY on that subset. No low-depth decision tree computes (approximates) PARITY, as it always depends on all inputs. Furthermore, this trivially completes to uniform, as 0 and 1 are symmetric. It remains to show the first property that any circuit simplifies, which is the result of Hastad's original switching lemma.

Proving depth hierarchy theorems is harder. In this case, we consider the target Sipser_d circuit, which is, in some sense the "bushiest" (and simplest) depth-d circuit. Layers are alternating, each layer's gates have fan-in about $n^{1/d}$, and each variable is read exactly once (we define this more rigorously below). Unfortunately, our simple random restrictions do not preserve the structure of Sipser_d (Property 2) (in fact, by the switching lemma, they destroy Sipser_d).

Instead, Hastad proposed using certain *blockwise random restrictions*, which are built specifically to preserve Sipser_d. Essentially, we consider restrictions which are not directly product distributions: we process the restriction in blocks, corresponding to sub-trees of the Sipser_d circuit. In each block, our restriction always leaves at most 1 variable unfixed. Thus, these restrictions match the structure of Sipser, but are chosen to still reduce general circuits.

The above blockwise restriction is sufficient to prove worst-case bounds, but due to the strong correlation in each block, it is difficult to complete to the uniform distribution. The solution by [RST15] is to generalize randomized restrictions to randomized *projections*, which restrict some variables and fix others as equal (instead of setting only 1 unfixed variable). The desired properties still hold, and the projection actually gives us enough flexibility to complete to the uniform distribution.

3 Inapproximability of PARITY: standard random restrictions

In this section, we prove that PARITY cannot be approximated by any constant-depth circuits using "vanilla" or "standard" random restrictions. We begin by formalizing the definition of a restriction.

Definition 5 (Restriction)**.** Let A be a set. Given a map $\rho : \{x_i\}_{i\in A} \to \{0,1,*\}^A$ and function $f : \{0,1\}^A \to \{0,1\}$ on variables x_i, the restriction $(f\restriction\rho) : \{0,1\}^{\rho^{-1}(*)} \to \{0,1\}$ is defined as $(f\restriction\rho)(y) = f(x)$, where $x_i = \rho(x_i)$ if $\rho(x_i) = 0,1$ and leaves $x_i = y_i$ otherwise.

Essentially, a restriction fixes some subset of the variables.

Restriction ρ' is said to *refine* ρ if ρ''s labeling of x_i does not disagree with any of ρ's labels. We can *randomize* restrictions by specifying a distribution $\mathcal{R}$ of such maps and letting $\rho \leftarrow \mathcal{R}$.

Definition 6 (Standard random restrictions)**.** We define the "standard" distribution $\mathcal{R}_p$ of random restrictions over $\{0,1\}^A$ for $0 \le p \le 1$ to be: for each $i \in A$, choose ρ_i independently according to

$$\rho_i = \begin{cases} * & \text{with probability } p \\ 1 & \text{with probability } \frac{1-p}{2} \\ 0 & \text{with probability } \frac{1-p}{2} \end{cases}$$

Our main power for these restrictions comes from the following switching lemma:

Theorem 7 (Hastad's switching lemma)**.** *Consider a width-t DNF (or CNF) F (each clause has at most t variables), and consider a random restriction $\rho \leftarrow \mathcal{R}_p$. Then, for sufficiently small constants p, with probability at least $1-(pt)^{O(s)}$, $F' = F\restriction\rho$ can be written as a decision tree of depth at most s. Note that this decision tree can be converted to either a width-s CNF or DNF.*

There are a variety of proofs of the switching lemma, all of which are fairly tedious, and none of which are very enlightening. As such, we will defer the proof to Section 6.

Corollary 8. *Consider any alternating Boolean circuit C with depth $d \ge 3$, and consider a random restriction $\rho \leftarrow \mathcal{R}_p$. Let g be the number of gates in the second-to-last level. Then, with probability at least $1 - g(pt)^{O(s)}$, $C' = C\restriction\rho$ can be written as a depth $d-1$ circuit with bottom fan-in s (and the same upper $d-2$ levels).*

Proof. Consider each of the gates in the second-to-last layer of C. Without loss of generality, let them be OR gates. Then, the subtree at these gates form width-t DNFs, so by the switching lemma, we can replace them with width-s CNFs, which can be merged into the layer above. Our new probability bounds comes from a union bound over these g gates. □

We now prove a slightly modified version of Theorem 2.

Theorem (Modified Theorem 2)**.** *There exists a universal constant c_0 such that for any $\epsilon > 0$, any depth d circuit C of size at most $2^{n^{c_0(\frac{1}{d})}}$ with all bottom gates with fan-in at most $n^{\frac{1}{2d}}$ does not agree with PARITY on at least $(\frac{1}{2} - \epsilon) \cdot 2^n$ inputs.*

Note that this still implies Theorem 2, as any depth-d circuit is a depth-$d+1$ circuit with bottom fan-in 1.

Proof. We choose $p = O(n^{-\frac{1}{2d}})$, and set the restrictions for each level to be $\mathcal{R}_p$. (Recall that we perform $d-1$ restrictions in sequence). Then, we can prove the following properties:

- **Property 1:** Circuits do simplify under random restrictions. After $d-1$ restrictions, by the corollary with $s = t = n^{\frac{1}{2d}}$, we end with a decision tree of depth $n^{\frac{1}{2d}}$. This occurs with probability at least $1 - g(pt)^{O(n^{\frac{1}{2d}})}$, where g is the size of the C (actually only the first $d-1$ levels of C). We chose $pt = O(1)$, so this is negligible for $g < 2^{n^{c(\frac{1}{d})}}$ for some universal constant c.
- **Property 2:** PARITY reduces to PARITY under random restrictions. In particular, after some restriction ρ, $\mathsf{PARITY}\restriction\rho$ corresponds to either PARITY or the complement of PARITY on the remaining variables. With high probability (by a simple Chernoff bound), we have approximately $p^{d-1}n = O(\sqrt{n})$ variables left.
- **Property 3:** PARITY (or its complement) on k variables cannot be approximated by decision trees of depth $k-1$. This follows from the observation that along each path of the decision tree, we can read at most $k-1$ of the variables. Then, each (reachable) branch of the tree is correct exactly $\frac{1}{2}$ of the time, so the decision tree cannot approximate PARITY better than $\frac{1}{2}$ of the time.
- **Property 4:** The random restrictions complete to uniform. This is trivial, as $\mathcal{R}_p$ selects 0 and 1 with equal probability.

Thus, with all but negligible probability, the restriction of C matches PARITY on exactly $\frac{1}{2}$ of the inputs, so C does not match PARITY on at least $\frac{1}{2} - \mathsf{negl}(n)$ fraction of inputs. □

4 Incomputability of Sipser: Hastad's blockwise restrictions

We now extend our restrictions to sketch the proof the worst-case depth hierarchy theorem (Theorem 3). We omit many of the details, as they can be seen as simplifications of the average-case depth hierarchy theorem we will prove in Section 5.

We must introduce two new concepts: a new target function, and a new restriction.

Our target function is essentially the "biggest and baddest" possible depth-d circuit. All gates (except the top and bottom) have the same fan-in, and no variables are repeated.

Definition 9 (Sipser function)**.** Sipser_d is a "read-once monotone" depth-d Boolean formula on n variables consisting of AND and OR operations. It is the function computed by a depth d polynomial sized circuit C with n leaves, each taking a distinct input variable. Every path in C alternates between AND and OR gates starting with AND gates on the bottom-most layer.

Let the gates at depth k have fan-in w_k. Then, set $w_{d-1} = m$, define $w = w_{d-2} = ... = w_1 = \lfloor m2^m / \log(e) \rfloor$, and choose w_0 so that the output is as balanced as possible (probability of outputting 0 and probability of outputting 1 are both about $\frac{1}{2}$). This is necessary for there to not be a trivial approximator for Sipser_d.

We also introduce some notation to describe Sipser.

- First, we give a system to label gates (or wires) of Sipser_d. Let $A_0 := \{output\}$ and recursively define $A_k := A_{k-1} \times [w_{k-1}]$ where w_{k-1} is the uniform fan-in of the depth $k-1$ gates of the Sipser_d. An element of A_k specifies a gate of the circuit at depth k, and A_d are the input wires.
- $\mathsf{Sipser}_d^{(k)} : \{0,1\}^{A_k} \to \{0,1\}$ is the depth-k subformula obtained by letting the depth k gates of Sipser_d be input variables.

Now unfortunately, our original standard restrictions do not preserve the structure of Sipser_d. This is somewhat unavoidable, as they were designed to destroy constant-depth circuits, such as Sipser_d itself.

Instead, we design a new set of restrictions specifically to match (and thus preserve) the structure of Sipser.

Definition 10. Let $\{\circ_p, \bullet_{1-p}\}$ be the distribution over $\{\circ, \bullet\}$ with $\circ$ weighted p and $\bullet$ weighted $1-p$.

Definition 11 (Blockwise restrictions)**.** We define restrictions over the set $A \times [w]$, which we view as $|A|$ blocks of size w.

We define $\mathcal{R}_p^+ : \{\rho : \{x_{a,i}\}_{(a,i)\in A\times[w]} \to \{0,1,*\}^{A\times[w]}\}$ to be the following distribution of restrictions: independently for each block $a \in A$,

- With probability q, sample $\rho_a \leftarrow g(\{*_q, 1_{1-q}\}^w)$. Here, $g : \{*,1\}^w \to \{*,1\}^w$ turns all terms except the first $*$ into 1s.
- With probability $1-q$, sample $\rho_a \leftarrow \{0_q, 1_{1-q}\}^w$.

We define $\mathcal{R}_p^-$ in the same way, except swapping 1s and 0s.

We use $\mathcal{R}_p^+$ in layers with AND gates, and $\mathcal{R}_p^-$ in layers with OR gates.

Thus, we get the **key property** of these blockwise restrictions: each restriction collapses about $1-p$ fraction of the AND gates in the bottom layer of Sipser to value 0, which are absorbed by the OR gates above it, while the rest are collapsed to *a single variable* (by g).

Thus, for appropriate choice of p, we essentially collapse Sipser_d to Sipser_{d-1}, so after $d-1$ restrictions, Sipser_d is collapsed to only the top level AND or OR gate with about a fraction p of the fan-in, which cannot be computed by any decision tree of smaller depth.

In contrast, the switching lemma still holds for circuits with small fan-in, so these circuits collapse to a shallow decision tree. The proof is once again involved and is only a slight variation of the previous switching lemma.

Together, these statements prove worst-case bounds for computing Sipser.

Unfortunately, our blockwise restrictions cannot complete to uniform. Because we apply g to restrict the number of variables passed through Sipser, the restrictions of different variables become quite correlated. In the next section, we show how to redefine these restrictions to avoid correlation and to obtain an average-case bound.

5 Inapproximability of Sipser: RST's adaptive projections

This section presents the proof of Theorem 4 given in [RST15]. We begin by defining projections and stating our random projections, which are adaptive based on previous projections. Then we show these projections satisfy **Properties 2-4** of Section 2.1. **Property 1** is handled in Section 6. Together, these compose a proof of Theorem 4.

5.1 Defining random projections

We first collect the parameters we will use to define our random projections.

Definition 12 (Parameters)**.** Recall Sipser_d is formally defined in terms of its bottom fan-in m and subsequent fan-in $w := \left\lfloor \frac{m2^m}{\log(e)} \right\rfloor \approx n^{1/d}$.

1. $\lambda := \frac{(\log w)^{3/2}}{w^{5/4}}$
2. $p := 2^{-m}$ and $q := 2^{-m/2}$.
3. t_k for $1 \le k \le d$ are defined recursively: let $t_d := 1/2$, $t_{d-1} := \frac{(1-t_d)^{w_{d-1}}-\lambda}{q} = \frac{p-\lambda}{q}$, and $t_{k-1} := \frac{(1-t_k)^{qw}-\lambda}{q}$ for $1 \le k-1 \le d-2$.

Roughly, t_k corresponds to the probability that a wire in the kth layer is fixed. We choose λ and q so that t_k is always close to q. We will motivate the rest of the definitions later.

Definition 13 (Projection)**.** We consider functions f mapping $|A|$ blocks of length w. Then, a projection merely sets all variables in a single block to be equal.

More formally, consider $f : \{0,1\}^{A\times[w]} \to \{0,1\}$. Then, we define $\mathrm{proj}(f) : \{0,1\}^A \to \{0,1\}$ to be $(\mathrm{proj}(f))(y) = f(x)$ where $x_{a,i} = y_a$ for all a.

For a restriction $\rho : \{x_{a,i}\}_{a\in A, i\in[w]} \to \{0,1,*\}^{A\times[w]}$ of function $f : \{0,1\}^{A\times[w]} \to \{0,1\}$, the projection $\mathrm{proj}_\rho(f)$ first applies the restriction ρ. Then for each block $a \in A$, we replace all unlabeled variables $x_{a,i}$ with a fresh formal variable y_a. Alternatively, $\mathrm{proj}_\rho(f) = \mathrm{proj}(f\restriction\rho)$.

Projections can be thought of as generalizations of the blockwise restrictions used above: instead of limiting the number of $*$'s in each block, we merely make them all equal.

Definition 14 (Adaptive Random Restrictions)**.** For $\tau \in \{0,1\}^{A_k}$ define the distribution $\mathcal{R}(\tau)$ over refinements $\boldsymbol{\rho} \in \{0,1\}^{A_k}$ of τ as follows:

For each block $a \in A_{k-1}$, define $S_a = \tau_a^{-1}(*) \subseteq [w_{k-1}]$ and define $\boldsymbol{\rho}(S_a)$ as the substring of $\boldsymbol{\rho}$ with coordinates in S_a, i.e. the substring of $\boldsymbol{\rho}$ we need to restrict. Without loss of generality assume layer $k-1$ consists of AND gates. Independently for each block $a \in A_{k-1}$:

If $\widehat{\tau}_a = 1$, then the block a in layer k consists of all 1s so there is nothing to set (S_a is empty).

If $S_a \not\approx qw$ or if $\widehat{\tau}_a = 0$ (meaning block a in layer k has at least one 0), then

$$\boldsymbol{\rho}(S_a) \leftarrow \{0_{t_k}, 1_{1-t_k}\}^{S_a}$$

If $S_a \approx qw$ and $\widehat{\tau}_a = *$ in layer $k-1$, then

$$\boldsymbol{\rho}(S_a) \leftarrow \begin{cases} \{1\}^m & \text{with probability } \lambda \\ \{*_{t_k}, 1_{1-t_k}\}^m \ \{1\}^m & \text{with probability } q_a \\ \{0_{t_k}, 1_{1-t_k}\}^m \ \{1\}^m & \text{with probability } 1-\lambda-q_a \end{cases}$$

where q_a is chosen to satisfy $(1-t_k)^{|S_a|} = \lambda + q_a t_{k-1}$.

In general, we should have $|S_a| \approx qw$, which motivates the recursive definition of t_k.

Definition 15 (RST Projections)**.** The sequence of random projections in [RST15] is defined by the random restrictions $\boldsymbol{\rho}^{(d)} \leftarrow \mathcal{R}_{init}$ and $\boldsymbol{\rho}^{(k)} \leftarrow \mathcal{R}(\widehat{\boldsymbol{\rho}^{(k+1)}})$.

Let $\boldsymbol{\Psi} := proj_{\boldsymbol{\rho}^{(2)}} \circ proj_{\boldsymbol{\rho}^{(3)}} \circ ... \circ proj_{\boldsymbol{\rho}^{(d)}}$ denote the composition of all projections.

For the distribution $\mathcal{R}_{init}$ of restrictions applied to the dth layer, $S_a = [w_{d-1}]$ so we set $\boldsymbol{\rho}(S_a)$ according to the last case in Definition 14 recalling that $t_d = 1/2$.

Remark 16. The RST restrictions are adaptive in the sense that the restrictions in layer $k-1$ depend on the outcome (or lift) of the restriction in layer k. This adaptiveness, the choice of q_a, and the definition of $\mathcal{R}_{init}$ permit $\boldsymbol{\Psi}$ to complete to uniform; see Section 5.2.

Definition 17 (Lift)**.** For a restriction $\tau \in \{0,1,*\}^{A_k}$, its lift $\widehat{\tau}$ is the string in $\{0,1,*\}^{A_{k-1}}$ corresponding to setting the gates at level k according to τ.

For example, say layer $k-1$ consist of AND gates. For gate $a \in A_{k-1}$, if $\tau_{a,i} = 0$ for some $i \in w_{k-1}$, then $\widehat{\tau}_a = 0$. If $\tau_a = \{1\}^{w_{k-1}}$, then $\widehat{\tau}_a = 1$. If $\tau_a \in \{*,1\}^{w_{k-1}} \setminus \{1\}^{w_{k-1}}$, then $\widehat{\tau}_a = *$.

5.2 Random projections complete to uniform

In this section we show **Property 4** of Section 2.1. This is the *key property* of the random projections which was difficult to achieve for random restrictions. More formally, we show that the outputs of function $\boldsymbol{\Psi}(\mathsf{Sipser}_d)$ on $\mathbf{Y} \leftarrow \{0_{1-t_1}, 1_{t_1}\}^{w_0} =: \mathcal{D}$ have the same distribution as the outputs of Sipser_d on $\mathbf{X} \leftarrow \{0_{1/2}, 1_{1/2}\}^n$. Then we can essentially simulate the latter by the former.

Recall $\boldsymbol{\Psi}(\mathsf{Sipser}_d)$ is Sipser_d with depth k gates labeled according to $\boldsymbol{\rho}^{(k)}$ for all $k \geq 2$.

Definition 18 (Percolate)**.** To percolate from layer 0 to layer d, label the depth 1 gates according to $\widehat{\rho^{(2)}}$. Fill in any unlabeled gates by sampling independently from $\{0_{1-t_1}, 1_{t_1}\}$. We let the labels of layer 1 "percolate" downwards so that any unlabeled children in the tree receive the same label as their parent. We can similarly define percolating from layer $k-1$ to layer k given a restriction τ for layer k and a label distribution of $\{0_p, 1_{1-p}\}$ for layer $k-1$.

Lemma 19. *We consider the case for k even. Let $\tau \in \{0,1,*\}^{A_k}, \boldsymbol{\rho} \leftarrow \mathcal{R}(\tau)$. Percolate from layer $k-1$ to layer k. Each label at layer k resulting from percolation (not from τ) is distributed according to $\{0_{t_k}, 1_{1-t_k}\}$.*

Proof. This follows from Definition 14 by computation. Recall q_a is chosen so that $(1-t_k)^{|S_a|} = \lambda + q_a t_{k-1}$.

Showing the result for block $a \in A_k$ suffices. Let $S_a = \tau_a^{-1}(*) \subseteq [w_{k-1}]$ be the unlabeled gates in layer k. Let $\boldsymbol{Z}_a \leftarrow \{0,1\}^{S_a}$ be the random variable corresponding to the percolated labels. By Definition 14,

$$\mathbb{P}[\boldsymbol{Z}_a = 1^{S_a}] = \lambda + q_a \,\mathbb{P}[\text{percolated 1}] = \lambda + q_a t_{k-1} = (1-t_k)^{|S_a|}.$$

Now for any $Z \in \{0,1\}^{|S_a|} \setminus \{1\}^{S_a}$,

$$\mathbb{P}[\boldsymbol{Z}_a = Z] = (1-\lambda-q_a)\cdot \frac{t_k^u (1-t_k)^{|S_a|-u}}{1-(1-t_k)^{|S_a|}} + q_a(1-t_{k-1})\cdot \frac{t_k^u(1-t_k)^{|S_a|-u}}{1-(1-t_k)^{|S_a|}} = t_k^u (1-t_k)^{|S_a|-u}.$$

□

The following theorem follows from applying Lemma 19 several times (recall $t_d = 1/2$):

Theorem 20. *If we percolate $\boldsymbol{\Psi}(\mathsf{Sipser}_d)$ from layer 0 to layer d as described in Definition 18, then the labels of the dth layer are distributed uniformly (i.e. according to $\{0_{1/2}, 1_{1/2}\}^n$).*

Corollary 21. $\mathbb{P}[\mathsf{Sipser}_d(\boldsymbol{X}) \neq C(\boldsymbol{X})] = \mathbb{P}[(\boldsymbol{\Psi}(\mathsf{Sipser}_d))(\boldsymbol{Y}) \neq (\boldsymbol{\Psi}(C))(\boldsymbol{Y})]$
for $\boldsymbol{X} \leftarrow \{0_{1/2}, 1_{1/2}\}^n, \boldsymbol{Y} \leftarrow \{0_{1-t_1}, 1_{t_1}\}^{w_0}$.

This follows by applying Theorem 20 and viewing $(\boldsymbol{\Psi}(C))(\boldsymbol{Y})$ as $C(\boldsymbol{\Psi}(\boldsymbol{Y}))$.

The remaining sections show $\boldsymbol{\Psi}(C)$ and $\boldsymbol{\Psi}(\mathsf{Sipser}_d)$ are of certain forms with high probability (taken over $\boldsymbol{\Psi}$) and that these forms do not correlate well over $\mathcal{D}$.

5.3 Sipser remains biased under random projections

In this section we show **Property 2** of Section 2.1. We define typical restrictions to have limited consequences for our function (even under composition) and show the RST restrictions are all typical. This shows $\boldsymbol{\Psi}(\mathsf{Sipser}_d)$ remains biased (i.e. the distribution of outputs is close to $\{0_{1/2}, 1_{1/2}\}$).

Definition 22 (Typical Restriction)**.** $\tau \in \{0,1\}^{A_k}$ is typical if

(1) $|\tau_a^{-1}(*)| \approx qw$ for all $a \in A_{k-1}$ and

(2) $|(\widehat{\tau}_a)^{-1}(*)| \geq w_{k-2} - w^{4/5}$ for all $a \in A_{k-2}$.

The first claim is pretty clear and follows from the fact that the probability of not restricting a subtree is precisely $q_a \approx q$. The second claim merely corresponds to the low probability λ of fixing a subtree entirely at 1s (for an AND level).

Without loss of generality let $k-1$ consist of AND gates. If τ is typical, these conditions imply that inputs to $a \in A_{k-1}$ are not all 1 so layer $k-1$ evaluates to $\{0,*\}^{w_{k-2}} \setminus \{0\}^{w_{k-2}}$. Thus no OR gate in layer $k-2$ has a set value. In general, typical restrictions to set variables at depth k while leaving variables at depth $k-2$ untouched.

Proposition 23. *With respect to RST restrictions:*

(a) $\widehat{\boldsymbol{\rho}^{(d)}}$ *is typical with high probability*

(b) if $\widehat{\boldsymbol{\rho}^{(k)}}$ *is typical then* $\widehat{\boldsymbol{\rho}^{(k-1)}} \leftarrow \mathcal{R}(\widehat{\boldsymbol{\rho}^{(k)}})$ *is typical with high probability.*

For an idea of why (a) holds, note that by definition $\mathbb{P}[\boldsymbol{\rho}_{a,i}^{(d)} = *] = q$ so we expect qw inputs in block a to be $*$ (remain undetermined/unassigned). A Chernoff bound will show that deviation from expectation is small with high probability. For (b), we have a lower bound on the number of $*$ in layer $k-1$ and using the adaptive nature of $\boldsymbol{\rho}^{(k-1)}$ we can bound the expected number of $*$ in any block of $\widehat{\boldsymbol{\rho}^{(k-1)}}$ to be $\approx qw$. Again Chernoff bounds will show that deviation is small with high probability.

Applying $\boldsymbol{\Psi}$ to Sipser_d allows us to reduce the depth of Sipser_d without affecting its output distribution.

Proposition 24. $\boldsymbol{\Psi}(\mathsf{Sipser}_d) \equiv \mathsf{Sipser}_d^{(1)} \restriction \widehat{\boldsymbol{\rho}^{(2)}}$

Proof. Note that $proj\mathsf{Sipser}_d^{(k)} \equiv \mathsf{Sipser}_d^{(k-1)}$ (here we are projecting without restricting). Therefore $proj_{\rho^{(d)}}\mathsf{Sipser}_d^{(d)} \equiv \mathsf{Sipser}_d^{(d-1)} \restriction \widehat{\rho^{(d)}}$.

Note that $proj_{\rho^{(k)}}(\mathsf{Sipser}_d^{(k)} \restriction \widehat{\rho^{(k+1)}}) \equiv proj(\mathsf{Sipser}_d^{(k)} \restriction \rho^{(k)}) \equiv \mathsf{Sipser}_d^{(k-1)} \restriction \widehat{\rho^{(k)}}$ where the first equivalence follows from recalling that $\rho^{(k)}$ refines $\widehat{\rho^{(k+1)}}$. Repeatedly applying this fact on top of $proj_{\rho^{(d)}}\mathsf{Sipser}_d^{(d)} \equiv \mathsf{Sipser}_d^{(d-1)} \restriction \widehat{\rho^{(d)}}$ suffices. □

Theorem 25. *For* $\boldsymbol{Y} \leftarrow \{0_{1-t_1}, 1_{t_1}\}^{w_0}$, $\mathbb{E}_{\boldsymbol{\Psi}}\left[\mathbb{P}_{\boldsymbol{Y}}\left[(\boldsymbol{\Psi}(\mathsf{Sipser}_d))(\boldsymbol{Y}) = 0\right]\right] \approx 1/2$. *Formally, this expected value is bounded between* $1/2 \pm \tilde{O}(w^{-1/12})$.

Proof. With (very) high probability by Proposition 23, $\widehat{\boldsymbol{\rho}^{(d)}}, ..., \widehat{\boldsymbol{\rho}^{(3)}}$ are typical. It suffices to bound $\gamma := \mathbb{P}_{\boldsymbol{Y}}[(\mathsf{Sipser}_d^{(1)} \restriction \widehat{\boldsymbol{\rho}^{(2)}})(\boldsymbol{Y}) = 0]$. By typicality, $|(\widehat{\boldsymbol{\rho}^{(2)}})^{-1}(*)| \approx qw_0$ with high probability. Recall that we assume d is even so the top gate of $\boldsymbol{\Psi}(\mathsf{Sipser}_d)$ is an OR gate. So $\gamma \approx (1-t_1)^{qw_0} \approx 1/2$ by Definition 9. □

5.4 Projected Sipser cannot be computed by simple circuits

In this section we show **Property 3** of Section 2.1.

Theorem 26. *For any width-s CNF $F : \{0,1\}^{w_0} \to \{0,1\}$ and restriction $\tau\{0,*\}^{w_0} \setminus \{0\}^{w_0}$,*

$$\mathbb{P}[(\mathsf{OR} \restriction \tau)(\boldsymbol{Y}) \neq F(\boldsymbol{Y})] \geq \mathbb{P}[(\mathsf{OR} \restriction \tau)(\boldsymbol{Y}) = 0] - st_1$$

for $\boldsymbol{Y} \leftarrow \{0_{1-t_1}, 1_{t_1}\}^{w_0}$.

Proof. Let S be the subset of $[w_0]$ that are not set by τ. Here, let OR refer to the OR of elements of S. There exists ρ, a restriction on variables $[w_0] \setminus S$, such that

$$\mathbb{P}[\mathsf{OR}(\boldsymbol{Y}) \neq F(\boldsymbol{Y})] \geq \mathbb{P}[\mathsf{OR}(\boldsymbol{Y}) \neq F'(\boldsymbol{Y})]$$

where $F' = F \restriction \rho$ (depending only on variables in S). The lower bound holds in either of the following cases:

If for all clauses in F', there is a negated variable, then $F'(0^S) = 1 \neq 0 = \mathsf{OR}(0^S)$. This occurs with probability $\mathbb{P}[(\mathsf{OR} \restriction \tau)(\boldsymbol{Y}) = 0]$.

Else some clause does not contain a negation of a variable. In this case, the probability that $\mathsf{OR}(\boldsymbol{Y})$ and $F'(\boldsymbol{Y})$ do not agree is at least

$$\mathbb{P}[\mathsf{OR}(\boldsymbol{Y}) = 1] - \mathbb{P}[F'(\boldsymbol{Y}) = 1] \geq \mathbb{P}[\mathsf{OR}(\boldsymbol{Y}) = 1] - rt_1$$

(recall all literals of this clause are true with probability $\leq rt_1$ by union bound). □

5.5 Proof of RST's Hierarchy Theorem

We now state the projection switching lemma. This is used to show the RST projections satisfy **Property 1** of Section 2.1, namely that approximating circuits reduce to decision trees with high probability. The proof is once again similar to the previous switching lemmas, and is breifly discussed in Section 6.

Theorem 27. *Consider a width-t DNF (or CNF) $F : \{0,1\}^{A_k} \to \{0,1\}$ (each clause has at most t variables), and consider a random restriction $\rho \leftarrow \mathcal{R}(\tau)$. Then, for sufficiently small constants p, with probability at least $1 - O(re^{rt_k/(1-t_k)} \cdot w^{-1/4})^s$, $F' = F \restriction \rho$ can be written as a decision tree of depth at most s. Note that this decision tree can be converted to either a width-s CNF or DNF.*

We now have **Properties 1-4** (specifically results 21, 24, 25, and 26) for Sipser_d, so we can conclude that Sipser_d cannot be approximated by bottom fan-in depth d circuits with small bottom fan-in (Theorem 4).

6 Proofs of Hastad's Switching Lemma

We now present a proof of Hastad's switching lemma (Theorem 7) for the original random restrictions. The same proof techniques generalize fairly easily to switching lemmas for our restrictions and projections.

Recall from the statement of the lemma that we have width-t DNF F and a random restriction $\rho \leftarrow \mathcal{R}_p$. Then, we would like to construct a depth-s decision tree for $F' = F\restriction\rho$ with probability at least $1 - (pt)^{O(s)}$.

Hastad's original proof of this statement is fairly complex and involves computing various conditional probabilities. Instead, we present a proof method by Rasborov.

Proof of switching lemma. Our proof will explicitly exhibit a "canonical decision tree" which computes F', and we will bound the probability that ρ is a "bad" restriction, i.e. that the canonical decision tree has depth at least s.

Definition 28 (Canonical decision tree)**.** We recursively construct a decision tree for a DNF G. First, if G is identically 0 or 1, just output G. Otherwise, consider first clause T in G which is not guaranteed to be either 0 or 1 (we impose some arbitrary canonical order). Then, condition on all unset variables x_i in T as the first layers of the decision tree, and recursively build the rest.

Clearly, this canonical decision tree computes G.

Our proof will now use the structure of the decision tree to make an injective mapping (or encoding) from the set of bad restrictions to $\{0,1,*\}^A \times \{0,1\}^{O(s\log(t))}$.

The way this proof is generally phrased is that, for $\rho \to (\rho', enc)$, the probability (under $\mathcal{R}_p$) of ρ' is at least $p^{-O(s)}$ times that of ρ, so taking a union bound over strings enc, the probability of a bad restriction is at most

$$\sum_{enc}\sum_{\rho'} \mathop{\mathbb{P}}_{\mathcal{R}_p}[\rho']p^{O(s)} = 2^{O(s\log(t))} \cdot 1 \cdot p^{O(s)} = (pt)^{O(s)}$$

Remark. This view of the proof is slightly flawed: ρ' is really only a restriction in the sense that it lives in $\{0,1,*\}^A$, and is in fact used to "hide" information just like enc. As such, considering the probability of ρ' under $\mathcal{R}_p$ is misleading, as ρ' does not represent a possible restriction at all; the probability is just a useful measure which sums to 1.

Unfortunately, removing this from the proof results in a significantly more tedious proof, so we will not attempt this. Instead, keep the following in mind as we proceed:

- We could replace the $\mathbb{P}_{\mathcal{R}_p}[\rho']$ with many other functions, but this particular probability matches the exponential nature of $\mathbb{P}_{\mathcal{R}_p}[\rho]$ quite well.
- The key part of this proof is changing the order of summation and bounding the outer sum (part of the "extra information") with only $2^{O(s\log(t))}$ possibilities, while hiding the remaining information in the gaps of ρ itself.

In this proof, we treat the negations of variables as formal variables themselves. Thus, if $x = \neg y$ appears in clause T, we will treat setting $x = 1$ as setting $y = 0$.

We now specify our encoding. Consider any bad encoding ρ. Then, there is some path in the canonical decision tree with depth at least s (setting at least s variables). Let the path correspond to setting variables $x_{v_1}, \ldots, x_{v_s}$ to be $\pi_1, \ldots, \pi_s$. Our encoding consists of the following procedure:

- Initialize $\rho' = \rho$, and keep an encoding string.
- For each node i in the tree, consider F_i', defined as F' restricted up to that point in the decision tree. Let x_{v_i} be the kth term in clause T. By construction of the decision tree, at this point, no terms are entirely true, and T is the first term without any variables set to false.

  Then, set $\rho'(x_{v_i}) = 1$, and add π_i and k_i to the encoding (with $1 + \log(t)$ bits).

It remains to show this is injective. We prove this by showing a procedure for inverting this encoding. We invert the process from iteration 1 to s.

- Initialize $\rho = \rho'$. At any point, ρ will essentially match the restriction for F_i, with some additional bits set from ρ'.
- First, note that our procedure only sets values in $\rho'(x_i)$ to be true, and only sets them in left-to-right order. Thus, the first term T in $F \restriction \rho$ which is has no variable set to false is actually the term in F_i' which contains x_{v_1}.

  Thus, given the position k_i of x_{v_i} in T, we can recover v_i. Finally, we can replace $\rho(x_{v_i}) = \pi_i$ to descend into the next level.

Thus, we are done, as ρ' has exactly s more bits set than ρ, so the probability is higher by the at least $\left(\frac{1-p}{2p}\right)^s = p^{-O(s)}$ for sufficiently small p. □

Some final notes: our encoding essentially cleverly hid the locations of the variables set on the path by breaking symmetry in the restriction (setting $\rho'(x) = 1$), and then storing less information than previously necessary by the structure of bounded-width gates.

We conclude this section with a brief discussion of how to generalize this proof to random projections.

This proof method generalizes well to random projections. Our decision tree structure is modified to set the new projected variables y_a, and we need some additional bits in the encoding to tell us which of the $x_{a,i}$ were fixed in the projection and which were only fixed in the decision tree. Overall, the structure of the bounds remains essentially the same.

7 From circuits to oracles

We now discuss and formalize the connections between circuits and oracles developed in [FSS84], following the explanations in [Hås86]. We show how to translate both worst-case (incomputability) and average-case (inapproximability) circuit hierarchy theorems to results in relativized worlds.

Recall the characterization of Σ_k^P alternating Turning machines as circuits. Define the *alternation depth* of a circuit to be 1 plus the maximal number of switches between AND and OR gates along any path. A machine is in Σ_k^P if it has alternation depth at most k. We can merge layers with the same gate type so that the depth of the circuit equals the alternation depth.

7.1 Worst-case bounds construction

Armed with several circuit lower bounds, we show that there *exists* relativized worlds where there are separations between PSPACE and PH and between levels of the PH. We begin by introducing the idea of a weak oracle Turing machine to simplify the form of our circuits.

Definition 29. A weak oracle machine is allowed to query its oracle once at the end of each computation branch.

Lemma 30. *For any oracle A and any machine $M^A \in \Sigma_k^{P,A}$, M^A decides the same language as some weak oracle machine $M_1^A \in \Sigma_{k+2}^{P,A}$.*

Proof. M_1^A has similar structure as M^A except whenever M^A queries, M_1^A guesses the answer to the query by branching out into two branches. It checks all guesses made along a path using its extra alternations at the end. For instance, if M_A makes query Q_i at a AND gate which is the root of subcircuit C. M_1^A creates two wires at this gate connected to copies of C. It guesses Q_i; it assumes the answer to the query Q_i was 0 along one branch and 1 along the other. The OR gate case is similar.

Then M_1^A uses its extra alternations as follows:
M_1^A accepts iff $(M^A \text{accepts} \wedge \forall i, Q_i \text{ is correct}) \vee (\exists i, Q_i \text{ is incorrect} \wedge \forall j < i, Q_j \text{ is correct})$. Casework on the type of gate whether the first incorrect guess was made shows that M_1^A rejects if M_1 rejects. If M_1 accepts on some branch, then M_1^A will guess correctly on an analogous branch and accept. □

With respect to oracle A, define Boolean variables y_z^A for all strings z such that y_z^A has value 1 iff $z \in A$. Note that any weak oracle machine M^A of alternation depth k and (normal) depth t on any input x corresponds to a circuit C of depth k and size 2^t taking a subset of y_z^A as inputs. x becomes hardwired into C and any computation in M_A depending on a query is replaced with the appropriate y_z^A or $\overline{y_z^A}$. In other words, y_z^A is what we answer when A is queried on z.

We use diagonalization to construct an oracle A such that all weak oracle machines do not compute a language in PSPACE.

Theorem 31. *[FSS84]. There exists an oracle A such that $\mathsf{PSPACE}^A \neq \mathsf{PH}^A$.*

Proof. For any oracle A, the language

$$L(A) := \{1^n | A \text{ contains an odd number of length } n \text{ strings}\}$$

is decidable in PSPACE^A since we can query for all $y_z^A, |z| = n$ and directly compute $\bigoplus_{|z|=n} y_z^A$.

We will construct A in rounds such that in each round, we ensure another machine in PH^A cannot decide $L(A)$. By Lemma 30, it suffices to ensure all weak oracle machines in PH^A cannot decide $L(A)$. There are countably such machines because there are countably many machines in $\Sigma_k^{P,A}$ and $\mathbb{Z}^2$ is countable. Enumerate all of these machines as $M_1^A, M_2^A, ...$ Let $n_0 = 1$. Say M_i^A has runtime cn^c. Find $m_i > n_{i-1}$ such that $cm_i^c \ll 2^{c_0 m_i/d}$ and consider the circuit C of M_i^A taking size m_i inputs. This circuit must depend on all $y_z^A, |z| = m_i$ or else we can trivially set y_z^A so that C incorrectly computes $\bigoplus_{|z|=n} y_z^A$. The circuit may depend on y_z^A for $|z| \neq m_i$. For these, hard-wire any previously set y_z^A and set any remaining y_z^A to 0. By Theorem 1, our depth k circuit on 2^{m_i} inputs has size at most $\exp^{cm_i^c} \ll 2^{2^{c_0 m_i/d}}$ so it cannot compute PARITY on all inputs. Thus there exists some setting of $y_z^A, |z| = m_i$ so that our circuit fails, so set $y_z^A, |z| = m_i$ to these bits. Let n_i be the maximum length z such that y_z^A has been set so far. Set any unset $y_z^A, |z| \leq n_i$ to 0 and continue with round $i + 1$.

On round i, we force M_i^A to be incorrect on 1^{m_i}. This process sets y_z^A for all z and since the sequence of m_i is increasing, our values for y_z^A never conflict. Therefore oracle A is well-defined. □

Similarly, the proof showing a separation between relativized levels of the PH follows from defining an appropriate $L(A)$ and applying Theorem 4 in each round.

Theorem 32. *[Sip83, Yao85]. There exists an oracle A such that $\Sigma_{k-1}^{P,A} \neq \Sigma_k^{P,A}$ for all k.*

Proof. It suffices to construct A such that $\Sigma_k^{P,A} \neq \Sigma_{k-3}^{P,A}$ for all $k > 3$ because this implies $\Sigma_k^{P,A} \neq \Sigma_{k-1}^{P,A}$ for all k. This is because if the latter were not true for some k, then PH^A collapses to $\Sigma_{k-1}^{P,A}$. By Lemma 30, any $\Sigma_{k-3}^{P,A}$ machine is equivalent to a weak $\Sigma_{k-1}^{P,A}$ oracle machine. Thus it suffices to separate $\Sigma_k^{P,A}$ from all weak oracle machines in $\Sigma_{k-1}^{P,A}$.

For any A and any k, let $t = n/k$ and define language

$$L_k(A) := \{1^n | \exists x_1 ... x_t \forall x_{t+1} ... x_{2t} \exists ... x_n \text{ such that } x_i \in A\}$$

This is computable by a $\Sigma_k^{P,A}$ machine. As a function of y_z^A, it is at least as hard to compute as Sipser_k.

Again, we can use a diagonalization argument to determine y_z^A in rounds. Enumerate all weak oracle machines in PH^A as $M_1^A, M_2^A, \ldots$. On round i, say M_i^A is a $\Sigma_{k-1}^{P,A}$ weak oracle machine. For an appropriate m_i, the computation of M_i^A on 1^{m_i} is a small, bounded fan-in, depth $k-1$ circuit taking y_z^A's as inputs. By Theorem 3, no such circuit can compute Sipser_k (or $L_k(A)$) on all inputs. As before, this gives us a setting of some y_z^A in each round. □

7.2 Average-case bounds construction

It's natural to ask whether these separations hold for the *average* oracle. We focus on the proof for separation of levels of the PH in Theorem 35. In fact, this separation holds relative to a random oracle with probability 1.

Remark 33. Formally, any subset of random oracles can be given a measure from $[0, 1]$ and we show that the subset of oracles for which the separation holds has measure 1.

Theorem 34. *[Cai86, Hås86] For a random oracle A, $\mathsf{PSPACE}^A \neq \mathsf{PH}^A$ with probability 1.*

Theorem 35. *For a random oracle A, $\Sigma_{k-1}^{P,A} \neq \Sigma_k^{P,A}$ for all k with probability 1.*

Proof. Recall the definition of $L_k(A)$, a language in $\Sigma_k^{P,A}$ for any A. We show that for random oracles A, any weak $\Sigma_{k-1}^{P,A}$ oracle machine computes $L_k(A)$ with probability 0. We can consider this class of machines by the same explanation given for Theorem 32, and we can compute this probability for one such machine because there are countably many.

Consider M^A, a weak $\Sigma_{k-1}^{P,A}$ oracle machine with runtime cm^c. Let m_1 be a constant such that $cm_1^c \ll 2^{m_1}$. Define the sequence m_i recursively: $m_i = cm_{i-1}^c + 1$. Note that if $cm_j^c \ll 2^{m_j}$, then $cm_{j+1}^c \ll c2^{cm_j} \ll 2^{m_{j+1}}$.

Lemma 36. *For any i, the probability that M^A agrees with $L_k(A)$ on input 1^{m_i} given that it agrees on 1^{m_j} for $j < i$ is at most $1/2 + \epsilon$.*

Proof of Lemma 36. This probability is taken over random settings of y_z^A for $|z| = m_i$ since whether a random oracle contains strings s_1, s_2 are independent events. In particular, by choice of m_i, M^A does not have enough runtime to query about $y_z^A, |z| = m_i$ when trying to decide 1^{m_j} for $j < i$ so the space we take our probability over is "independent" of the values of y_z^A for $|z| < m_i$. Note that M^A's computation on input 1^{m_i} is a depth $k-1$ circuit which must depend on 2^{m_i} inputs. This circuit has size at most $2^{cm_i^c} \ll 2^{2^{m_i}}$. Since depth is being kept constant, the lemma follows from Theorem 4 by viewing this circuit as a depth k circuit with bottom fan-in 1. □

Therefore, the probability that M^A agrees with $L_k(A)$ on all inputs is 0, concluding the proof of Theorem 35. □

8 Conclusion

Our survey presents several worst-case and average-case circuit lower bounds and highlights the underlying ideas used in their proofs. We then translate these results into separations of relativized complexity classes for a "worst-case" oracle and for an "average-case" (random) oracle. Such a connection was realized in [FSS84] but it remained open to show an average case depth hierarchy theorem until 2015 when [RST15] solved the last piece of the puzzle.

The idea underlying all of these circuit bounds is the restriction, a randomized setting of gates circuit gates. Repeated use of restrictions has the effect of simplifying sufficiently small and shallow circuits with high probability while allowing delicate functions to remain structured. In particular, the former results from switching lemmas showing that we can turn CNFs into DNFs (or vice versa) and essentially absorb the bottom layer of gates into the layer above it. The latter relies on choosing the right functions and choosing the right restrictions to preserve these functions. As discussed, PARITY and Sipser are delicate in the sense that their outputs are sensitive to small changes in their inputs. Showing that the restricted forms of such circuits and functions disagree/do not correlate suffices to conclude results about incomputability/inapproximability of these functions by circuits. The well-known "standard" restrictions are enough to show incomputability of PARITY. To show certain circuits cannot compute Sipser, Hastad proposed more complicated blockwise restrictions that leave variables fixed in subcircuits of Sipser. These restrictions are correlated amongst subcircuits so they could not extend to inapproximability. Hence, in order to get a result about inapproximability of Sipser, RST proposed a generalization of blockwise restrictions called projections which leave more variables unfixed and are adaptive. As we saw, the projections are defined in such a way that the Sipser function and circuits both simplify and over a distribution $\mathcal{D}$ of inputs, these simplified forms do not correlate well. This lack of correlation over $\mathcal{D}$ then implies lack of correlation for the original function and circuit over the original space of inputs.

We show the connection between circuit bounds and relativized complexity by viewing oracle Turing machines as circuits computing PARITY or Sipser. Here the input variables are the answers to oracle queries. By our circuit bounds, only the larger/deeper circuits, and hence the larger relativized complexity classes, can compute these functions.

Restrictions are a general method for obtaining lower bounds. In particular, they have been applied to obtain size bounds on threshold circuits and Nick's circuits. It would be interesting to see whether projections would be useful in these or other areas and what additional implications circuits bounds may have for relativized complexity.

References

[Bar15] Boaz Barak. Average case depth hierarchy for $\wedge, \vee, \neg$-boolean circuits. Guest lecture at MIT, 2015.

[Cai86] Jin-Yi Cai. With probability one, a random oracle separates pspace from the polynomial-time hierarchy. *Journal of Computer and System Sciences*, 38(1):68–85, 1986.

[FSS84] Merrick Furst, James B Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984.

[Hås86] Johan Håstad. *Computational Limitations of Small-depth Circuits*. MIT Press, Cambridge, MA, USA, 1986.

[RST15] Benjamin Rossman, Rocco A Servedio, and Li-Yang Tan. An average-case depth hierarchy theorem for boolean circuits. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 1030–1048. IEEE, 2015.

[Sip83] Michael Sipser. Borel sets and circuit complexity. In *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing*, pages 61–69, 1983.

[Yao85] Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles. In *Foundations of Computer Science, 1985., 26th Annual Symposium on*, pages 1–10. IEEE, 1985.