Adaptivity vs. Postselection, and Hardness Amplification for Polynomial Approximation*

Lijie Chen

Tsinghua University, Beijing, China
wjmzbmr@gmail.com

Abstract

We study the following problem: with the power of postselection (classically or quantumly), what is your ability to answer adaptive queries to certain languages? More specifically, for what kind of computational classes $\mathcal{C}$, we have $\mathsf{P}^{\mathcal{C}}$ belongs to $\mathsf{PostBPP}$ or $\mathsf{PostBQP}$? While a complete answer to the above question seems impossible given the development of present computational complexity theory. We study the analogous question in *query complexity*, which sheds light on the limitation of *relativized* methods (the relativization barrier) to the above question.

Informally, we show that, for a partial function f, if there is no efficient[1] *small bounded-error* algorithm for f classically or quantumly, then there is no efficient *postselection bounded-error* algorithm to answer adaptive queries to f classically or quantumly. Our results imply a new proof for the classical oracle separation $\mathsf{P}^{\mathsf{NP}^{\mathcal{O}}} \not\subset \mathsf{PP}^{\mathcal{O}}$, which is arguably more elegant. They also lead to a new oracle separation $\mathsf{P}^{\mathsf{SZK}^{\mathcal{O}}} \not\subset \mathsf{PP}^{\mathcal{O}}$, which is close to an oracle separation between SZK and PP – an open problem in the field of oracle separations.

Our result also implies a hardness amplification construction for polynomial approximation: given a function f on n bits, we construct an adaptive-version of f, denoted by F, on $O(m \cdot n)$ bits, such that if f requires large degree to approximate to error $2/3$ in a certain one-sided sense, then F requires large degree to approximate even to error $1/2 - 2^{-m}$. Our construction achieves the same amplification in the work of Thaler (ICALP, 2016), by composing a function with $O(\log n)$ *deterministic query complexity*, which is in sharp contrast to all the previous results where the composing *amplifiers* are all hard functions in a certain sense.

1998 ACM Subject Classification F.1.1 Models of Computation

Keywords and phrases approximate degree, postselection, hardness amplification, adaptivity

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2016.26

1 Introduction

1.1 Background

The idea of postselection has been surprisingly fruitful in theoretical computer science and quantum computing [3, 11, 6]. Philosophically, it addresses the following question: if you believe in the Many-worlds interpretation[2] and can condition on a rare event (implemented by killing yourself after observing the undesired outcomes), then what would you be able to compute in a reasonable amount of time? The complexity classes $\mathsf{PostBPP}$ [12] and $\mathsf{PostBQP}$ [1] are defined to represent the computational problems you can solve with the ability of postselection in a classical world or a quantum world.

* The full version is available at http://arxiv.org/abs/1606.04016.
[1] In the world of query complexity, being efficient means using $O(\text{polylog}(n))$ time.
[2] https://en.wikipedia.org/wiki/Many-worlds_interpretation

However, even with that seemingly omnipotent power of postselection, your computational power is still bounded. It is known that $\mathsf{PostBPP} \subseteq \mathsf{PH}$ [12], and (surprisingly) $\mathsf{PostBQP} = \mathsf{PP}$ [1]. Hence, it seems quite plausible that even with the postselection power, you are still not able to solve a PSPACE-complete problem, as it is widely believed that PH and PP are strictly contained in PSPACE.

Another more non-trivial (and perhaps unexpected) weakness of those postselection computation classes, is their inability to simulate *adaptive queries* to certain languages. For example, it is known that $\mathsf{P}^{\mathsf{NP}[O(\log n)]}$[3] is contained in $\mathsf{PostBPP}$ [12], and this result relativizes. But there is an oracle separation between $\mathsf{P}^{\mathsf{NP}[\omega(\log n)]}$ and $\mathsf{PostBQP}$ [4]. In other words, there is no relativized $\mathsf{PostBQP}$ algorithm that can simulate $\omega(\log n)$ adaptive queries to a certain language in NP. In contrast, we know that $\mathsf{P}^{\|\mathsf{NP}} \subseteq \mathsf{PostBPP} \subseteq \mathsf{PP}$ [12], hence they are capable of simulating *non-adaptive* queries to NP.

Then a natural question follows:

▶ **Question 1.1.** *What is the limit of the abilities of these postselection classes on simulating adaptive queries to certain languages? More specifically, is there any characterization of the complexity class $\mathcal{C}$ such that $\mathsf{P}^{\mathcal{C}}$ is contained in PostBPP or PostBQP?*

Arguably, a complete answer to this problem seems not possible at the present time: even determining whether $\mathsf{P}^{\mathsf{NP}} \subseteq \mathsf{PP}$ is already extremely hard, as showing $\mathsf{P}^{\mathsf{NP}} \subseteq \mathsf{PP}$ probably requires some new non-relativized techniques, and proving $\mathsf{P}^{\mathsf{NP}} \not\subset \mathsf{PP}$ implies $\mathsf{PH} \not\subset \mathsf{PP}$, which is a long-standing open problem.

1.2 Relativization and the analogous question in query complexity

So in this paper, inspired by the oracle separation in [4], we study this problem from a relativization point of view. *Relativization*, or *oracle separations* are ultimately about the *query complexity*. Given a complexity class $\mathcal{C}$, there is a canonical way to define its analogue in query complexity: partial functions which are computable by a *non-uniform* $\mathcal{C}$ machine with polylog(n) queries to the input. For convenience, we will use $\mathcal{C}^{\mathsf{dt}}$ to denote the query complexity version of $\mathcal{C}$. We adopt the convention that $\mathcal{C}^{\mathsf{dt}}$ denotes the query analogue of $\mathcal{C}$, while $\mathcal{C}^{\mathsf{dt}}(f)$ denotes the $\mathcal{C}^{\mathsf{dt}}$ complexity of the partial function f.

For a partial function f, we use $\mathsf{len}(f)$ to denote its input length. We say a family of partial functions $\mathbf{f} \in \mathcal{C}^{\mathsf{dt}}$, if $\mathcal{C}^{\mathsf{dt}}(f) = O(\text{polylog}(\mathsf{len}(f)))$ for all $f \in \mathbf{f}$.

In order to study this question in the query complexity setting, given a partial function f, we need to define its adaptive version.

▶ **Definition 1.2** (Adaptive Construction)**.** Given a function $f : D \to \{0,1\}$ with $D \subseteq \{0,1\}^M$ and an integer d, we define $\mathsf{Ada}_{f,d}$, its depth d adaptive version, as follows:

$$\mathsf{Ada}_{f,0} := f \quad \text{and} \quad \begin{array}{c} \mathsf{Ada}_{f,d} : D \times D_{d-1} \times D_{d-1} \to \{0,1\} \\ \mathsf{Ada}_{f,d}(w,x,y) := \begin{cases} \mathsf{Ada}_{f,d-1}(x) & f(w) = 0 \\ \mathsf{Ada}_{f,d-1}(y) & f(w) = 1 \end{cases} \end{array}$$

where D_{d-1} denotes the domain of $\mathsf{Ada}_{f,d-1}$.

The input to $\mathsf{Ada}_{f,d}$ can be encoded as a string of length $(2^{d+1}-1) \cdot M$. Thus, $\mathsf{Ada}_{f,d}$ is a partial function from $D^{(2^{d+1}-1)} \to \{0,1\}$.

[3] $O(\log n)$ stands for the P algorithm can only make $O(\log n)$ queries to the oracle.

Then, given a family of partial function $\mathbf{f}$, we define $\mathsf{Ada}_{\mathbf{f}} := \{\mathsf{Ada}_{f,d} \mid f \in \mathbf{f}, d \in \mathbb{N}\}$.

Notice that when you have the ability to adaptively solve $d+1$ queries to f (or with high probability), then it is easy to solve $\mathsf{Ada}_{f,d}$. Conversely, in order to solve $\mathsf{Ada}_{f,d}$, you need to be able to adaptively answer $d+1$ questions to f, as even *knowing what is the right i^{th} question to answer* requires you to correctly answer all the previous $i-1$ questions.

Now, everything is ready for us to state the analogous question in query complexity.

▶ **Question 1.3.** *What is the characterization of the partial functions family* $\mathbf{f}$ *such that* $\mathsf{Ada}_{\mathbf{f}} \in \mathsf{PostBPP}^{\mathsf{dt}}$ *(*$\mathsf{PostBQP}^{\mathsf{dt}}$*)?*

There are at least two reasons to study Question 1.3. First, it is an interesting question itself in query complexity. Second, an answer to Question 1.3 also completely characterizes the limitation on the *relativized techniques* for answering Question 1.1, i.e., the limitation of relativized methods for simulating *adaptive queries* to certain complexity classes with the power of postselection.

This paper provides some interesting results toward resolving Question 1.3.

1.3 Our results

Despite that we are not able to give a complete answer to Question 1.3. We provide some interesting lower bounds showing that certain functions' adaptive versions are hard for these postselection classes.

Formally, we prove the following two theorems.

▶ **Theorem 1.4** (Quantum Case). *For a family of partial function* $\mathbf{f}$*,* $\mathsf{Ada}_{\mathbf{f}} \notin \mathsf{PostBQP}^{\mathsf{dt}}(\mathsf{PP}^{\mathsf{dt}})$ *if* $\mathbf{f} \notin \mathsf{SBQP}^{\mathsf{dt}} \cap \mathsf{coSBQP}^{\mathsf{dt}}$*.*

▶ **Theorem 1.5** (Classical Case). *For a family of partial function* $\mathbf{f}$*,* $\mathsf{Ada}_{\mathbf{f}} \notin \mathsf{PostBPP}^{\mathsf{dt}}$ *if* $\mathbf{f} \notin \mathsf{SBP}^{\mathsf{dt}} \cap \mathsf{coSBP}^{\mathsf{dt}}$*.*

Roughly speaking, SBP is a relaxation of BPP, it is the set of languages L such that there exists a BPP machine M, which accepts x with probability $\geq 2\alpha$ if $x \in L$; and with probability $\leq \alpha$ if $x \notin L$ for a positive real number α. And SBQP is the quantum analogue of SBP, where you are allowed to use a polynomial time quantum algorithm instead.[4]

Our theorems show that, for a partial function f, if there is no efficient classical (quantum) algorithm which accepts all the 1-inputs with a slightly better chance than all the 0-inputs, then there is no efficient PostBPP (PostBQP) algorithm that can answer adaptive queries to f.

In fact, we prove the following two quantitatively tighter theorems, from which Theorem 1.4 and Theorem 1.5 follows easily.

▶ **Theorem 1.6.** *Let* f *be a partial function and* T *be a non-negative integer. Suppose* $\widehat{\deg}_+(f) > T$ *or* $\widehat{\deg}_-(f) > T$*, then we have*

$$\mathsf{PP}^{\mathsf{dt}}(\mathsf{Ada}_{f,d}) > \min(T/4, 2^{d-1}).^5$$

▶ **Theorem 1.7.** *Let* $f : D \to \{0,1\}$ *with* $D \subseteq \{0,1\}^M$ *be a partial function and* d *be a non-negative integer. Suppose* $\mathsf{SBP}^{\mathsf{dt}}(f) > T$ *or* $\mathsf{coSBP}^{\mathsf{dt}}(f) > T$*, then we have*

$$\mathsf{PostBPP}^{\mathsf{dt}}(\mathsf{Ada}_{f,d}) > \min(T/5, (2^d - 1)/5).$$

[4] For the formal definitions of SBP, PostBPP, PostBQP, SBQP and their equivalents in query complexity, see the preliminaries.

1.4 Applications in oracle separations

Our results have several applications in oracle separations.

- A new proof for $\mathsf{P}^{\mathsf{NP}^{\mathcal{O}}} \not\subset \mathsf{PP}^{\mathcal{O}}$:
  We prove that $\mathsf{SBQP}^{\mathsf{dt}}(f)$ is indeed equivalent to *one-sided low-weight approximate degree*, denoted by $\widetilde{\deg}_+(f)$ (cf. Definition 2.8), which is lower bounded by one-sided approximate degree $\deg_+(f)$ (cf. Definition 1.8).
  Using the fact that $\deg_+(\mathsf{AND}_n) \geq \Omega(\sqrt{n})$, Theorem 1.4 implies that $\mathsf{Ada}_{\mathsf{AND}} \not\subset \mathsf{PP}^{\mathsf{dt}}$, yielding a simpler proof for the classical oracle separation between P^{NP} and PP in [4].
  Our proof is arguably simpler and more elegant. Also, unlike the seemingly artificial problem ODD-MAX-BIT[6] in [4], $\mathsf{Ada}_{\mathsf{AND}}$ looks like a more natural hard problem in P^{NP}.
- The new oracle separation $\mathsf{P}^{\mathsf{SZK}^{\mathcal{O}}} \not\subset \mathsf{PP}^{\mathcal{O}}$:
  Since the *Permutation Testing Problem*, denoted by PTP_n (see Problem 2.12 for a formal definition), satisfies $\deg_+(\mathsf{PTP}_n) \geq \Omega(n^{1/3})$ and has a $\log(n)$-time SZK protocol. Theorem 1.4 implies that $\mathsf{Ada}_{\mathsf{PTP}} \not\subset \mathsf{PP}^{\mathsf{dt}}$, which in turn shows an oracle separation between $\mathsf{P}^{\mathsf{SZK}}$ and PP.
  It has been an open problem [2] that whether there exists an oracle separation between SZK and PP, our result is pretty close to an affirmative answer to that.[7]
  Also, note that $\mathsf{P}^{\mathsf{SZK}} \subseteq \mathsf{P}^{\mathsf{AM}\cap\mathsf{coAM}} = \mathsf{AM} \cap \mathsf{coAM}$, so our result improves on the oracle separation between $\mathsf{AM} \cap \mathsf{coAM}$ and PP by Vereschchagin [18].

1.5 Applications in hardness amplification for polynomial approximation

Our construction also leads to a hardness amplification theorem for polynomial approximation. In order to state our result, we need to introduce the definition of two approximate degrees first.

▶ **Definition 1.8.** The ϵ-approximate degree of a partial function of $f : D \to \{0,1\}$, denoted as $\widetilde{\deg}_\epsilon(f)$, is the least degree of a real polynomial p such that $|p(x) - f(x)| \leq \epsilon$ when $x \in D$, and $|p(x)| \leq 1 + \epsilon$ when $x \notin D$.

We say a polynomial p one-sided ϵ-approximates a partial Boolean function f, if $p(x) \in [0, \epsilon]$ when $f(x) = 0$, and $p(x) \geq 1$ when $f(x) = 1$.[8] Then the one-sided ϵ-approximate degree of a partial function f, denoted by $\deg_+^\epsilon(f)$, is the minimum degree of a polynomial one-sided ϵ-approximating f.

Now we are in a position to state our amplification theorem.

▶ **Theorem 1.9.** *Let f be a partial function such that $\deg_+^{2/3}(f) > T$ and d be a positive integer, we have $\widetilde{\deg}_\epsilon(\mathsf{Ada}_{f,d}) > T$ for $\epsilon = 0.5 - 2^{-2^d+1}$.*

That is, given a function with high one-sided approximate degree for an error constant bounded away from 1, it can be transformed to a function with high approximate degree even for ϵ *doubly exponentially* close to $1/2$ in d.[9]

---

[6] Given a binary input x, it asks whether the rightest 1 in x is in an odd position.

[7] Partially inspired by this work, an oracle separation between SZK and PP (in fact, UPP) has been constructed in a very recent work of Bouland, Chen, Holden, Thaler and Vasudevan [5], thus resolved this open problem.

[8] Our definition of one-sided approximation is slightly different from the standard one [15, 8, 16], but it greatly simplifies several discussions in our paper, and they are clearly equivalent up to a linear transformation in ϵ.

[9] Which is *single exponential* in the input length of the *amplifier* AdaQ, see the discussion below.

Comparison with previous amplification results

There have been a lot of research interest in hardness amplification for polynomial approximation, many amplification results are achieved through *function composition* [9, 15, 17]. We use $f \circ g$ to denote the block composition of f and g, i.e. $f(g, g, \ldots, g)$.

Our result can also be viewed as one of them. Let $\mathsf{AdaQ}_d := \mathsf{Ada}_{\mathsf{id},d}$, where id is just the identity function from $\{0,1\}$ to $\{0,1\}$. Then we can see that in fact $\mathsf{Ada}_{f,d}$ is equivalent to $\mathsf{AdaQ}_d \circ f$. Let $n = 2^{d+1} - 1$, which is the input length of AdaQ_d.

However, all the previous amplification results are achieved by letting the *amplifier* f to be a *hard* function. We list all these results for an easy comparison.

- In the work of Bun and Tahler [9], they showed that for a function g such that $\deg_+(g) > T$, $\widetilde{\deg}_\epsilon(\mathsf{OR}_n \circ g) > T$ for $\epsilon = 1/2 - 2^{-\Omega(n)}$. This is further improved by Sherstov [15] to that $\deg_\pm(\mathsf{OR}_n \circ g) = \Omega(\min(n, T))$. Here, the amplifier OR_n is a hard function in the sense that $\deg_+(\mathsf{OR}_n) \geq \Omega(\sqrt{n})$ [14].
- In [17], Thaler showed that for a function g such that $\deg_+(g) > T$, $\widetilde{\deg}_\epsilon(\mathsf{ODD\text{-}MAX\text{-}BIT}_n \circ g) > T$ for $\epsilon = 1/2 - 2^{-\Omega(n)}$.[10] In this case, the amplifier $\mathsf{ODD\text{-}MAX\text{-}BIT}_n$ is even harder in the sense that it has a $\mathsf{PP}^{\mathsf{dt}}$ query complexity of $\Omega(\sqrt[3]{n})$ [4].
- Moreover, it is easy to see that the *randomized query complexity* of both OR_n and $\mathsf{ODD\text{-}MAX\text{-}BIT}_n$ is the maximum possible $\Omega(n)$.

In contrast, our amplifier AdaQ, is *extremely* simple – it has a *deterministic query complexity* of $O(\log n)$![11]

This is a rather surprising feature of our result. That means AdaQ also has an *exact degree* of $O(\log n)$. Intuitively, composing with such a simple and innocent function seems would not affect the hardness of the resulting function. Our result severely contradicts this intuition. But from the view point of Theorem 1.4, composing with AdaQ indeed "adaptivize" the function, makes it hard for $\mathsf{PostBQP}$ algorithms, which is in turn closely connected to PP algorithms and therefore polynomial approximate degree. So this result is arguably natural under that perspective, which illustrates a recurring theme in TCS: a new perspective can lead to some unexpected results.

1.6 Paper organization

In Section 2 we introduce some preliminaries, due to the space constraints, some of the formal definitions of those partial function classes in query complexity can be found in the full version. We prove Theorem 1.4 and Theorem 1.6 in Section 3, and defer the proof for Theorem 1.5 and Theorem 1.7 to the full version. Theorem 1.9 is proved in Section 3.4. And we provide formal proofs for the two oracle separation results in the full version.

2 Preliminaries

2.1 Decision trees and quantum query algorithms

A (randomized) decision tree is the analogue of a deterministic (randomized) algorithm in the query complexity world, and a quantum query algorithm is the analogue of a quantum algorithm. See [7] for a nice survey on query complexity.

[10] This construction is further improved in a very recent work [10] by Bun and Thaler, with a more sophisticated construction which does not follow the composition paradigm.

[11] A simple $O(\log n)$-query algorithm just follows from the definition.

Let $\mathcal{T}$ be a randomized decision tree, we use $\mathcal{C}(\mathcal{T})$ to denote the maximum number of queries incurred by $\mathcal{T}$ in the worst case[12]. Let $\mathcal{Q}$ be a quantum query algorithm, we use $\mathcal{C}(\mathcal{Q})$ to denote the number of queries taken by $\mathcal{Q}$.

We assume a randomized decision tree $\mathcal{T}$ (or a quantum query algorithm $\mathcal{Q}$) outputs a result in $\{0,1\}$, and we use $\mathcal{T}(x)$ ($\mathcal{Q}(x)$) to denote the (random) output of $\mathcal{T}$ ($\mathcal{Q}$) given an input x.

2.2 Complexity classes and their query complexity analogues

We assume familiarity with some standard complexity classes like PP. Due to space constraint, we only introduce the most relevant classes $\mathsf{A0PP}^{\mathsf{dt}}$ and $\mathsf{PP}^{\mathsf{dt}}$ here, and defer the formal definitions of the partial function complexity classes $\mathsf{SBP}^{\mathsf{dt}}$, $\mathsf{SBQP}^{\mathsf{dt}}$, $\mathsf{PostBPP}^{\mathsf{dt}}$ and $\mathsf{PostBQP}^{\mathsf{dt}}$ to the full version.

Recall that $\mathcal{C}^{\mathsf{dt}}$ is the set of the partial function family $\mathbf{f}$ with $\mathcal{C}^{\mathsf{dt}}(f) = O(\mathrm{polylog}(\mathsf{len}(f)))$ for all $f \in \mathbf{f}$, hence we only need to define $\mathcal{C}^{\mathsf{dt}}(f)$ for a partial function f.

$\mathsf{PP}^{\mathsf{dt}}$

We first define $\mathsf{PP}^{\mathsf{dt}}(f)$.

▶ **Definition 2.1.** Let $f : D \to \{0,1\}$ with $D \subseteq \{0,1\}^M$ be a partial function. Let $\mathcal{T}$ be a randomized decision tree which computes f with a probability better than $1/2$. Let α be the maximum real number such that

$$\Pr[\mathcal{T}(x) = f(x)] \geq \frac{1}{2} + \alpha$$

for all $x \in D$.

Then we define $\mathsf{PP}^{\mathsf{dt}}(\mathcal{T}; f) := C(\mathcal{T}) + \log_2(1/\alpha)$, and $\mathsf{PP}^{\mathsf{dt}}(f)$ as the minimum of $\mathsf{PP}^{\mathsf{dt}}(\mathcal{T}; f)$ over all $\mathcal{T}$ computing f with a probability better than $1/2$.

A0PP and $\mathsf{A0PP}^{\mathsf{dt}}$

In this subsection we review the definition of $\mathsf{A0PP}$, and define its analogue in query complexity. There are several equivalent definitions for $\mathsf{A0PP}$, we choose the most convenient one here.

▶ **Definition 2.2.** $\mathsf{A0PP}$ (defined by Vyalyi [19]) is the class of languages $L \subseteq \{0,1\}^*$ for which there exists a BPP machine M and a polynomial p, such that for all inputs x:

(i) $x \in L \Longrightarrow \Pr[M(x) \text{ accepts}] \geq \frac{1}{2} + 2^{-p(|x|)}$.

(ii) $x \notin L \Longrightarrow \Pr[M(x) \text{accepts}] \in \left[\frac{1}{2}, \frac{1}{2} + 2^{-p(|x|)-1}\right]$.

▶ **Definition 2.3.** Let $f : D \to \{0,1\}$ with $D \subseteq \{0,1\}^M$ be a partial function. We say a randomized decision tree $\mathcal{T}$ $\mathsf{A0PP}$-computes f if there is a real number $\alpha > 0$ such that

- $\Pr[\mathcal{T}(x) = 1] \geq 1/2 + 2\alpha$ when $f(x) = 1$.
- $\Pr[\mathcal{T}(x) = 1] \in [1/2, 1/2 + \alpha]$ when $f(x) = 0$.

Fix a $\mathcal{T}$ $\mathsf{A0PP}$-computing f, let α be the maximum real number satisfying above conditions. Then we define $\mathsf{A0PP}^{\mathsf{dt}}(\mathcal{T}; f) = C(\mathcal{T}) + \log_2(1/\alpha)$ for $\mathcal{T}$ $\mathsf{A0PP}$-computing f and $\mathsf{A0PP}^{\mathsf{dt}}(f)$ as the minimum of $\mathsf{A0PP}^{\mathsf{dt}}(\mathcal{T}; f)$ over all $\mathcal{T}$ $\mathsf{A0PP}^{\mathsf{dt}}$-computing f. And we simply let $\mathsf{coA0PP}^{\mathsf{dt}}(f) := \mathsf{A0PP}^{\mathsf{dt}}(\neg f)$.

[12] i.e. the maximum height of a decision tree in the support of $\mathcal{T}$

Two relativized facts

We also introduce two important relativized results here. In [1], Aaronson showed that PostBQP is indeed PP in disguise.

▶ **Theorem 2.4** ([1]). *PostBQP = PP.*

And in [13], Kuperberg showed that SBQP is in fact equal to A0PP.

▶ **Theorem 2.5** ([13]). *SBQP = A0PP.*

These two theorems relativize, hence we have the following corollaries.

▶ **Corollary 2.6.** $\mathit{SBQP}^{dt} = \mathit{A0PP}^{dt}$.

▶ **Corollary 2.7.** $\mathit{PostBQP}^{dt} = \mathit{PP}^{dt}$.

2.3 Low-weighted one-sided approximate degree

In this subsection, we introduce a new notion of one-sided approximate degree, which is closely connected to $\mathsf{A0PP}^{\mathsf{dt}}(f)$.

▶ **Definition 2.8.** Write a polynomial $p(x) := \sum_{i=1}^{m} a_i \cdot M_i(x)$ as a sum of monomials, we define $\mathsf{weight}(p) := \sum_{i=1}^{m} |a_i|$. The one-sided low-weight ϵ-approximate degree of a partial function f denoted by $\widehat{\deg}_+^{\epsilon}(f)$, is defined by

$$\widehat{\deg}_+^{\epsilon}(f) := \min_p \max\{\deg(p), \log_2(\mathsf{weight}(p))\},$$

where p goes over all polynomials which one-sided ϵ-approximates f.[13]

We simply let $\widehat{\deg}_-^{\epsilon}(f) := \widehat{\deg}_+^{\epsilon}(\neg f)$. We also define $\widehat{\deg}_+(f)$ as $\widehat{\deg}_+^{1/2}(f)$. $\widehat{\deg}_-$ is defined similarly.

Clearly $\widehat{\deg}_+^{\epsilon}(f) \geq \deg_+^{\epsilon}(f)$. And the choice of constant $1/2$ is arbitrary, as we can reduce the approximation error by the following lemma.

▶ **Lemma 2.9.** *For any* $0 < \epsilon_1 < \epsilon_2 < 1$, $\widehat{\deg}_+^{\epsilon_1}(f) \leq \left\lceil \frac{\ln \epsilon_1^{-1}}{\ln \epsilon_2^{-1}} \right\rceil \cdot \widehat{\deg}_+^{\epsilon_2}(f)$.

Proof. We can just take the $\left\lceil \frac{\ln \epsilon_1^{-1}}{\ln \epsilon_2^{-1}} \right\rceil^{th}$ power of the polynomial corresponding to $\widehat{\deg}_+^{\epsilon_2}(f)$. ◀

We show that $\widehat{\deg}_+(f)$ is in fact equivalent to $\mathsf{A0PP}^{\mathsf{dt}}(f)$ up to a constant factor.

▶ **Theorem 2.10.** *Let f be a partial function, then*

$$\widehat{\deg}_+(f) \leq 2 \cdot \mathit{A0PP}^{dt}(f) \text{ and } \mathit{A0PP}^{dt}(f) \leq 2 \cdot \widehat{\deg}_+(f) + 2.$$

The proof is based on a simple transformation between a decision tree and the polynomial representing it, we defer the details to the full version.

And the following corollary follows from the definitions.

▶ **Corollary 2.11.** *Let f be a partial function, then*

$$\widehat{\deg}_-(f) \leq 2 \cdot \mathit{coA0PP}^{dt}(f) \text{ and } \mathit{coA0PP}^{dt}(f) \leq 2 \cdot \widehat{\deg}_-(f) + 2.$$

[13] Recall that a polynomial p one-sided ϵ-approximates a partial Boolean function f, if $p(x) \in [0, \epsilon]$ when $f(x) = 0$, and $p(x) \geq 1$ when $f(x) = 1$ as in Definition 1.8.

2.4 The permutation testing problem

Finally, we introduce the permutation testing problem.

▶ **Problem 2.12** (Permutation Testing Problem or PTP). *Given black-box access to a function $f : [n] \to [n]$, and promised that either*

(i) *f is a permutation (i.e., is one-to-one), or*

(ii) *f differs from every permutation on at least $n/8$ coordinates.*

The problem is to accept if (i) holds and reject if (ii) holds.

Assume n is a power of 2, we use PTP_n to denote the Permutation Testing Problem on functions from $[n] \to [n]$. PTP_n can be viewed as a partial function $D \to \{0,1\}$ with $D \subseteq \{0,1\}^{n \cdot \log_2 n}$.

3 Proof for the quantum case

In this section we prove Theorem 1.4 and Theorem 1.6.

Let $f : D \to \{0,1\}$ with $D \subseteq \{0,1\}^M$ be a partial function, we say a polynomial p on M variables *computes* f, if $p(x) \geq 1$ whenever $f(x) = 1$, and $p(x) \leq -1$ whenever $f(x) = 0$.

3.1 Existence of the hard distributions

In this subsection we show that if $\widehat{\deg}_+(f)$ is large, there must exist some input distributions witness this fact in a certain sense.

▶ **Lemma 3.1.** *Let f be a partial function and T be a non-negative integer. For convenience, we say a polynomial p is* valid*, if it is of degree at most T, and satisfies* $\mathsf{weight}(p) \leq 2^T$.

If $\widehat{\deg}_+^{2/3}(f) > T$, there exist two distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ supported on $f^{-1}(0)$ and $f^{-1}(1)$ respectively, such that

$$-p(\mathcal{D}_0) > 2 \cdot p(\mathcal{D}_1),$$

where $p(\mathcal{D}) = \mathbb{E}_{x \sim \mathcal{D}}[p(x)]$, for all valid polynomial p computing f.

In order to establish the above lemma, we need the following simple lemma.

▶ **Lemma 3.2.** *For any valid polynomial p computing f, if $\widehat{\deg}_+^{2/3}(f) > T$, then there exist $x \in f^{-1}(0)$ and $y \in f^{-1}(1)$ such that $-p(x) > 2 \cdot p(y)$.*

The proof is based on a simple calculation, the details can be found in the full version.

Then we prove Lemma 3.1.

Proof of Lemma 3.1. By Lemma 3.2, we have

$$\min_p \max_{(x,y) \in f^0 \times f^1} -p(x) - 2 \cdot p(y) > 0,$$

where p is a valid polynomial which computes f, $f^0 := f^{-1}(0)$ and $f^1 := f^{-1}(1)$. By the minimax theorem, and note that all the valid polynomials form a *compact convex set*, there exists a distribution $\mathcal{D}_{xy}$ on $f^0 \times f^1$ such that for any valid polynomial p computing f, we have

$$\mathbb{E}_{(x,y) \sim \mathcal{D}_{xy}}[-p(x) - 2 \cdot p(y)] > 0.$$

Then we simply let $\mathcal{D}_0$ ($\mathcal{D}_1$) be the marginal distribution of $\mathcal{D}_{xy}$ on f^0 (f^1), which completes the proof. ◀

And the following corollary follows by the definition of $\widehat{\deg}_-$.

▶ **Corollary 3.3.** *Let f be a partial function and T be a non-negative integer, if $\widehat{\deg}_-^{2/3}(f) > T$, then there exist two distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ supported on $f^{-1}(0)$ and $f^{-1}(1)$ respectively, such that for all valid polynomial p computing f,*

$p(\mathcal{D}_1) > -2 \cdot p(\mathcal{D}_0)$.

3.2 Proof for Theorem 1.4 and Theorem 1.6

We first show Theorem 1.6 implies Theorem 1.4.

Proof of Theorem 1.4. Suppose $\mathbf{f} \notin \mathsf{SBQP}^{\mathsf{dt}}$, the case that $\mathbf{f} \notin \mathsf{coSBQP}^{\mathsf{dt}}$ is similar.

By Corollary 2.6 and Theorem 2.10, there exists a sequence of function $\{f_i\}_{i=1}^{\infty} \subseteq \mathbf{f}$ such that $\widehat{\deg}_+(f_i) > \log(\mathsf{len}(f_i))^i$. Then we consider the partial function sequence $\{\mathsf{Ada}_{f_i,\lceil\log(\mathsf{len}(f_i))\rceil}\}_{i=1}^{\infty} \subseteq \mathsf{Ada}_{\mathbf{f}}$.

By Theorem 1.6, we have

$$\mathsf{PP}^{\mathsf{dt}}(\mathsf{Ada}_{f_i,\lceil\log(\mathsf{len}(f_i))\rceil}) > \min(\log(\mathsf{len}(f_i))^i/4, \mathsf{len}(f_i)/2).$$

Note that $\mathsf{len}(\mathsf{Ada}_{f_i,\lceil\log(\mathsf{len}(f_i))\rceil}) \leq 2 \cdot \mathsf{len}(f_i)^2$, we can see $\mathsf{Ada}_{\mathbf{f}} \notin \mathsf{PP}^{\mathsf{dt}}$ due to the above partial function sequence. ◀

Now, we are going to prove Theorem 1.6. We begin by introducing some consequences of a function having low $\mathsf{PP}^{\mathsf{dt}}$ complexity.

▶ **Lemma 3.4.** *Let f be a partial function, T be a positive integer. Suppose $PP^{dt}(f) \leq T$, then there exists a degree T-polynomial p computing f and satisfying $\mathsf{weight}(p) \leq 2^{2T}$.*

The proof is based on a direct analysis of the polynomial representing the decision tree for $\mathsf{PP}^{\mathsf{dt}}(f)$, we defer the details to the full version.

Our proof relies on the following two key lemmas.

▶ **Lemma 3.5.** *Let f be a partial function with $\widehat{\deg}_+^{2/3}(f) > T$. Then for each integer d, there exist two distributions $\mathcal{D}_1^d$ and $\mathcal{D}_0^d$ supported on $\mathsf{Ada}_{f,d}^{-1}(1)$ and $\mathsf{Ada}_{f,d}^{-1}(0)$ respectively, such that $-p(\mathcal{D}_0) > 2^{2^d} \cdot p(\mathcal{D}_1)$ for any degree-T polynomial p computing $\mathsf{Ada}_{f,d}$ and satisfying $\mathsf{weight}(p) \leq 2^T$.*

▶ **Lemma 3.6.** *Let f be a partial function with $\widehat{\deg}_-^{2/3}(f) > T$. Then for each integer d, there exist two distributions $\mathcal{D}_1^d$ and $\mathcal{D}_0^d$ supported on $\mathsf{Ada}_{f,d}^{-1}(1)$ and $\mathsf{Ada}_{f,d}^{-1}(0)$ respectively, such that $p(\mathcal{D}_1) > -2^{2^d} \cdot p(\mathcal{D}_0)$ for any degree-T polynomial p computing $\mathsf{Ada}_{f,d}$ and satisfying $\mathsf{weight}(p) \leq 2^T$.*

We first show these two lemmas imply Theorem 1.6 in a straightforward way.

Proof of Theorem 1.6. We prove the case when $\widehat{\deg}_+(f) > T$ first.

Otherwise, suppose $\mathsf{PP}^{\mathsf{dt}}(\mathsf{Ada}_{f,d}) \leq \min(T/4, 2^{d-1})$. By Lemma 3.4, we have a degree-$T/4$ polynomial p computing $\mathsf{Ada}_{f,d}$ with $\mathsf{weight}(p) \leq \min(2^{T/2}, 2^{2^d})$. From Lemma 2.9, $\widehat{\deg}_+(f) = \widehat{\deg}_+^{1/2}(f) \leq 2 \cdot \widehat{\deg}_+^{2/3}(f)$, hence $\widehat{\deg}_+^{2/3}(f) > T/2$. Then by Lemma 3.5, there exist two distributions $\mathcal{D}_1^d$ and $\mathcal{D}_0^d$ supported on $\mathsf{Ada}_{f,d}^{-1}(1)$ and $\mathsf{Ada}_{f,d}^{-1}(0)$ respectively, such that $-p(\mathcal{D}_0) > 2^{2^d} \cdot p(\mathcal{D}_1)$ as p is of degree at most $T/4$ and satisfies $\mathsf{weight}(p) \leq 2^{T/2}$.

But this means that $-p(\mathcal{D}_0) > 2^{2^d}$, which implies there exists an x such that $p(x) < -2^{2^d}$, therefore $\mathsf{weight}(p) \geq 2^{2^d}$, contradiction.

The case when $\widehat{\deg}_-(f) > T$ follows exactly in the same way by using Lemma 3.6 instead of Lemma 3.5. ◀

3.3 Proof for Lemma 3.5

Finally we prove Lemma 3.5. The proof for Lemma 3.6 is completely symmetric using Corollary 3.3 instead of Lemma 3.1.

Proof of Lemma 3.5. Recall that a polynomial p is valid, if it is of degree at most T, and satisfies $\mathsf{weight}(p) \leq 2^T$. Let $f_d := \mathsf{Ada}_{f,d}$ and D_d be the domain of f_d. We are going to construct these distributions $\mathcal{D}_0^d$'s and $\mathcal{D}_1^d$'s by an elegant induction.

Construction of $\mathcal{D}_0$ and $\mathcal{D}_1$ from Lemma 3.1. By Lemma 3.1 there exist two distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ supported on $f^{-1}(0)$ and $f^{-1}(1)$ respectively, such that $-p(\mathcal{D}_0) > 2 \cdot p(\mathcal{D}_1)$ for all valid polynomial p computing f.

The base case: construction of $\mathcal{D}_0^0$ and $\mathcal{D}_1^0$. For the base case $d = 0$, as f_0 is just f, we simply set $\mathcal{D}_0^0 = \mathcal{D}_0$ and $\mathcal{D}_1^0 = \mathcal{D}_1$. Then for all valid polynomial p computing f_0, we have $-p(\mathcal{D}_0^0) > 2 \cdot p(\mathcal{D}_1^0) = 2^{2^0} \cdot p(\mathcal{D}_1^0)$.

Construction of $\mathcal{D}_0^d$ and $\mathcal{D}_1^d$ for $d > 0$. When $d > 0$, suppose that we have already constructed the required distributions $\mathcal{D}_0^{d-1}$ and $\mathcal{D}_1^{d-1}$ for f_{d-1}. Decompose the input to f_d as $(w, x, y) \in D \times D_{d-1} \times D_{d-1}$ as in the definition, we claim that

$$\mathcal{D}_0^d = (\mathcal{D}_0, \mathcal{D}_0^{d-1}, \mathcal{D}_0^{d-1})^{14} \text{ and } \mathcal{D}_1^d = (\mathcal{D}_1, \mathcal{D}_1^{d-1}, \mathcal{D}_1^{d-1})$$

satisfy our conditions.

Analysis of $\mathcal{D}_0^d$ and $\mathcal{D}_1^d$. Note that D_i^d is supported on $f_d^{-1}(i)$ for $i \in \{0, 1\}$ from the definition. Let $p(w, x, y)$ be a valid polynomial computing f_d. We set

$$p(\mathcal{D}_w, \mathcal{D}_x, \mathcal{D}_y) := \mathbb{E}_{w \sim \mathcal{D}_w, x \sim \mathcal{D}_x, y \sim \mathcal{D}_y}[p(w, x, y)]$$

for simplicity, where $\mathcal{D}_w, \mathcal{D}_x, \mathcal{D}_y$ are distributions over D, D_{d-1}, D_{d-1} respectively.

Then we have to verify that for all valid polynomial p computing f_d,

$$-p(\mathcal{D}_0^d) = -p(\mathcal{D}_0, \mathcal{D}_0^{d-1}, \mathcal{D}_0^{d-1}) > 2^{2^d} \cdot p(\mathcal{D}_1, \mathcal{D}_1^{d-1}, \mathcal{D}_1^{d-1}) = 2^{2^d} \cdot p(\mathcal{D}_1^d).$$

We proceed by incrementally changing $(\mathcal{D}_0, \mathcal{D}_0^{d-1}, \mathcal{D}_0^{d-1})$ into $(\mathcal{D}_1, \mathcal{D}_1^{d-1}, \mathcal{D}_1^{d-1})$, and establish inequalities along the way.

Step 1: $(\mathcal{D}_0, \mathcal{D}_0^{d-1}, \mathcal{D}_0^{d-1}) \Rightarrow (\mathcal{D}_0, \mathcal{D}_1^{d-1}, \mathcal{D}_0^{d-1})$. By the definition, we can see that for any fixed $W \in \mathbf{support}(\mathcal{D}_0)$ and $Y \in \mathbf{support}(\mathcal{D}_0^{d-1})$, the polynomial in x defined by $p_L(x) := p(W, x, Y)$ is a valid polynomial computing f_{d-1}, hence $-p_L(\mathcal{D}_0^{d-1}) > 2^{2^{d-1}} \cdot p_L(\mathcal{D}_1^{d-1})$. By linearity, we have

$$-p(\mathcal{D}_0, \mathcal{D}_0^{d-1}, \mathcal{D}_0^{d-1}) > 2^{2^{d-1}} \cdot p(\mathcal{D}_0, \mathcal{D}_1^{d-1}, \mathcal{D}_0^{d-1}).$$

Step 2: $(\mathcal{D}_0, \mathcal{D}_1^{d-1}, \mathcal{D}_0^{d-1}) \Rightarrow (\mathcal{D}_1, \mathcal{D}_1^{d-1}, \mathcal{D}_0^{d-1})$. Similarly, for any fixed $X \in \mathbf{support}(\mathcal{D}_1^{d-1})$ and $Y \in \mathbf{support}(\mathcal{D}_0^{d-1})$, by the definition, we can see that the polynomial in w defined by $p_M(w) := -p(w, X, Y)$ is a valid polynomial computing f, hence $-p_M(\mathcal{D}_0) > 2 \cdot p_M(\mathcal{D}_1)$. Again by linearity, we have

$$p(\mathcal{D}_0, \mathcal{D}_1^{d-1}, \mathcal{D}_0^{d-1}) > -2 \cdot p(\mathcal{D}_1, \mathcal{D}_1^{d-1}, \mathcal{D}_0^{d-1}) > -p(\mathcal{D}_1, \mathcal{D}_1^{d-1}, \mathcal{D}_0^{d-1}).$$

Step 3: $(\mathcal{D}_1, \mathcal{D}_1^{d-1}, \mathcal{D}_0^{d-1}) \Rightarrow (\mathcal{D}_1, \mathcal{D}_1^{d-1}, \mathcal{D}_1^{d-1})$. Finally, for any fixed $W \in \mathbf{support}(\mathcal{D}_1)$ and $X \in \mathbf{support}(\mathcal{D}_1^{d-1})$, the polynomial in y defined by $p_R(y) := p(W, X, y)$ is a polynomial computing f_{d-1}, hence $-p_R(\mathcal{D}_0^{d-1}) > 2^{2^{d-1}} \cdot p_R(\mathcal{D}_1^{d-1})$. By linearity, we have

$$-p(\mathcal{D}_1, \mathcal{D}_1^{d-1}, \mathcal{D}_0^{d-1}) > 2^{2^{d-1}} \cdot p(\mathcal{D}_1, \mathcal{D}_1^{d-1}, \mathcal{D}_1^{d-1}).$$

Putting the above three inequalities together, we have

$$-p(\mathcal{D}_0^d) = -p(\mathcal{D}_0, \mathcal{D}_0^{d-1}, \mathcal{D}_0^{d-1}) > 2^{2^d} \cdot p(\mathcal{D}_1, \mathcal{D}_1^{d-1}, \mathcal{D}_1^{d-1}) = 2^{2^d} \cdot p(\mathcal{D}_1^d).$$

This completes the proof. ◀

3.4 Application in hardness amplification for polynomial approximation

In this subsection, we slightly adapt the above proof in order to show Theorem 1.9.

For a polynomial p on n variables, let $\|p\|_\infty := \max_{x \in \{0,1\}^n} |p(x)|$. Lemma 3.5 shows that, fix a partial function f with $\widehat{\deg}_+(f) > T$, then for any polynomial computing $\mathsf{Ada}_{f,d}$ with $\mathsf{weight}(p) \leq 2^T$, we must have $\|p\|_\infty > 2^{2^d}$. The restriction on $\mathsf{weight}(p)$ is essential for us to establish the connection between $\mathsf{A0PP}^{\mathsf{dt}}$ and $\widehat{\deg}_+$, but it becomes troublesome when it comes to proving a hardness amplification result.

Luckily, we can get rid of the restriction on $\mathsf{weight}(p)$ by making a stronger assumption that $\deg_+(f) > T$. Formally, we have the following analogous lemma for Lemma 3.5.

▶ **Lemma 3.7.** *Let f be a partial function with $\deg_+^{2/3}(f) > T$. Then for each integer d, there exist two distributions $\mathcal{D}_1^d$ and $\mathcal{D}_0^d$ supported on $\mathsf{Ada}_{f,d}^{-1}(1)$ and $\mathsf{Ada}_{f,d}^{-1}(0)$ respectively, such that for any degree-T polynomial p computing $\mathsf{Ada}_{f,d}$, $-p(\mathcal{D}_0^d) > 2^{2^d} \cdot p(\mathcal{D}_1^d)$ and consequently $\|p\|_{+\infty} > 2^{2^d}$.*

Proof. Using nearly the same proof for Lemma 3.1, we can show that for a partial function f, if $\deg_+^{2/3}(f) > T$, there exist two distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ supported on $f^{-1}(0)$ and $f^{-1}(1)$ respectively, such that $-p(\mathcal{D}_0) > 2 \cdot p(\mathcal{D}_1)$ for all degree-T polynomial p computing f. Then we can proceed exactly as in the proof for Lemma 3.5 to get the desired distributions. ◀

Finally, we are ready to prove Theorem 1.9.

Proof of Theorem 1.9. Let $F := \mathsf{Ada}_{f,d}$. Suppose otherwise $\widetilde{\deg}_\epsilon(F) \leq T$ for $\epsilon = 0.5 - 2^{-2^d+1}$. Then there exists a polynomial p such that $\|p\|_\infty \leq 1 + \epsilon$, $p(x) \leq 0.5 - 2^{-2^d+1}$ when $F(x) = 0$, and $p(x) \geq 0.5 + 2^{-2^d+1}$ when $F(x) = 1$.

Then we define polynomial $q(x) := (p(x) - 0.5) \cdot 2^{2^d-1}$. It is easy to see $q(x)$ computes F. Also, we have $\|q\|_\infty \leq (\|p\|_\infty + 0.5) \cdot 2^{2^d-1} < 2^{2^d}$, which contradicts Lemma 3.7, and this completes the proof. ◀

Acknowledgment. I would like to thank Scott Aaronson, Adam Bouland, Dhiraj Holden and Prashant Vasudevan for several helpful discussions during this work, Ruosong Wang for many comments on an early draft of this paper, Justin Thaler for the suggestion on the application in hardness amplification for polynomial approximation, and Mika Göös and Thomas Watson for pointing out an issue in the proof of Theorem 1.7.

References

1. Scott Aaronson. Quantum computing, postselection, and probabilistic polynomial-time. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 461(2063):3473–3482, 2005.
2. Scott Aaronson. Impossibility of succinct quantum proofs for collision-freeness. *Quantum Information & Computation*, 12(1-2):21–28, 2012.
3. Scott Aaronson and Alex Arkhipov. The computational complexity of linear optics. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 333–342. ACM, 2011.
4. Richard Beigel. Perceptrons, PP, and the polynomial hierarchy. *Computational Complexity*, 4(4):339–349, 1994.
5. Adam Bouland, Lijie Chen, Dhiraj Holden, Justin Thaler, and Prashant Nalini Vasudevan. On SZK and PP. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 23, page 140, 2016.
6. Michael J. Bremner, Richard Jozsa, and Dan J. Shepherd. Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 467(2126):459–472, 2011.
7. Harry Buhrman and Ronald De Wolf. Complexity measures and decision tree complexity: a survey. *Theoretical Computer Science*, 288(1):21–43, 2002.
8. Mark Bun and Justin Thaler. Dual polynomials for collision and element distinctness. *arXiv preprint arXiv:1503.07261*, 2015.
9. Mark Bun and Justin Thaler. Hardness amplification and the approximate degree of constant-depth circuits. In *International Colloquium on Automata, Languages, and Programming*, pages 268–280. Springer, 2015.
10. Mark Bun and Justin Thaler. Approximate degree and the complexity of depth three circuits. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 23, page 121, 2016.
11. Andrew Drucker and Ronald de Wolf. Quantum proofs for classical theorems. *arXiv preprint arXiv:0910.3376*, 2009.
12. Yenjo Han, Lane A Hemaspaandra, and Thomas Thierauf. Threshold computation and cryptographic security. *SIAM Journal on Computing*, 26(1):59–78, 1997.
13. Greg Kuperberg. How hard is it to approximate the Jones polynomial? *arXiv preprint arXiv:0908.0512*, 2009.
14. Noam Nisan and Mario Szegedy. On the degree of boolean functions as real polynomials. *Computational complexity*, 4(4):301–313, 1994.
15. Alexander A. Sherstov. Breaking the Minsky-Papert barrier for constant-depth circuits. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pages 223–232. ACM, 2014.
16. Alexander A Sherstov. The power of asymmetry in constant-depth circuits. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 431–450. IEEE, 2015.
17. Justin Thaler. Lower bounds for the approximate degree of block-composed functions. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 21, page 150, 2014.
18. NK Vereschchagin. On the power of pp. In *Structure in Complexity Theory Conference, 1992., Proceedings of the Seventh Annual*, pages 138–143. IEEE, 1992.
19. Mikhail N. Vyalyi. QMA = PP implies that PP contains PH. In *ECCCTR: Electronic Colloquium on Computational Complexity, technical reports*, 2003. URL: `http://eccc.hpi-web.de/report/2003/021/`.