On the Pseudo-Deterministic Query Complexity of NP Search Problems

Shafi Goldwasser
University of California, Berkeley, CA, USA

Russell Impagliazzo
University of California, San Diego, CA, USA

Toniann Pitassi
University of Toronto, Canada
Columbia University, New York, NY, USA
Institute of Advanced Study, Princeton, NJ, USA

Rahul Santhanam
University of Oxford, UK

Abstract

We study *pseudo-deterministic* query complexity – randomized query algorithms that are required to output the *same* answer with high probability on all inputs. We prove $\Omega(\sqrt{n})$ lower bounds on the pseudo-deterministic complexity of a large family of search problems based on unsatisfiable random CNF instances, and also for the promise problem (FIND1) of finding a 1 in a vector populated with at least half one's. This gives an exponential separation between randomized query complexity and pseudo-deterministic complexity, which is tight in the quantum setting. As applications we partially solve a related combinatorial coloring problem, and we separate random tree-like Resolution from its pseudo-deterministic version. In contrast to our lower bound, we show, surprisingly, that in the zero-error, average case setting, the three notions (deterministic, randomized, pseudo-deterministic) collapse.

2012 ACM Subject Classification Theory of computation → Complexity classes; Theory of computation → Oracles and decision trees; Theory of computation → Proof complexity

Keywords and phrases Pseudo-determinism, Query complexity, Proof complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2021.36

Funding *Shafi Goldwasser*: Research supported in part by DARPA under Contract No. HR001120C0015.
Russell Impagliazzo: Research supported by NSF and the Simons Foundation.
Toniann Pitassi: Research supported by NSERC, the IAS School of Mathematics and NSF Grant No. CCF-1900460.

Acknowledgements We thank Ofer Grossman, Ran Raz, Avi Wigderson and Ryan Williams for helpful discussions. The quantum query upper bound for FIND1 was pointed out to the fourth author by Igor Oliveira. We also thank the anonymous CCC reviewers for very helpful comments.

1 Introduction

The natural and beautiful notion of *pseudo-determinism* which formalizes random search algorithms that are required on every input, to output the *same* solution with high probability, was introduced by Gat and Goldwasser in [12]. A motivating example is the problem of finding an n-bit prime number in time polynomial in n. Since primality testing is in P, and the primes are dense within the natural numbers, we can efficiently find a prime with high probability by repeatedly selecting a random number, test it for primality, and halt if a prime is found. In contrast the fastest *deterministic* algorithm for finding primes is exponential

in n. A pseudo-deterministic algorithm lies between a randomized search algorithm (which on each input may output a large number of different solutions as we vary the random coins), and a deterministic algorithm. Here we are allowed unlimited use of randomness, but the search algorithm is required to output a *canonical* answer $f(x)$ on each input x (with very high probability).

Pseudodeterminism is important, both because of the intrinsic nature of the underlying questions that it raises, and because of its strong connections to other phenomena. First, it relates to the *reproducibility* question in science – empirical research has unavoidable randomness in many phases of research, from data generation/collection, to experiment design and testing. Pseudodeterministic algorithms correspond to *reproducible* experiments where the same (or a very similar) outcome will usually be obtained if the experiment is reproduced under a different set of (random) conditions [12, 22]. Pseudodeterminism also is related to the notion of *global stability* in machine learning, which is closely tied to generalization in machine learning

Starting with [12], a growing body of research has laid much of the groundwork for a theory of pseudo-deterministic complexity theory, establishing the power and limitations of pseudo-determinism for a variety of computational models (See for example [12, 13, 22, 14, 15, 16].) Assuming $\mathsf{P} = \mathsf{BPP}$, polynomial-time pseudo-deterministic search is equivalent to deterministic polynomial-time search. This implies for example that finding an n-bit prime is in polytime assuming $\mathsf{P} = \mathsf{BPP}$, but this is far from giving a efficient deterministic or pseudo-deterministic algorithm that generates primes. Oliveira and Santhanam [30] demonstrated the power of pseudo-determinism by proving unconditionally that finding primes could be carried out (for infinitely many n) in subexponential-time.

1.1 Our Results

In this paper we study the power of pseudo-determinism in the context of *query complexity*, which was first defined and studied by Goldreich, Goldwasser and Ron [13]. We focus on search problems with solutions that can be verified easily by deterministic query algorithms, similarly to the complexity class FNP, and that have an abundance of solutions[1]. In other words, we consider search problems where a solution can be found randomly simply by guessing and then verifying the guess, but for which deterministically finding a solution is difficult. The most natural problems we consider are promise problems, but we prove lower bounds for these via reduction to problems which have the above property on the full domain, i.e., we prove lower bounds for the analogs of TFNP problems with an abundance of witnesses.

This scenario is of central importance in complexity theory, where many longstanding open problems are closely connected to explicit constructions of objects that exist in abundance. For example, explicit constructions of rigid matrices imply circuit lower bounds, and explicit constructions of functions that are hard to compute (or approximate) imply derandomization.

1. We define an elementary promise search problem, FIND1: given an n bit string with the promise that it contains at least $n/2$ 1's, output a coordinate i such that $x_i = 1$. FIND1 is easy for randomized query complexity, and we observe (Section 3) that FIND1 is complete for easily verifiable search problems with randomized query algorithms. [2]

[1] In contrast, the linear query lower bounds of [13] are not for a problem with easily verifiable solutions.
[2] A similar problem titled Find-Support-Elem was considered in the context of studying the space complexity of pseudo-deterministic streaming algorithms [17]

2. We prove (Section 4) a lower bound of $\Omega(\sqrt{n})$ on the pseudo-determinsitic query complexity of a broad class of search problems associated with random unsatisfiable CNF formulas, a problem in the query analog of TFNP. As a corollary we prove the same lower bound for FIND1, thus separating randomized from pseudo-deterministic query complexity for a problem in the analog of FNP. Our lower bound also holds in the quantum setting where a simple binary search plus Grover's result shows that our lower bound is tight. A key idea in our proof is to look at a different *structured* family of search problems associated with highly unsatisfiable CNF formulas. Our lower bound for these structured search problems follows by combining Huang's Sensitivity Theorem with known linear lower bounds on the Nullstellensatz/SOS degree for refuting random unsatisfiable CNF instances.
3. Applications. We study two questions related to our lower bound in Section 5. First as a corollary, we obtain a lower bound for a related combinatorial coloring problem that we define and find independently interesting. Secondly, we extend our results to give an exponential separation between the *size* of randomized decision trees and the *size* of pseudo-deterministic decision trees. Our size separation in turn implies an exponential separation between *pseudo-deterministic* tree-like Resolution refutations and *random* tree-like Resolution refutations (defined in [7]).
4. In contrast to our lower bounds which expose the limitations of pseudo-deterministic query algorithms, we prove (Section 6) that in the zero-error average setting, the three notions (deterministic, randomized, and pseudo-deterministic) collapse.

1.2 Our Ideas

We discuss our results and the ideas behind them at a high level.

Our observation that FIND1 is a canonical problem for pseudo-deterministic query complexity for problems in FNP follows from the fact that every randomized query algorithm can be assumed to have as support a linear-size set B of deterministic decision trees. Assume that FIND1 has an efficient pseudodeterministic query algorithm, and let $\mathcal{S}$ be a problem in FNP. We define a pseudodeterministic algorithm for $\mathcal{S}$ by simulating the protocol for FIND1. Every time the protocol for FIND1 queries a bit, we run the corresponding decision tree in the linear-size set B and return 1 iff the decision tree returns a valid solution to $\mathcal{S}$. Note that since $\mathcal{S}$ is in FNP, we can check that a solution is valid efficiently. When the protocol for FIND1 concludes and outputs an index j of a bit, we simulate the corresponding decision tree in B and return the solution for $\mathcal{S}$ that it outputs.

Our lower bound for a TFNP problem is for the search problem associated with a randomly chosen k-CNF formula ϕ of linear size. The main property we require from this formula is that the factor graph is a strong enough expander. The search problem associated with ϕ is to return the index of an unsatisfied clause, given an assignment to the variables. We choose the size of the CNF large enough so that for each assignment to variables, a constant fraction of clauses are violated. Thus there is a trivial randomized protocol for the search problem with cost $O(1)$: output a random clause.

We show that any pseudo-deterministic query algorithm for this problem requires $\Omega(\sqrt{n})$ queries, using a novel connection to proof complexity. We use the known result [21, 6, 3] that the random CNFs we consider require linear degree to refute in the Nullstellensatz proof system to show a lower bound on the Fourier degree of the search problem associated with these CNFs. We then use the recent breakthrough of Huang on the Sensitivity Conjecture [25] to lower bound the sensitivity by $\Omega(\sqrt{n})$, and show that the pseudo-deterministic query complexity is lower bounded by the sensitivity. By using the very recent result of [2] instead

of [25], we can even lower bound the pseudodeterministic quantum query complexity by $\Omega(\sqrt{n})$. For quantum query complexity, this is actually tight, as a matching upper bound follows from combining binary search with Grover's algorithm.

Our quest for an improved linear lower bound for FIND1 raises an interesting combinatorial question: given any coloring of the hypercube (omitting the all zeroes vertex) with n colors such that each vertex is colored with the index of one of its 1s, must there be vertex with a constant fraction of 1s so that a constant fraction of its neighbours are colored differently from it? If the answer to this question is yes, we would be able to show that FIND1 requires linear pseudo-deterministic query complexity. The question above is about the sensitivity of a coloring; we can ask an analogous question for block-sensitivity and in this case, it turns out that we can prove an $\Omega(\sqrt{n})$ lower bound, which also implies our $\Omega(\sqrt{n})$ lower bound for FIND1.

Our proof of a pseudo-deterministic query lower bound uses ideas from proof complexity. We show that there is a connection in the reverse direction too, by defining pseudo-deterministic versions of propositional proof systems such as Resolution. A broad question in proof complexity is whether we can use proof systems to capture the behaviour of randomized algorithms. Motivated in part by this and in part by a question about bounded-depth Frege proof systems, [7] defined Random Resolution: a randomized version of Resolution. This is quite a powerful system which even refutes random k-CNFs in constant size, contrary to our intuition that random k-CNFs should be hard to solve. We define pseudo-deterministic Resolution and pseudo-deterministic Tree Resoluton, and we show that pseudo-deterministic Tree Resolution is efficiently verifiable, suggesting that it is a more viable candidate for capturing the behaviour of randomized algorithms. We apply the ideas of our separation between randomized query complexity and pseudo-deterministic query complexity to get a strong separation between Random Tree Resolution and pseudo-deterministic Tree Resolution: random k-CNFs can be refuted in linear size in Random Tree Resolution but require $2^{\Omega(\sqrt{n})}$ size in pseudo-deterministic Tree Resolution.

Finally, we turn our attention from lower bounds to algorithms. We show that perhaps surprisingly, there is a close connection between randomized query complexity and pseudo-deterministic query complexity on average. Specifically, for zero-error algorithms (where the query algorithm is not allowed to make a mistake), we show that over any distribution D, the randomized, pseudo-deterministic and deterministic query complexity are all within a polylogarithmic factor of each other. Similarly, we show that for any approximation problem (such as the problem of approximating the Hamming weight of an input considered in [13], for which there is a constant-query randomized algorithm) and distribution D, there is an efficient bounded-error pseudo-deterministic query algorithm which asks few queries on average over D. Note that we require the algorithm to be pseudo-deterministic on *every input*, which is a pretty strong guarantee.

As a toy problem for our result on zero-error query algorithms, consider the FIND1 problem, which does have a very efficient zero-error randomized algorithm. Given any distribution D, we can use an averaging argument to identify a small set of decision trees from the support of our randomized query algorithm such that at least one of the trees from this set outputs a correct solution with probability at least $1 - 1/n$ over the distribution. We can also efficiently check if this is indeed the case. If not, we simply query every bit, and this doesn't cost too much on average because this case happens with very low probability.

Generalizing to efficient average-case zero-error algorithms is somewhat more involved, and requires an interleaving simulation of decision trees together with a Markov argument at different scales. We use similar ideas for our bounded-error pseudo-deterministic algorithms - the challenge is to meet the pseudo-determinsitic guarantee on every input.

1.3 Related Work

Optimal query separations were already proven by [13] but their search problem is not in FNP – that is, for the problem that they studied, solutions are not verifiable with a polylogarithmic number of queries. In particular, they studied the search problem of estimating the number of ones in a binary string to within an additive ϵn. They proved that this search problem has low randomized query complexity, but requires linear pseudodeterministic query complexity.

2 Definitions

▶ **Definition 1.** *A search problem over domain $\mathcal{X}$ and range $\mathcal{O}$ is defined to be a relation $\mathcal{S} \subseteq \mathcal{X} \times \mathcal{O}$. For $x \in \mathcal{X}$, the feasible solutions for $\mathcal{S}$ on x are the elements $o \in \mathcal{O}$ such that $(x, o) \in \mathcal{S}$. $\mathcal{S}$ is total if there is at least one feasible solution for every $x \in \mathcal{X}$. A function $f : \mathcal{X} \to \mathcal{O}$ solves the search problem $\mathcal{S}$ if for every $x \in \mathcal{X}$ with at least one feasible solution for $\mathcal{S}$, $(x, f(x)) \in \mathcal{O}$.*

Deterministic Query Complexity. Let $\mathcal{X} = \{0,1\}^n$. A determininistic decision tree T over $x_1, \ldots, x_n$ with outputs from $\mathcal{O}$ is a binary tree where each internal node is labelled with a variable x_i, and with outedges labelled by $x_i = 0$ and $x_i = 1$. Each leaf of the tree is labelled with some $o \in \mathcal{O}$. A deterministic decision tree T computes $f : \{0,1\}^n \to \mathcal{O}$ if for every input $x \in \{0,1\}^n$, the (unique) path in T consistent with x has leaf label $f(x)$. Let $\mathsf{P}^{\mathsf{dt}}(f)$ be the minimum depth of a deterministic decision tree computing f. [3] For a search problem $\mathcal{S} \subseteq \{0,1\}^n \times \mathcal{O}$, The (deterministic) query complexity of $\mathcal{S}$, $\mathsf{P}^{\mathsf{dt}}(\mathcal{S})$ is the minimum of $\mathsf{P}^{\mathsf{dt}}(f)$ over all functions f solving $\mathcal{S}$.

Randomized and Quantum Query Complexity. A randomized decision tree over $x_1, \ldots, x_n$ with outputs from $\mathcal{O}$ is a distribution $\mathcal{T}$ over deterministic decision trees. A randomized decision tree $\mathcal{T}$ computes $f : \{0,1\}^n \to \mathcal{O}$ with error at most ϵ if for every input x, the probability (over T drawn from $\mathcal{T}$) that $T(x)$ ouputs $f(x)$ is at least $1 - \epsilon$. The bounded-error randomized query complexity of search problem $\mathcal{S}$, denoted by $\mathsf{BPP}^{\mathsf{dt}}(\mathcal{S})$, is the minimum over all functions f computing $\mathcal{S}$ of the depth of a randomized decision tree computing f with error $1/3$.

We can also define zero-error randomized query complexity for f and $\mathcal{S}$. In this case $\mathcal{T}$ is a distribution over decision trees, but with the property that for every x, the probability that $\mathcal{T}(x) = f(x)$ is one. Whereas before the depth was defined to be the maximum depth over all decision trees in the distribution, in the zero-error case, we define the depth to be the expected depth. The quantum query complexity for functions and search problems is defined analogously. (e.g., see [9].)

Nondeterministic Query Complexity. Let $\mathcal{S} \subseteq \{0,1\}^n \times [m]$ be a search problem. A *verification* decision tree for f is a decision tree $\mathcal{T}$ over the Boolean variables $x_1, \ldots, x_n$, $y_1, \ldots, y_{\log m}$ with outputs $\{0,1\}$ such that for every input pair $(x, y) \in \{0,1\}^n \times [m]$, $\mathcal{T}(x, y) = 1$ if and only if $(x, y) \in \mathcal{S}$. The verification query complexity of $\mathcal{S}$ is the minimum depth over all verification decision trees for $\mathcal{S}$. A search problem $\mathcal{S} \subseteq \{0,1\}^n \times [m]$ with $m = O(n)$ is an NP-search problem if there is a verification decision tree for $\mathcal{S}$ of depth polynomial in $\log m$.

[3] We note that since f may not be Boolean, $\mathsf{FP}^{\mathsf{dt}}(f)$ is a more accurate notation, but we slightly abuse notation and use P^{dt} to be consistent with prior work/notation.

Pseudodeterministic Query Complexity. Finally we define the bounded-error and zero-error pseudo-deterministic query complexity for total search problems $\mathcal{S}$. A bounded-error pseudo-deterministic decision tree for S is a distribution over decision trees with the following property: For every input x, there is a *canonical* value $o \in \mathcal{O}$ such that with probability at least $2/3$, $\mathcal{T}(x) = o$. In other words, $\mathcal{T}$ is a bounded-error randomized decision tree for a particular function f that solves $\mathcal{S}$. Let $\mathsf{psP}^{\mathsf{dt}}(\mathcal{S})$ denote the (bounded-error) pseudo-deterministic query complexity of $\mathcal{S}$. Similarly let $\mathsf{psQ}^{\mathsf{dt}}(\mathcal{S})$ denote the pseudo-deterministic bounded-error quantum query complexity of $\mathcal{S}$.

We note that for bounded-error randomized and pseudo-determinstic query algorithms, by repeatedly running the query algorithm $O(log(1/\delta))$ times, we can amplify the success probability from $2/3$ to $1 - \delta$.

Sensitivity and Block Sensitivity. Let $f : \{0,1\}^n \to \mathcal{O}$. A block $B \subseteq [n]$ is *sensitive* for f on input x if $f(x \oplus 1_B) \neq f(x)$, where 1_B is the n-bit string that is 1 on bits in B and 0 otherwise. In other words, if we change x by flipping all of the bits in B to get x^B, then the value of f changes (so $f(x) \neq f(x^B)$). The *block sensitivity* of x with respect to f, $\boldsymbol{bs}_x(f)$, is the maximal number of disjoint blocks that are all sensitive for x. We define $\boldsymbol{bs}(f) = max_{x\in\{0,1\}^n} \boldsymbol{bs}_x(f)$.

A bit $i \in [n]$ is sensitive for x with respect to f if the block $\{i\}$ is sensitive for x. The sensitivity of x with respect to f, $\mathbf{s}_x(f)$, is the maximal number of sensitive bits for x, and $\mathbf{s}(f) = max_{x\in\{0,1\}^n} \mathbf{s}_x(f)$.

Degree. A polynomial $q \in \mathbb{R}[x_1, \ldots, x_n]$ is said to *represent* the function $f : \{0,1\}^n \to \{0,1\}$ if $q(x) = f(x)$ for all $x \in \{0,1\}^n$. The (Fourier) degree of f, $\boldsymbol{d}(f)$ is the degree of the (unique) polynomial representing f. A multioutput function $f : \{0,1\}^n \to [m]$, induces a partition of $\{0,1\}^n$ into m classes, where the i^{th} class contains those inputs that are mapped to i (i.e., those x such that $f(x) = i$). Thus we can define m associated Boolean functions, f^i, $i \in [m]$, where $f^i(x)$ is 1 if and only if $f(x) = i$. The Fourier degree of $f : \{0,1\}^n \to [m]$ is defined as $max_{i\in[m]} \boldsymbol{d}(f^i)$, and the Fourier degree of a total search problem $\mathcal{S}$ is the minimum of $\boldsymbol{d}(f)$ over all functions f solving the search problem $\mathcal{S}$.

Known Relationships. Pioneering work of Nisan [28], Nisan and Szegedy [29] and Beals-et-al [4] studied the above query measures and showed that nearly all of them are polynomially equivalent. (See [5] for a nice exposition.) The two exceptions are pseudo-deterministic complexity (which was defined later) and sensitivity, which remained a longstanding open problem for thirty years. In recent breakthrough work, Huang [25] resolved the conjecture by proving $s(f) \geq deg(f)^{1/2}$. The exact quantitative relationships between the measures has been intensively studied; a table summarizing the state-of-the-art pairwise relationsihps (pre-Huang) is given in [1]. Post-Huang, [2] improved the relationships between deterministic query complexity, quantum query complexity and degree to near-optimal (ignoring polylog factors).

We summarize here the relationships that will be important for us. First, the following basic relationships are known:

$$\mathsf{Q}^{\mathsf{dt}}(f) = O(\mathsf{BPP}^{\mathsf{dt}}(f)) = O(\mathsf{P}^{\mathsf{dt}}(f))$$

$$\boldsymbol{d}(f) = O(\mathsf{P}^{\mathsf{dt}}(f))$$

$$\boldsymbol{s}(f) = O(\boldsymbol{bs}(f)) = O(\mathsf{BPP}^{\mathsf{dt}}(f)).$$

The following nontrivial relationships have recently been proven using Huang's theorem [2]:

$\boldsymbol{d}(f) = O(\mathsf{Q}^{\mathsf{dt}}(f)^2)$

$\mathsf{P}^{\mathsf{dt}}(f) = O(\mathsf{Q}^{\mathsf{dt}}(f)^4)$.

These results are know to be tight within polylog factors. Before these results the best known (pre-Huang) was $\boldsymbol{d}(f), \mathsf{P}^{\mathsf{dt}}(f) = O(\mathsf{Q}^{\mathsf{dt}}(f)^6)$.

We now consider the relationship between the pseudo-deterministic, deterministic and randomized query classes. Let $\mathcal{S}$ be a FNP search problem, we have the easy inclusions:

$\mathsf{P}^{\mathsf{dt}}(\mathcal{S}) \geq \mathsf{psBPP}^{\mathsf{dt}}(\mathcal{S}) \geq \mathsf{BPP}^{\mathsf{dt}}(\mathcal{S})$

$\mathsf{P}^{\mathsf{dt}}(\mathcal{S}) \geq \mathsf{psQ}^{\mathsf{dt}}(\mathcal{S}) \geq \mathsf{Q}^{\mathsf{dt}}(\mathcal{S})$.

3 Search Problems in TFNP

We define $\mathsf{TFNP}^{\mathsf{dt}}$, the query analog of TFNP to be the class of all search problems $f : \{0,1\}^n \to [m]$ that admit a nondeterministic decision tree of complexity $\mathsf{polylog}(n)$. (Equivalently, f can be written as a $\mathsf{polylog}(n)$-width DNF.)

▶ **Definition 2.** *Let $X = \{x \in \{0,1\}^n, \mid |x| \geq n/2\}$ where $|x|$ is the number of 1's in x. The search problem FIND1 $\subseteq X \times [n]$ is defined by: $(x,i) \in$ FIND1 if and only if $x \in X$ and $x_i = 1$.*

It is not hard to see that the deterministic query complexity of FIND1 is $\Omega(n)$, but the randomized query complexity (and therefore also the quantum query complexity) is constant. Here we show that for any search problem in $\mathsf{TFNP}^{\mathsf{dt}}$ for which solutions are verifiable using few queries, a gap between randomized and pseudo-deterministic query complexity implies a gap between randomized and pseudo-deterministic query for FIND1.

Call a function $f : \mathbb{N} \to \mathbb{N}$ *reasonable* if $f(\Theta(n)) = \Theta(f(n))$. Note that functions such as $f(n) = n^\epsilon$ for $\epsilon < 1$, $f(n) = \log(n)$ and $f(n) = O(1)$, which often occur as bounds on query complexity, are all reasonable.

▶ **Theorem 3.** *Let $r, q, v : \mathbb{N} \to \mathbb{N}$ be reasonable functions. Let $\mathcal{S}$ be a search problem verifiable with $v(n)$ queries such that $\mathsf{BPP}^{\mathsf{dt}}(\mathcal{S}) \leq r(n)$ and $\mathsf{psP}^{\mathsf{dt}}(\mathcal{S}) \geq q(n)$. Then $\mathsf{psP}^{\mathsf{dt}}(\text{FIND1}) = \Omega(q(n)/(r(n) + v(n)))$.*

Proof. Since $\mathcal{S}$ has randomized decision tree complexity at most $r(n)$, there is a family $\mathcal{F}$ of deterministic decision trees of depth $r(n)$ such that for each $x \in \mathcal{I}$ of length n, a uniformly chosen tree from $\mathcal{F}$ solves $\mathcal{S}$ on x with probability at least $3/4$. If we uniformly and independently pick a subfamily $\mathcal{F}'$ of cn trees from $\mathcal{F}$ for large enough constant c, it follows using Chernoff bounds and a union bound that with positive probability over the choice of $\mathcal{F}'$, for each $x \in \mathcal{X}$ of length n, a uniformly chosen tree from $\mathcal{F}'$ solves $\mathcal{S}$ on x with probability at least $2/3$. Hence, by the probabilistic method, there must exist such a subfamily $\mathcal{F}'$. Fix such a subfamily, and let $T_1 \ldots T_m$ be an arbitrary enumeration of the decision trees in $\mathcal{F}'$, where $m = cn$.

Assume that FIND1 can be solved pseudo-deterministically with at most $p(m)$ queries on inputs of length m. We show how to solve $\mathcal{S}$ pseudo-deterministically on inputs of length n with at most $p(m)(r(n) + v(n))$ queries. The pseudo-deterministic query algorithm A for $\mathcal{S}$ on input x of length n is as follows. We simulate the pseudo-deterministic query algorithm A' for FIND1 that makes at most $p(m)$ queries. If A' asks whether bit $i \in [m]$ is 1 in the

input to FIND1, we run the query algorithm for $\mathcal{S}$ corresponding to tree T_i. By assumption, at most $r(n)$ queries are made, and some output y is produced. We verify that $(x, y) \in \mathcal{S}$ by using the $v(n)$ query verification algorithm for the search problem $\mathcal{S}$. If the verification succeeds, we assume the answer to the query made by A' is 1 and proceed, otherwise we proceed with the simulation of A assuming that the answer is 0. When we finish simulating A', some index $j \in [m]$ is output. We proceed to run the query algorithm corresponding to T_j on x and return the output z of this algorithm.

The cost of this query algorithm A is at most $p(m)(r(n) + v(n))$ since the simulation of each query of A' has cost at most $r(n) + v(n)$, and there are at most $p(m)$ queries along any computation path. It remains to argue that A pseudo-deterministically solves $\mathcal{S}$. By assumption, a uniformly chosen tree from $\mathcal{F}'$ solves $\mathcal{S}$ on x with probability at least 2/3 - this implies that for at least 2/3 fraction of indices $i \in [m]$, the simulation of a query made by A' to i returns 1. By assumption, A' pseudo-deterministically solves FIND1, hence there is a fixed $j \in [m]$ for which the query made by A' to j returns 1 such that A' outputs j with probability at least 2/3. But since T_j solves $\mathcal{S}$ correctly, this means that A outputs a fixed solution to the search problem $\mathcal{S}$ with probability at least 2/3.

Thus we have that $p(m) \geq q(n)/(r(n) + v(n))$ This implies that $p(m) = \Omega(q(m)/(r(m) + v(m))$ using $m = \Theta(n)$ and our assumption that the functions r, q, v are all reasonable. ◀

4 Lower Bounds for Pseudo-deterministic Query Complexity

▶ **Theorem 4.** *There is a $\sqrt{n}$ gap between the randomized and pseudo-deterministic query complexity of FIND1:*
(1) $\mathsf{BPP}^{\mathsf{dt}}(FIND1) = O(1)$*, and therefore* $\mathsf{Q}^{\mathsf{dt}}(FIND1) = O(1)$ *as well;*
(2) $\mathsf{psQ}^{\mathsf{dt}}(FIND1) = \Omega(\sqrt{n})$ *and thus* $\mathsf{psP}^{\mathsf{dt}}(FIND1) = \Omega(\sqrt{n})$ *as well.*

The proof of the above theorem follows from Theorem 3 together with our main theorem below which proves a $\sqrt{n}$ separation between randomized and pseudo-deterministic quantum query complexity for a broad family of $\mathsf{TFNP}^{\mathsf{dt}}$ search problems that are associated with expanding unsatisfiable CNF formulas.

▶ **Definition 5.** *Let $C = C_1 \wedge \ldots \wedge C_m$ be an unsatisfiable k-CSP problem over Boolean variables $x_1, \ldots, x_n$, where each C_i is a constraint involving at most k variables. The search problem associated with C, $\mathcal{S}_C \subseteq \{0,1\}^n \times [m]$, consists of all pairs (x, i) such that $x \in \{0,1\}^n$, and $C_i(x) = 0$. A query algorithm for $\mathcal{S}_C$ on input x outputs a constraint C_i that is falsified by x.*

$\mathcal{S}_C$ has been studied extensively in proof complexity and communication complexity, where lower bounds on its deterministic query complexity have been used to obtain, via lifting, exponential lower bounds on the monotone circuit size of a monotone function associated with C. Similarly, these search problems play a prominent role in lower bounds in proof complexity and extended formulations (e.g., [10, 8, 11]).

▶ **Definition 6.** *Let $C = C_1 \wedge \ldots \wedge C_m$ be a k-CSP over Boolean variables $x_1, \ldots, x_n$. Consider the bipartite graph with m left vertices (one for each constraint) and n right vertices (one for each variable), such that (i, j) is an edge if and only if variable x_j occurs in constraint C_i. C is (r, s)-expanding if for every subset $S \subseteq [m]$ of left vertices, $|S| \leq r$, the set of right elements adjacent to S, $N(S)$, has size at least s.*

▶ **Theorem 7.** *Let C be a k-CNF or k-XOR over $x_1, \ldots, x_n$, that is $(\epsilon n, c)$-expanding for $\epsilon = 1/100$, $c \geq k/2$. Then $\mathsf{psQ}^{\mathsf{dt}}(\mathcal{S}_C) = \Omega(\sqrt{n})$.*

▶ **Corollary 8.** *Let $k \geq 3$, $c = c(k)$ a sufficiently large constant, n sufficiently large and $m = cn$. Let $\mathcal{C}_n^m$ be the distribution over random k-CNF (k-XOR) formulas with m constraints, where each constraint is chosen uniformly at random from the set of all size-k clauses (size-k XOR formulas). Then with probability $1 - o(1)$, a random C drawn from $\mathcal{C}_n^m$ will have $\mathsf{BPP}^{\mathsf{dt}}(\mathcal{S}_C) \in \mathcal{O}(1)$, and $\mathsf{psQ}^{\mathsf{dt}}(\mathcal{S}_C) \in \Omega(\sqrt{n})$.*

Proof of Corollary 8. For $c = c(k)$ a sufficiently large constant, with high probability a random k-CNF from $\mathcal{C}_n^m$ will have the property that every assignment x falsifies a constant fraction of the clauses of C. Assuming that C drawn from $\mathcal{C}_n^m$ satisfies this property, there is a constant depth randomized query algorithm for $\mathcal{S}_C$. Namely, pick a random subset S of $O(1)$ clauses from C, and query all of the variables underlying these clauses. Output the first clause from S that is falsified, if one exists, and otherwise output error. Since every assignment falsifies a constant fraction, ϵ, of clauses, the probability that all clauses in S are satisfied (so the algorithm errs) is at most $(1-\epsilon)^{|S|}$, so we can choose $|S|$ to be a sufficiently large constant so that the probability of error is at most $1/3$. Therefore with probability $1 - o(1)$, $\mathsf{BPP}^{\mathsf{dt}}(\mathcal{S}_C) = O(1)$. For the lower bound, a standard calculation shows that a random k-CNF (or k-XOR) formula will be $(n/100, k/2)$ expanding with high probability. Therefore by Theorem 7, $\mathsf{psQ}^{\mathsf{dt}}(\mathcal{S}_C) = \Omega(\sqrt{n})$. ◀

Our lower bound proceeds by first proving linear lower bounds on the Fourier degree of $\mathcal{S}_C$, by a reduction to known lower bounds on the Nullstellensatz degree of refuting C. With this linear degree bound at hand, we obtain our lower bound by applying Huang's sensitivity theorem (showing that sensitivity and degree are quadratically related) together with the fact that sensitivity lower bounds randomized query complexity.

A alternative proof which also gives us the $\sqrt{n}$ quantum pseudo-deterministic lower bound can be obtained by combining our linear degree bound for $\mathcal{S}_C$ with the result of [2], showing that quantum query complexity is quadratically related to degree. We begin with the definition of Nullstellensatz degree.

▶ **Definition 9.** *For $C = C_1 \wedge \ldots \wedge C_m$ be an unsatisfiable k-CNF formula, we define the standard representation of C by a set of $m + n$ polynomial equations (each of degree at most k) such that C is satisfiable if and only if there is an assignment such that all polynomials evaluate to zero. For a clause C_i, let C_i^+ denote the set of variables occurring positively in C_i and let C_i^- denote the set of variables occurring negatively in C_i; with this notation we can write $C_i = \bigvee_{x_j \in C_i^+} x_j \vee \bigvee_{x_j \in C_i^-} \overline{x}_j$. From C_i define the polynomial*

$$Q(C_i) = \Pi_{x_j \in C_i^+}(1 - x_j)\Pi_{x_j \in C_i^-} x_j.$$

Let $\mathcal{Q}(C) = \{Q_1, \ldots, Q_{m+n}\}$ denote the set of polynomials $\{Q(C_i) : C_i \in C\} \cup \{x_i^2 - x_i : i \in [n]\}$.

▶ **Definition 10.** *Let C be an unsatisfiable k-CNF formula and let $\mathcal{Q}(C)$ be the associated set of polynomials as in Definition 9. A Nullstellensatz refutation of C (over a field F) is a set of polynomials $\{P_i\}, i = 1 \ldots m + n$ such that*

$$\sum_{i \in [m+n]} P_i Q_i = 1$$

holds over the ring $F[x_1 \ldots x_n]$. Any such sequence $\{P_i\}$ is called a Nullstellensatz refutation of C, and the degree of the refutation is $max_{i \in [m+n]} \boldsymbol{d}(P_i)$. The Nullstellensatz degree of C, $\mathsf{NS}(C)$, is the minimum degree over all Nullstellensatz refutations of C.

We will use the following linear lower bounds on the Nullstellensatz degree for random formulas.

▶ **Theorem 11** ([21, 6, 3]). *Let $C = C_1 \wedge \ldots \wedge C_m$ be a k-CNF or k-XOR formula over $x_1, \ldots, x_m$, with $m = O(n)$ and such that C is $(\epsilon n, k/2)$-expanding. Then $\mathsf{NS}(C) = \Omega(n)$ (over any field).*

The next lemma shows that $\boldsymbol{d}(\mathcal{S}_C)$ is lower bounded by Nullstellensatz degree (over any field).

▶ **Lemma 12.** *Let C be an unsatisfiable k-CNF formula, and let f be any function solving the search problem S_C. Then $\mathsf{NS}(C) \leq \boldsymbol{d}(f)$. Conversely, for any finite field $\mathbb{F}$, $O(\boldsymbol{d}(\mathcal{S}_C) \log n) \leq \mathsf{NS}(C) \leq \boldsymbol{d}(\mathcal{S}_C)$.*

Proof of Lemma 12. Suppose that $f : \{0,1\}^m \to [m]$ solves the search problem for C, and let $d = \boldsymbol{d}(f) = \max_i \boldsymbol{d}(f^i)$. Consider the polynomial $\sum_{i\in[m]} f^i Q_i$. First, we claim that the polynomial $\sum_{i\in[m]} f^i Q_i$ evaluates to 1 on all inputs in $\{0,1\}^n$. Since the functions $\{f^i \mid i \in [m]\}$ form a partition of $\{0,1\}^n$, for every $\alpha \in \{0,1\}^n$, there is exactly one $i \in [m]$ such that $f^i(\alpha) = 1$, and for all other $j \neq i$, $f^j(\alpha) = 0$. Since $f^i(\alpha) = 1$ implies $C_i(\alpha) = 0$, it follows that $Q_i(\alpha) = 1$. Thus, $\sum_{i\in[m]} f^i Q_i$ evaluates to 1 for all $\alpha \in \{0,1\}^n$ as claimed. Now using the axioms $\{Q_{m+1}, \ldots, Q_{m+n}\} = \{x_i^2 - x_i \mid i \in [n]\}$, we can derive the identically 1 polynomial as:

$$\sum_{i\in[m]} f^i Q_i + \sum_{i\in[m+1,m+n]} h_i Q_i,$$

where each h_i is of degree at most d. Thus we have a degree d Nullstellsatz refutation of C, so $\mathsf{NS}(C) \leq \boldsymbol{d}(f)$.

In the other direction, let $\mathcal{Q}(C)$ be the set of polynomials associated with C, and assume that we have degree-d polynomials $P_1, \ldots, P_m$ such that $\sum_i P_i Q_i = 1 (mod 2)$, where $\mathbb{F} = GF(2)$. (A similar argument works over any finite field.) We want to define polynomials f^i such that: $f^i(\alpha) = 1$ implies that $C_i(\alpha) = 1$ and for all C_j, $j < i$, $C_j(\alpha) = 0$. For any α, we know that $\sum_i P_i(\alpha) Q_i(\alpha) = 1$. In order to determine whether or not $f^i(\alpha) = 1$, we want to do a binary search in order to find a term $P_i(\alpha)Q_i(\alpha)$ that evaluates to 1. For example suppose that $m = 16$. Then since $\sum_{i=1}^{16} P_i(\alpha)Q_i(\alpha)$ is odd either (a) $\sum_{i=1}^{8} P_i Q_i$ is odd, or (b) $\sum_{i=9}^{16} P_i Q_i$ is odd. If (a) is odd, then we recurse on the left (smaller) side and otherwise if (a) is even then we recurse on the right side. Viewing the binary search as a decision tree, at the root we query $\sum_{i=1}^{8} P_i Q_i$ and if it evaluates to 1 we go left and otherwise we go right. This gives a height $\log m$ decision tree where internal vertices are labelled with degree d polynomials, and the leaves are labelled with the index $i \in [m]$ such that $P_i(\alpha) = 1$. Let p_i be the path from the root to the leaf labelled by i. We can define a polynomial f^i associated with p_i which is the product of $\log m$ polynomials (along the path) such that $f^i(\alpha) = 1$ if and only if α is consistent with the path p_i. Thus the f^i's solve the search problem $\mathcal{S}_C$ and have degree $d \log m$. ◀

Proof of Theorem 7. Let $C = C_1 \wedge \ldots \wedge C_m$ be a k-CNF or k-XOR CSP over $x_1, \ldots, x_n$ that is $(\epsilon n, k/2)$ expanding, where $m = O(n)$. By Theorem 11, $\mathsf{NS}(C) = \Omega(n)$, and thus by Theorem 12, $\boldsymbol{d}(\mathcal{S}_C) = \Omega(n)$.

Assume that $\mathcal{T}$ is a pseudo-deterministic query algorithm for $\mathcal{S}_C$. Then for every input $x \in \{0,1\}^n$, there is a canonical solution $f(x)$ such that $\mathcal{T}(x)$ outputs $f(x)$ with probability at least $2/3$. Thus $\mathcal{T}$ is a randomized query algorithm for f. By Nisan [28] $\boldsymbol{s}(f) = O(\mathsf{BPP}^{\mathsf{dt}}(f))$, and by Huang [25], $\boldsymbol{s}(f) \geq \sqrt{\boldsymbol{d}(f)}$. Thus since $\boldsymbol{d}(\mathcal{S}_C) = \Omega(n)$, it follows that $\mathsf{BPP}^{\mathsf{dt}}(f) = \Omega(\sqrt{n})$, and thus $\mathsf{psP}^{\mathsf{dt}}(\mathcal{S}_C) = \Omega(\sqrt{n})$. ◀

Proof of Theorem 4. This follows from Theorem 7 and Theorem 3. We apply Theorem 3 to the search problem $S_{\mathcal{C}}$. By Theorem 7, we have that $r(n) = O(1)$ and $q(n) = \Omega(\sqrt{n})$. Also $v(n) = O(1)$ since we can verify a solution to $S_{\mathcal{C}}$ by just querying the variables in the clause that is the candidate solution. Clearly, r, q, v are all reasonable, hence it follows from Theorem 3 that FIND1 has pseudo-deterministic query complexity $\Omega(\sqrt{n})$. ◀

We observe that our $\Omega(\sqrt{n})$ separation between pseudo-deterministic quantum query complexity and quantum query complexity is tight. Grover [23] discovered a quantum query algorithm of complexity $O(\sqrt{n})$ for solving the following search problem: Given an n-bit binary string x, the goal is to find a coordinate i such that $x_i = 1$ (or to indicate that no such i exists). (See e.g., [9] for a survey.) This implies that FIND1 has pseudo-deterministic quantum query complexity $\tilde{O}(\sqrt{n})$, using a simple binary search algorithm to find the lexicographically first 1. For the quantum lower bound, we combine the result of [2] that quantum query complexity is at least $\sqrt{deg(f)}$ with Theorem 7.

5 Applications

5.1 A Related Combinatorial Problem

Our pseudo-deterministic query lower bound is related to a natural problem in extremal graph theory, which states that any proper coloring of the hypercube has high (block) sensitivity.

▶ **Definition 13.** *A proper coloring of the m-dimensional Boolean cube is any function $c : \{0,1\}^m - \{0^m\} \to [m]$ such that for all $\beta \in \{0,1\}^m - \{0^m\}$, $\beta_{c(\beta)} = 1$.*

▶ **Theorem 14.** *Let c be any proper coloring of the Boolean cube. Then there must exist $\beta \in \{0,1\}^m$ such that: (i) β contains at least a constant fraction of 1's, and (ii) β has block sensitivity $d = \Omega(\sqrt{m})$. That is, there are d disjoint blocks of inputs, $B_1, \ldots, B_d$ such that for all $i \in [d]$, $c(\beta) \neq c(\beta^{B_i})$.*

We remark that the above theorem implies a lower bound of $\Omega(\sqrt{n})$ on the pseudo-deterministic query complexity of FIND1.

Proof. At a high level, we will convert our sensitivity lower bound for the search problem associated with a random unsat k-XOR formula into a block sensitivity lower bound for the above coloring problem. Fix an expanding k-XOR formula C with $m = O(n)$ constraints and n variables such that for any assignment $\alpha \in \{0,1\}^n$, at least a constant fraction of the parity constraints are falsified by α. Further we will assume that the constraint-to-variable graph is expanding and in particular, for any subset $S \subseteq [n]$, there exists a large subset $S' \subseteq S$, $|S'| = O(|S|)$ such that for all $i \neq j \in S'$, the constraints containing x_i are disjoint from the constraints containing x_j.

First, we define a simple transformation that maps each input $\alpha \in \{0,1\}^n$ to an associated m-dimensional Boolean vector, $\beta(\alpha) \subseteq \{0,1\}^m$.

▶ **Definition 15.** *Let $\alpha \in \{0,1\}^n$. The constraint vector, $\beta(\alpha) \in \{0,1\}^m$ associated with α is defined as follows. For each $j \in [m]$, $\beta(\alpha)_m = 1$ if and only if $C_j(\alpha) = 0$. That is, the constraint vector associated with α has a 1 in coordinate j exactly when the j^{th} constraint of C is falsified by α. Let $S(C)$ denote the image of this map; that is, $S(C) \subseteq \{0,1\}^m$ is the set of all length m vectors that are constraint vectors for some $\alpha \in \{0,1\}^n$.*

Since C has the property that every assignment falsifies a constant fraction of the constraints in C, it follows that for every α, $\beta(\alpha)$ contains at least a constant fraction of 1's. Now consider a pair of adjacent assignments α and α^i where α^i is obtained from α by toggling the value of x_i, $i \in [n]$. Let $B(x_i) \subseteq [m]$ denote the set of coordinates j such that constraint C_j in C contains x_i. Because the constraints in C are all parity constraints, the constraint vector, $\beta(\alpha^i)$ associated with α^i can be obtained from $\beta(\alpha)$ by toggling the coordinates in $B(x_i)$. Thus for every $\alpha \in \{0,1\}^n$ and $i \in [n]$, we have:

$$\beta(\alpha^i) = (\beta(\alpha))^{B(x_i)},$$

where $\beta(\alpha)^{B(x_i)}$ is obtained by starting with $\beta(\alpha)$ and flipping the coordinates in $B(x_i)$.

Now suppose that $c : \{0,1\}^m \to [m]$ is a proper coloring of the m-dimensional Boolean hypercube. Then c restricted to the constraint vectors $S(C)$ defines a function $f_c : \{0,1\}^n \to [m]$ that solves the search problem associated with C. By the proof of Theorem 7, any function that solves the search problem for C has sensitivity $\Omega(\sqrt{n})$. Let $\alpha \in \{0,1\}^n$ be an input of maximal sensitivity, and let $S \subseteq [n]$, $|S| = \Omega(\sqrt{n})$, be the set of sensitive coordinates: for all $i \in S$, $f_c(\alpha) \neq f_c(\alpha^i)$.

By our assumption on C (which follows by expansion), there exists a subset $S' \subseteq S$ of size at least $\epsilon|S|$ such that the sets of coordinates/constraints, $\{B(x_i) \mid i \in S'\}$ are pairwise disjoint. Now we claim that $\beta(\alpha) \in \{0,1\}^m$ has block sensitivity $|S'|$, where the sensitive blocks are: $\{B(x_i) \mid i \in S'\}$.

First, by construction the blocks are pairwise disjoint. Secondly we want to show that for each $i \in [S']$, $c(\beta(\alpha)) \neq c(\beta(\alpha)^{B(x_i)})$. Since the constraints of C are parity constraints, flipping the value of any variable x_i flips the value of each constraint containing x_i. That is, the assignment α^i corresponds to the constraint vector $\beta(\alpha^i) = \beta(\alpha)^{B(x_i)}$. Since i is a sensitive coordinate for f_c with respect to α, $f_c(\alpha) \neq f_c(\alpha^i)$, and therefore $c(\beta(\alpha)) \neq c(\beta(\alpha)^{B(x_i)})$. ◀

We leave open the following conjecture which is a strengthening of the above theorem.

▶ **Conjecture 16.** *Let c be any proper coloring of the Boolean cube. Then there exists an assignment $\beta \in \{0,1\}^m$ such that β has at least a constant fraction of 1's and such that β has $\Omega(n)$ sensitivity.*

We also state another conjecture that strengthens the theorem in a different way. A vector $\beta \in \{0,1\}^m$ is *b-colorful* with respect to a proper coloring c if the set of colors associated with β plus all of the neighbors of β is at least b.

▶ **Conjecture 17.** *Let c be any proper coloring of the m-dimensional Boolean hypercube. then there exists $\beta \in \{0,1\}^m$ such that β has a constant fraction of 1's, and β is $\Omega(n)$-colorful.*

5.2 Size Lower Bounds and Pseudo-deterministic Resolution

The rich theory of TFNP and its subclasses (PPA, PPAD, PLS, etc) are defined based on the underlying combinatorial axiom required to *prove* the totality of functions in the class. Thus it is not surprising that there are strong connections between many TFNP subclasses and corresponding proof systems. For example it is known that FP is complete for the bounded arithmetic theory S^1_2 (in the sense that the TFNP problems definable in S^1_2 are the functions in FP), and similarly PLS is complete for the theory T^1_2.

The query complexity of subclasses of TFNP corresponds to studying the subclasses relative to an oracle. In the query world FP becomes P^{dt} and PLS becomes $\mathsf{PLS}^{\mathsf{dt}}$. The corresponding relativized systems of bounded arithmetic, $S^1_2(R)$ and $T^1_2(R)$, are uniform versions of the propositional proof systems TreeRes (Tree-like Resolution) and Res (dag-like Resolution).

For many weak propositional proof systems, there is an *equivalence* between minimal-size proofs of unsatisfiable formulas C and the query complexity of solving the search problem $\mathcal{S}_C$ in a corresponding query model. In this section we will use this equivalence to define *pseudo-deterministic* Resolution – a new notion that lies between ordinary Resolution and the much stronger notion of *Random Resolution*. Building on our pseudo-deterministic query lower bound, we exponentially separate *pseudo-deterministic* tree-like Resolution from Random Resolution.

5.2.1 Pseudo-deterministic Resolution

We start by defining some dag-like query models and review the known equivalences between Resolution and its common subsystems and their query model counterparts.

▶ **Definition 18** (Conjunction DAGs). *Consider the n-bit input domain $\{0,1\}^n$ and let $\mathcal{F}$ be the set of all conjunctions of literals over the n input variables. An $\mathcal{F}$-DAG, Π, solving a search problem $\mathcal{S} \subseteq \{0,1\}^n \times [m] \in \mathsf{TFNP}^{\mathsf{dt}}$ is a directed acyclic graph of fanout at most two, where each node v is associated with a function $f_v \in \mathcal{F}$. (The set $f_v^{-1}(1)$ is called the feasible set for v) and satisfying the following conditions:*

- *There is a distinguished root node r and $f_r = 1$ (the constant 1 function).*
- *For each non-leaf node v with children u, u', we have $f_v^{-1}(1) \subseteq f_u^{-1}(1) \cup f_{u'}^{-1}(1)$.*
- *Each leaf node v is labelled with an output $o_v \in [m]$ such that $f_v^{-1}(1) \subseteq \mathcal{S}^{-1}(o_v)$.*

The size of Π is the number of vertices in the dag. The width of Π is the maximum width of a conjunction associated with a node of Π.

▶ **Theorem 19.** *Let C be an unsatisfiable k-CNF formula and let $\mathcal{S}_C$ be the associated search problem. The following equivalences hold:*

1. *The minimum width Resolution refutation of C is equivalent (to within constant factors) to the minimum width of a conjuction-DAG for $\mathcal{S}_C$ [31, 32].*
2. *The minimum size Resolution refutation of C is equivalent (to within constant factors) to the minimum size conjunction-DAG for $\mathcal{S}_C$. [31, 32].*
3. *The minimum size Regular Resolution refutation of C is equivalent to the minimum-size read-once Branching program for $\mathcal{S}_C$ [27].*
4. *The minimum size tree-like Resolution refutation of C is equivalent to the minimum size deterministic decision tree for $\mathcal{S}_C$.*

With these equivalences in hand, we easily obtain natural *pseudo-deterministic* versions of these proof systems, stated next for Resolution and its common subsystems.

▶ **Definition 20.** *Let $\mathcal{C}$ be an unsatisfiable k-CNF formula. A pseudo-deterministic tree-like Resolution refutation of C is a pseudo-deterministic decision tree for $\mathcal{S}_C$. Let the minimal-size pseudo-deterministic* TreeRes *refutation for C be equal to $\mathsf{psP}^{\mathsf{dt}}(\mathcal{S}_C)$. Similarly the pseudo-deterministic regular Resolution complexity of C is the pseudo-deterministic read-once branching program size for $\mathcal{S}_C$, and the pseudo-deterministic Resolution complexity of C is the pseudo-deterministic dag-like query complexity of $\mathcal{S}_C$.*

It is not hard to see that pseudo-deterministic TreeRes, Res refutations are sound, and at least for TreeRes, pseudo-deterministic proofs can be efficiently verified. We want to compare pseudo-deterministic Resolution (and its subsystems) to Random Resolution (defined in [7] (following a suggestion by S. Danchev), where it was motivated by the open problem of proving a strict depth hierarchy for bounded-depth Frege systems.

▶ **Definition 21.** *A random Resolution refutation* (RR) *of an unsat CNF formula F over $x_1, \ldots, x_n$ is a distribution π on pairs (w_i, E_i), $i \in [q]$ such that:*
1. *Each E_i is a CNF formula in $x_1, \ldots, x_n$;*
2. *For each $i \in [q]$, w_i is a Resolution refutation of $F \wedge E_i$;*
3. *For all $\alpha \in \{0,1\}^n$, $Pr_{i \sim \pi}[E_i(\alpha) = 1] \geq 3/4$*

The size of the proof is $\sum_i (|w_i| + size(E_i))$.

Similarly one can define random tree-like and regular) Resolution proofs, where now each w_i is a tree-like (regular) Resolution refutation of $F \wedge E_i$. Random Cutting Planes refutations were also defined in a similar manner by Sokolov [32].

Random Resolution turns out to be quite powerful, as is evidenced by the fact that random unsatisfiable k-CNF formulas have short RR refutations, and even short random tree-like refutations. For a random k-CNF with sufficiently many clauses, every assignment will falsify a constant fraction of the clauses and thus we can create the distribution $\{(w_i, E_i), i \in [q]\}$ to mimic the randomized strategy for finding a violated clause: for each clause C_i in F, let E_i be the negation of C_i. Clearly each formula $F \wedge E_i$ is unsatisfiable and has a very short tree-like proof, since C_i together with E_i is contradictory. Secondly since every assignment is falsified by $1 - \epsilon$ fraction of clauses, $Pr_i[E_i(\alpha) = 1] \geq 1 - \epsilon$. Using this fact together with the PCP theorem, Pudlak and Thapen [7] observed that no polynomial-time verifier, or even a randomized verifier, can check a RR refutation (or even a tree-like refutation) efficiently unless P = NP (or BPP = NP).

The following theorem shows that a natural random distribution of formulas exponentially separates pseudo-deterministic TreeRes size from random TreeRes size.

▶ **Theorem 22.** *For all constant $k \geq 3$, there exists a family of k-CNF (k-XOR) formulas $\{F_n\}_{n \in \mathbb{N}}\}$ such that:*
- *The formulas F_n admit linear-size random* TreeRes *refutations;*
- *For n sufficiently large and $m = O(n)$ sufficiently large, any pseudo-deterministic* TreeRes *refutation of F_n requires size $exp(\Omega(\sqrt{n}))$.*

Proof. The formula F_n will be obtained by two steps. First we will choose a $k/2$-CNF ($k/2$-XOR) formula, f_n, such that its clause variable graph is expanding. For example a random formula chosen with $m = O(n)$ clauses (XOR equations) will suffice. Secondly we obtain F_n by composing f_n with a 2-bit gadget g. That is, each variable x_i will be replaced by $g(x_i^a, x_i^b)$, where x_i^a, x_i^b are twin variables replacing x_i. For f_n an expanding CNF formula, we define the gadget g to be the parity function, $g(a,b) = a \oplus b$ and for f_n an XOR formula, $g(a,b) = a \vee b$. We then rewrite $f_n \circ g_n$ as a k-CNF, clause-by-clause. Since f_n is a $k/2$-CNF formula with n variables and m clauses, F_n will be a k-CNF formula with $2n$ variables and $m \cdot 2^k$ clauses.

Fix n sufficiently large, and let $\mathcal{T}$ be a pseudo-deterministic TreeRes refutation of F_n, where each tree $T_i \in \mathcal{T}$ has size at most s. Define $size(\mathcal{T})$ to be the sum of the sizes of all trees in $\mathcal{T}$. First we remark that by Newman's theorem, we can assume that the number of trees (i.e. the amount of randomness required) is polynomial in the size of each tree, and thus counting the total size of all trees combined, rather than the max tree size, is justified.

Let $\mathcal{R} \subseteq \{0, 1, *\}^{2n}$ be the uniform distribution over the family of restrictions ρ such that: for all $i \in n$, exactly one variable in the pair (x_i^a, x_i^b) is set to 0 or 1 and the other variable in the pair is set to $*$. That is, $(x_i^a|_\rho, x_i^b|_\rho) \subseteq \{(*,0), (*,1), (0,*), (1,*)\}$. Let $\mathcal{T}$ be a size s pseudo-deterministic TreeRes refutation of F_n. Let $terms(\mathcal{T})$ be the set of all terms (partial assignments) associated with all paths in all trees, T_i, and let $wide \subseteq terms(\mathcal{T})$ be those terms in $terms(\mathcal{T})$ of width at least w, $w = O(\sqrt{n})$.

For a fixed term $t \in wide(\mathcal{T})$, the probability that a random $\rho \in \mathcal{R}$ does not set t to zero is at most $(3/4)^w$. By the union bound, the probability that there exists $\rho \in \mathcal{R}$ that sets all wide terms to zero is at least $1 - s(3/4)^w$ which is greater than zero for $\log s = O(w)$. Thus there exists a restriction setting all wide terms of $\mathcal{T}$ to zero.

Applying ρ to $\mathcal{T}$, and to F_n, we obtain a pseudo-deterministic TreeRes refutation $\mathcal{T}'$ of $F_n|_\rho$ of size at most n and of depth at most w. Since $F_n|_\rho$ is just a copy of f_n, by the expansion properties of f_n, we can apply Theorem 7 which states that any pseudo-deterministic decision tree for f_n must have depth $\Omega(\sqrt{n})$, and thus $s = \Omega(exp(\sqrt{n}))$. ◀

5.2.2 Pseudo-deterministic Algebraic Proofs

By the relationship between low-degree polynomials solving $\mathcal{S}_C$ and low-degree Nullstellensatz refutations of C given in Lemma 12, we can define pseudo-deterministic Nullstellensatz refutations to be pseudo-deterministic polynomials solving $\mathcal{S}_C$.

▶ **Definition 23.** *Let C be an unsatisfiable k-CNF and let $\mathcal{S}_C$ be the corresponding search problem. Then a pseudo-deterministic degree d Nullstellensatz refutation over $\mathbb{F}$ is a distribution over polynomials $\mathcal{P} = \{P^1, \ldots P^q\}$ over $\mathbb{F}$ such that each $P^i : \{0,1\}^n \to [m]$ has degree at most d and such that there exists a function f solving $\mathcal{S}_C$ such that $\mathcal{P}$ probabilistically computes f: for all inputs $x \in \{0,1\}^n$, $Pr_{i\in[q]}[P^i(x) = f(x)] \geq 3/4$.*

We note that the degree of Nullstellensatz refutations of C over $\mathbb{F}_2$ have also been shown to be equivalent to the $\mathsf{PPA}^{\mathsf{dt}}$ query complexity of $\mathcal{S}_C$ [19]. (Intuitively there is a degree-d $\mathsf{PPA}^{\mathsf{dt}}$ query algorithm for $\mathcal{S}$ if there is a depth-d decision tree reduction from $\mathcal{S}$ to an instance of PPA. See [19] for a formal definition.)

Over the reals, $\Omega(n^\epsilon)$ lower bounds for pseudo-deterministic Nullstellensatz refutations follow from our pseudo-deterministic query lower bound for $\mathcal{S}_C$ for random C. This is because a family of polynomials computing a function f that solves the search problem implies the existence of an *approximate* polynomial of the same degree for solving f (that is, polynomials p_i that pointwise are within ϵ of $f^i(x)$ for all x.) And polynomial degree is polynomially related to approximate-degree for Boolean functions over the reals [29].

It is interesting to study similar relationships for other, stronger algebraic proof systems such as Sherali Adams (SA) and Sum-of-Squares. Can low degree proofs be characterized or lower bounded by the complexity of a family of pseudo-deterministic algebraic objects for solving the associated search problem?

6 Average Case Pseudo-deterministic Simulations

In this section, we study pseudo-deterministic simulations of randomized query algorithms in the average-case setting. We first show that for any search problem $\mathcal{S}$, the existence of zero-error randomized algorithms with low query complexity on average over a distribution D implies the existence of *deterministic* algorithms with low query complexity on average over D (and hence also of zero-error pseudo-deterministic algorithms). In the bounded-error setting, we show that for any search problem $\mathcal{S}$ solving an approximation problem, the existence of bounded-error randomized algorithms with low query complexity implies that for any D, there is a bounded-error pseudo-deterministic algorithm with low query complexity on average over D.

We first define what it means to solve search problems efficiently on average by a pseudo-deterministic algorithm. We adopt the strongest reasonable definition of average-case solvability: the algorithm must be pseudo-deterministic and solve the problem correctly on

every input, and must have low query complexity on average over the distribution on inputs (and randomness of the algorithm). Adopting a strong notion of solvability makes our results stronger, as our results are mainly simulation results.

▶ **Definition 24.** *Let D be a distribution over $\mathcal{X} \subseteq \{0,1\}^n$. We say that a search problem $\mathcal{S}$ over domain $\mathcal{X}$ is solvable on average over D by a bounded-error pseudo-deterministic query algorithm with complexity q if there is a randomized query algorithm A that is bounded-error pseudo-deterministic and solves $\mathcal{S}$ correctly with probability $\geq 2/3$ on each input in $\mathcal{X}$, and moreover the expected number of queries of A (over the randomness of A and the distribution D) is at most q. Similarly, we say that a search problem $\mathcal{S}$ over domain $\mathcal{X}$ is solvable on average over D by a zero-error pseudo-deterministic query algorithm with complexity q if there is a randomized query algorithm A that is zero-error pseudo-deterministic and solves $\mathcal{S}$ correctly with probability 1 on each input in $\mathcal{X}$, and moreover the expected number of queries of A (over the randomness of A and the distribution D) is at most q. If A is deterministic, we say that $\mathcal{S}$ is solvable on average over D by a deterministic query algorithm with complexity q.*

We first show that the canonical problem FIND1 (which is solvable efficiently by zero-error query algorithms) has low average-case deterministic query complexity over any distribution.

▶ **Proposition 25.** *Let D be any distribution on the domain of FIND1 restricted to n-bit inputs. FIND1 is solvable on average over D by a deterministic query algorithm with complexity $\log(n) + 1$.*

Proof. Let D be any distribution on the domain of FIND1 restricted to n-bit inputs. Let R be a subset of $[n]$ of size $\log(n)$ where R is chosen uniformly at random over all such subsets. Since FIND1 is defined over inputs x with $|x| \geq n/2$, we have that for each x in the domain of FIND1, the probability that there is a $j \in R$ such that $x_j = 1$ is at least $1 - 1/n$. By averaging, there is a subset B of $[n]$ of size $\log(n)$ such that with probability at least $1 - 1/n$ over D, $x_j = 1$ for some $j \in B$ when x is chosen from D.

Consider the following deterministic query algorithm A. A queries the indices in B in lexicographic order, and outputs the first such index j for which $x_j = 1$, if such an index exists. If no such index exists, A queries the indices in $[n] \setminus B$ in lexicographic order, and outputs the first index j for which $x_j = 1$. Since FIND1 is only defined over n-bit inputs with at least one 1, this query algorithm is correct. Call an input x in the domain of FIND1 "good" if there is a $j \in B$ such that $x_j = 1$. With probability at least $1 - 1/n$ over x chosen from D, x is good and the query algorithm A uses at most $\log(n)$ queries. When x is not good, A uses at most n queries. Thus the query complexity is at most $(1 - 1/n) \cdot \log(n) + n \cdot 1/n \leq \log(n) + 1$ on average over D. ◀

Next we significantly generalize Proposition 25 and show that efficient average-case solvability by zero-error randomized algorithms is in fact equivalent to efficient average-case solvability by deterministic algorithms (and hence also by zero-error pseudo-deterministic algorithms).

▶ **Theorem 26.** *Let $\mathcal{S}$ be a total search problem over domain $\mathcal{X} \subseteq \{0,1\}^n$, D a distribution over $\mathcal{X}$, and $q : \mathbb{N} \to \mathbb{N}$ a function. The following are equivalent:*

1. *$\mathcal{S}$ is solvable on average over D by a zero-error query algorithm with complexity $O(q(n)\mathsf{polylog}(n))$.*
2. *$\mathcal{S}$ is solvable on average over D by a zero-error pseudo-deterministic query algorithm with complexity $O(q(n)\mathsf{polylog}(n))$.*
3. *$\mathcal{S}$ is solvable on average over D by a deterministic query algorithm with complexity $O(q(n)\mathsf{polylog}(n))$.*

Proof. The third item trivially implies the second, and the second item trivially implies the first. We show that the first item implies the third.

Suppose $\mathcal{S}$ is solvable on average over D by a zero-error query algorithm with complexity $r(n) = O(q(n)\mathsf{polylog}(n))$. This implies that there is a distribution D' over deterministic query algorithms such that for every input x in $\mathcal{X}$, a query algorithm A chosen from D' solves $\mathcal{S}$ with probability at least $2/3$ over D', and moreover the expected number of queries over A chosen from D' and x chosen from D is at most $r(n)$. Without loss of generality, we can assume that D' is uniform over a multi-set Y of deterministic query algorithms. If this multi-set has size K, sampling from D' is equivalent to sampling uniformly from $[K]$. From now on, we assume a bijection between $[K]$ and Y, and also assume without loss of generality that $K \geq 4\log(n)$.

For positive integral t, define an input $x \in \mathcal{X}$ to be t-good if it is the case that with probability at least $1/6$ over choice of A from D', A solves S correctly on x making at most $2tr(n)$ queries. We argue that for each t, x chosen from D is t-good with probability at least $1 - 1/t$. The proof is by contradiction. Suppose this were not the case. Then for some positive integer t, with probability greater than $1/t$ over x chosen from D, x is not t-good. If x is not t-good, then with probability at least $5/6$ over choice of A from D', A either does not return a solution for S or makes more than $2tr(n)$ queries. Since for any $x \in \mathcal{X}$, A solves x with probability at least $2/3$, it must be the case that with probability at least $1/2$ over choice of A from D', A makes more than $2tr(n)$ queries on x when x is not t-good. Since the probability over D that x is not t-good is greater than $1/t$, this implies that when x is sampled from D and A from D', the expected number of queries is greater than $r(n)$, in contradiction to the assumption that the zero-error query algorithm corresponding to D' has complexity at most $r(n)$.

Now consider a t-good $x \in \mathcal{X}$. Say that $k \in [K]$ is t-suitable for x if running the k'th deterministic query algorithm from Y on x succeeds in solving $\mathcal{S}$ on x while making at most $2tr(n)$ queries. Since x is t-good, k chosen uniformly from $[K]$ is suitable for x with probability at least $1/6$. Let R be a subset of $[K]$ of size $4\log(n)$ chosen uniformly at random from all subsets of this size. With probability at least $1 - 1/n$, R contains $j \in [K]$ such that j is t-suitable for x. Say that R is t-suitable for x if this is the case.

Let $\mu(x)$ be the smallest positive integer t such that x is t-good. For any x, we have that $\mu(x) \leq n$.

By averaging, there is a subset B of $[K]$ of size $4\log(n)$ such that with probability at least $1 - 1/n$ over x sampled from D, B is $\mu(x)$-suitable for x. Consider the query algorithm A that works as follows. It runs the query algorithms corresponding to the elements of B in an interleaving fashion. Namely, if the elements of B are $b_1 \dots b_{4\log(n)}$, it makes the first query of the b_j'th algorithm for each $j \in [4\log(n)]$ in order, then the second query for each of these algorithms, and so on until it has made enough queries for a given algorithm so that the algorithm outputs an answer. Naturally, it never repeats a query that it has already been made. Note that A halts after making at most $8\mu(x)\log(n)r(n)$ queries.

We bound the expected number of queries made by A for x chosen from distribution D. With probability at most $1/n$, B is not $\mu(x)$-suitable for x, and in this case A makes at most n queries on x. When B is $\mu(x)$-suitable, A halts and outputs a correct solution for $\mathcal{S}$ on x after making at most $8\mu(x)\log(n)r(n)$ queries. For each integer $i \in [\lceil \log(n) \rceil]$, we have that the probability over x sampled from D that $\mu(x) \leq 2^i$ is at least $1 - 1/2^i$. Computing the expectation of the running time of A by summing over $1 \leq i \leq \log(n)$ such that $2^i < \mu(x) \leq 2^{i+1}$, we have that the contribution to the expectation when B is $\mu(x)$-suitable is at most $(1/2 \cdot 2 + 1/4 \cdot 4 + \dots)8\log(n)r(n) \leq 16(\log(n))^2 r(n)$. Thus, the total expectation is at most $16(\log(n))^2 r(n) + 1 = O(q(n)\mathsf{polylog}(n))$, as desired. ◀

Next, we turn to bounded-error average-case solvability. We show that the ϵ-HWE problem of approximating the Hamming weight of a string to within an additive term ϵ is solvable efficiently on average by bounded-error pseudo-deterministic query algorithms. We note that Goldreich, Goldwasser and Ron [13] showed an $\Omega(n)$ query lower bound for *worst-case* bounded-error pseudo-deterministic algorithms solving this problem.

▶ **Theorem 27.** *For any distribution D and any constant $\epsilon > 0$, ϵ-HWE is solvable on average over D by bounded-error pseudo-deterministic algorithms of complexity $O(\log(n)/\epsilon^2)$.*

Proof. We use the fact that, on any x, if we take a random sample of bits of size $q = O(\log(n)/\epsilon^2)$, the empirical average of ones of this sample differs from that of x by $\epsilon/4$ with probability at most $1/5n$, using a standard Chernoff-Hoeffding bound. By averaging, for every distribution D there must be some fixed such subset of bits with this property, when we take the expectation over random x from D. Call this subset A, and let $d_A(x)$ be the empirical estimate of the density of x based on the bits in A. Let B represent a uniform random subset of bits of size q, and let $d_B(x)$ represent the empirical estimate of the density of x based on the bits in B. Let $d(x)$ represent the actual density of x.

Let $p(x)$ be the function : $p(x) = d_A(x)$ if $Prob_B[|d_B(x) - d_A(x)| \leq \epsilon/2] > 1/5$, and $p(x) = d(x)$ otherwise. $p(x)$ is a fixed function of x, and it is always a good approximation to $d(x)$, since if it is not literally $d(x)$, it is $\epsilon/2$ close to $d_B(x)$ for most B, and a random B has $d_B(x)$ $\epsilon/2$ close to $d(x)$.

Consider the following algorithm for computing $p(x)$:

1. Compute $d_A(x)$.
2. Choose a random B of size q.
3. Compute $d_B(x)$
4. If $|d_B(x) - d_A(x)| \leq \epsilon/2$, return $d_A(x)$
5. Otherwise, query all bits of x, and compute $p(x)$. Return $p(x)$.

We claim that this algorithm returns $p(x)$ on any x except with probability at most $1/5$. Case 1: If $Prob_B[|d_B(x) - d_A(x)| \leq \epsilon/2] > 1/5$, then $p(x) = d_A(x)$. Then we either return the correct value in step 4, or we go on to compute the correct value in step 5. Either way, the algorithm is always correct.

Case 2: If $Prob_B[|d_B(x) - d_A(x)| \leq \epsilon/2] \leq 1/5$, then by definition, we return a value in step 4 with probability at most $1/5$. Thus, on such an input, with probability at least $4/5$, we go on to compute and return $p(x)$ by brute force in step 5.

Finally, we bound the expected number of bits queried by the algorithm over a random x from D. Over such random x, with probability $1 - 1/5n$, $|d_A(x) - d(x)| \leq \epsilon/4$, and for any x, with the same probability over B, $|d_B(x) - d(x)| \leq \epsilon/4$. If both of these happen, $|d_A(x) - d_B(x)| \leq \epsilon/2$ and the algorithm terminates in line 4 after making $2q$ queries. So the expected number of queries is at most $2q + 2/5n \cdot n = O(q)$. ◀

We observe that Theorem 27 generalizes to yield an efficient bounded-error pseudo-deterministic algorithm on average for any *approximation* problem with low bounded-error randomized query complexity. Given a metric Δ on a space $\mathcal{O}$ and a function $f : \mathcal{X} \to \mathcal{O}$, a search problem $\mathcal{S}$ with domain $\mathcal{X} \subset \{0,1\}^n$ and range $\mathcal{O}$ is said to be the ϵ-approximation problem for f if the solutions to $\mathcal{S}$ on input $x \in \mathcal{X}$ are all points $y \in \mathcal{O}$ for which $\Delta(y, f(x)) \leq \epsilon$.

▶ **Theorem 28.** *Let ϵ be a constant, $\mathcal{O}$ be a space with metric Δ and $f : \mathcal{X} \to \mathcal{O}$ be a function such that there is a randomized query algorithm with complexity q to $\epsilon/4$-approximate f. Then for any distribution D over $\mathcal{X}$ there is a pseudo-deterministic query algorithm A that ϵ-approximates f with query complexity $O(q \log(n))$ on average over D.*

The proof is a straightforward generalization of the proof of Theorem 27, and we therefore omit it.

We note that unlike with zero-error randomized query complexity, efficient bounded-error query algorithms are not in general efficiently simulated on average by deterministic query algorithms.

▶ **Proposition 29.** *Let D be any distribution assigning positive weight to every n-bit input. For any $\epsilon < 1/2$, ϵ-HWE has zero-error average-case query complexity $\Omega(n)$ over D.*

Proof. Let A be any zero-error query algorithm solving ϵ-HWE on average over D. A is a distribution over deterministic query algorithms. We note that for any deterministic query algorithm in the support of A, there is no path of length $< (1-2\epsilon)n$ with an output. If there were such a path, then the output would not be a correct ϵ-approximation either for the input on which all unqueried bits are 0 or for the input on which all unqueried bits are 1. Since both of these inputs have positive probability according to D, this would imply that A is not a correct zero-error algorithm for ϵ-HWE.

Now for any input x, since A is a correct zero-error query algorithm, it must return an output with probability at least $2/3$. By the previous paragraph, this means that the average number of queries over D is $\Omega(n)$. ◀

7 Open Problems

Here we record some open problems and directions that we leave open.

First, our lower bound is tight for pseudo-deterministic quantum query complexity. We conjecture that the bound for both FIND1 and $\mathcal{S}_C$ can be improved to $\Omega(n)$ for pseudo-determnistic query complexity. Such an improvement would have to bypass sensitivity (and approximate degree) since both incur a quadratic loss. Secondly, we leave open the question of proving superpolynomial or exponential lower bounds for pseudo-deterministic Resolution refutations.

More generally, it is very interesting to study pseudo-determinism in the realm of communication complexity. A pseudo-deterministic communication protocol for a search problem $\mathcal{S} = \{0,1\}^n \times \{0,1\}^n \times [m]$ is a distribution $\Pi = \{\pi_1, \ldots, \pi_q\}$ over deterministic protocols with the property that there exists a function $f_\Pi : \{0,1\}^n \times \{0,1\}^n \to [m]$ solving $\mathcal{S}$, where Π is a randomized protocol for f. That is, for every input $(x,y) \in \{0,1\}^n \times \{0,1\}^n$, $Pr_{i\in[q]}[\pi_i(x,y) = f(x,y)] \geq 3/4$.

Pseudo-deterministic communication complexity is interesting for several reasons. For Boolean functions an exciting body of work has culminated in what is now a nearly complete understanding of many query/degree measures and their pairwise relationships. In turn these query measures for Boolean functions have natural analogs in communication complexity, and lifting theorems give a way to lift query upper and lower bounds to their communication counterparts. However for search problems, we lack a good understanding of query measures and the relationships between them, and this in turn leads to a lack of clarity with respect to their communication analogs. For example, what is the analog of sensitivity and block-sensitivity for search problems? In [26] a notion called critical block sensitivity was defined, and used in [20, 18] to prove strong lower bounds on dynamic SOS and extended formulations on the exact compution of certain functions. Unfortunately critical block sensitivity is only defined for search problems containing inputs with a unique solution and therefore these tools cannot be used to prove inapproximability results. As a second example, extended formulation lower bounds have been proven by lifting semialgebraic degree lower bounds, but

applying the lifting framework to prove *inapproximability* lower bounds is quite subtle, in large part due to a lack of relaxed/approximate/pseudo-deterministic notions of query complexity for search problems (e.g., approximate notions of Sherali-Adams (SA) and Sum-of-Squares (SOS) degree.) Since pseudo-deterministic algorithms are just randomized algorithms for computing *some* function solving the search problem, they are central to the study of relaxed query measures for search problems.

Secondly, the pseudo-determinism communication complexity of Karchmer-Wigderson search problems is particularly interesting. It is well known that deterministic communication complexity lower bounds on the KW search problems associated with a Boolean function is equivalent to formula size lower bounds (and dag-like communication lower bounds are equivalent to circuit lower bounds). This equivalence has been quite successful for proving lower bounds in monotone models of computation where lifting theorems in communication complexity have been applied to prove a variety of state-of-the art lower bounds for monotone formulas, monotone span programs, monotone circuits, as well as extended formulations (which are also a monotone model as they relate to nonnegative rank).

An exciting direction towards proving *nonmonotone* circuit/formula lower bounds is to further develop lower bound techniques for monotone models to apply to more functions – such as slice functions or all small "perturbations" of the function [24]. Related to this, we note that the communication complexity of monotone KW games is quite different than that of non-monotone KW games: whereas the (nonmonotone) KW game for *any* f has a trivial $O(\log n)$ pseudo-deterministic protocol, the *monotone* KW game (for monotone f) in general appears to be hard pseudo-deterministically.

A reasonable approach for separating pseudo-determininistic from randomized communication is lifting. We conjecture that the lifted/composed functions $\text{FIND1} \circ g^n$ and $\mathcal{S}_C \circ g^n$ require large pseudo-deterministic communication complexity for good choices of g (such as the index function). We note that standard lifting theorems won't work in a black-box way since the pseudo-deterministic protocol can have different canonical solutions for different inputs $(\vec{x}, \vec{y}), (\vec{x}', \vec{y}')$ such that $g^n(\vec{x}, \vec{y}) = g^n(\vec{x}'\vec{y}')$. Nonetheless, pseudo-deterministic communication lower bounds should be possible by combining lifting (in a non-blackbox way) with the right pseudo-deterministic query lower bound argument. In this respect we view our pseudo-deterministic query lower bounds as a first step towards obtaining a similar separation in communication complexity.

.

References

1 Scott Aaronson, Shalev Ben-David, and Robin Kothari. Separations in query complexity using cheat sheets. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 863–876. ACM, 2016. doi:10.1145/2897518.2897644.

2 Scott Aaronson, Shalev Ben-David, Robin Kothari, and Avishay Tal. Quantum implications of huang's sensitivity theorem. *CoRR*, abs/2004.13231, 2020. arXiv:2004.13231.

3 Michael Alekhnovich and Alexander A. Razborov. Lower bounds for polynomial calculus: Non-binomial case. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 190–199, 2001.

4 Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials. *J. ACM*, 48(4):778–797, 2001. doi:10.1145/502090.502097.

5 Harry Buhrman and Ronald de Wolf. Complexity measures and decision tree complexity: a survey. *Theor. Comput. Sci.*, 288(1):21–43, 2002. doi:10.1016/S0304-3975(01)00144-X.

6 Samuel R. Buss, Dima Grigoriev, Russell Impagliazzo, and Toniann Pitassi. Linear gaps between degrees for the polynomial calculus modulo distinct primes. *J. Comput. Syst. Sci.*, 62(2):267–289, 2001.

7 Samuel R. Buss, Leszek Aleksander Kolodziejczyk, and Neil Thapen. Fragments of approximate counting. *J. Symb. Log.*, 79(2):496–525, 2014. doi:10.1017/jsl.2013.37.

8 Siu On Chan, James R. Lee, Prasad Raghavendra, and David Steurer. Approximate constraint satisfaction requires large LP relaxations. *J. ACM*, 63(4):34:1–34:22, 2016. doi:10.1145/2811255.

9 Richard Cleve. An introduction to quantum complexity theory. *Quantum Computation and Quantum Information Theory*, page 103–127, January 2001. doi:10.1142/9789810248185_0004.

10 Noah Fleming, Pravesh Kothari, and Toniann Pitassi. Semialgebraic proofs and efficient algorithm design. *Found. Trends Theor. Comput. Sci.*, 14(1-2):1–221, 2019. doi:10.1561/0400000086.

11 Ankit Garg, Mika Göös, Pritish Kamath, and Dmitry Sokolov. Monotone circuit lower bounds from resolution. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 902–911. ACM, 2018. doi:10.1145/3188745.3188838.

12 Eran Gat and Shafi Goldwasser. Probabilistic search algorithms with unique answers and their cryptographic applications. *Electron. Colloquium Comput. Complex.*, 18:136, 2011. URL: http://eccc.hpi-web.de/report/2011/136.

13 Oded Goldreich, Shafi Goldwasser, and Dana Ron. On the possibilities and limitations of pseudodeterministic algorithms. In *4th Innovations in Theoretical Computer Science Conference, ITCS*, pages 127–138, 2013.

14 Shafi Goldwasser and Ofer Grossman. Bipartite perfect matching in pseudo-deterministic NC. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, pages 87:1–87:13, 2017.

15 Shafi Goldwasser, Ofer Grossman, and Dhiraj Holden. Pseudo-deterministic proofs. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 17:1–17:18, 2018.

16 Shafi Goldwasser, Ofer Grossman, Sidhanth Mohanty, and David P. Woodruff. Pseudo-deterministic streaming. *CoRR*, abs/1911.11368, 2019. arXiv:1911.11368.

17 Shafi Goldwasser, Ofer Grossman, Sidhanth Mohanty, and David P. Woodruff. Pseudo-deterministic streaming. In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA*, volume 151 of *LIPIcs*, pages 79:1–79:25. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ITCS.2020.79.

18 Mika Göös, Rahul Jain, and Thomas Watson. Extension complexity of independent set polytopes. *SIAM J. Comput.*, 47(1):241–269, 2018. doi:10.1137/16M109884X.

19 Mika Göös, Pritish Kamath, Robert Robere, and Dmitry Sokolov. Adventures in monotone complexity and TFNP. In Avrim Blum, editor, *10th Innovations in Theoretical Computer Science Conference, ITCS 2019, January 10-12, 2019, San Diego, California, USA*, volume 124 of *LIPIcs*, pages 38:1–38:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ITCS.2019.38.

20 Mika Göös and Toniann Pitassi. Communication lower bounds via critical block sensitivity. *SIAM J. Comput.*, 47(5):1778–1806, 2018. doi:10.1137/16M1082007.

21 Dima Grigoriev. Tseitin's tautologies and lower bounds for nullstellensatz proofs. In *39th Annual Symposium on Foundations of Computer Science, FOCS '98, November 8-11, 1998, Palo Alto, California, USA*, pages 648–652, 1998.

22 Ofer Grossman and Yang P. Liu. Reproducibility and pseudo-determinism in log-space. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 606–620, 2019.

23 Lov K. Grover. A fast quantum mechanical algorithm for database search. In Gary L. Miller, editor, *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 212–219. ACM, 1996. doi:10.1145/237814.237866.

24 Pavel Hrubes. On ϵ-sensitive monotone computations. *Comput. Complex.*, 29(2):6, 2020. doi:10.1007/s00037-020-00196-6.

25 Hao Huang. Induced subgraphs of hypercubes and a proof of the sensitivity conjecture. *CoRR*, abs/1907.00847, 2019. arXiv:1907.00847.

26 Trinh Huynh and Jakob Nordström. On the virtue of succinct proofs: amplifying communication complexity hardness to time-space trade-offs in proof complexity. In Howard J. Karloff and Toniann Pitassi, editors, *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 233–248. ACM, 2012. doi:10.1145/2213977.2214000.

27 László Lovász, Moni Naor, Ilan Newman, and Avi Wigderson. Search problems in the decision tree model. *SIAM J. Discret. Math.*, 8(1):119–132, 1995. doi:10.1137/S0895480192233867.

28 Noam Nisan. CREW PRAMs and decision trees. *SIAM Journal on Computing*, 20(6):999–1007, 1991.

29 Noam Nisan and Mario Szegedy. On the degree of boolean functions as real polynomials. *Comput. Complex.*, 4:301–313, 1994. doi:10.1007/BF01263419.

30 Igor Carboni Oliveira and Rahul Santhanam. Pseudodeterministic constructions in subexponential time. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 665–677. ACM, 2017. doi:10.1145/3055399.3055500.

31 A. A. Razborov. Unprovability of lower bounds on circuit size in certain fragments of bounded arithmetic. *Izvestiya RAN. Ser. Mat.*, pages 201–224, 1995.

32 Dmitry Sokolov. Dag-like communication and its applications. In Pascal Weil, editor, *Computer Science - Theory and Applications - 12th International Computer Science Symposium in Russia, CSR 2017, Kazan, Russia, June 8-12, 2017, Proceedings*, volume 10304 of *Lecture Notes in Computer Science*, pages 294–307. Springer, 2017. doi:10.1007/978-3-319-58747-9_26.