Fine-Grained Derandomization: From Problem-Centric to Resource-Centric Complexity

Marco L. Carmosino[1]
Department of Computer Science, University of California San Diego, La Jolla, CA, USA
marco@ntime.org

Russell Impagliazzo[2]
Department of Computer Science, University of California San Diego, La Jolla, CA, USA
russell@cs.ucsd.edu

Manuel Sabin[3]
Computer Science Division, University of California Berkeley, Berkeley, CA, USA
msabin@berkeley.edu

Abstract

We show that popular hardness conjectures about problems from the field of fine-grained complexity theory imply structural results for resource-based complexity classes. Namely, we show that if either k-Orthogonal Vectors or k-CLIQUE requires $n^{\epsilon k}$ time, for some constant $\epsilon > 1/2$, to *count* (note that these conjectures are significantly weaker than the usual ones made on these problems) on randomized machines for all but finitely many input lengths, then we have the following derandomizations:

- BPP can be decided in polynomial time *using only* n^{α} *random bits* on average over *any* efficient input distribution, for any constant $\alpha > 0$
- BPP can be decided in polynomial time with *no randomness* on average *over the uniform distribution*

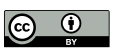
This answers an open question of Ball et al. (STOC '17) in the positive of whether derandomization can be achieved from conjectures from fine-grained complexity theory. More strongly, these derandomizations improve over all previous ones achieved from *worst-case uniform* assumptions by succeeding on all but finitely many input lengths. Previously, derandomizations from worst-case uniform assumptions were only know to succeed on infinitely many input lengths. It is specifically the structure and moderate hardness of the k-Orthogonal Vectors and k-CLIQUE problems that makes removing this restriction possible.

Via this uniform derandomization, we connect the problem-centric and resource-centric views of complexity theory by showing that exact hardness assumptions about specific problems like k-CLIQUE imply quantitative and qualitative relationships between randomized and deterministic time. This can be either viewed as a barrier to proving some of the main conjectures of fine-grained complexity theory lest we achieve a major breakthrough in unconditional derandomization or, optimistically, as route to attain such derandomizations by working on very concrete and weak conjectures about specific problems.

2012 ACM Subject Classification Theory of computation → Complexity classes

Keywords and phrases Derandomization, Hardness vs Randomness, Fine-Grained Complexity, Average-Case Complexity, k-Orthogonal Vectors, k-CLIQUE

[1] Supported by the Simons Foundation
[2] Supported by the Simons Foundation
[3] Supported by the National Science Foundation Graduate Research Fellowship under Grant #DGE-1106400



45th International Colloquium on Automata, Languages, and Programming (ICALP 2018).
Editors: Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella;
Article No. 27; pp. 27:1–27:16

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Digital Object Identifier 10.4230/LIPIcs.ICALP.2018.27

Related Version A full version of the paper is available at `https://eccc.weizmann.ac.il/report/2018/092/`.

Acknowledgements We would like to thank Benjamin Caulfield, Andrea Lincoln, Luca Trevisan, Prashant Nalini Vasudevan, Ryan Williams, and Virginia Vassilevska Williams for helpful discussions and comments.

1 Introduction

Computational complexity can be viewed through two main perspectives: problem-centric or resource-centric. Problem-centric complexity theory asks what resources are required to solve *specific problems*, while resource-centric complexity deals with the relative power of different computational models given different resource budgets such as time, memory, non-determinism, randomness, etc. (see [17] for a discussion). Through complete problems, these two perspectives often coincide, so that a resource-centric view acts as a fine proxy for answering questions about the complexity of specific problems. The rapidly progressing field of fine-grained complexity theory, however, brings attention back to the problem-centric viewpoint, raising fine distinctions even between problems complete for the same complexity class, and making connections between problems at very different levels of complexity. To what extent are these two approaches linked, i.e., to what extent can inferences about the fine-grained complexities of specific problems be made from general assumptions about complexity classes, and vice versa?

Here, we examine such links between the fine-grained complexity of specific problems such as the k-Orthogonal Vectors and k-CLIQUE problems and general results about derandomization of algorithms. Derandomization has been a very fruitful study in complexity theory, with many fascinating connections between lower bounds, showing that problems require large amounts of resources to solve, and upper bounds, showing that classes of probabilistic algorithms can be 'derandomized' by simulating them deterministically in a non-trivial fashion. In particular, the hardness-to-randomness framework shows that in many cases, the existence of any "hard" problem can be used to derandomize classes of algorithms. We reconsider this framework from the fine-grained, problem-centric perspective. We show that replacing a generic hard problem with specific hardness conjectures from fine-grained complexity leads to quantitatively and qualitatively stronger derandomization results than one gets from the analogous assumption about a generic problem. In particular, we show that starting from these assumptions, we can simulate any polynomial-time probabilistic algorithm (on any samplable distribution on inputs with a very small fraction of errors) by a polynomial time probabilistc algorithm that uses only n^α random coins, for any $\alpha > 0$. This type of derandomization previously either assumed the existence of cryptographic One-Way Functions or *exponential non-uniform* hardness of Boolean functions.

Thus, the problem-centric conjectures of fine-grained complexity cannot live in isolation from classical resource-centric consequences about the power of randomness. Viewed another way, our results can be seen as a barrier to proving some of the key hardness assumptions used by fine-grained complexity theory. That is, despite recent progress towards proving hardness for k-Orthogonal Vectors, one of fine-grained complexity's key problems, in restricted models of computation [29], doing so for general randomized algorithms would immediately prove *all* problems in BPP are easy on average (over, say, uniformly chosen inputs).

1.1 Our Results

We obtain two main theorems about the power of BPP from uniform worst-case assumptions about well-studied problems from fine-grained complexity theory. We consider the k-Orthogonal Vectors (k-OV) and the k-CLIQUE problems, defined and motivated in Section 2.1, and show that (even weaker versions of) popular conjectures about their hardness give two flavors of average-case derandomization that improve over the classical *uniform* derandomizations.

All previous derandomizations from *uniform* assumptions on worst-case hardness only succeed on *infinitely many input lengths*. Our work is the first to use worst-case uniform assumptions to derandomize BPP *for all but finitely many input lengths*, giving a standard inclusion. The only other worst-case uniform assumptions known to imply such results are those so strong as to imply cryptographic assumptions or circuit lower bounds, fitting closer to the cryptographic or non-uniform derandomization literature. In contrast, our *uniform derandomizations* are from extremely weak worst-case uniform conjectures on simple, natural, combinatorial problems. Informally, we prove the following:

▶ **Informal Theorem 1.** *If k-OV or k-CLIQUE requires $n^{\epsilon k}$ time, for some constant $\epsilon > 1/2$, to **count** on randomized machines in the worst-case for all but finitely many input lengths, then BPP can be decided in polynomial time **using only** n^{α} **random bits** on average over **any** efficient input distribution, for any constant $\alpha > 0$.*

Randomness can be removed entirely by simply brute-forcing all random bits and taking the majority of the outputs to give the following more standard full derandomization.

▶ **Corollary.** *If k-OV or k-CLIQUE requires $n^{\epsilon k}$ time, for some constant $\epsilon > 1/2$, to **count** on randomized machines in the worst-case for all but finitely many input lengths, then BPP can be decided with **no randomness** in sub-exponential time on average over **any** efficient input distribution.*

This conclusion is strictly stronger than the classic uniform derandomizations of [25, 38]. The weakest uniform assumption previously known to imply such a conclusion was from those already strong enough to imply the cryptographic assumption of the existence of One-Way Functions that are hard to invert for polynomial time adversaries [9, 19, 20, 23, 42] or those implying *non-uniform* circuit lower bounds [4].

Our second main theorem, using techniques from [31], shows how to remove all randomness *within polynomial time* if the distribution over inputs is uniform. The only stronger derandomization from uniform assumptions were, again, from those already strong enough to imply circuit lower bounds or the cryptographic assumption of the existence of One-Way Permutations that require *exponential* time to invert [9, 20, 42].

▶ **Informal Theorem 2.** *If k-OV or k-CLIQUE require $n^{\epsilon k}$ time, for some constant $\epsilon > 1/2$, to **count** on randomized machines in the worst-case for all but finitely many input lengths, then BPP can be decided in polynomial time with **no randomness** on average over **the uniform distribution**.*

1.2 Related Work

Connections Between Problem-Centric and Resource-Centric Complexity. Most connections from problem-centric to resource-centric complexity show that faster algorithms for OV or related problems give circuit lower bounds. For instance, improvements in EDIT-DISTANCE algorithms imply circuit lower bounds [2] and solving OV faster (and thus

CNF-SAT [40]) implies circuit lower bounds [26]. These are all non-uniform results, however, whereas in this paper we are concerned with *machines* and their resource-bounds as opposed to circuits. On the uniform side, [18] recently showed that the exact complexity of k-Orthogonal Vectors is closely related to the complexity of uniform AC^0, although a connection between more powerful machine models and fine-grained assumptions was still not known until now. Further, most of these results follow from OV being *easy*. Our work shows instead that there are interesting resource-centric consequences if our fine-grained problems are *hard*.

Uniform Derandomization Framework. The uniform derandomization framework was introduced in [25], a breakthrough paper that showed the first derandomization from a uniform assumption ($\mathsf{EXP} \neq \mathsf{BPP}$) in the *low-end* setting: a weak assumption gives a *slow* (subexponential-time) deterministic simulation of BPP. This is in contrast to our simulation which retains small amounts of randomness but is *fast* (this is a strictly stronger result as it recovers the [25] derandomization as a corollary).

We build on [38], which simplifies the proof of [25] to prove that $\mathsf{PSPACE} \neq \mathsf{BPP}$ implies a non-trivial deterministic simulation of BPP. The technique of [38] carefully arithmetizes the PSPACE-complete problem TQBF and uses this as a hard function in the generator of [25]. Our proof substitutes a carefully-arithmetized k-OV problem from [8]. Numerous other works study derandomization from uniform assumptions ([27, 33, 24, 22, 37]), but these all focus on assumptions and consequences about *nondeterministic* classes.

All worst-case uniform derandomizations, including [38] and [25], seem to only be able to achieve simulations of BPP that succeed for infinitely many input lengths because of how their proofs use downward self-reductions. Our is the first work to achieve simulations on *all but finitely many input lengths*, because the k-OV and k-CLIQUE-inspired problems have very parallelizable downward self reductions so that we can reduce to a *single* much smaller input length rather than recurse through a chain of incrementally smaller input lengths in our downward self-reduction.

Heuristics by Extracting Randomness From the Input. A separate line of work began when [21] introduced the idea of using the *input itself* as a source of randomness to heuristically simulate randomized algorithms over uniformly-distributed inputs. While their assumptions contain oracles and are mostly non-uniform and average-case, they construct an algebraic problem inside P whose worst-case uniform hardness can be used in the framework of [25] to get an *infinitely-often* simulation of BPP in polynomial time. Our work differs in that we achieve an *almost-everywhere* simulation, that our assumptions are based on canonical fine-grained problems, and that our assumptions aren't against machines with SAT-oracles. Further, the downward self-reduction of their problem requires an expansion by minors of the determinant and so they cannot also obtain an *almost-everywhere* heuristic using our techniques without placing the determinant in NC^1 (as our modification to [25] exploits *embarrassingly parallel* downward self reductions).

The work of [31], generalizing [36], removes the SAT-oracles needed in the assumptions of [21] by showing that the Nisan-Wigderson generator (see [35]) remains secure against *non-uniform* adversaries even if the seed is revealed to potential distinguishers. In Section 3.2.2 we will show their arguments can be made *uniform* so we can derandomize from uniform assumptions. Seed-revealing Nisan-Wigderson generators are used in [31] to obtain polynomial-time heuristics for randomized algorithms, where the uniformly distributed input is used as a seed to the generator. The derandomizations in [31] are achieved from *non-uniform* assumptions of polynomial *average-case* hardness. From *worst-case uniform* assumptions we achieve the same derandomizations.

2 Preliminaries

All complexity measures of fine-grained problems will refer to time on a randomized word RAM with $O(\log(n))$-bit word length, as is standard for the fine-grained literature [41, 8]. Specifically, we will consider two-sided bounded error as in [8].

2.1 Fine-Grained Hardness

The problem-centric field of fine-grained complexity theory has had impressive success in showing the fixed polynomial time ("fine-grained") hardness of many practical problems by assuming the fine-grained hardness of four "key" well-studied problems, as discussed in [8]. We obtain our results under hardness conjectures about two of these four key problems: the k-Orthogonal Vectors (k-OV) problem and the k-CLIQUE problem.

***k*-CLIQUE.** Denote the matrix multiplication constant by ω. The fastest known algorithm for deciding if a graph has a k-CLIQUE (given its adjacency matrix) runs in time $O(n^{\omega k/3})$, and was discovered in 1985 [34] for k a multiple of three (for other k different ideas are needed [14]). It is conjectured that no algorithm can improve the exponent to a better constant. The parameterized version of the famous NP-hard MAX-CLIQUE problem [30], k-CLIQUE is one of the most heavily studied problems in theoretical computer science and is the canonical intractable (W[1]-complete) problem in parameterized complexity (see [1] for a review of the copious evidence of k-CLIQUE's hardness and consequences of its algorithm's exponent being improved). Recent work has shown that conjecturing k-CLIQUE to require $n^{\omega k/3-o(1)}$ time, for k a multiple of three, leads to interesting hardness results for other important problems such as parsing languages and RNA folding [1, 12, 5, 7], and it is known that refuting this conjecture deterministically would give a faster exact algorithm for MAX-CUT [40]. Our results hold under a *much weaker version of the conjecture*:

▶ **Definition 1** (Weak k-CLIQUE Conjecture)**.** There exists an absolute constant $\epsilon_0 > 1/2$ such that, for all $k \in \mathbb{N}$ a multiple of three, any randomized algorithm that *counts* the number of k-CLIQUE's in an n node graph requires $n^{\epsilon_0 k}$ time.

Note that this conjecture gives leeway for the exponent of the k-CLIQUE algorithm to be improved so long as it doesn't get down to $k/2$; even finding a linear time algorithm for Boolean matrix multipliaction ($\omega = 2$) would not contradict this conjecture! Further, even if it is possible to *decide* the k-CLIQUE problem that quickly, this conjecture still holds unless it is possible to *count all of the k-CLIQUE's in that time.*

***k*-Orthogonal Vectors.** Although the k-CLIQUE problem is certainly *at least* as important as the k-OV problem, for concreteness we will use the k-OV problem to demonstrate our techniques throughout the paper. Proofs based on hardness of k-CLIQUE follow identically.

▶ **Definition 2** (k-Orthogonal Vectors Problem, k-OV$_{n,d}$)**.** For an integer $k \geq 2$, the k-OV$_{n,d}$ problem on vectors of dimension d is to determine, given k sets $(U_1, \ldots, U_k)$ of n vectors from $\{0,1\}^d$ each, whether there exist $u_i \in U_i$ for each i such that over $\mathbb{Z}$,

$$\sum_{\ell \in [d]} u_{1\ell} \cdots u_{k\ell} = 0$$

If left unspecified, d is to be taken to be $d(n) = \lceil \log^2 n \rceil$.

▶ **Definition 3** (k-Orthogonal Vectors Conjecture, k-OVC)**.** For any $d = \omega(\log n)$, for all $k \geq 2$, any randomized algorithm for the k-$\mathsf{OV}_{n,d}$ problem requires $n^{k-o(1)}$ time.

For $k = 2$ the Orthogonal Vectors conjecture for deterministic algorithms has been extensively studied and is supported by the *Strong Exponential Time Hypothesis* (SETH) [40], which states that there is no $\epsilon > 0$ such that t-SAT can be solved in time $\widetilde{O}(2^{n(1-\epsilon)})$ for all values of t. The natural generalization to k-OV is studied in [8, 18] and its deterministic hardness is also supported by SETH. While SETH has been controversial , the deterministic k-OV conjecture seems to be a much weaker assumption and is independently believable and supported: it has been shown that it holds unless *all* first-order graph properties become easy to decide [18] and the 2-OV conjecture has recently been proven unconditionally when the model of computation is restricted to branching programs [29]. This conjecture has also been used to support the hardness of many practical and well-studied fine-grained problems [3, 6, 13]. As with k-CLIQUE, our main results will hold *using a much weaker version of the randomized k-OV conjecture* introduced below.

▶ **Definition 4** (Weak k-Orthogonal Vectors Conjecture)**.** For any $d = \omega(\log n)$, there exists an absolute constant $\epsilon_0 > 1/2$ such that, for all $k \geq 2$, any randomized algorithm *counting* the number of k-$\mathsf{OV}_{n,d}$ solutions requires $n^{\epsilon_0 k}$ time.

▶ Remark. For all of these conjectures we will also consider the strengthened versions that assume that all algorithms running in time less than what is required will fail *on all but finitely many input lengths*, as opposed to only on infinitely many input lengths. For natural problems we expect that hardness grows, instead of oscillates, asymptotically.

For the purposes of derandomization, for a given k, we will use a family of polynomials introduced in [8], $\left\{f^k_{n,d,p} : \mathbb{F}_p^{knd} \to \mathbb{F}_p\right\}_{n,d,p\in\mathbb{N}}$, such that the variables are grouped into sets of size nd in the form of a matrix $U_i \in \mathbb{F}_p^{n\times d}$ where the n rows $u_i \in U_i$ are each collections of d variables:

$$f^k_{n,d,p}(U_1, \ldots, U_k) = \sum_{u_1\in U_1,\ldots,u_k\in U_k} \prod_{\ell\in[d]} (1 - u_{1\ell} \cdots u_{k\ell})$$

The worst-case hardness of evaluating these polynomials was related to the worst-case hardness of k-OV_n in [8].

▶ **Lemma 5.** *Let p be the smallest prime number larger than n^k and $d = \lceil \log^2(n) \rceil$. If $f^k_{n,d,p}$ can be computed in $O(n^{k/2+c})$ time for some $c > 0$, then k-OV_n can be **counted** in time $\widetilde{O}(n^{k/2+c})$*

Derandomization from uniform assumptions typically requires two other properties of the assumed hard problem: random self-reducibility and downward self-reducibility. We recall from [8] that $f^k_{n,d,p}$ satisfies both of these properties. We give a polynomial for k-CLIQUE and show that it also has the necessary properties in the full version.

Random Self-Reducible. $f^k_{n,d,p}$ is *random self-reducible* by the following classical lemma [32, 15] (see [8] for a proof). Note that degree $\log^2 n$ adds negligibly to the random self-reduction time.

▶ **Lemma 6** (Random Self-Reducibility of Polynomials)**.** *If $f : \mathbb{F}_P^N \to \mathbb{F}_P$ is a degree $9 < D < P/12$ polynomial, then there exists a randomized algorithm that takes a circuit $\widehat{C}$ 3/4-approximating f and produces a circuit C exactly computing f, such that the algorithm succeeds with high probability and runs in time $poly(N, D, \log P, |\widehat{C}|)$.*

Downward Self-Reducible. We will show that $f^k_{n,d,p}$ is downward self-reducible in the sense that, if we we have a way to produce an oracle for $f^k_{\sqrt{n},d,p}$, we can quickly compute $f^k_{n,d,p}$ with it. Compare this to downward self reducibility going from input size n to $n-1$ in previous uniform derandomizations. We exploit our more dramatic shrinkage and parallelism to later give an almost-everywhere derandomization, instead of an infinitely-often one. The proof of the following lemma is in the full version.

▶ **Lemma 7.** *If there exists an algorithm A that, on input 1^n, outputs a circuit C computing $f^k_{\sqrt{n},d,p}$, then there exists an algorithm that computes $f^k_{n,d,p}$ in time $O(n^{k/2}|C| + \mathsf{TIME}(A))$.*

2.2 Derandomization

We now define pseudorandom generators (PRGs) in terms of their distinguishers.

▶ **Definition 8** (Distinguishers)**.** A test $T : \{0,1\}^{m^\ell} \to \{0,1\}$ is an ϵ-*distinguisher* against $G : \{0,1\}^m \to \{0,1\}^{m^\ell}$, denoted $T \in \mathsf{DIS}(G, \epsilon)$, if:

$$\left| \Pr_{r \sim \mathcal{U}_{m^\ell}} [T(r)] - \Pr_{z \sim \mathcal{U}_m} [T(G(z))] \right| > \epsilon$$

We also will consider the seemingly weaker object of distinguishers that succeed if they are also given the seed to the PRG. These were studied in [38] to relate uniform derandomization to average-case hardness and in [31] to derandomize over the uniform distribution by *using the random input itself as the seed* to the PRG.

▶ **Definition 9** (Seed-Aware Distinguishers)**.** A test $T : \{0,1\}^m \times \{0,1\}^{m^\ell} \to \{0,1\}$ is an ϵ-*seed-aware distinguisher* against G, denoted $T \in \mathsf{DIS}(G, \epsilon)$, if:

$$\left| \Pr_{x \sim \mathcal{U}_m, r \sim \mathcal{U}_{m^\ell}} [T(x, r)] - \Pr_{x \sim \mathcal{U}_m} [T(x, G(x))] \right| > \epsilon$$

Standard hardness-to-randomness arguments typically derandomize using generators that are based on some 'hard' function by contrapositive: if derandomization fails, then a distinguisher for the generator can be produced. Further, from a distinguisher, we can create a small circuit for the supposedly hard function that the generator was based on. For our purposes, we require an algorithmic version of this argument for derandomization from *uniform* hardness assumptions. More specifically, we will use the following lemma which was originally proved for distinguishers [38, 25] but Lemma 2.9 of [31] proves that it also holds for seed-aware distinguishers (while the proof of [31] is non-uniform, it is easy to see that it can be made constructive, in the same way that [25] gave a constructive version of [35]). Thus, $\mathsf{DIS}(G, \epsilon)$ in the lemma below can be thought to refer to either regular or seed-aware distinguishers (which justifies overloading this notation).

▶ **Lemma 10** (Algorithmic Distinguishers to Predictors ([38, 25]))**.** *For every random self-reducible f, there exists a function G with stretch m bits to m^ℓ bits and a constant c such that*

- *$G(z)$ is in time $(|z|^\ell)^c$ with oracle access to f on inputs of length at most $|z|$*
- *There exists a polynomial-time randomized algorithm A that, with high probability, given as input circuit $D \in \mathsf{DIS}(G, \epsilon)$ for ϵ at least inverse polynomial and an oracle for f, prints a circuit computing f exactly.*

2.3 Uniform Derandomization

Average-Case Tractability. We give standard definitions of average-case tractability (for an extensive survey of these notions, see [10]).

▶ **Definition 11** ($t(n)$-Samplable Ensemble)**.** An ensemble $\mu = \{\mu_n\}$ is $t(n)$-samplable if there is a randomized algorithm A that, on input a number n, outputs a string in $\{0,1\}^*$ and:

- A runs in time at most $t(n)$ on input n, regardless of its internal coin tosses
- for every n and for every $x \in \{0,1\}^*$, $\Pr[A(n) = x] = \mu_n(x)$

With this notion of samplable ensemble we can now consider *heuristic algorithms* that perform well on some language $\mathcal{L} : \{0,1\}^* \to \{0,1\}$ over some μ. The pair $(\mathcal{L}, \mu)$ is a *distributional problem.*

▶ **Definition 12** (Heuristics for Distributional Problems)**.** For $t : \mathbb{N} \to \mathbb{N}$, $\delta : \mathbb{N} \to \mathbb{R}^+$, we say $(\mathcal{L}, \mu) \in \text{Heur}_{\delta(n)}\mathsf{DTIME}[t(n)]$ if there is a time $t(n)$ deterministic algorithm A such that, for all but finitely many n: $\Pr_{x\sim\mu_n}[A(x) \neq \mathcal{L}(x)] \leq \delta(n)$.

For a class of languages $\mathcal{C}$ we say $(\mathcal{C}, \mu) \in \text{Heur}_{\delta(n)}\mathsf{DTIME}[t(n)]$ if $(\mathcal{L}, \mu) \in \text{Heur}_{\delta(n)}\mathsf{DTIME}[t(n)]$ for all $\mathcal{L} \in \mathcal{C}$. As in [10], $\text{Heur}_\delta\mathsf{P}$ is defined as the union over all polynomials p of $\text{Heur}_\delta\mathsf{DTIME}(p(n))$ and HeurP is the intersection over all inverse polynomial $\delta(n)$ of $\text{Heur}_\delta\mathsf{P}$. $\mathsf{HeurSUBEXP}$ is defined similarly where $\mathsf{SUBEXP} = \cap_{\epsilon>0}\mathsf{DTIME}\left[2^{n^\epsilon}\right]$. Finally, to discuss the *randomness-reduced* simulations we construct, we define BPTIME with a limited number of random coins in the natural way.

▶ **Definition 13** (Randomized Heuristics with Bounded Coins)**.** For $t : \mathbb{N} \to \mathbb{N}$, $\delta : \mathbb{N} \to \mathbb{R}^+$, and coin bound $r : \mathbb{N} \to \mathbb{N}$ we say $(\mathcal{L}, \mu) \in \text{Heur}_{\delta(n)}\mathsf{BPTIME}_{[r(n)]}[t(n)]$ if there is randomized algorithm A running in time $t(n)$ and flipping $r(n)$ coins such that, for all but finitely many n: $\Pr_{x\sim\mu_n}\left[\Pr_{r\sim\mathcal{U}_{r(n)}}[A(x,r) \neq \mathcal{L}(x)] > 1/3\right] \leq \delta(n)$

For example, $\text{Heur}\mathsf{BPP}_{[r(n)]}$ denotes the class of distributional problems that, for every inverse polynomial error, have a polynomial time randomized algorithm using only $r(n)$ random coins.

Infinitely-Often Simulation. As opposed to an algorithm that decides a language (possibly on average) "for all but finitely many n" as in Definition 12, an infinitely-often (io-) qualifier can be added to any complexity class to specify that an algorithm need only succeed on infinitely many input lengths within the time and error bounds. Thus, to derandomize BPP into $\mathsf{io\text{-}HeurP}$ over the uniform distribution is to say that every language in BPP can be simulated on average in polynomial time by an algorithm that is only guaranteed to succeed for infinitely many input lengths. *There is no guarantee on what those input lengths are or how large the gaps could be between them.* This is obviously a very undesirable notion of 'tractability'.

Non-uniform hardness to randomness trade-offs can derandomize almost-everywhere (the desired notion of tractability for asymptotics) by assuming almost-everywhere hardness: that no algorithm works *for all sufficiently large input lengths.* That is, the 'infinitely-often' qualifier on the consequent can be *flipped* across the implication to be an 'almost-everywhere' qualifier on the assumption and vice-versa. Thus, the unrealistic 'infinitely-often' notion of tractability can be dropped by slightly strengthening the assumption to the (as argued in Section 2.1's remark) realistic 'almost-everywhere' hardness. For *non-uniform* derandomizations this is possible.

Starting with [25] and the techniques it introduced, all *uniform* derandomizations have been infinitely-often derandomizations *without* being able to flip the io- qualifier to an 'almost-everywhere' assumption. Our work is the first that is able to do this in the uniform derandomization framework, thus removing the 'infinitely-often' qualifier from our derandomizations.

3 Fine-Grained Derandomization

We will prove our main derandomization results (Theorems 17 and 20) here. Under either the (weak) k-OV or k-CLIQUE conjectures, we derandomize BPP on average, where 'on average' will have two different flavors. Although all techniques apply to k-CLIQUE, for concreteness we will use k-OV throughout this section.

We show in Section 3.1 that if we base pseudorandom generators on $f^k_{n,d,p}$, then an algorithm printing distinguishers for this PRG can be used to count k-OV solutions quickly. We will then show in Section 3.2 how to attain these distinguisher-printing algorithms if derandomization *doesn't* work on average (for both flavors of on average). Thus, a failed derandomization using these PRGs refutes the k-OV conjecture (similarly for k-CLIQUE).

3.1 Counting k-OV from Distinguishers

In this section we show that any algorithm producing a distinguisher for $G^{f^k_{m,d,p}}$ (the generator guaranteed to exist from Lemma 14, using the hard function $f^k_{m,d,p}$) can be used to quickly count k-OV solutions. First, Lemma 14 follows immediately by combining the distinguisher to predictor algorithm of Lemma 10 with the fact that $f^k_{m,d,p}$ is random self-reducible as in Lemma 6.

▶ **Lemma 14.** *There is a randomized algorithm $A^{f^k_{m,d,p}}$ that takes any circuit D that is a distinguisher for $G^{f^k_{m,d,p}}$ and produces a circuit C exactly computing $f^k_{m,d,p}$, such that A succeeds with high probability and runs in time* $\mathsf{poly}(m, d, \log p, |D|)$

As $f^k_{m,d,p}$ is efficiently computable (unlike the hard problems of [25]) in time $O(m^k \mathsf{poly}(d, \log p))$, we get the following theorem *without* an oracle by running the algorithm guaranteed in Lemma 14 with each oracle call answered by naïve brute force computation of $f^k_{m,d,p}$.

▶ **Lemma 15.** *There is a randomized algorithm B that takes any circuit D that is a distinguisher for $G^{f^k_{m,d,p}}$ and produces a circuit C of size* $\mathsf{poly}(m, d, \log p, |D|)$ *exactly computing $f^k_{m,d,p}$. B succeeds with high probability and runs in time $O(m^k \mathsf{poly}(m, d, \log p, |D|))$.*

Now we show that, if we have an algorithm producing a distinguisher, then we have an algorithm counting k-OV.

▶ **Theorem 16.** *Let p be the smallest prime number larger than n^k and $d = \lceil \log^2(n) \rceil$. If there is an algorithm A that, on input 1^n, outputs a distinguisher D of* $\mathsf{poly}(n)$ *size for $G^{f^k_{\sqrt{n},d,p}}$, then there exists a randomized algorithm counting k-OV_n that runs in time $O(n^{k/2+c} + \mathsf{TIME}(A))$, where c only depends on $|D|$.*

Proof. Using A, we print a distinguisher circuit D for $G^{f^k_{\sqrt{n},d,p}}$. Then, by Lemma 15, we know there exists a randomized algorithm running in time $O(n^{k/2}\mathsf{poly}(\sqrt{n}, d, \log p, |D|)) = O(n^{k/2+c_1})$ that yields a circuit exactly computing $f^k_{\sqrt{n},d,p}$ of size only $\mathsf{poly}(\sqrt{n}, d, \log p, |D|) = O(n^{c_2})$, where c_1 and c_2 only depend on $|D|$. Thus, by Lemma 7, there exists an algorithm

computing $f^k_{n,d,p}$ in time $O(n^{k/2+c_2} + (n^{k/2+c_1} + \mathsf{TIME}(A))) = O(n^{k/2+c} + \mathsf{TIME}(A))$ for $c = \max\{c_1, c_2\}$. Finally, this gives us an algorithm running in time $\widetilde{O}(n^{k/2+c} + \mathsf{TIME}(A))$ to count k-OV_n by Lemma 5. ◀

3.2 Printing Distinguishers from Failed Derandomization

3.2.1 Randomness-Reduced Heuristics Over Any Efficient Distribution

Our first main result in derandomizing BPP is to reduce the amount of randomness required to arbitrarily small quantities, over any efficient distribution of inputs. This simulation trades time for reduced randomness under fine-grained hardness assumptions.

▶ **Theorem 17.** *If the weak k-OV conjecture holds almost everywhere, then, for all polynomially samplable ensembles μ and for all constants $\alpha > 0$, $(\mathsf{BPP}, \mu) \in \mathrm{Heur}\mathsf{BPP}_{[n^\alpha]}$.*

Thus, for any efficient distribution over inputs that nature might be drawing from and for any inverse polynomial error rate we specify, we can simulate BPP using only n^α random bits for any constant $\alpha > 0$ we want. In contrast to typical full derandomizations which brute-force all seeds to a pseudorandom generator and take majority answer (which we can also do with our randomness-reduced derandomization to get a subexponential-time full derandomization), we show that choosing a *single random seed* and using the generator's output as our randomness yields randomness-reduced simulations so long as the generator is efficient enough (which it typically is not; 'quick' complexity-theoretic PRGs are usually given exponential time in their seed length).

▶ **Definition 18** (Randomness-Reduced Simulations)**.** Let $A : \{0,1\}^N \times \{0,1\}^{N^\ell} \to \{0,1\}$ be a randomized algorithm that uses N^ℓ random bits and let $G : \{0,1\}^{N^\alpha} \to \{0,1\}^{N^\ell}$ be a function. Then for constant $\alpha > 0$, define the randomness-reduced simulation to be a randomized algorithm $B : \{0,1\}^N \times \{0,1\}^{N^\alpha} \to \{0,1\}$ using only N^α random bits as $B(x,r) = A(x, G(r))$.

Lemma 19 states that if this simulation fails, we can uniformly print a distinguisher for the function G. This proof is identical to that of Lemma 18 in [25] and is recalled in the full version.

▶ **Lemma 19** (Failed Randomness-Reduction to Distinguishers)**.** *Let A, B, and G be as in Definition 18 such that for language $\mathcal{L} : \{0,1\}^N \to \{0,1\}$, $\Pr_{r \sim \mathcal{U}_{N^\ell}}[A(x,r) \neq \mathcal{L}(x)] \leq 1/10$. That is, that A as a good randomized algorithm deciding $\mathcal{L}$ for all $x \in \{0,1\}^N$. Yet, also assume that, for μ samplable in time N^{a_1} and $\delta(n) = 1/N^{a_2}$, it holds that*

$$\Pr_{x \sim \mu_N}\left[\Pr_{r \sim \mathcal{U}_{N^\alpha}}[B(x,r) \neq \mathcal{L}(x)] > 1/3\right] \geq \delta(N)$$

So B is a (randomness-reduced) randomized algorithm that does not decide $\mathcal{L}$ on average over μ. Then $1^N \mapsto \mathsf{DIS}(G, 1/5)$ is in randomized time N^c $\mathsf{TIME}(G)$ for c depending on a_1 and a_2.

Randomness-Reduced Simulation from k-OV. To finish defining a randomness-reduced simulation, we need to use a specific pseudorandom generator G that, for input length N, stretches N^α coins to N^ℓ. Thus, consider the family of simulations B_k using the standard generators $G^{f^k_{\sqrt{n},d,p}}$ of Lemma 10 that map $\sqrt{n}^s$ bits to $\sqrt{n}^b$ bits, for some fixed s and any b we choose, using $f^k_{\sqrt{n},d,p}$ as our hard function, for $d = \log^2 n$ and p the smallest

prime number larger than n^k. Set $b = s\ell/\alpha$ and $\sqrt{n} = N^{\alpha/s}$. Note that $\mathsf{TIME}(G^{f^k_{\sqrt{n},d,p}}) = \mathsf{poly}(N)\ n^{k/2} = \mathsf{poly}(N)$ by naïvely evaluating $f^k_{\sqrt{n},d,p}$ at each oracle call, giving an efficient randomness-reduced simulation. Further, since $N = \mathsf{poly}(n)$, $\mathsf{TIME}(G^{f^k_{\sqrt{n},d,p}})$ also equals $n^{k/2+c}$ for some constant c not depending on k (this will be useful in quickly counting k-OV_n using downward self-reducibility in the following proof). Thus, given an N^ℓ-coin machine A, we have the N^α-coin machine $B_k(x,r) = A\left(x, G^{f^k_{\sqrt{n},d,p}}(r)\right)$. We now prove our main Theorem 17 using this simulation and the above lemma.

Proof of Theorem 17. We proceed by contradiction. Assume that the weak k-OV_n conjecture holds for all but finitely many input lengths, where $\epsilon_0 = 1/2 + \gamma$ for some constant $\gamma > 0$, but that there exists $\mathcal{L} \in \mathsf{BPP}$, a polynomially samplable distribution μ, constant α, and an inverse polynomial function $\delta(N)$ such that *any* polynomial-time randomness-reduced algorithm with coin bound N^α fails in deciding $\mathcal{L}$ on average over μ within $\delta(N)$ error for infinitely many input lengths N.

Since $\mathcal{L} \in \mathsf{BPP}$ there is a randomized algorithm A deciding $\mathcal{L}$ with probability of error at most $1/10$ over its randomness yet, since *any* polynomial-time randomness-reduced algorithm fails to decide $\mathcal{L}$ on average, B_k, the randomness-reduced simulation of A described above, fails on average infinitely often, for *any* constant k. Thus, the antecedents of Lemma 19 are satisfied and we can uniformly print $D \in \mathsf{DIS}(G^{f^k_{\sqrt{n},d,p}}, 1/5)$ in time $n^{c_1}\ \mathsf{TIME}\left(G^{f^k_{\sqrt{n},d,p}}\right) = n^{c_1}\ n^{c_2} n^{k/2}$.

This uniform printing of D allows us to apply Theorem 16 to count k-OV_n in time $O(n^{k/2+c_3} + n^{k/2+c_1+c_2}) = O(n^{k/2+c}) = O(n^{(\frac{1}{2}+\frac{c}{k})k})$ for *any* k, where $c = \max\{c_1 + c_2, c_3\}$. Setting k such that $\frac{c}{k} < \gamma$ yields our contradictions: on the infinitely many input lengths where B_k fails to derandomize $\mathcal{L}$, the algorithm counts k-OV faster than $n^{\epsilon_0 k}$ time. ◀

3.2.2 Fast Heuristics for BPP Over the Uniform Distribution

Here we present our second flavor of derandomization: a fully deterministic heuristic for BPP when inputs are sampled according to the uniform distribution.

▶ **Theorem 20.** *If the weak k-OV conjecture holds almost everywhere, then $(\mathsf{BPP}, \mathcal{U}) \in \mathrm{HeurP}$.*

This strictly improves previous uniform derandomizations over the uniform distribution. Specifically, [21] can be seen to achieve our derandomization identically from a worst-case uniform assumption if combined with techniques from [31] *except only on infinitely many input lengths.*

We proceed by showing that if a PRG fails to give a good heuristic for BPP over the uniform distribution, a seed-aware distinguisher for the PRG can be produced uniformly and efficiently, which can then be used to count k-OV solutions quickly using Theorem 16.

▶ **Definition 21** (Input-As-Seed Heuristics)**.** Let $A : \{0,1\}^N \times \{0,1\}^{N^\ell} \to \{0,1\}$ be a polynomial-time randomized algorithm using N^ℓ random bits. Let $G : \{0,1\}^N \to \{0,1\}^{N^\ell}$ be a deterministic function. Define the heuristic $B : \{0,1\}^N \to \{0,1\}$ that uses its input as G's seed as $B(x) = A(x, G(x))$.

We prove a uniform analog of the Main Lemma of [31], which gave the consequences of failed heuristics in the *non-uniform* setting. Namely, we prove:

▶ **Lemma 22** (Failed Heuristics to Distinguishers). *Let* $A : \{0,1\}^N \times \{0,1\}^{N^\ell} \to \{0,1\}$ *and* $\mathcal{L} : \{0,1\}^N \to \{0,1\}$ *be functions such that* $\Pr_{x\sim\mathcal{U}_N, r\sim\mathcal{U}_{N^\ell}}[A(x,r) \neq \mathcal{L}(x)] \leq \rho$. *Let* B *be the input-as-seed heuristic for* A *using function* G. *Then, if* B *does* ***not*** *succeed on a* $(5\rho+\epsilon)$ *fraction of the inputs of a given length, the map* $1^N \mapsto \mathsf{DIS}(G,\epsilon)$ *is uniform and in randomized polynomial time, for infinitely many* N.

We sketch the proof here (see the full version for a proof). If B is a *bad* heuristic for $\mathcal{L}$, then we could use $B(x) = A(x, G(x))$ as a seed-aware distinguisher for G by comparing $B(x)$ to $\mathcal{L}(x)$. Unfortunately we cannot afford to print distinguishers with $\mathcal{L}$-oracles. But since we are guaranteed that A is a *good* heuristic for $\mathcal{L}$, we can obtain a deterministic circuit close to $\mathcal{L}$ from A, by fixing a string of good random bits r'. The proof of the analogous lemma in [31] uses non-uniformity to obtain a good r' for distinguishing, but we can instead obtain good strings r' by showing that there are *many* good random strings. We find a good r' by a sample-and-test procedure. If we compare $B(x)$ and the fixed-coin algorithm $A(x,r')$, they will also tend to disagree, giving the necessary distinguishing gap.

Fully Deterministic Heuristics from k-OV. Here we specify a family of heuristics B_k, by specifying the generator G, that stretches a seed of length N to N^ℓ, as the generators $G^{f^k_{\sqrt{n},d,p}}$ of Lemma 10. These map $\sqrt{n}^s$ bits to $\sqrt{n}^b$ bits, for some fixed s and any b we choose, using $f^k_{\sqrt{n},d,p}$, for $d = \log^2 n$ and p the smallest prime number larger than n^k. Set $b = s\ell$ and $\sqrt{n} = N^{1/s}$. All comments about the runtime of the randomness-reduced heuristic in Section 3.2.1 also apply to this fully deterministic heuristic. Thus, given an N^ℓ-coin machine A, we have the *deterministic* machine $B_k(x) = A\left(x, G^{f^k_{\sqrt{n},d,p}}(x)\right)$.

This can now be used to prove Theorem 20, although we defer this proof to the full version as it is very similar to the proof of Theorem 17 in Section 3.2.1.

4 Open Questions

- We derandomize under hardness conjectures about two of four 'key' problems in fine-grained complexity: k-OV and k-CLIQUE. What about k-SUM and APSP? APSP doesn't seem to have a natural hierarchy and so doesn't fit our framework (although it does reduce to ZERO-TRIANGLE which generalizes to ZERO-k-CLIQUE and should easily work in our framework using polynomials similar to those in [8]). k-SUM however is actually computable in $O(n^{\lceil k/2\rceil})$ time and so our downward self-reducibility techniques are not fast enough to break this conjecture in the contrapositive. The clearest path we see to getting derandomization without reintroducing the io- qualifier is to find a polynomial for k-SUM that is also computable in $\widetilde{O}(n^{\lceil k/2\rceil})$ time (unlike the one found in [8]).
- Our derandomizations hold under (randomized) SETH, since SETH implies the k-OV conjecture. Can a better derandomization be obtained directly from SETH, the stronger assumption? A stumbling block here is the random self-reduction, an ingredient in all known uniform derandomization techniques: If t-SAT has a straightforward and efficient random-self-reduction, PH collapses [16, 11]. So derandomizing from SETH directly could require new ideas, or a strange random self-reduction. An *inefficient* random self-reduction for t-SAT shouldn't collapse PH except to say that t-SAT has a mildly exponential MA proof which is already known to be true [39], although most random self-reductions we know are through arithmetization which seems to always have 'low' degree to the point that such a polynomial would still collapse PH.
- Is a strong "derandomization to hardness" converse possible for these heuristic simulations of BPP? In the full version of this paper, we show a weak converse: our simulation

is impossible without separting DTIME[$n^{\omega(1)}$] from BPP. But this is a very different statement from the k-OV or k-CLIQUE conjectures. In [31], they show that herusitic simulations of BPP with inverse-subexponential error rates imply circuit lower bounds, by generalizing techniques of [28]. Do the efficient inverse-polynomial error heuristics we obtain imply any circuit lower bounds?

References

1 Amir Abboud, Arturs Backurs, and Virginia Vassilevska Williams. If the current clique algorithms are optimal, so is valiant's parser. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 98–117. IEEE Computer Society, 2015. doi:10.1109/FOCS.2015.16.

2 Amir Abboud, Thomas Dueholm Hansen, Virginia Vassilevska Williams, and Ryan Williams. Simulating branching programs with edit distance and friends or: A polylog shaved is a lower bound made. *CoRR*, abs/1511.06022, 2015. URL: http://arxiv.org/abs/1511.06022.

3 Amir Abboud, Virginia Vassilevska Williams, and Oren Weimann. Consequences of faster alignment of sequences. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, volume 8572 of *Lecture Notes in Computer Science*, pages 39–51. Springer, 2014. doi:10.1007/978-3-662-43948-7_4.

4 László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3:307–318, 1993. doi:10.1007/BF01275486.

5 Arturs Backurs, Nishanth Dikkala, and Christos Tzamos. Tight hardness results for maximum weight rectangles. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, volume 55 of *LIPIcs*, pages 81:1–81:13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016. doi:10.4230/LIPIcs.ICALP.2016.81.

6 Arturs Backurs and Piotr Indyk. Edit distance cannot be computed in strongly subquadratic time (unless SETH is false). In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 51–58. ACM, 2015. doi:10.1145/2746539.2746612.

7 Arturs Backurs and Christos Tzamos. Improving viterbi is hard: Better runtimes imply faster clique algorithms. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 311–321. PMLR, 2017. URL: http://proceedings.mlr.press/v70/backurs17a.html.

8 Marshall Ball, Alon Rosen, Manuel Sabin, and Prashant Nalini Vasudevan. Average-case fine-grained hardness. 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 483–496. ACM, 2017. doi:10.1145/3055399.3055466.

9 Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J. Comput.*, 13(4):850–864, 1984. doi:10.1137/0213053.

10 Andrej Bogdanov and Luca Trevisan. Average-case complexity. *Foundations and Trends in Theoretical Computer Science*, 2(1), 2006. doi:10.1561/0400000004.

11 Andrej Bogdanov and Luca Trevisan. On worst-case to average-case reductions for NP problems. *SIAM J. Comput.*, 36(4):1119–1159, 2006. doi:10.1137/S0097539705446974.

12 Karl Bringmann, Allan Grønlund, and Kasper Green Larsen. A dichotomy for regular expression membership testing. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 307–318. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.36.

13 Karl Bringmann and Marvin Künnemann. Quadratic conditional lower bounds for string problems and dynamic time warping. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 79–97. IEEE Computer Society, 2015. doi:10.1109/FOCS.2015.15.

14 Friedrich Eisenbrand and Fabrizio Grandoni. On the complexity of fixed parameter clique and dominating set. *Theor. Comput. Sci.*, 326(1-3):57–67, 2004. doi:10.1016/j.tcs.2004.05.009.

15 Joan Feigenbaum and Lance Fortnow. On the random-self-reducibility of complete sets. In *Proceedings of the Sixth Annual Structure in Complexity Theory Conference, Chicago, Illinois, USA, June 30 - July 3, 1991*, pages 124–132. IEEE Computer Society, 1991. doi:10.1109/SCT.1991.160252.

16 Joan Feigenbaum and Lance Fortnow. Random-self-reducibility of complete sets. *SIAM J. Comput.*, 22(5):994–1005, 1993. doi:10.1137/0222061.

17 Jiawei Gao and Russell Impagliazzo. Orthogonal vectors is hard for first-order properties on sparse graphs. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:53, 2016. URL: http://eccc.hpi-web.de/report/2016/053.

18 Jiawei Gao, Russell Impagliazzo, Antonina Kolokolova, and R. Ryan Williams. Completeness for first-order properties on sparse structures with algorithmic applications. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2162–2181. SIAM, 2017. doi:10.1137/1.9781611974782.141.

19 Oded Goldreich, Hugo Krawczyk, and Michael Luby. On the existence of pseudorandom generators. *SIAM J. Comput.*, 22(6):1163–1175, 1993. doi:10.1137/0222069.

20 Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 25–32, 1989. doi:10.1145/73007.73010.

21 Oded Goldreich and Avi Wigderson. Derandomization that is rarely wrong from short advice that is typically good. In José D. P. Rolim and Salil P. Vadhan, editors, *Randomization and Approximation Techniques, 6th International Workshop, RANDOM 2002, Cambridge, MA, USA, September 13-15, 2002, Proceedings*, volume 2483 of *Lecture Notes in Computer Science*, pages 209–223. Springer, 2002. doi:10.1007/3-540-45726-7_17.

22 Dan Gutfreund, Ronen Shaltiel, and Amnon Ta-Shma. Uniform hardness vs. randomness tradeoffs for arthur-merlin games. In *18th Annual IEEE Conference on Computational Complexity (Complexity 2003), 7-10 July 2003, Aarhus, Denmark*, pages 33–47. IEEE Computer Society, 2003. doi:10.1109/CCC.2003.1214408.

23 Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999. doi:10.1137/S0097539793244708.

24 Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. In search of an easy witness: exponential time vs. probabilistic polynomial time. *J. Comput. Syst. Sci.*, 65(4):672–694, 2002. doi:10.1016/S0022-0000(02)00024-7.

25 Russell Impagliazzo and Avi Wigderson. Randomness vs time: Derandomization under a uniform assumption. *J. Comput. Syst. Sci.*, 63(4):672–688, 2001. doi:10.1006/jcss.2001.1780.

26 Hamid Jahanjou, Eric Miles, and Emanuele Viola. Local reductions. In Magnús M. Halldórsson, Kazuo Iwama, Naoki Kobayashi, and Bettina Speckmann, editors, *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, volume 9134 of *Lecture Notes in Computer Science*, pages 749–760. Springer, 2015. doi:10.1007/978-3-662-47672-7_61.

27 Valentine Kabanets. Easiness assumptions and hardness tests: Trading time for zero error. *J. Comput. Syst. Sci.*, 63(2):236–252, 2001. doi:10.1006/jcss.2001.1763.

28 Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004. doi:10.1007/s00037-004-0182-6.

29 Daniel M. Kane and R. Ryan Williams. The orthogonal vectors conjecture for branching programs and formulas. *CoRR*, abs/1709.05294, 2017. URL: http://arxiv.org/abs/1709.05294.

30 Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, editors, *Proceedings of a symposium on the Complexity of Computer Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York.*, The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972. URL: http://www.cs.berkeley.edu/~luca/cs172/karp.pdf.

31 Jeff Kinne, Dieter van Melkebeek, and Ronen Shaltiel. Pseudorandom generators, typically-correct derandomization, and circuit lower bounds. *Computational Complexity*, 21(1):3–61, 2012. doi:10.1007/s00037-011-0019-z.

32 Richard J. Lipton. New directions in testing. In Joan Feigenbaum and Michael Merritt, editors, *Distributed Computing And Cryptography, Proceedings of a DIMACS Workshop, Princeton, New Jersey, USA, October 4-6, 1989*, volume 2 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 191–202. DIMACS/AMS, 1989.

33 Chi-Jen Lu. Derandomizing arthur-merlin games under uniform assumptions. *Computational Complexity*, 10(3):247–259, 2001. doi:10.1007/s00037-001-8196-9.

34 Jaroslav Nešetřil and Svatopluk Poljak. On the complexity of the subgraph problem. *Commentationes Mathematicae Universitatis Carolinae*, 26(2):415–419, 1985.

35 Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994. doi:10.1016/S0022-0000(05)80043-1.

36 Ronen Shaltiel. Weak derandomization of weak algorithms: Explicit versions of yao's lemma. *Computational Complexity*, 20(1):87–143, 2011. doi:10.1007/s00037-011-0006-4.

37 Ronen Shaltiel and Christopher Umans. Low-end uniform hardness versus randomness tradeoffs for AM. *SIAM J. Comput.*, 39(3):1006–1037, 2009. doi:10.1137/070698348.

38 Luca Trevisan and Salil P. Vadhan. Pseudorandomness and average-case complexity via uniform reductions. *Computational Complexity*, 16(4):331–364, 2007. doi:10.1007/s00037-007-0233-x.

39 Richard Ryan Williams. Strong ETH breaks with merlin and arthur: Short non-interactive proofs of batch evaluation. In Ran Raz, editor, *31st Conference on Computational Complexity, CCC 2016, May 29 to June 1, 2016, Tokyo, Japan*, volume 50 of *LIPIcs*, pages 2:1–2:17. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016. doi:10.4230/LIPIcs.CCC.2016.2.

40 Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theor. Comput. Sci.*, 348(2-3):357–365, 2005. doi:10.1016/j.tcs.2005.09.023.

41 Virginia Vassilevska Williams. Hardness of easy problems: Basing hardness on popular conjectures such as the strong exponential time hypothesis (invited talk). In Thore Husfeldt and Iyad A. Kanj, editors, *10th International Symposium on Parameterized and Exact Computation, IPEC 2015, September 16-18, 2015, Patras, Greece*, volume 43 of *LIPIcs*, pages 17–29. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015. doi:10.4230/LIPIcs.IPEC.2015.17.

42 Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 80–91. IEEE Computer Society, 1982. doi:10.1109/SFCS.1982.45.