Distinguishing, Predicting, and Certifying: On the Long Reach of Partial Notions of Pseudorandomness

Jiatu Li
CSAIL
MIT
Cambridge, USA
jiatuli@mit.edu

Edward Pyne
CSAIL
MIT
Cambridge, USA
epyne@mit.edu

Roei Tell
Theory Group
University of Toronto
Toronto, Canada
roei@cs.toronto.edu

2024 IEEE 65th Annual Symposium on Foundations of Computer Science (FOCS) | 979-8-3315-1674-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/FOCS61266.2024.00095

***Abstract*—This paper revisits the study of two classical technical tools in theoretical computer science: Yao's transformation of distinguishers to next-bit predictors (FOCS 1982), and the "reconstruction paradigm" in pseudorandomness (e.g., as in Nisan and Wigderson, JCSS 1994). Recent works of Pyne, Raz, and Zhan (FOCS 2023) and Doron, Pyne, and Tell (STOC 2024) showed that both of these tools can be derandomized in the specific context of read-once branching programs (ROBPs), but left open the question of derandomizing them in more general settings.**

Our main contributions give appealing evidence that derandomization of the two tools is possible in general settings, show surprisingly strong consequences of such derandomization, and reveal several new settings where such derandomization is unconditionally possible for algorithms stronger than ROBPs (with useful consequences). Specifically:

- **We show that derandomizing these tools is equivalent to general derandomization. Specifically, we show that derandomizing distinguish-to-predict transformations is equivalent to prBPP=prP, and that derandomized reconstruction procedures (in a more general sense that we introduce) is equivalent to prBPP=prZPP. These statements hold even when scaled down to weak circuit classes and to algorithms that run in super-polynomial time.**
- **Our main technical contributions are unconditional constructions of derandomized versions of Yao's transformation (or reductions of this task to other problems) for classes and for algorithms beyond ROBPs. Consequently, we deduce new results: A significant relaxation of the hypotheses required to derandomize the isolation lemma for logspace algorithms and deduce that NL=UL; and proofs that derandomization necessitates targeted PRGs in catalytic logspace (unconditionally) and in logspace (conditionally).**

In addition, we introduce a natural subclass of prZPP that has been implicitly studied in recent works (Korten FOCS 2021, CCC 2022): The class of problems reducible to a problem called "Lossy Code". We provide a structural characterization for this class in terms of derandomized reconstruction procedures, and show that this characterization is robust to several natural variations.

Lastly, we present alternative proofs for classical results in the theory of pseudorandomness (such as two-sided derandomization reducing to one-sided), relying on the notion of deterministically transforming distinguishers to predictors as the main technical tool.

***Index Terms*—pseudorandomness, complexity**

I. Introduction

This paper revisits the study of two classical technical tools in theoretical computer science: Yao's transformation of distinguishers to next-bit predictors [Yao82], and the "reconstruction paradigm", both of which will be explained next.

More than four decades ago, Yao introduced a very simple probabilistic transformation of any distinguisher for a distribution into a **next-bit predictor** for the same distribution; that is:

Definition I.1 (distinguisher)**.** We say that $C\colon \{0,1\}^n \to \{0,1\}$ is an ε-distinguisher for a distribution $\mathbf{D}$ over $\{0,1\}^n$ if $\left|\mathbb{E}[C(\mathbf{U}_n)] - \mathbb{E}[C(\mathbf{D})]\right| \geq \varepsilon$, where $\mathbf{U}_n$ is the uniform distribution.

Definition I.2 (next-bit predictor)**.** For $i \in [n]$, we say that $P\colon \{0,1\}^{i-1} \to \{0,1\}$ is a δ-next-bit-predictor for a distribution $\mathbf{D}$ over $\{0,1\}^n$ if $\Pr_{x \leftarrow \mathbf{D}}\left[P(x_{<i}) = x_i\right] \geq \frac{1}{2} + \delta$.

Lemma I.3 (Yao's next-bit-predictor)**.** *For any $C\colon \{0,1\}^n \to \{0,1\}$ and distribution $\mathbf{D}$ over $\{0,1\}^n$, if C is an ε-distinguisher for $\mathbf{D}$, then there exists $i \in [n]$ and $\sigma_1, \sigma_2 \in \{0,1\}$ such that with noticeable probability over $z \in \{0,1\}^{n-i+1}$ it holds that $P(x_{<i}) = C(x_{<i} \circ \sigma_1 \circ z) \oplus \sigma_2$ is a $(1/O(n))$-next-bit-predictor for $\mathbf{D}$.*

The simplicity and generality of this transformation have made it an invaluable tool, most prominently in cryptography and in pseudorandomness (expositions

2575-8454/24/$31.00 ©2024 IEEE
DOI 10.1109/FOCS61266.2024.00095

appear in standard textbooks, e.g. [Gol08], [AB09], [Gol01]). The canonical example for its use is in analyses of pseudorandom generator constructions: Assuming that the output distribution $\mathbf{D}$ of a generator does not "fool" C (i.e., C is a distinguisher for $\mathbf{D}$), one obtains a next-bit-predictor for $\mathbf{D}$, and the argument uses the latter to contradict the results or assumptions on which the generator is based.

Another ubiquitous technical tool is the **"reconstruction paradigm"**, which appeared explicitly in the works of Nisan and Wigderson [Nis91], [NW94] and can be traced back to prior works (e.g., to [Sip88]). Loosely speaking, this is a general way of constructing algorithms from "hard strings" (e.g., truth-tables of hard functions, or incompressible strings). One designs an algorithm that transforms an input string f into the desired object $\mathcal{O}_f$, and this algorithm is coupled with a **reconstruction procedure**, which supports the following claim: For any string f, if $\mathcal{O}_f$ does not have the desired properties, then f is "not hard". Indeed, the reconstruction procedure outputs an "easy" representation of f, such as an efficient algorithm or a small circuit.

Our main focus in this context is trying to construct a distribution $\mathcal{O}_f$ that is pseudorandom for a class of efficient procedures. The canonical example is the "hardness versus randomness" paradigm, introduced in [Nis91], [NW94]: In this context, the string f represents the truth-table of a hard function, the algorithm transforms f into a (hopefully pseudorandom) multiset $\mathcal{O}_f$, and the reconstruction procedure shows that if $\mathcal{O}_f$ is not pseudorandom, then the function represented by f can be computed efficiently (e.g., by a small circuit).[1] Closely related examples exist in numerous areas, such as extractor theory [Tre01], expanders [TSUZ07], and error-correcting codes [STV01].[2]

A. Recent Progress: Derandomizing Yao's Transformation and Reconstruction Arguments for Read-Once Branching Programs

The computational complexity of these two technical tools is crucial. This is because the tools are used in analyses that contradict an initial assumption or result, and the higher the complexity of the tools, the stronger the assumption or result that we need. As an illustrative example, suppose that we want to use Yao's transformation to show that a distribution $\mathbf{D}$ is pseudorandom for a class $\mathcal{C}$. In the analysis, we assume towards a contradiction that $\mathbf{D}$ is not pseudorandom for some $C \in \mathcal{C}$, and use Lemma I.3 to obtain a predictor P; the rest of the argument, which proceeds to contradict an initial assumption or result, will carry on the overhead of transforming C to P.

In essentially all classical applications we are aware of, both tools are modeled as probabilistic procedures, or worse, as non-uniform procedures (which are stronger than probabilistic algorithms). However, very recent works showed that in the *specific context of read-once branching programs*, we can do better. To be more concrete, let us recall a definition of Doron, Pyne, and Tell [DPT24]:

Definition I.4 (D2P, simplified)**.** An algorithm A is a distinguish to predict (D2P) transformation for a class $\mathcal{C}$ if A gets as input a description of a circuit $C\colon \{0,1\}^n \to \{0,1\}$ from $\mathcal{C}$, and prints a list of circuits $P_1, ..., P_m\colon \{0,1\}^* \to \{0,1\}$ such that for *every* distribution $\mathbf{D}$ over $\{0,1\}^n$ the following holds. If C is an ε-distinguisher for $\mathbf{D}$, then there is an $i \in [m]$ such that P_i is an $(\varepsilon/O(n))$-predictor for $\mathbf{D}$.

Some choices in Definition I.4 may seem arbitrary at this point (e.g., we could also define a non-black-box transformation that takes the distribution $\mathbf{D}$ as part of its input, or require the transformation to work only for certain classes of distributions, or consider more general parameter regimes). Nevertheless, these choices will be justified by showing algorithms that satisfy Definition I.4 as well as matching lower bounds.

Indeed, Yao's transformation (i.e., Lemma I.3) can be thought of as a probabilistic D2P transformation for general circuits. The works of Pyne, Raz, and Zhan [PRZ23] and [DPT24], building on [Nis94], [CH22], [GRZ23], showed that there is a *deterministic logspace D2P* algorithm for the class of *read-once branching programs* (ROBPs). While it is not surprising that their algorithm runs in logarithmic space (since Lemma I.3 already yields a probabilistic logspace D2P for ROBPs), the crucial novel point is that the D2P transformation algorithm can be made deterministic.

As a consequence of their D2P algorithm, they deduced the existence of a *derandomized reconstruction procedure* for the classical Nisan-Wigderson PRG [NW94] when the distinguisher is an ROBP.[3] This resulted in what they called **"certified derandomization"**: A deterministic logspace algorithm that gets as

[1] More recent versions of the "hardness vs randomness" paradigm, following [Gol11a], [CT21], work in an instance-wise fashion: Given input x, they compute $f = g(x)$ for some function g, and the reconstruction procedure shows that if $\mathcal{O}_f$ is not pseudorandom, then g is easy to compute on the specific input x. See, e.g., [CT23] for more details.

[2] Many standard reconstruction procedures (e.g., [NW94], [TSZS06], [SU05], [Uma03]) use Yao's next-bit-predictor lemma as a key step. This can be viewed as *reducing* reconstruction to constructing a next-bit-predictor. We further explain this point in Section I-D.

[3] The notion of "derandomized" here means that the procedure uses only $O(\log(|f|))$ random coins, where f is the truth-table of the function on which the generator is based. Indeed, the reconstruction procedure is deterministic, but its complexity may be higher than that of computing f to begin with (where the point is that it outputs a small circuit for f).

input a truth-table f and an ROBP C, and either confirms that the PRG instantiated with f is pseudorandom for C, or prints a small circuit whose truth-table is f.

Beyond the ROBP setting, however, the notions of deterministic D2P transformation and certified derandomization have not been studied in detail. For general circuits, it is not a-priori clear whether to expect impossibility results (on the one hand) or easy constructions (on the other hand). In fact, it is not even a-priori clear that *non-explicit* deterministic D2P transformations exist for general circuits, since the algorithm in Definition I.4 is required to work for *all* distributions $\mathbf{D}$.

B. Our contributions: A bird's eye

In this work we give appealing evidence that derandomization of the two tools is possible, show surprisingly strong consequences of such derandomization, and reveal several new settings where such derandomization is unconditionally possible for algorithms stronger than ROBPs (and has useful consequences). Specifically:

- We show that derandomizing these tools is equivalent to general derandomization. In particular, we show that derandomized D2P is equivalent to $\mathbf{prBPP} = \mathbf{prP}$ and that certified derandomization (in a more general sense that we introduce) is equivalent to $\mathbf{prBPP} = \mathbf{prZPP}$. These results appear in Sections I-C and I-D.
- Our main technical contributions are unconditional constructions of derandomized D2P transformations (or reductions of this task to other problems) for classes and for algorithms beyond ROBPs. Consequently, we deduce new results: A significant relaxation of the hypotheses required to derandomize the isolation lemma for logspace algorithms and deduce that $\mathbf{NL} = \mathbf{UL}$; and proofs that derandomization necessitates targeted PRGs in catalytic logspace (unconditionally) and in logspace (conditionally). These contributions appear in Sections I-C1 and I-C2.

In addition, we introduce a natural subclass of $\mathbf{prZPP}$ that has been implicitly studied in recent works on the range avoidance problem [Kor21], [Kor22], [ILW23]: The class of problems reducible to a problem called LossyCode (see Problem I.16). We provide a structural characterization for this class using the notion of certified derandomization, and show that this characterization is robust to several natural variations (see Section I-D).

As a last contribution, we present alternative proofs of two classical results in the theory of pseudorandomness: The reduction of derandomization of $\mathbf{prBPP}$ to derandomization of $\mathbf{prRP}$ [Sip83], [Lau83], [ACR98], [ACRT99], [GVW11], [GZ11], [CH22], and the fact that $\mathbf{MA} \subseteq \mathbf{S_2P}$ [RS98]. Our proofs are technically simple and appealing, and rely on D2P transformations as a main ingredient.

C. Derandomized D2P Transformations and Their Consequences

Should we expect D2P transformations to exist, and should we expect to explicitly construct them any time soon? A recent result of Korten [Kor22, Corollary 41] implies the following statement: If there is a deterministic D2P transformation for general circuits, then $\mathbf{BPP} \subseteq \mathbf{NP}$. Moreover, it was implicitly proved by Goldreich [Gol11a, Appendix A] (following ideas in [GW00]) that D2P transformation exists with respect to a fixed universal distribution, assuming $\mathbf{prBPP} = \mathbf{prP}$. Both works, however, do not settle the existence of this transformation, even in the non-explicit setting.

The first result, which motivates the result of our work, asserts that a derandomized D2P algorithm for general circuits follows from general derandomization (i.e., from $\mathbf{prBPP} = \mathbf{prP}$), and is in fact *equivalent* to it.

Theorem I.5 (D2P $\iff$ derandomization)**.** *The following are equivalent:*

1) $\mathbf{prBPP} = \mathbf{prP}$.
2) *There exists a deterministic polynomial-time D2P algorithm for general circuits.*

Moreover, there unconditionally exists a polynomial-sized family of non-uniform circuits for D2P of general circuits.

The surprisingly simple proof of Theorem I.5 combines an idea of Goldreich and Wigderson [GW00] with the recent "instance-wise" approach to derandomization (following [Gol11a], [CT21]).

The equivalence in Theorem I.5 has a positive aspect and a discouraging one: The result means that D2P exists under the widely believed conjecture $\mathbf{prBPP} = \mathbf{prP}$, but it also means that constructing a D2P algorithm requires proving this conjecture. We focus on the positive aspect.

a) A natural challenge: Derandomizing D2P beyond ROBPs.: Motivated by Theorem I.5, we consider the following challenge:

Open Problem I.6. Unconditionally construct deterministic D2P transformations (or deterministic reductions of D2P to other tasks) for algorithms beyond the ROBP setting, and leverage these constructions to make progress on long-standing questions.

Our main technical contributions are two solutions to Open Problem I.6: We construct new D2P transformations, going beyond the ROBP setting, and leverage them to make progress on the following two long-standing questions: making nondeterministic logspace unambiguous (Section I-C1), and reducing targeted PRGs to derandomization (Section I-C2). We view these positive

results as suggesting that more positive answers to Open Problem I.6 may be found.

1) The Isolation Lemma And Unambiguous Logspace: The isolation lemma of Mulmuley, Valiant, and Vazirani [VV86], [MVV87] (see also [CRS95]) gives a randomized procedure to reduce a search problem with many solutions to one with a single valid solution. This procedure has found many uses in algorithms and complexity; among the well-known examples are [Tod91], [BDCGL92]. We focus on its application in reducing nondeterminism to *unambiguous* nondeterminism, where each "yes" instance has exactly one valid witness (as in [Val77]).

In the general case, there is evidence that derandomizing the isolation lemma for this purpose is impossible (since it was shown in [DKvMW13] to be equivalent to $\mathsf{NP} \subseteq \mathsf{P}/\operatorname{poly}$), and even derandomizing restricted versions of it implies circuit lower bounds [AM08]. Part of the difficulty is that it is not clear how to identify a good candidate (i.e., an instance that has exactly one satisfying solution) in an unambiguous way, so even strong PRGs are not known to imply $\mathsf{NP} = \mathsf{UP}$.

In the bounded-space setting, however, there is evidence that we *can* make nondeterminism unambiguous. In contrast to recognizing circuits with a unique satisfying assignment, recognizing if a graph has unique shortest paths can be done in UL [RA00], [GW96]. Leveraging this fact, Reinhardt and Allender [RA00] showed that to prove $\mathsf{NL} = \mathsf{UL}$, it suffices to construct weight functions that induce *unique shortest paths* in UL:

Problem I.7 (path isolation)**.** Construct in UL a set of weight functions $\{w_1, \ldots, w_{n^c}\}$ with $w_i : E \to [n^{10}]$ such that for every graph $G = (V, E)$ on n vertices, there is some i such that the weighted graph (G, w_i) has *unique shortest paths (USPs)*.

Allender et al. [ARZ99] showed that such a construction is possible assuming strong circuit lower bounds: in particular, hardness of $\mathsf{SPACE}[n]$ for general circuits of size $2^{\varepsilon n}$. Subsequently, there has been extensive work designing space-efficient unambiguous verifiers for various problems. In particular, placing connectivity for restricted families of directed graphs in UL [BTV09], [KV10], [GST19], reducing the complexity of connectivity for general graphs [vMP19], [KT16], [Hoz19].

The reason prior conditional results of [GW96], [ARZ99] required strong circuit lower bounds is precisely because they apply the generic probabilistic D2P transformation of Yao to the distinguisher $T_G(w)$ that checks if (G, w) has unique shortest paths.

a) Our results.: Motivated by this observation, we unconditionally construct a deterministic logspace D2P transformation for this particular distinguisher T_G:

Theorem I.8 (Informal)**.** *There is a logspace-computable D2P transformation for the class of distinguishers $\{T_G\}_G$, where G is a directed graph and $T_G(w) = \mathbb{I}[w$ induces USPs in $G]$.*

We leverage this transformation to significantly weaken the assumptions needed in previous work [ARZ99] to deduce that non-deterministic bounded-space computation can be made unambiguous. Specifically, instead of strong circuit lower bounds, we show that lower bounds for *uniform and deterministic* (or near-deterministic) algorithms suffice.

We show two results, one for a "scaled-up" parameter setting, and one for the logspace setting. For context, recall that [ARZ99] deduced that $\mathsf{NL} = \mathsf{UL}$ from hardness of $\mathsf{SPACE}[n]$ for general non-uniform circuits of size $2^{\varepsilon n}$ (for some $\varepsilon > 0$).

Our first result deduces a scaled-up version of their conclusion from a lower bound for *uniform and deterministic procedures*; specifically, from a lower bound for circuits that are printable by a nondeterministic logspace algorithm, that moreover prints a circuit on only one guess sequence.[4] Specifically:

Theorem I.9 (Informal)**.** *Suppose there exists $\varepsilon > 0$ such that $\mathsf{SPACE}[n]$ is hard for UL-uniform circuits of size $2^{\varepsilon n}$. Then $\mathsf{NSPACE}[O(n)] = \mathsf{USPACE}[O(n)]$.*

We stress that the hypotheses in Theorem I.9 (and Assumption I.10) are considerably weaker than hypotheses that are typically required for hardness-vs-randomness results. In particular, the latter rely on lower bounds either for non-uniform circuits (as in, say, [NW94], [IW97]) or for probabilistic algorithms (as in, say, [CT21], [CTW23]).[5]

In fact, our technical result is stronger, and shows that the conclusion of Theorem I.9 follows from a weaker hypothesis; namely, from hardness of $\mathsf{USPACE}[n]$ against (deterministic, uniform) TC^0 circuits with low-space oracles.

For our second result, to deduce that $\mathsf{NL} = \mathsf{UL}$ (i.e., a scaled-down conclusion as in [ARZ99]), we will assume hardness of functions in NC^1 for uniform algorithms that use only $\operatorname{polylog}(n)$ random coins. Specifically:

Assumption I.10 (hardness in NC^1 for uniform near-deterministic algorithms)**.** For every $c \in \mathbb{N}$, there exists

[4]The uniformity requirement is significantly weaker than the standard requirement that the circuit is printable in polynomial time (i.e., P-uniformity), let alone from models such as NTIME-uniformity (e.g., as in [SW13], [CRTY20]).

[5]One recent exception is the work of Doron, Pyne, and Tell [DPT24] on derandomizing BPL, and another exception is the study of pseudorandomness for deterministic observers by Goldreich and Wigderson [GW00]. We build on both works, and in particular we extend the results of [DPT24] the specific setting of ROBPs to more general settings. See Section II for further details.

$C \in \mathbb{N}$ and a family of functions $\{f : \{0,1\}^n \to \{0,1\}^n\}_{n\in\mathbb{N}}$ computable by logspace-uniform $\mathbf{NC}^1$-circuits of size n^C, such that there is no time n^c algorithm using $\mathrm{polylog}(n)$ many coins that on infinitely many x prints $f(x)$ with probability at least $2/3$.

Theorem I.11. *Suppose that Assumption I.10 holds. Then* $\mathbf{NL} = \mathbf{UL}$.

The technical contribution underlying the foregoing results is two-fold: Constructing the D2P transformation stated in Theorem I.8, and leveraging it via new "hardness-vs-randomness" tradeoffs to obtain Theorems I.9 and I.11. The latter contribution further develops very recent work on targeted pseudorandom generators with randomness-efficient reconstruction procedures [PRZ23], [DPT24]. In particular, in the proof of Theorem I.9 we show that such reconstruction procedures can be made to satisfy $\mathbf{UL}\cap\mathbf{coUL}$ uniformity, and in the proof Theorem I.11, we construct a version of the Chen-Tell targeted HSG [CT21] that works with a hard function in $\mathbf{NC}^1$, is computable in logspace, and has a derandomized reconstruction. See Section II for details.

2) Derandomization Requires Targeted Generators in $\mathbf{CL}$ *and in* $\mathbf{L}$*:* Assuming that we can solve CAPP for a class $\mathcal{C}$ of circuits,[6] can we also output a a distribution $\mathbf{D}$ that fools a given $C \in \mathcal{C}$? In particular, do $\mathbf{BPP}$-*search* problems (a-la [Gol11a]) reduce to the *decision* problem CAPP? This question was first posed by Goldreich [Gol11a], [Gol11b], who phrased it as the question of whether derandomization requires *targeted PRGs*.

Goldreich [Gol11b], [Gol11a] proved such a result for $\mathbf{prBPP}$ (i.e., when the CAPP algorithm is a general probabilistic algorithm), and posed the open question of obtaining analogous results for classes such as $\mathbf{AM}$ and $\mathbf{L}$. For recent progress see, e.g., [HU22], [vMS23a], [vMS23b], [PR23].

We *resolve this question for the catalytic logspace (*$\mathbf{CL}$*) model* of Buhrman et al. [BCK+14], and weaken the assumptions required to resolve this question for $\mathbf{L}$.

a) Derandomization in $\mathbf{CL}$ *requires targeted PRGs.:* In the catalytic logspace model, we are given $O(\log n)$ bits of standard workspace, and a *catalytic* tape $\mathbf{w}$ of length n^c, which functions as follows. The tape $\mathbf{w}$ is initialized to an arbitrary value, and we may edit it during the computation, but must exactly reset the tape to the original configuration at the end. The work of [BCK+14] proved that logspace-uniform $\mathbf{TC}^1$ is contained in $\mathbf{CL}$, so in particular $\mathbf{NL} \subseteq \mathbf{CL}$. Since this intriguing result there has been extensive work on the model [BKLS18], [GJST19], [DGJ+12], [CM20], [CM23], [DPT24], [Pyn24], [CLM+24] (see the survey of Mertz [Mer23] for an excellent exposition).

Despite extensive recent interest, many basic structural questions remain open. In particular, prior to this work it was not known whether solving $\mathbf{BPP}$-search problems reduces to CAPP in $\mathbf{CL}$. As mentioned above, we resolve this question in the affirmative:

Theorem I.12 (informal)**.** *Suppose that there is a* $\mathbf{CL}$*-computable* CAPP *algorithm for a* $\mathbf{CL}$*-evaluable class of circuits* $\mathcal{C}$*. Then:*

1) *There is a* $\mathbf{CL}$*-computable D2P transformation for* $\mathcal{C}$ *circuits.*
2) *There is a* $\mathbf{CL}$ *algorithm that, given* $C \in \mathcal{C}$*, outputs a distribution* $\mathbf{D}$ *that* $(1/3)$*-fools* C*.*

Our proof proceeds in two steps: We first reduce the task of producing $\mathbf{D}$ to D2P in $\mathbf{CL}$, and then reduce D2P to CAPP in $\mathbf{CL}$. The first and main step combines the "compress or random" approach in catalytic computation (which tries to use the catalytic tape as a hard truth-table; see, e.g., [DPT24] and [Mer23, Section 3.2.1]) with ideas from the proof of Theorem I.5. The second step shows that the reduction of D2P to CAPP from Theorem I.5 can be implemented in $\mathbf{CL}$. Crucially, our proof relies on the fact that our reductions of D2P to CAPP are "instance-wise", in the sense that a CAPP algorithm for a fixed circuit C yields a deterministic D2P transformation for C specifically. Details appear in Section II-C.

In addition, combining the ideas in the proofs of Theorems I.8 and I.12 with the main result of [BCK+14], we show that we can derandomize the path isolation lemma in $\mathbf{CL}$. Note that this is the first $\mathbf{CL}$ algorithm for a natural problem that combines *both* main algorithmic techniques for $\mathbf{CL}$, namely the algebraic computation approach [BCK+14], [CM23] and the compress-or-random approach [Pyn24], [DPT24].

b) Derandomization in $\mathbf{L}$ *requires targeted PRGs, under weak assumptions.:* Finally, we show that derandomization in logspace indeed implies targeted logspace PRGs, assuming lower bounds against uniform algorithms that use only $\mathrm{polylog}(n)$ random coins:

Theorem I.13 (informal)**.** *Suppose that Assumption I.10 holds. Let* $\mathcal{C}$ *be an arbitrary circuit class that is evaluable in* $\mathbf{L}$*, and suppose there is a logspace* CAPP *algorithm for* $\mathcal{C}$*. Then, there is a logspace algorithm that, given* $C \in \mathcal{C}$*, outputs a distribution* D *that* $(1/3)$*-fools* C*.*

The conditional statement in Theorem I.13 is the first result indicating that the answer to the open problem is affirmative (i.e., that logspace derandomization necessitates logspace targeted PRGs) without relying on assumptions that are sufficiently strong to immediately yield (by themselves) logspace targeted PRGs.

[6]Recall that CAPP is the promise problem whose "yes" instances are circuits C such that $\Pr_r[C(r) = 1] \geq 2/3$ and whose "no" instances are circuits C such that $\Pr_r[C(r) = 1] \leq 1/3$. Also recall that CAPP is complete for $\mathbf{prBPP}$.

D. Certified Derandomization and the Class LOSSY

We now turn our attention to a notion of *derandomized reconstruction procedures*. Specifically, we focus on what was coined by Pyne, Raz, and Zhan [PRZ23] as certified derandomization: Given a circuit $C : \{0,1\}^n \to \{0,1\}$ and a truth-table $f \in \{0,1\}^{\mathrm{poly}(n)}$, either estimate the acceptance probability of C to an additive error of $1/6$ (i.e. solve CAPP for C), or construct a small circuit for f. (Indeed, we think of such an algorithm as executing a derandomized version of the classical reconstruction procedure a-la [NW94]: When the algorithm is unable to use f to obtain a pseudorandom distribution for C, it deterministically finds a small circuit for f.)

a) Context: Certified derandomization, derandomized D2P, and reconstruction procedures.: Certified derandomization is a natural notion in and of itself, which was constructed for ROBPs in [PRZ23] and which can be constructed for more general classes (see next).

An additional motivation for studying certified derandomization arises from understanding reconstruction procedures in the "hardness vs randomness" paradigm. Classical reconstruction procedures (e.g., in [NW94], [TSZS06], [SU05], [Uma03]) use D2P as a key technical step. However, it is not clear if this approach (i.e., designing reconstruction procedures that go through D2P) is necessary, or if there are approaches that avoid D2P altogether.[7]

To be more concrete, we know that certified derandomization deterministically reduces to constructing a D2P transformation: Either using the derandomized reconstruction procedure for the Nisan-Widgerson PRG of [PRZ23], or using our equivalence between derandomizing D2P and **prBPP** $=$ **prP** (the latter trivially implies certified derandomization, as we can simply ignore the truth-table f provided and solve CAPP). Nevertheless, the characterization of certified derandomization stated below implies that *reducing certified derandomization to D2P may be an overkill*, as the latter is equivalent to **prBPP** $=$ **prP** (by Theorem I.5) whereas the former is only equivalent to **prBPP** $=$ **prZPP** (see Theorem I.15).

b) A general notion of certified derandomization, and **prZPP** $=$ **prBPP**.*:* The certified derandomization algorithm of [PRZ23] uses strings f that are truth-tables with high circuit complexity (i.e., if f has high circuit complexity, then the algorithm estimates the acceptance probability the given circuit). One can think of the set of truth-tables with high circuit complexity as a dense property of strings (i.e., inspired by Razborov and Rudich [RR97]), where this property is in **coNP**. It is not, however, clear why we should use this specific property for the purpose of certified derandomization, rather than any other property in **coNP**.

Accordingly, we define a general notion of certified derandomization using *an arbitrary dense property* $\mathcal{P} \in$ **coNP**: The certified derandomization algorithm is given C and a string τ, and is required to either solve CAPP for C (which it should be able to do whenever $\tau \in \mathcal{P}$), or provide a witness w that $\tau \notin \mathcal{P}$.

We prove that certified derandomization in this more general sense is *equivalent* to **prBPP** $=$ **prZPP**; that is, **prBPP** $=$ **prZPP** if and only if there exists a dense property $\mathcal{P} \in$ **coNP** and a certified derandomization algorithm using $\mathcal{P}$. As explained above, this result yields a conceptual separation between certified derandomization and D2P.

Definition I.14 (certified derandomization)**.** For $\ell = \ell(n) = 2^{o(n)}$, let $\mathcal{P} = \{\mathcal{P}_n \subseteq \{0,1\}^\ell\}_{n\in\mathbb{N}} \in$ **coNP** such that $\mathcal{P} \cap \{0,1\}^n \neq \varnothing$ for every $n \in \mathbb{N}$. Let V be a **coNP** verifier of $\mathcal{P}$. An algorithm A is a certified derandomization algorithm using $\mathcal{P}$ (with respect to the verifier V) if for every linear-size circuit $C : \{0,1\}^n \to \{0,1\}$ and every $\tau \in \{0,1\}^\ell$,

- If $\tau \in \mathcal{P}$ then $A(C,\tau)$ solves CAPP on C.
- If $\tau \notin \mathcal{P}$ then either $A(C,\tau)$ solves CAPP on C, or $A(C,\tau)$ prints w such that $V(\tau,w) = 0$.

Theorem I.15 (certified derandomization $\iff$ **prBPP** $=$ **prZPP**)**.** *The following statements are equivalent.*

- **prBPP** $=$ **prZPP**.
- *There is a deterministic polynomial-time certified derandomization algorithm using a dense property* $\mathcal{P} = \{\mathcal{P}_n \subseteq \{0,1\}^\ell\}_{n\in\mathbb{N}} \in$ **coNP**, *where* $\ell = \ell(n) = \mathrm{poly}(n)$.

The proof of Theorem I.15 is elementary, and appears in the full version.

c) The class LOSSY.*:* It turns out, however, that the more restricted notion of certified derandomization with hard truth-tables (as in [PRZ23]) is interesting in and of itself. A recent work of Korten [Kor22] introduced a search problem called LossyCode, which admits a randomized polynomial-time zero-error algorithm, and asked what is the set of problems reducible to LossyCode.

Problem I.16 (LossyCode)**.** Given a pair of circuits $C : \{0,1\}^n \to \{0,1\}^m$, $D : \{0,1\}^m \to \{0,1\}^n$, where $m < n$, find a string $x \in \{0,1\}^n$ such that $D(C(x)) \neq x$.

We define the subclass **LOSSY** $\subseteq$ **ZPP** as the class of languages reducible to LossyCode in deterministic

[7] A closely related question, focusing on the *hybrid argument*, was studied by Fefferman *et al.* [FSUV13] motivated by avoiding the $1/n$ advantage loss.

polynomial time.[8] This is an interesting class, with one motivation coming from proof complexity: Loosely speaking, if a statement of the form "$\forall x\ \exists y\ \varphi(x,y)$" (where φ is a quantifier-free formula) can be proved in the bounded theory $\mathbf{APC}_1$ (see [Jeř04], [Jeř07] for the definition and related discussion), then the corresponding search problem can be solved in **FLOSSY** (i.e., in the functional version of **LOSSY**).

We provide additional motivation for studying **LOSSY**, by showing that this class has an interesting structural characterization, which relies on the notion of certified derandomization with hard truth tables. Specifically, we prove the following:

Theorem I.17 (informal)**.** *The following statements are equivalent.*

(1) $\mathbf{prBPP} = \mathbf{prLOSSY}$.
(2) *There is a deterministic polynomial-time certified derandomization algorithm using hard truth tables.*

In fact, we further extend Theorem I.17, by showing that the two statements are also equivalent to the existence of a deterministic polynomial-time certified derandomization algorithm using any property defined by an efficient *Range Avoidance* problem.[9] These equivalences significantly strengthen results from [Kor22]; see the full version for further details.

II. Overview of Proofs

In Section II-A we describe the equivalence between D2P and derandomization (i.e., the proof of Theorem I.5). Our main technical contributions are described in Sections II-B and II-C:

- In Section II-B we explain our construction of a deterministic logspace D2P algorithm for unique shortest paths (i.e., Theorem I.8) and how it allows deducing $\mathbf{NL} = \mathbf{UL}$ and $\mathbf{NSPACE}[n] = \mathbf{USPACE}[n]$ from weaker assumptions (i.e., Theorems I.9 and I.11).
- In Section II-C we explain our reduction of targeted PRGs to CAPP in catalytic logspace (i.e., Theorem I.12) and the analogous conditional reduction in **L** (i.e., Theorem I.13).

Finally, in Section II-D we describe the equivalences between certified derandomization and zero-error derandomization.

[8]It turns out that the definition of **LOSSY** is robust with respect to the type of the reduction. Specifically, we prove that any language that is Cook-type reducible to LossyCode (i.e. the reduction could call the LossyCode oracle multiple times adaptively) is also Karp-type reducible to LossyCode (i.e. the reduction calls the LossyCode oracle only once).

[9]That is, any property of strings that are outside the range of an efficiently computable function $g : \{0,1\}^n \to \{0,1\}^m$ such that $m > n$ (i.e. a string $y \in \{0,1\}^m$ such that $g^{-1}(y) = \varnothing$). For more details about the range avoidance problem, see [Kor21], [RSW22], [GLW22], [GGNS23], [CHLR23], [ILW23], [CHR24], [Li24], [CL24].

A. D2P is Equivalent to Derandomization

Our goal is to show that deterministic D2P implies $\mathbf{prBPP} = \mathbf{prP}$, and vice versa. At a high level, we build on ideas from a sequence of works by Goldreich and Wigderson [GW00], [Gol11b], [Gol11a], who examined what they called "deterministic observers". We prove the equivalence by combining their ideas with the "instance-wise" approach to derandomization (i.e., derandomization that uses targeted PRGs instead of classical PRGs), following Goldreich [Gol11a] and Chen and Tell [CT21].

1) D2P implies derandomization: Goldreich and Wigderson [GW00] constructed a distribution ensemble $\mathbf{D} = \{\mathbf{D}_n\}_{n\in\mathbb{N}}$ that is *unpredictable* by uniform deterministic Turing machines. In a gist, on input length n they consider the first (say) n machines, and using a diagonalization-style approach, they build $\mathbf{D}_n$ bit-by-bit. In each iteration the distribution $\mathbf{D}_n^{(i)}$ consists of i-bit strings, and they search a pseudorandom sample space to find an extension of the strings in $\mathbf{D}_n^{(i)}$ such that none of the n machines can predict the new distribution $\mathbf{D}_n^{(i+1)}$.[10]

At a high level, if deterministic D2P is possible, then any efficient distinguisher yields a deterministic predictor for $\mathbf{D}_n$. Since $\mathbf{D}_n$ is unpredictable by such machines (by its construction), we intuitively expect to deduce that $\mathbf{D}_n$ is also pseudorandom.

The only issue is that (in contrast to [GW00]) we are trying to obtain worst-case derandomization. That is, in our setting the machine M that tries to distinguish $\mathbf{D}_n$ from uniform also has access to a (worst-case) input $x \in \{0,1\}^n$. Hence, instead of trying to construct a distribution that is unpredictable by such procedures, we simply adapt the approach to yield *non-black-box* derandomization. Specifically, given input $x \in \{0,1\}^n$, we build $\mathbf{D}_x$ that is unpredictable by any *efficient machine that also gets access to* x (using the same diagonalization-style approach). Assuming that deterministic D2P is possible, $\mathbf{D}_x$ is indistinguishable from uniform by efficient machines that get access to x, and in particular by $M(x,\cdot)$. Indeed, this construction is a targeted PRG, mapping x to a multiset $\mathbf{D}_x$ that is pseudorandom for distinguishers of the form $M(x,\cdot)$.[11]

This simple argument is versatile, and yields several interesting corollaries, for instance an equivalence between D2P and superfast derandomization under OWFs.

[10]They show that even a pairwise-independent distribution suffices for this purpose, while we apply stronger tools to obtain a stronger equivalence.

[11]The "missing observations" in [GW00] seems to be two-fold: First, clearly defining the notion of D2P and asking about its implications; and secondly, considering non-black-box derandomization by targeted PRGs, which is a notion that was introduced in a follow-up work [Gol11a] and extensively studied only recently (following [CT21]).

We present the technical details as well as extensions and corollaries in the full version.

2) Derandomization implies D2P: To discuss the other direction, recall that by Yao's Lemma, for every circuit $C : \{0,1\}^n \to \{0,1\}$ and distribution $\mathbf{D}_n$ that C distinguishes from uniform, there is an index $i \in [n]$ and $\sigma \in \{0,1\}^2$ such that

$$\mathop{\mathbb{E}}_{z}\left[\Pr_{x\leftarrow\mathbf{D}_n}[x_i = C(x_{<i} \circ \sigma_1 \circ z) \oplus \sigma_2]\right] \geq \frac{1}{2} + \frac{1}{\mathrm{poly}(n)}. \tag{II.1}$$

A natural strategy is to try and find z that approximately achieves this expectation. For example, this is indeed the strategy undertaken in all uses of Yao's transformation in the "hardness vs randomness" paradigm (see, e.g., [NW94], [STV01], [CT21], [DPT24], for a collection of arguments that all rely on finding such a z). Unfortunately, we show that explicitly constructing a good family of these strings is equivalent to the explicit construction of *hitting sets* (and thus circuit lower bounds), and hence we are unlikely to deduce such a result from $\mathbf{prBPP} = \mathbf{prP}$.

Intuitively, the key difficulty is that testing whether z maintains the advantage in Eq. (II.1) for every distribution $\mathbf{D}_n$ over $\{0,1\}^n$ for which C is a distinguisher seems to require doubly exponential time (since there can be doubly exponentially many such distributions). Thus, it is unclear how to apply a CAPP algorithm to a polynomial-sized circuit to find such a z.

Our key observation is as follows. Instead of trying to use the CAPP algorithm to find a good z that will be hard-wired into a predictor P_z, we will construct a predictor that *uses the* CAPP *algorithm to predict the next bit*. This way, instead of considering all possible distributions over inputs $x_{<i}$ to the predictor, we *fix an input* $x_{<i}$ that is explicitly given to the CAPP algorithm, and the CAPP algorithm only needs to consider the uniform distribution over suffixes z to approximate the LHS of Eq. (II.1) on this given prefix. (This approach is technically reminiscent of a proof in [Gol11a, Appendix A], although the settings and notions are different.)

In more detail, fixing $i \in [n]$ and $\sigma \in \{0,1\}$, let us first construct a "non-Boolean predictor" $\tilde{P}_{i,\sigma}$. For simplicity of presentation, let us assume that $\sigma = 00$, and denote $\tilde{P}_i = \tilde{P}_{i,\sigma}$. Given input $x_{<i}$, the predictor estimates the value $\mathbb{E}_z[C(x_{<i} \circ z)]$, up to a polynomially small error.[12] By Eq. (II.1) and linearity of expectation, for some i, σ (again, assume $\sigma = 0$) we have

$$\begin{aligned} &1/2 + 1/\mathrm{poly}(n) \\ &\quad\leq \mathop{\mathbb{E}}_{x}\Big[\mathbf{1}[x_i = 1] \cdot \Pr_z[C(x_{<i} \circ z) = 1] \\ &\qquad + \mathbf{1}[x_i = 0] \cdot \Pr_z[C(x_{<i} \circ z) = 0]\Big] \\ &\quad\approx_{1/\mathrm{poly}(n)} \mathop{\mathbb{E}}_{x}\Big[\mathbf{1}[x_i = 1] \cdot \tilde{P}_i(x_{<i}) \\ &\qquad + \mathbf{1}[x_i = 0] \cdot (1 - \tilde{P}_i(x_{<i}))\Big] \ ; \end{aligned} \tag{II.2}$$

in other words, $\tilde{P}$ computes a real value whose correlation with the event "$x_i = 1$" is non-trivial.

This almost finishes the construction, since now we just need to convert $\tilde{P}$ into a Boolean predictor. This can be done in a generic way: An elementary argument shows that for any real-valued function $\tilde{P}_i$ with correlation as in Eq. (II.2) there is a threshold $\tau \in \{i \cdot \varepsilon\}_{i=1,\ldots,1/\varepsilon}$ (where $\varepsilon = 1/\mathrm{poly}(n)$) such that the Boolean function $P_{i,\tau}(x_{<i}) = \mathbf{1}[\tilde{P}_i(x_{<i}) \leq \tau]$ has correlation $1/2 + 1/\mathrm{poly}(n)$ with the event "$x_i = 1$" (i.e., $\Pr_{x_{<i}}[P_{i,\tau}(x_{<i}) = x_i] \geq 1/2 + 1/\mathrm{poly}(n)$).[13]

Thus, our D2P algorithm outputs the collection $\{P_{i,\sigma,\tau}\}$. This collection is indeed of polynomial size, and for every distribution $\mathbf{D}_n$ for which Eq. (II.1) holds (in particular, for every distribution for which C is a distinguisher), the collection has a predictor for $\mathbf{D}_n$.

Remark II.1. Indeed, this direction is a-priori far less obvious than the first one. Note that the argument is "instance-wise": Solving CAPP for (prefixes of) a certain circuit C yields a deterministic D2P transformation for C specifically. We will crucially use this property in Section II-C.

B. D2P for Unique Shortest Paths, and Derandomizing the Path Isolation Lemma

We now explain how to construct a specific deterministic D2P transformation and use it to deduce that $\mathbf{NL} = \mathbf{UL}$ (or $\mathbf{NSPACE}[n] = \mathbf{USPACE}[n]$) from weak hardness assumptions (i.e., for deterministic uniform algorithms).

Our argument has two parts. We first construct a deterministic logspace-computable D2P for a distinguisher $T = T_G$ that decides whether a weight assignment induces unique shortest paths in the graph G. The reason for focusing on this specific T is that finding such a weight assignment suffices to deduce that $\mathbf{UL} = \mathbf{NL}$ (see [RA00], [GW96]). The second part of our argument "lifts" this D2P to a proof that $\mathbf{UL} = \mathbf{NL}$, under weak assumptions.

[12] That is, the predictor constructs the circuit $D(z) = C(x_{<i} \circ z)$, and outputs the real value obtained by applying the CAPP algorithm to D (with sufficiently small error $1/\mathrm{poly}(n)$).

[13] More generally, for any two random variables $\mathbf{x} \in [0,1]$ and $\mathbf{y} \in [-1,1]$ such that $\mathbb{E}[\mathbf{x} \cdot \mathbf{y}] \geq \delta$, there is $\tau \in [1/\varepsilon]$ such that $\mathbb{E}[\mathbf{1}_{\mathbf{x} \leq \varepsilon \cdot \tau} \cdot \mathbf{y}] \geq \delta - \varepsilon$.

a) A deterministic logspace D2P for unique shortest paths.: Consider the function T_G that takes in a weight assignment $w : E \to [n^{10}]$ and accepts if the weighted graph (G, w) has *unique shortest paths*.[14] Note that T_G does not seem to be computable by an ROBP (since checking unique shortest paths between every pair of vertices seemingly requires reading edge weights many times over), and thus the previously known D2P transformation for ROBPs does not suffice.

In order to obtain our D2P transformation, we first make the distinguisher *stricter*. We place a fixed ordering $e_1, \ldots, e_m$ on the edges of $G = (V, E)$, define $G_i = (V, E_i = \{e_1, \ldots, e_i\})$, and let w_i be the restriction of $w : E \to [n^{10}]$ to E_i. Then we define:

$$T_G(w) = \bigwedge_{i \in [m]} \mathbb{I}\left[w_i \text{ induces USPs in } G_i\right],$$

i.e. every prefix of the weight function w likewise induces unique shortest paths. A random weight assignment still satisfies this stricter condition with high probability.

Recall that in Section II-A2 we showed a construction of D2P that uses a CAPP algorithm; this can be viewed as a reduction of D2P to CAPP. As mentioned in Remark II.1, this reduction is instance-wise, in the sense that solving CAPP for a specific circuit yields D2P for that circuit. We now use the same instance-wise reduction of D2P to CAPP, while performing the hybrid argument of Yao (that yields Eq. (II.1)) in the same order as T_G reads the edge weights. We deduce that constructing a D2P transform for T_G reduces to solving the following problem: Given an arbitrary partial assignment $w_i : E_i \to [n^{10}]$, estimate $\mathbb{E}_z[T_G(w_i \circ z)]$.

The key point is that with the stricter distinguisher and the choice of hybrid order, we obtain a *polarization* effect. In particular, one of the following holds:

- The partial assignment has *already* failed to induce USPs in a subgraph G_j for $j \leq i$. In this case, $T_G(w_i \circ z) = 0$ for every z.
- The partial assignment has not already failed to induce USPs in a subgraph. In this case, we show that *almost all suffixes* z will successfully induce USPs, no matter the current prefix.

Due to this polarizing effect, we can estimate $\rho = \mathbb{E}_z[T(w_i \circ z)]$ by determining if w_i has already failed to induce USPs (in which case $\rho = 0$) or not (in which case $\rho \approx 1$). Allender and Reinhardt [RA00] constructed a $\mathsf{UL} \cap \mathsf{coUL}$ algorithm for this task (i.e., testing if a fixed assignment induces USPs), and thus we obtain a deterministic logspace D2P transformation where the predictors it outputs are $\mathsf{UL} \cap \mathsf{coUL}$ algorithms.

[14]A simple argument shows that for every graph G, $\mathbb{E}[T_G(\mathbf{U})] \geq 1 - n^{-8}$.

b) From D2P to disambiguation of non-deterministic logspace.: Using the D2P above, we now deduce that $\mathsf{NL} = \mathsf{UL}$ (and $\mathsf{NSPACE}[n] = \mathsf{USPACE}[n]$) from hardness for uniform algorithms that are either deterministic or use only $\mathrm{polylog}(n)$ coins. The idea, following Pyne, Raz, and Zhan [PRZ23], is to use a (targeted) pseudorandom generator with a *near-deterministic reconstruction procedure* conditioned on a deterministic D2P for the relevant distinguisher.

In our case, we let the distinguisher be the test T_G that accepts if the weight set induces unique shortest paths. Given our deterministic D2P for this distinguisher, such a generator transforms hardness for near-deterministic procedures into a set of pseudorandom strings for the distinguisher (which, in our setting, will include a weight assignment that induces unique shortest paths on the input graph). The original generator of [PRZ23] was based on circuit lower bounds, and later on Doron, Pyne, and Tell [DPT24] (building on the framework of [CT21]) constructed a targeted PRG based on lower bounds for deterministic uniform procedures.

Using these works as our starting point, we will need to construct yet another version of the targeted PRG of Chen and Tell [CT21] (following [CRT22], [CTW23], [CLO$^+$23], [DPT24]).[15] We build a logspace-computable targeted PRG that is based on a hard function in logspace-uniform NC^1, where the hardness is for uniform algorithms that use only polylogarithmically many random coins (and that have access to a deterministic D2P for the relevant distinguisher). Since the technical details are quite involved (and are not the conceptual focus of the current paper), let us focus mostly on two key differences, postponing the details to the full version.

- As in all previous works, our targeted PRG encodes the computation of the hard function as a sequence of polynomials. Previous works did so relying either on a hard function in TC^0; or on a preprocessing step that incurs a $\mathrm{polylog}(n)$ depth blowup (using an idea from [Gol18]), which would prohibit evaluating the generator in logspace. To resolve this, we preprocess the circuit in a more careful way, which still incurs a $\mathrm{polylog}(n)$ depth blowup but nevertheless allows us to evaluate the resulting polynomials in logspace.

[15]Each previous version has shortcomings making it unsuitable for the current purpose. Specifically, the targeted PRGs of [CT21], [CLO$^+$23] are not evaluable in logspace (even if the hard function is computable in constant depth), and their reconstruction is probabilistic. The targeted PRG in [CTW23] is logspace-computable, but it is based on hardness in TC^0, and its reconstruction is still probabilistic. The targeted PRG in [DPT24] is also based on hardness in TC^0 rather than in NC^1, and does not yield worst-case derandomization (as we explain below).

- The conclusion in [DPT24] was average-case derandomization, whereas we are interested in worst-case derandomization. The reason for their weaker conclusion is that their reconstruction algorithm was a logspace-uniform circuit, where the logspace machine constructing the circuit had higher space complexity than that of the hard function; this prohibits making the assumptions necessary to conclude worst-case derandomization.[16] To resolve this, we replace parts of their argument as follows. Instead of modeling the reconstruction as a logspace-uniform circuit, we model it as a probabilistic machine; and then, following [PRZ23], we show how to significantly reduce the randomness complexity of this machine, relying on a combination of derandomized D2P transformation with standard sampler-based techniques.

C. Derandomization Requires Targeted PRGs in Catalytic Logspace and in Logspace

Next, we focus on the question of whether solving BPP-search problems in a certain class (specifically, in CL or in L) reduces to solving the decision problem CAPP in that class. Equivalently, we ask whether derandomization in the class requires targeted PRGs. A straightforward search-to-decision reduction in [Gol11a] establishes this for P, but it is highly space-inefficient, and thus unsuitable for CL and for L, where its existence is an open problem [PR23] (the reduction from [Gol11a] also fails for AM, due to other reasons; see [Gol11a], [vMS23a], [vMS23b]).

For concreteness, throughout this section let us assume that all circuits are in some fixed circuit class $\mathcal{C}$ that can be evaluated in logspace (e.g., $\mathcal{C} = \mathsf{NC}^1$).

a) Catalytic logspace.: Recall the setting: We are given a circuit C, we can solve CAPP for C, and we want to construct a distribution $\mathbf{D}$ that is pseudorandom for C.[17]

Our result combines (a modification of) the result of [DPT24] reducing producing a targeted PRG for C to constructing a D2P transformation for C, with our instance-wise reduction from constructing a D2P transformation to solving CAPP. In more detail, we first modify the reduction of [DPT24], which uses the "compress or random" paradigm. They think of the catalytic tape $\mathbf{w}$ as a *hard truth table*, and instantiate a version of the Nisan-Wigderson generator with this truth table. Letting the generator be $\mathsf{NW}^{\mathbf{w}}$ (and note that it has seed length $O(\log n)$), either the generator is pseudorandom for C (in which case we can let $\mathbf{D}$ be its output set, and halt without modifying the tape), or the D2P transformation can be used to compress the tape $\mathbf{w}$, freeing up polynomially many bits on the tape. This enables us to use our (space inefficient) reduction from D2P to producing a targeted PRG (whereas their result used a time-efficient brute force derandomization).

The only missing piece is a D2P transformation in CL (assuming a CAPP algorithm in CL). Indeed, to obtain such an algorithm, we show that our reduction from D2P to CAPP can be implemented in catalytic logspace.

Remark II.2. By combining this search-to-decision reduction with the D2P transformation for the path isolation lemma (and the main result of [BCK+14], which implies that the algorithm of [RA00] can be implemented in CL), we unconditionally obtain a CL algorithm that, given a graph G, outputs a weight assignment w such that (G, w) has unique shortest paths. This constitutes the first result for CL that is proved by *combining* the algebraic computation perspective (to evaluate the D2P transformation) with the compress-or-random perspective (to reduce search to D2P).

b) Logspace machines.: Finally, let us briefly explain how to obtain our *conditional* result that derandomization of a class $\mathcal{C}$ in L necessitates targeted PRGs for $\mathcal{C}$ in L (again, we suggest thinking of $\mathcal{C} = \mathsf{NC}^1$ for concreteness). The main technical tool is the new version of the targeted PRG of [CT21], which was described in Section II-B.

Recall that our hardness assumption is a function $f : \{0,1\}^n \to \{0,1\}^n$ computable by n^C size NC^1 circuits, that is hard for n^c time algorithms that use only $\mathrm{polylog}(n)$ many random coins, for $c < C$. Assuming that we have a CAPP algorithm for $\mathcal{C}$-circuits in L, we use the reduction of D2P to CAPP (from Section II-A) to obtain a deterministic D2P for $\mathcal{C}$-circuits in L. The key observation is that this reduction is computable in logspace, and thus the D2P for $\mathcal{C}$ is a deterministic *logspace* algorithm (and hence is computable time n^c for some c).[18] Using this D2P, we instantiate our targeted PRG with this hard function. Supposing the generator is not pseudorandom for C, we can obtain a predictor for the generator using our deterministic logspace D2P transformation, and then compute the function quickly using only $\mathrm{polylog}(n)$ random coins and n^c time, contradicting the assumption.

[16] Specifically, to deduce worst-case derandomization in their framework (following [CT21]), we need to assume hardness on almost all inputs. If the hard function f is computable in space $c \cdot \log(n)$, and the machine in the reconstruction uses $C \cdot \log(n)$ space for some $C > c$, then the machine can hard-wire values of f (e.g., $f(1^n)$) into the circuit that it prints.

[17] A mistaken intuition is that since $\mathsf{BPL} \subseteq \mathsf{CL}$, we do not need a derandomization hypothesis. However, crucially, the derandomization hypothesis only applies to *decision* problems. Moreover, $\mathsf{BPL} \subseteq \mathsf{CL}$ only means that CAPP for ROBPs is in CL, and here we are concerned with richer circuit classes.

[18] Similarly to [DPT24], our targeted PRG construction uses D2P both when computing the generator and when computing the reconstruction.

D. Certified Derandomization Using Hard Truth-Tables, and the Class LOSSY

Recall that certified derandomization using a property $\mathcal{P}$ refers to a deterministic algorithm that gets a linear-size circuit $C : \{0,1\}^n \to \{0,1\}$ and a string $\tau \in \{0,1\}^{\ell(n)}$, and either estimates the acceptance probability of C or provides a witness that $\tau \notin \mathcal{P}$. We prove that such certified derandomization is equivalent to $\mathbf{prBPP} = \mathbf{prZPP}$.

We now focus on Theorem I.17 that shows an equivalence between a restricted type of certified derandomization – namely, when $\mathcal{P}$ is the property of truth-tables that do not have small circuits (e.g., truth-tables of length 2^ℓ without circuits of size $2^{\cdot 01 \cdot \ell}$) – and a simulation of **prBPP** in the class **LOSSY** of problems reducible to LossyCode. Recall that in LossyCode (see Problem I.16), we are given a pair of circuits $C : \{0,1\}^n \to \{0,1\}^m$ ("compression") and $D : \{0,1\}^m \to \{0,1\}^n$ ("decompression"), where $m < n$, and we want to find a string $x \in \{0,1\}^n$ such that $D(C(x)) \neq x$.

We first explain the direction ($\Rightarrow$), which is easier. If there is a certified derandomization algorithm A using hard truth tables, one can reduce an instance $C : \{0,1\}^n \to \{0,1\}$ of CAPP to the following instance of LossyCode: The compression circuit C' takes a candidate hard truth table τ, simulates the certified derandomization algorithm $A(C,\tau)$, and outputs a small circuit for τ if $A(C,\tau)$ fails to estimate $\mathbb{E}[C(\mathbf{U}_n)]$; the decompression circuit D' takes the description of a circuit and outputs its truth table. By definition, any solution τ to the LossyCode instance (C', D') is a truth-table such that A correctly estimates $\mathbb{E}[C(\mathbf{U}_n)]$.[19]

The other direction relies on an idea from [Kor21], which was implicit in earlier results in bounded arithmetic [Tha02], [Jeř04], [Jeř07] and cryptography [GGM86]. Korten [Kor21] proved that LossyCode can be efficiently reduced to the problem of finding truth tables of functions that are not computable by small circuits (for an explanation, see [ILW23, Appendix C]). The key observation leading to our results is that the reduction of LossyCode to finding hard truth-table can be thought of as a certified reduction: It either solves the LossyCode instances, or produces a certificate that the truth-table is not hard (in the form of a small circuit).[20]

Assume that $\mathbf{prBPP} = \mathbf{prLOSSY}$. Then, there is a polynomial-time algorithm M for CAPP with a LossyCode oracle. Our certified derandomization algorithm works as follows: Given a circuit C and a supposedly hard truth table τ, it simulates the algorithm $M(x)$, and attempts to answer the LossyCode oracle calls using the truth table τ and the certified reduction from LossyCode to finding hard truth tables. The certified reduction either solves the LossyCode oracle calls, in which case we can keep simulating $M(x)$, or prints a small circuit for the truth table τ. Therefore, our algorithm either prints a small circuit for τ, or successfully simulates $M(x)$; in the latter case, it will solve $\mathsf{CAPP}(C)$ by the correctness of $M(x)$.

[19] Indeed, this implication does not require that that A will use hard truth-tables, and a more general notion of a "range avoidance" property suffices.

[20] This was also observed (although phrased in a different context) in the literature of bounded arithmetic, see, e.g., [Tha02, Lemma 3.7].

ACKNOWLEDGMENT

The authors thank Oded Goldreich for a useful conversation about [GW00], and Lijie Chen, Dean Doron, and Ryan Williams for helpful conversations. Jiatu Li is supported by MIT Akamai Presidential Fellowship and NSF grant CCF-2127597. Edward Pyne is supported by a Jane Street Graduate Research Fellowship. Roei Tell is supported by an NSERC Discovery Grant.

REFERENCES

[AB09] Sanjeev Arora and Boaz Barak. *Computational complexity: A modern approach*. Cambridge University Press, Cambridge, 2009.

[ACR98] Alexander E. Andreev, Andrea E. F. Clementi, and José D. P. Rolim. A new general derandomization method. *Journal of the ACM*, 45(1):179–213, 1998.

[ACRT99] Alexander E. Andreev, Andrea E. F. Clementi, José D. P. Rolim, and Luca Trevisan. Weak random sources, hitting sets, and BPP simulations. *SIAM J. Comput.*, 28(6):2103–2116, 1999.

[AM08] Vikraman Arvind and Partha Mukhopadhyay. Derandomizing the isolation lemma and lower bounds for circuit size. In Ashish Goel, Klaus Jansen, José D. P. Rolim, and Ronitt Rubinfeld, editors, *RANDOM 2008, Boston, MA, USA, August 25-27*, 2008.

[ARZ99] Eric Allender, Klaus Reinhardt, and Shiyu Zhou. Isolation, matching, and counting uniform and nonuniform upper bounds. *J. Comput. Syst. Sci.*, 59(2):164–181, 1999.

[BCK+14] Harry Buhrman, Richard Cleve, Michal Koucký, Bruno Loff, and Florian Speelman. Computing with a full memory: catalytic space. In *Proc. 46 Annual ACM Symposium on Theory of Computing (STOC)*, pages 857–866, 2014.

[BDCGL92] Shai Ben-David, Benny Chor, Oded Goldreich, and Michel Luby. On the theory of average case complexity. *Journal of Computer and System Sciences*, 44(2):193–219, 1992.

[BKLS18] Harry Buhrman, Michal Koucký, Bruno Loff, and Florian Speelman. Catalytic space: Non-determinism and hierarchy. *Theory Comput. Syst.*, 62(1):116–135, 2018.

[BTV09] Chris Bourke, Raghunath Tewari, and N. V. Vinodchandran. Directed planar reachability is in unambiguous log-space. *ACM Trans. Comput. Theory*, 1, 2009.

[CH22] Kuan Cheng and William M. Hoza. Hitting sets give two-sided derandomization of small space. *Theory Comput.*, 18:1–32, 2022.

[CHLR23] Yeyuan Chen, Yizhi Huang, Jiatu Li, and Hanlin Ren. Range avoidance, remote point, and hard partial truth table via satisfying-pairs algorithms. In *STOC*, pages 1058–1066. ACM, 2023.

[CHR24] Lijie Chen, Shuichi Hirahara, and Hanlin Ren. Symmetric exponential time requires near-maximum circuit size. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 1990–1999. ACM, 2024.

[CL24] Yilei Chen and Jiatu Li. Hardness of range avoidance and remote point for restricted circuits via cryptography. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 620–629, 2024.

[CLM+24] James Cook, Jiatu Li, Ian Mertz, , and Edward Pyne. The structure of catalytic space: Capturing randomness and .... *Electron. Colloquium Comput. Complex.*, TR24-106, 2024.

[CLO+23] Lijie Chen, Zhenjian Lu, Igor Carboni Oliveira, Hanlin Ren, and Rahul Santhanam. Polynomial-time pseudodeterministic construction of primes. *arXiv preprint arXiv:2305.15140*, 2023.

[CM20] James Cook and Ian Mertz. Catalytic approaches to the tree evaluation problem. In *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020*, pages 752–760. ACM, 2020.

[CM23] James Cook and Ian Mertz. Tree evaluation is in space o(log n · log log n). *Electron. Colloquium Comput. Complex.*, TR23-174, 2023.

[CRS95] Suresh Chari, Pankaj Rohatgi, and Aravind Srinivasan. Randomness-optimal unique element isolation with applications to perfect matching and related problems. *SIAM Journal on Computing*, 24(5):1036–1050, 1995.

[CRT22] Lijie Chen, Ron D. Rothblum, and Roei Tell. Unstructured hardness to average-case randomness. In *Proc. 63rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 429–437, 2022.

[CRTY20] Lijie Chen, Ron D. Rothblum, Roei Tell, and Eylon Yogev. On exponential-time hypotheses, derandomization, and circuit lower bounds. In *Proc. 61st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 13–23, 2020.

[CT21] Lijie Chen and Roei Tell. Hardness vs randomness, revised: Uniform, non-black-box, and instance-wise. In *Proc. 62nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 125–136, 2021.

[CT23] Lijie Chen and Roei Tell. Guest column: New ways of studying the $\mathbf{BPL} = \mathbf{P}$ conjecture. *ACM SIGACT News*, 54(2):44–69, 2023.

[CTW23] Lijie Chen, Roei Tell, and Ryan Williams. Derandomization vs refutation: A unified framework for characterizing derandomization. In *Proc. 64 Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2023. To appear.

[DGJ+12] Samir Datta, Chetan Gupta, Rahul Jain, Vimal Raj Sharma, and Raghunath Tewari. Randomized and symmetric catalytic computation. In *Computer Science - Theory and Applications - 15th International Computer Science Symposium in Russia, CSR 2020*, 2012.

[DKvMW13] Holger Dell, Valentine Kabanets, Dieter van Melkebeek, and Osamu Watanabe. Is Valiant-Vazirani's isolation probability improvable? *Computational Complexity*, 22(2):345–383, 2013.

[DPT24] Dean Doron, Edward Pyne, and Roei Tell. Opening up the distinguisher: A hardness to randomness approach for BPL = L that uses properties of BPL. In *Proc. 56th Annual ACM Symposium on Theory of Computing (STOC)*, 2024.

[FSUV13] Bill Fefferman, Ronen Shaltiel, Christopher Umans, and Emanuele Viola. On beating the hybrid argument. *Theory of Computing*, 9:809–843, 2013.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.

[GGNS23] Karthik Gajulapalli, Alexander Golovnev, Satyajeet Nagargoje, and Sidhant Saraogi. Range avoidance for constant-depth circuits: Hardness and algorithms. *CoRR*, abs/2303.05044, 2023.

[GJST19] Chetan Gupta, Rahul Jain, Vimal Raj Sharma, and Raghunath Tewari. Unambiguous catalytic computation. In *39th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2019*, volume 150 of *LIPIcs*, pages 16:1–16:13, 2019.

[GLW22] Venkatesan Guruswami, Xin Lyu, and Xiuhan Wang. Range avoidance for low-depth circuits and connections to pseudorandomness. In *APPROX/RANDOM*, volume 245 of *LIPIcs*, pages 20:1–20:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[Gol01] Oded Goldreich. *The Foundations of Cryptography - Volume 1: Basic Techniques*. Cambridge University Press, 2001.

[Gol08] Oded Goldreich. *Computational Complexity: A Conceptual Perspective*. Cambridge University Press, New York, NY, USA, 2008.

[Gol11a] Oded Goldreich. In a world of $\mathbf{P} = \mathbf{BPP}$. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay Randomness and Computation*, pages 191–232, 2011.

[Gol11b] Oded Goldreich. Two comments on targeted canonical derandomizers. *Electronic Colloquium on Computational Complexity: ECCC*, 2011.

[Gol18] Oded Goldreich. On doubly-efficient interactive proof systems. *Foundations and Trends® in Theoretical Computer Science*, 13(3), 2018.

[GRZ23] Uma Girish, Ran Raz, and Wei Zhan. Is untrusted randomness helpful? In *Proc. 14 Conference on Innovations in Theoretical Computer Science (ITCS)*, volume 251 of *LIPIcs*, pages 56:1–56:18, 2023.

[GST19] Chetan Gupta, Vimal Raj Sharma, and Raghunath Tewari. Reachability in o(log n) genus graphs is in unambiguous logspace. In *36th International Symposium on Theoretical Aspects of Computer Science, STACS 2019*, 2019.

[GVW11] Oded Goldreich, Salil Vadhan, and Avi Wigderson. Simplified derandomization of BPP using a hitting set generator. In *Studies in complexity and cryptography*, volume 6650 of *Lecture Notes in Computer Science*, pages 59–67. Springer, Heidelberg, 2011.

[GW96] Anna Gál and Avi Wigderson. Boolean complexity classes vs. their arithmetic analogs. *Random Struct. Algorithms*, 9(1-2):99–111, 1996.

[GW00] Oded Goldreich and Avi Wigderson. On pseudorandomness with respect to deterministic observers. *Electron. Colloquium Comput. Complex.*, TR00-056, 2000.

[GZ11] Oded Goldreich and David Zuckerman. Another proof that bpp subset ph (and more). In Oded Goldreich, editor, *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*, volume 6650 of *Lecture Notes in Computer Science*, pages 40–53. Springer, 2011.

[Hoz19] William M. Hoza. Typically-correct derandomization for small time and space. In *Proc. 34th Annual IEEE Conference on Computational Complexity (CCC)*, pages 9:1–9:39, 2019.

[HU22] William M. Hoza and Chris Umans. Targeted pseudorandom generators, simulation advice generators, and derandomizing logspace. *SIAM J. Comput.*, 51(2):17–281, 2022.

[ILW23] Rahul Ilango, Jiatu Li, and R. Ryan Williams. Indistinguishability obfuscation, range avoidance, and bounded arithmetic. In *Proc. 55th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1076–1089, [2023] ©2023.

[IW97] Russell Impagliazzo and Avi Wigderson. $\mathbf{P} = \mathbf{BPP}$ if $\mathbf{E}$ requires exponential circuits: derandomizing the XOR lemma. In *Proc. 29th Annual ACM Symposium on Theory of Computing (STOC)*, pages 220–229, 1997.

[Jeř04] Emil Jeřábek. Dual weak pigeonhole principle, boolean complexity, and derandomization. *Ann. Pure Appl. Log.*, 129(1-3):1–37, 2004.

[Jeř07] Emil Jeřábek. Approximate counting in bounded arithmetic. *J. Symb. Log.*, 72(3):959–993, 2007.

[Kor21] Oliver Korten. The hardest explicit construction. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 433–444. IEEE, 2021.

[Kor22] Oliver Korten. Derandomization from time-space tradeoffs. In *Proc. 37th Annual IEEE Conference on Computational Complexity (CCC)*, 2022.

[KT16] Vivek Anand T. Kallampally and Raghunath Tewari. Trading determinism for time in space bounded computations. In Piotr Faliszewski, Anca Muscholl, and Rolf Niedermeier, editors, *Proc. 41st International Symposium on Mathematical Foundations of Computer Science*, 2016.

[KV10] Jan Kyncl and Tomás Vyskocil. Logspace reduction of directed reachability for bounded genus graphs to the planar case. *ACM Trans. Comput. Theory*, 1, 2010.

[Lau83] Clemens Lautemann. BPP and the polynomial hierarchy. *Information Processing Letters*, 17(4):215–217, 1983.

[Li24] Zeyong Li. Symmetric exponential time requires near-maximum circuit size: Simplified, truly uniform. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 2000–2007. ACM, 2024.

[Mer23] Ian Mertz. Reusing space: Techniques and open problems. *Bulletin of EATCS*, 141(3), 2023.

[MVV87] Ketan Mulmuley, Umesh V. Vazirani, and Vijay V. Vazirani. Matching is as easy as matrix inversion. *Comb.*, 7(1):105–113, 1987.

[Nis91] Noam Nisan. Pseudorandom bits for constant depth circuits. *Combinatorica*, 11(1):63–70, 1991.

[Nis94] Noam Nisan. $\mathbf{rl} \subseteq \mathbf{sc}$. *Computational Complexity*, 4:1–11, 1994.

[NW94] Noam Nisan and Avi Wigderson. Hardness vs. randomness. *Journal of Computer and System Sciences*, 49(2):149–167, 1994.

[PR23] Rafael Pass and Oren Renard. Characterizing the power of (persistent) randomness in log-space. *Electronic Colloquium on Computational Complexity: ECCC*, 2023.

[PRZ23] Edward Pyne, Ran Raz, and Wei Zhan. Certified hardness vs. randomness for log-space. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023*, 2023.

[Pyn24] Edward Pyne. Derandomizing logspace with a small shared hard drive. In Rahul Santhanam, editor, *39th Computational Complexity Conference, CCC 2024, July 22-25, 2024, Ann Arbor, MI, USA*, volume 300 of *LIPIcs*, pages 4:1–4:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

[RA00] Klaus Reinhardt and Eric Allender. Making nondeterminism unambiguous. *SIAM J. Comput.*, 29(4):1118–1131, 2000.

[RR97] Alexander A. Razborov and Steven Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55(1, part 1):24–35, 1997.

[RS98] Alexander Russell and Ravi Sundaram. Symmetric alternation captures BPP. *Comput. Complex.*, 7(2):152–162, 1998.

[RSW22] Hanlin Ren, Rahul Santhanam, and Zhikun Wang. On the range avoidance problem for circuits. In *Proc. 63rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2022.

[Sip83] Michael Sipser. A complexity theoretic approach to randomness. In *Proc. 15th Annual ACM Symposium on Theory of Computing (STOC)*, pages 330–335, 1983.

[Sip88] Michael Sipser. Expanders, randomness, or time versus space. *Journal of Computer and System Sciences*, 36(3):379–383, 1988.

[STV01] Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the XOR lemma. *Journal of Computer and System Sciences*, 62(2):236–266, 2001.

[SU05] Ronen Shaltiel and Christopher Umans. Simple extractors for all min-entropies and a new pseudorandom generator. *Journal of the ACM*, 52(2):172–216, 2005.

[SW13] Rahul Santhanam and R. Ryan Williams. On medium-uniformity and circuit lower bounds. In *Proc. 28th Annual IEEE Conference on Computational Complexity (CCC)*, pages 15–23. IEEE, 2013.

[Tha02] Neil Thapen. *The weak pigeonhole principle in models of bounded arithmetic*. PhD thesis, University of Oxford, 2002.

[Tod91] Seinosuke Toda. PP is as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 20, 1991.

[Tre01] Luca Trevisan. Extractors and pseudorandom generators. *Journal of the ACM*, 48(4):860–879, 2001.

[TSUZ07] Amnon Ta-Shma, Christopher Umans, and David Zuckerman. Lossless condensers, unbalanced expanders, and extractors. *Combinatorica*, 27(2):213–240, 2007.

[TSZS06] Amnon Ta-Shma, David Zuckerman, and Shmuel Safra. Extractors from Reed-Muller codes. *Journal of Computer and System Sciences*, 72(5):786–812, 2006.

[Uma03] Christopher Umans. Pseudo-random generators for all hardnesses. *Journal of Computer and System Sciences*, 67(2):419–440, 2003.

[Val77] Leslie G. Valiant. Relative complexity of checking and evaluating. *Information Processing Letters*, 5(1):20–23, 1976/77.

[vMP19] Dieter van Melkebeek and Gautam Prakriya. Derandomizing isolation in space-bounded settings. *SIAM J. Comput.*, 48(3):979–1021, 2019.

[vMS23a] Dieter van Melkebeek and Nicollas Sdroievski. Instance-wise hardness versus randomness tradeoffs for arthur-merlin protocols. In *Proc. 38 Annual IEEE Conference on Computational Complexity (CCC)*, 2023.

[vMS23b] Dieter van Melkebeek and Nicollas M. Sdroievski. Leakage resilience, targeted pseudorandom generators, and mild derandomization of arthur-merlin protocols. In Patricia Bouyer and Srikanth Srinivasan, editors, *43rd IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 284 of *LIPIcs*, pages 29:1–29:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

[VV86] Leslie G. Valiant and Vijay V. Vazirani. NP is as easy as detecting unique solutions. *Theor. Comput. Sci.*, 47(3):85–93, 1986.

[Yao82] Andrew C. Yao. Theory and application of trapdoor functions. In *Proc. 23rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 80–91, 1982.