On the Complexity of Avoiding Heavy Elements

Zhenjian Lu
Department of Computer Science
University of Warwick
Coventry, UK
zhenjian.lu@warwick.ac.uk

Igor C. Oliveira
Department of Computer Science
University of Warwick
Coventry, UK
igor.oliveira@warwick.ac.uk

Hanlin Ren
Department of Computer Science
University of Oxford
Oxford, UK
hanlin.ren@cs.ox.ac.uk

Rahul Santhanam
Department of Computer Science
University of Oxford
Oxford, UK
rahul.santhanam@cs.ox.ac.uk

***Abstract*—We introduce and study the following natural total search problem, which we call the *heavy element avoidance* (Heavy Avoid) problem: for a distribution on N bits specified by a Boolean circuit sampling it, and for some parameter $\delta(N) \geq 1/\mathsf{poly}(N)$ fixed in advance, output an N-bit string that has probability less than $\delta(N)$. We show that the complexity of Heavy Avoid is closely tied to frontier open questions in complexity theory about uniform randomized lower bounds and derandomization. Among other results, we show:**

1) **For a wide range of circuit classes $\mathcal{C}$, including ACC^0, TC^0, NC^1, and general Boolean circuits, EXP does not have uniform randomized $\mathcal{C}$-circuits if and only if Heavy Avoid for uniform implicit $\mathcal{C}$-samplers has efficient deterministic algorithms infinitely often. This gives the first *algorithmic characterization* of lower bounds for EXP against uniform randomized low-depth circuits. We show similar algorithmic characterizations for lower bounds in PSPACE, NP and $\mathsf{EXP}^{\mathsf{NP}}$.**
2) **Unconditionally, there are polynomial-time *pseudodeterministic* algorithms that work infinitely often for several variants of Heavy Avoid, such as for uniform samplers of small randomness complexity. In contrast, the existence of a similar algorithm that solves Heavy Avoid for arbitrary polynomial-time samplers would solve a long-standing problem about hierarchies for probabilistic time.**
3) **If there is a time and depth efficient deterministic algorithm for Heavy Avoid, then $\mathsf{BPP} = \mathsf{P}$. Without the depth-efficiency requirement in the assumption, we still obtain a non-trivial form of infinitely-often deterministic simulation of randomized algorithms. These results are shown using *non-black-box* reductions, and we argue that the use of non-black-box reductions is essential here.**

The full version is available on ECCC [1].

***Index Terms*—total search problems, lower bounds, pseudodeterministic algorithms, derandomization**

I. Introduction

Let C be a Boolean circuit sampling a distribution D on N-bit strings. Say that an N-bit string y is δ-heavy in D if y occurs with probability at least δ in D. Assuming that some 2δ-heavy string exists, for $\delta \geq 1/\mathsf{poly}(N)$, how hard is it to find a δ-heavy string given C as input?

We call this natural search problem the *heavy element finding* (Heavy Find) problem. It is not difficult to see that the complexity of Heavy Find is closely related to the complexity of derandomization. There is a simple randomized polynomial-time algorithm for Heavy Find: we use C to draw $O(N/\delta^2)$ independent samples from D and output the string that occurs with the greatest multiplicity in the multiset of samples. A standard application of Chernoff–Hoeffding bounds shows that assuming that a 2δ-heavy string exists, the output of the algorithm will be a string that is δ-heavy in D with high probability.

Moreover, a deterministic polynomial-time algorithm for Heavy Find implies $\mathsf{BPP} = \mathsf{P}$. Indeed, let M be a probabilistic polynomial-time Turing machine with error bounded by $1/4$ and x be an input to M. We can define a circuit sampler C_x which interprets its input as randomness r for the computation of M on x, outputting 1^N if M accepts on x using randomness r and 0^N otherwise. Observe that if M accepts x, the unique solution to Heavy Find on input C_x with parameter $\delta = 1/3$ is 1^N, and if M rejects x, the unique solution to Heavy Find on input C_x with parameter $1/3$ is 0^N. Thus, a deterministic polynomial-time algorithm for Heavy Find allows us to decide if M accepts x, also in deterministic polynomial time.[1]

We now turn our original question on its head: given C as input, how hard is it to find a string that is *not* δ-heavy? We call this the *heavy element avoidance* (Heavy Avoid) problem. Heavy Avoid is the complementary search problem to Heavy Find: a string $y \in \{0,1\}^N$ is a solution to Heavy Avoid if and only if it is not a solution to Heavy Find. The complexity of Heavy Avoid is the primary focus of this paper.

Igor C. Oliveira received support from the Royal Society University Research Fellowship URF\R1\191059; the UKRI Frontier Research Guarantee Grant EP/Y007999/1; and the Centre for Discrete Mathematics and its Applications (DIMAP) at the University of Warwick.

[1]Readers who are familiar with derandomization might already see that the derandomization also holds for the *promise* version of BPP (prBPP). In fact, it is not hard to show that Heavy Find can be solved in deterministic polynomial time if and only if prBPP collapses to prP, the promise version of P.

Superficially, Heavy Avoid seems to be a much simpler problem to solve than Heavy Find. First, when $\delta > 2^{-N}$, Heavy Avoid is a *total* search problem, i.e., the promise that a non-heavy N-bit string exists is automatically satisfied. In this paper, we mainly focus on the regime where $\delta \geq 1/\mathsf{poly}(N)$, hence this is always true if N is large enough. Second, there is a trivial algorithm that *list*-solves Heavy Avoid: Since the number of δ-heavy strings is at most $1/\delta$, at least one of the lexicographically first $\lceil 1/\delta \rceil + 1$ strings of length N is guaranteed to be a solution to Heavy Avoid. Third, there is a very efficient randomized algorithm for Heavy Avoid with overwhelming success probability: output a uniformly random string of length N. Note that by the previous observation that the number of δ-heavy strings is at most $1/\delta$, this randomized algorithm fails on at most $1/\delta \leq \mathsf{poly}(N)$ of its random choices.

Our main contribution is to introduce Heavy Avoid as a natural search problem of interest, and show that despite its seeming simplicity, Heavy Avoid has applications to several frontier questions in complexity theory regarding *uniform randomized lower bounds* and *derandomization*. Indeed, we show that in many settings the existence of algorithms for Heavy Avoid is *equivalent* to a complexity lower bound. The study of Heavy Avoid also illuminates recent *almost-all-inputs-hardness* assumptions in the theory of derandomization [2], and leads to novel *white-box* reductions in settings where black-box reductions are hard to show. Moreover, the connections between Heavy Avoid and complexity lower bounds can be used to derive efficient unconditional *pseudodeterministic* algorithms (in the sense of [3]) for Heavy Avoid in several settings, i.e., efficient randomized algorithms that output some *fixed* solution to Heavy Avoid with high probability.

II. Results

We now discuss our results in greater detail. We present algorithmic characterizations of uniform lower bounds via Heavy Avoid, unconditional pseudodeterministic algorithms for Heavy Avoid, and connections between derandomization with minimum assumptions and Heavy Avoid. Thus, our results can be divided naturally into three sets.

- In Section II-A, we present algorithmic characterizations of lower bounds against uniform probabilistic circuits via Heavy Avoid. That is, deterministic algorithms for Heavy Avoid (in certain settings and with certain parameters) are *equivalent* to such lower bounds. In fact, we obtain very general characterizations that hold for classes such as $\mathsf{EXP}, \mathsf{PSPACE}, \mathsf{EXP}^{\mathsf{NP}}$ and NP, against uniform randomized circuit classes such as ACC^0, TC^0, or $\mathsf{SIZE}[\mathsf{poly}]$. This suggests that the analysis of Heavy Avoid could be useful in attacking frontier open questions such as $\mathsf{EXP} \not\subseteq \mathsf{BP}\text{-}\mathsf{ACC}^0$ and $\mathsf{EXP}^{\mathsf{NP}} \not\subseteq \mathsf{BPP}$.
- In Section II-B, we use our algorithmic characterizations together with other ideas to give *unconditional* pseudodeterministic algorithms for several variants of Heavy Avoid.
- In Section II-C, we give applications of Heavy Avoid to derandomization, including novel *white-box* reductions from promise problems that are hard for prRP or prBPP to Heavy Avoid, as well as connections to "almost-all-inputs-hardness" assumptions that have been explored in recent work on derandomization.

We consider both *uniform* and *non-uniform* versions of Heavy Avoid. In the uniform version, the search algorithm is given N in unary, and needs to find a δ-light[2] element in D_N, where $\mathcal{D} = \{D_N\}_{N\in\mathbb{N}}$ is an ensemble of distributions over N-bit strings that are sampled by some *uniform* sequence of circuits from a circuit class. Since $\mathcal{D}$ is sampled by a uniform sequence of circuits, we do not need to give the circuit sampler explicitly to the search algorithm—the search algorithm can compute the circuit sampler by itself. In this uniform variant of the problem, fix a parameter $\delta \colon \mathbb{N} \to [0,1]$, $(\mathcal{D}, \delta)$-$\texttt{Heavy-Avoid}$ is the problem of finding a $\delta(N)$-light element in D_N, given 1^N as input. This variant is the one we consider in Sections II-A and II-B.

In the non-uniform variant of the problem, the search algorithm is given as input a circuit sampler C from some circuit class $\mathcal{C}$, and needs to output a δ-light element in the distribution sampled by C. This is the version we mostly consider for the results in Section II-C.

There are also two kinds of samplability we consider: *implicit* and *explicit*. In the implicit version, our sampler C is Boolean: given randomness r as input together with an index $i \in [N]$, it outputs the i'th bit of the string sampled on randomness r. In this setting, the circuit size is typically less than N. In the explicit version, the circuit C is given randomness r as input and has N output bits: it outputs the string sampled on randomness r. In this setting, the circuit size is at least N, since there are N output bits. Note that when we show an implication from solving Heavy Avoid to proving lower bounds, the implication is *stronger* when we consider implicit solvers, since the algorithmic problem is *easier* to solve for implicit samplers.[3] An implicit solver $C(r,i)$ can easily be converted to an equivalent explicit solver

$$C_{\mathsf{Explicit}}(r) := C(r,1)C(r,2)\ldots C(r,N).$$

A. Equivalences Between Complexity Separations and Algorithms for Heavy Avoid

It is a long-standing open question to prove lower bounds against non-uniform circuits – we still have not ruled out the possibility that every language computable in exponential time with an NP oracle ($\mathsf{EXP}^{\mathsf{NP}}$) has polynomial-size circuits. What is more embarrassing is our inability to separate $\mathsf{EXP}^{\mathsf{NP}}$ from BPP (see, e.g., [4], [5] for discussions), despite the belief shared by many researchers that $\mathsf{BPP} = \mathsf{P}$ [6], [7].[4] Moreover, the state of affairs is the same regarding lower bounds against

[2]A δ-light element is one that is not δ-heavy.

[3]We measure the complexity of solving the search problem as a function of N, even in the implicit-sampler setting.

[4]Since BPP is strictly contained in $\mathsf{SIZE}[\mathsf{poly}]$ [8], the open problem of separating $\mathsf{EXP}^{\mathsf{NP}}$ from BPP is *strictly more embarrassing* than separating $\mathsf{EXP}^{\mathsf{NP}}$ from $\mathsf{SIZE}[\mathsf{poly}]$! See also [5, Table 1] for a related perspective.

uniform probabilistic circuits from *restricted* circuit classes: for example, it is open whether EXP can be simulated by $\mathsf{DLOGTIME}$-uniform probabilistic ACC^0 circuits or $\mathsf{EXP}^{\mathsf{NP}}$ can be simulated by $\mathsf{DLOGTIME}$-uniform probabilistic TC^0 circuits.[5]

Our first set of results gives *equivalences* between such explicit lower bounds against uniform probabilistic circuits and efficient deterministic algorithms for Heavy Avoid. The equivalences work in a wide variety of settings, for a range of circuit classes including $\mathsf{ACC}^0, \mathsf{TC}^0, \mathsf{NC}^1$ and general Boolean circuits, and for explicit lower bounds in several standard complexity classes of interest such as $\mathsf{EXP}, \mathsf{EXP}^{\mathsf{NP}}, \mathsf{PSPACE}$ and NP. Notably, these results give new *algorithmic characterizations* of uniform lower bound questions by the existence of efficient algorithms for a natural search problem. Thus they could potentially be useful in attacking frontier open questions such as the EXP vs (uniform probabilistic) ACC^0 question, or the $\mathsf{EXP}^{\mathsf{NP}}$ vs BPP question.

We use $\mathsf{BP}\text{-}\mathcal{C}$ to denote the set of languages computed by $\mathsf{DLOGTIME}$-uniform probabilistic $\mathcal{C}$-circuits.

Theorem II.1 (Informal)**.** *Let $\mathcal{C}$ be a nice[6] class of Boolean circuits. The following equivalences hold:*

(i) $\mathsf{EXP} \not\subseteq \mathsf{BP}\text{-}\mathcal{C}$ *if and only if* $(\mathcal{D}, \delta)$-`Heavy-Avoid` *with* $\delta(N) = 1/\mathsf{polylog}(N)$ *can be solved in deterministic polynomial time on infinitely many input lengths for any $\mathcal{D}$ that admits implicit* $\mathsf{DLOGTIME}$*-uniform $\mathcal{C}$-samplers of size* $\mathsf{polylog}(N)$*.*

(ii) $\mathsf{EXP}^{\mathsf{NP}} \not\subseteq \mathsf{BP}\text{-}\mathcal{C}$ *if and only if* $(\mathcal{D}, \delta)$-`Heavy-Avoid` *with* $\delta(N) = 1/\mathsf{polylog}(N)$ *can be solved in deterministic polynomial time with an* NP *oracle on infinitely many input lengths for any $\mathcal{D}$ that admits implicit* $\mathsf{DLOGTIME}$*-uniform $\mathcal{C}$-samplers of size* $\mathsf{polylog}(N)$*.*

(iii) $\mathsf{PSPACE} \not\subseteq \mathsf{BP}\text{-}\mathcal{C}$ *if and only if* $(\mathcal{D}, \delta)$-`Heavy-Avoid` *with* $\delta(N) = 1/\mathsf{polylog}(N)$ *can be solved in deterministic logarithmic space on infinitely many input lengths for any $\mathcal{D}$ that admits implicit* $\mathsf{DLOGTIME}$*-uniform $\mathcal{C}$-samplers of size* $\mathsf{polylog}(N)$*.*

(iv) $\mathsf{NP} \not\subseteq \mathsf{BP}\text{-}\mathcal{C}$ *if and only if* $(\mathcal{D}, \delta)$-`Heavy-Avoid` *with* $\delta(N) = 1/\mathsf{polylog}(N)$ *can be solved by* $\mathsf{DLOGTIME}$*-uniform unbounded fan-in circuits of quasi-polynomial size and constant depth on infinitely many input lengths for any $\mathcal{D}$ that admits implicit* $\mathsf{DLOGTIME}$*-uniform $\mathcal{C}$-samplers of size* $\mathsf{polylog}(N)$*.*

For the PSPACE lower bounds, analogous algorithmic characterizations hold for *almost everywhere* uniform lower bounds and for lower bounds against uniform randomized *sub-exponential* size circuits. Perhaps interestingly, it follows from our arguments that the existence of efficient algorithms for $(\mathcal{D}, \delta)$-`Heavy-Avoid` in the settings considered in Theorem II.1 is *robust* with respect to the threshold parameter $\delta(N)$: the existence of algorithms for any $\delta(N) = o(1)$ yields the existence of algorithms of similar complexity for $\delta(N) = 1/\mathsf{polylog}(N)$.

Theorem II.1 has direct corollaries that characterize frontier open questions in complexity theory.

Corollary II.2 (Informal)**.** *The following results hold:*

(i) $\mathsf{EXP}^{\mathsf{NP}} \not\subseteq \mathsf{BP}\text{-}\mathsf{TC}^0$ *if and only if* Heavy-Avoid *for implicit* $\mathsf{DLOGTIME}$*-uniform* TC^0*-samplers can be solved in deterministic polynomial time with access to and* NP *oracle on infinitely many input lengths.*

(ii) $\mathsf{PSPACE} \not\subseteq \mathsf{BP}\text{-}\mathsf{ACC}^0$ *if and only if* Heavy-Avoid *for implicit* $\mathsf{DLOGTIME}$*-uniform* ACC^0*-samplers can be solved in logarithmic space on infinitely many input lengths.*

Previously, algorithmic characterizations of *non-uniform* lower bounds were known for classes such as NEXP [11], [12] and $\mathsf{EXP}^{\mathsf{NP}}$ [13], [14], and such characterizations for uniform randomized lower bounds against *general* circuits (that is, against BPP) were known for EXP [15] and NEXP [12]. We are not aware of any previous algorithmic characterization of super-polynomial non-uniform or uniform randomized lower bounds for NP.

As a consequence of our results, we also observe a sharp threshold phenomenon in the setting of *quantified derandomization* of search problems (see Section IV below for related work on quantified derandomization and some discussions, and Section 3.3 of the full version for a proof of this corollary).

Corollary II.3. *For every function $e(N) = \omega(1)$, there is a unary* BPP *search problem S such that:*

- *$S(1^N)$ can be solved in randomized* $\mathsf{poly}(N)$ *time by an algorithm that uses N random bits and errs on at most $e(N)$ random bit sequences;*
- *If there exists a randomized polynomial-time algorithm for S with non-zero success probability that errs on at most $O(1)$ random bit sequences for any input, then* $\mathsf{EXP} \neq \mathsf{BPP}$.

B. Unconditional Pseudodeterministic Algorithms for Heavy Avoid

Recall that a randomized algorithm is *pseudodeterministic* if it outputs the same answer with high probability. In the next result, we use the results of the previous subsection together with other ideas to obtain unconditional pseudodeterministic algorithms for different variants of Heavy Avoid.

Theorem II.4. *Let $\mathcal{D} = \{D_N\}_{N\geq 1}$ be a distribution ensemble, where each D_N is supported over $\{0,1\}^N$. The following results hold:*

(i) Let $\mathcal{C}$ be a nice class of Boolean circuits, and suppose $\mathcal{D}$ admits implicit $\mathsf{DLOGTIME}$*-uniform $\mathcal{C}$-samplers of size* $\mathsf{polylog}(N)$*. Then, for every function $\delta(N) = 1/(\log N)^k$, where $k \in \mathbb{N}$, either* $(\mathcal{D}, \delta)$-`Heavy-Avoid` *can be solved in logarithmic space on infinitely many input lengths, or* $(\mathcal{D}, \delta)$-`Heavy-Avoid` *can be solved*

[5] It follows from $\mathsf{EXP}^{\mathsf{NP}} \not\subseteq \mathsf{ACC}^0$ [9], [10], which is a *non-uniform* circuit lower bound, that $\mathsf{EXP}^{\mathsf{NP}}$ cannot be simulated by $\mathsf{DLOGTIME}$-uniform probabilistic ACC^0 circuits. (Note that we do not know how to prove such lower bounds by exploiting the circuit *uniformity* condition.)

[6] In brief, a nice circuit class is one that contains $\mathsf{AC}^0[\oplus]$, is closed under composition, and admits universal circuits for the corresponding class.

pseudodeterministically by DLOGTIME*-uniform* BP-$\mathcal{C}$-*circuits of size* $\mathsf{polylog}(N)$ *on all input lengths.*[7]

In particular, when $\mathcal{C}$ is the class of general Boolean circuits and $\delta(N) = 1/(\log N)^k$, there is a polynomial-time pseudodeterministic algorithm for the $(\mathcal{D}, \delta)$-`Heavy-Avoid` *problem that succeeds on infinitely many inputs.*

(ii) *Suppose $\mathcal{D}$ admits a polynomial-time sampler of randomness complexity $(\log N)^{O(1)}$. Then, for every function $\delta(N) = 1/(\log N)^k$, where $k \in \mathbb{N}$, there is a polynomial-time pseudodeterministic algorithm for the $(\mathcal{D}, \delta)$-*`Heavy-Avoid` *problem that succeeds on infinitely many inputs.*[8]

(iii) *Suppose $\mathcal{D}$ admits a polynomial-time sampler. Then, for every function $\delta(N) = 1/N^k$, where $k \in \mathbb{N}$, and for every constant $\varepsilon > 0$, there is a pseudodeterministic algorithm for the $(\mathcal{D}, \delta)$-*`Heavy-Avoid` *problem that runs in time 2^{N^ε} and succeeds on infinitely many inputs.*

Moreover, in all items the corresponding algorithm behaves pseudodeterministically on every input.

While the pseudodeterministic algorithms above are for natural algorithmic problems about samplers, the design and analysis of the algorithms rely heavily on connections to complexity theory.

The first item of Theorem II.4 provides a pseudodeterministic polynomial-time infinitely-often algorithm for Heavy Avoid for *implicit samplers*. Moreover, this algorithm computes pseudodeterministically on all input lengths. We observe that the existence of a pseudodeterministic algorithm with these properties for the larger class of *explicit samplers* (as in the third item of Theorem II.4) would solve the longstanding open problem of showing a tight hierarchy theorem for probabilistic time [16], [17], [18], [19], [20]. This is captured by the following result (see Section 4.4 of the full version for a more detailed discussion).

Proposition II.5. *Suppose that for every polynomial-time samplable distribution ensemble $\mathcal{D} = \{D_N\}_{N \geq 1}$, the corresponding $(\mathcal{D}, \delta)$-*`Heavy-Avoid` *problem for $\delta(N) = 1/\log N$ admits a pseudodeterministic polynomial-time algorithm that succeeds on infinitely many input lengths and behaves pseudodeterministically on all input lengths. Then, for every constant $k \geq 1$, we have $\mathsf{BPE} \not\subseteq \mathsf{BPTIME}[2^{k \cdot n}]$ (in particular, for every constant $c \geq 1$, $\mathsf{BPP} \not\subseteq \mathsf{BPTIME}[n^c]$).*

C. Connections to Derandomization

Our final set of results explore relations between the complexity of Heavy Avoid and fundamental questions in derandomization. We consider the *non-uniform* variant of Heavy Avoid, where a Boolean circuit sampler is given as input to the algorithm solving Heavy Avoid. For $\delta\colon \mathbb{N} \to [0, 1]$, `Implicit-`$\delta$`-Heavy-Avoid` is the problem where we are given as input a circuit C implicitly sampling a distribution on N bits (as explained at the beginning of Section II), and would like to output a δ-light element in the distribution.

Our first result shows that the existence of efficient deterministic algorithms for Heavy Avoid that in addition can be implemented by sub-polynomial depth uniform circuits leads to a complete derandomization of prBPP. Note that in this result, to obtain the desired conclusion it is sufficient for this algorithm to solve the problem for implicit samplers.

Theorem II.6. *Let $\delta(N) = o(1)$ be any function. Suppose there is a constant $\epsilon > 0$ and a deterministic algorithm $\mathcal{A}$ that solves the* `Implicit-`δ`-Heavy-Avoid` *problem on implicit samplers of size N^ϵ. Moreover, assume that $\mathcal{A}$ can be implemented as a logspace-uniform circuit of size $\mathsf{poly}(N)$ and depth $N^{o(1)}$. Then $\mathsf{prBPP} = \mathsf{prP}$.*

If we could eliminate the circuit depth constraint from the statement of Theorem II.6, it would be possible to establish an equivalence between the derandomization of prBPP and algorithms for Heavy Avoid (in both the implicit and explicit settings). While obtaining this strong characterization remains elusive, in the next result we get a non-trivial derandomization consequence from the existence of an efficient algorithm for Heavy Avoid without assuming a circuit depth bound.

Let GAP-SAT denote the promise problem where YES instances are Boolean circuits with at least half of assignments being satisfying, and NO instances are unsatisfiable Boolean circuits. It is well known that GAP-SAT is complete for the promise version of RP.

Theorem II.7 (Informal)**.** *Let $\delta(N) = o(1)$ be any function. Suppose there is an algorithm for* `Implicit-`δ`-Heavy-Avoid` *on maps $G\colon \{0,1\}^{\mathsf{poly}(n)} \to \{0,1\}^N$ (where $N = 2^{n^\epsilon}$) implicitly computable by an input circuit of size $\mathsf{poly}(n)$, where the Heavy Avoid algorithm runs in $\mathsf{poly}(N)$ time and is infinitely-often correct. Then there is an algorithm for* GAP-SAT *that runs in subexponential time and is infinitely-often* correct.*[9]

Theorem II.6 and Theorem II.7 are both established using *non-black-box reductions* that make use of recent hardness-randomness tradeoffs. In more detail, as explained in Section III below, Theorem II.6 crucially relies on the instance-wise hardness-randomness tradeoff for low-depth circuits of Chen and Tell [2], while Theorem II.7 combines the framework of [2] and the "leakage-resilient" hardness-randomness framework of Liu and Pass [21]. In contrast to the non-black-

[7]In other words, the corresponding DLOGTIME-uniform BP-$\mathcal{C}$-circuit $E(i)$ is given $i \in [N] = \{0,1\}^{\log N}$ and outputs the i-th bit of the solution with high probability.

[8]Note that the result in this item subsumes the "in particular" result from the previous item.

[9]In Theorem II.7, we only obtain GAP-SAT algorithms satisfying a technical condition called infinitely-often* correctness, which is a nonstandard variant of infinitely-often correctness. The crucial difference is that, for a sequence of inputs $\{x_n\}_{n \in \mathbb{N}}$, given 1^n, the algorithm is allowed to inspect every input $x_1, x_2, \ldots, x_{\mathsf{poly}(n)}$, and needs to provide a solution for x_n. In other words, the algorithm is correct *infinitely-often** if it outputs the correct answer on infinitely many input lengths n while having access to all input strings from the sequence that have length polynomial in n. We refer the reader to Definition 5.7 and Theorem 5.8 of the full version of this paper for more details.

box nature of the proofs given for these two results, we show that it will be quite difficult to obtain them using *black-box* reductions. In particular, we show that improving Theorem II.7 to a polynomial-time Levin reduction [22] would derandomize prBPP.[10] Stated more precisely, if there is an efficient black-box Levin reduction from the search version of GAP-SAT to Heavy Avoid (even with respect to non-uniform explicit samplers), then prBPP = prP holds *unconditionally*. We refer to Section 5.3 of the full version for more details.

Finally, we establish a deeper connection between the implicit non-uniform variant of Heavy Avoid considered in this section and the recent paradigm of *instance-wise* hardness-randomness tradeoffs alluded to above [2], [23], [21], [24]. Roughly speaking, in this paradigm, we convert a hard function $f : \{0,1\}^n \to \{0,1\}^{\mathsf{poly}(n)}$ with multiple output bits into pseudorandomness, where the obtained derandomization is *instance-wise*: for every $x \in \{0,1\}^n$, if f is hard to compute on x, then the derandomization of the corresponding computation over input x succeeds. Naturally, the derandomization assumptions used in these results need *almost-all-inputs hardness*, meaning that f is hard on *all but finitely many* inputs (instead of input lengths).[11] In Section 5.4 of the full version of this paper, we prove that the existence of efficient deterministic algorithms for Heavy Avoid in the implicit non-uniform setting is *equivalent* to the existence of functions f with multiple output bits that are easy to compute deterministically but are hard against fixed polynomial-size randomized algorithms. This result sheds light into the relevance of the techniques that we employ to prove Theorem II.6 and Theorem II.7, and suggest that developing further connections between Heavy Avoid and these modern hardness-randomness tradeoffs paradigms could be a fruitful research direction.

III. Techniques

We now discuss the proofs of Theorem II.1, Theorem II.4, Theorem II.6, and Theorem II.7. We make use of a variety of techniques to establish these results:

- The proof of Theorem II.1 Item (*iii*) relies on extremely efficient *instance checkers* for a special PSPACE-complete problem investigated in [26]. This allows us to establish equivalences for very weak circuits classes $\mathcal{C}$ at the frontier of existing separations. Extending the equivalence result to NP, EXP, and $\mathsf{EXP}^{\mathsf{NP}}$ in the context of weak circuit classes poses some additional challenges that we address through different ideas and techniques.
- In the proof of Theorem II.4 Item (*i*), we use an instance checker to design a polynomial-time pseudodeterministic algorithm for the $(\mathcal{D},\delta)$-`Heavy-Avoid` problem. To our knowledge, this is the first application of instance checkers in the *design* of an algorithm for a natural problem. On the other hand, the algorithms in Items (*ii*) and (*iii*) of Theorem II.4 explore a connection to *randomized time-bounded Kolmogorov complexity* [27], [28] and its *source coding theorem* [29], [30].
- The proof of Theorem II.6 relies on a novel application of the Chen–Tell *non-black-box hitting set generator* construction from [2], [31]. In contrast to previous applications, here the *reconstruction procedure* of the generator itself, as well as the assumed algorithm for Heavy Avoid, plays a key role in the specification of a "hard" function.
- Finally, the proof of Theorem II.7 builds on the proof of Theorem II.6. It combines for the first time the Chen–Tell derandomization framework [2] with the *leakage resilience* derandomization framework of [21], using a win-win analysis. We show that either the Heavy Avoid algorithm is leakage resilient, which allows us to use the framework of [21], or it can be implemented by a low-depth circuit, which allows us to use the framework of [2]. This enables us to derive a non-trivial derandomization consequence without the circuit depth constraint present in the hypothesis of Theorem II.6.

Next, we describe some of our proofs and techniques in more detail.

a) Sketch of the Proof of Theorem II.1: We first explain the proof of Item (*iii*), i.e., the equivalence between the complexity separation $\mathsf{PSPACE} \not\subseteq \mathsf{BP}\text{-}\mathcal{C}$ and the existence of (infinitely often) logarithmic-space algorithms for Heavy-Avoid over implicit DLOGTIME-uniform $\mathcal{C}$-samplers.

First, we show how to obtain the separation using algorithms for the implicit heavy avoid problem. Using standard arguments, it suffices to show that for every choice of $k \geq 1$, there is $L \in \mathsf{DSPACE}[n^2]$ such that L cannot be computed by $\mathsf{DTIME}[k \cdot \log n]$-uniform randomized $\mathcal{C}$-circuits of size n^k.

Let $N = 2^n$. We consider a map $G_N \colon \{0,1\}^{n^{O(k)}} \to \{0,1\}^N$ that views its input string x as a pair (M, r), where M is a short encoding (say, $\log n$ bits) of a clocked machine running in time $10k \cdot \log n$, and r is a random string. Let D_M be the $\mathcal{C}$-circuit of size at most n^{2k} whose direct connection language is encoded by the machine M. For $i \in [N]$, we define the i-th output bit of $G_N(x)$ as $D_M(r, i)$. Due to its running time, the computation of M can be uniformly converted into an AC^0 circuit of size at most n^{10k}. Using that $\mathcal{C}$ is a nice circuit class, G_N can be implicitly computed by a DLOGTIME-uniform probabilistic $\mathcal{C}$-circuit C_N of size at most $n^{O(k)}$.

Let $B(1^N)$ be an algorithm of space complexity $O(\log N)$ that solves $\mathcal{C}$-`Implicit`-δ-`Heavy-Avoid` on infinitely many values of N for the sequence G_N, and let L_B be the language defined by B. Note that L_B is in $\mathsf{DSPACE}[O(n)] \subseteq \mathsf{DSPACE}[n^2]$. To argue that L_B cannot be computed by $\mathsf{DTIME}[k \cdot \log n]$-uniform randomized $\mathcal{C}$-circuits of size n^k, it is enough to show that for every language L computed by such circuits, each string in the sequence $\{y_N^L\}_N$ of truth-tables obtained from L is δ-heavy in $G_N(\mathcal{U}_{m(N)})$. Since B solves $\mathcal{C}$-`Implicit`-δ-`Heavy-Avoid` for the sequence $\{G_N\}$, it follows that $L_B \neq L$.

[10]Recall that in a Levin reduction between search problems we have a pair (f, g) of functions, where f maps to an instance of the other problem while g converts a given solution into a solution to the original problem.

[11]Compared with classical hardness-randomness frameworks such as [6], [7], [25], the advantage of the new paradigm is that lower bounds against *uniform algorithms* (instead of non-uniform circuits) suffice for *worst-case* derandomization.

The proof that the sequence $\{y_N^L\}_N$ of truth-tables obtained from L is δ-heavy in $G_N(\mathcal{U}_{m(N)})$ relies on the definition of G_N. In more detail, under the assumption that L admits $\mathsf{DTIME}[k \cdot \log n]$-uniform randomized $\mathcal{C}$-circuits of size n^k, it is not hard to show that its truth-table is produced with probability comparable to $2^{-|M|}$. However, this probability is sufficiently large under the assumption that the encoding length $|M|$ is small in the definition of G_N.

The proof of the other direction in Theorem II.1 is more interesting. We establish the contrapositive. Suppose that for some $G_N \colon \{0,1\}^{\mathsf{poly}(n)} \to \{0,1\}^N$ implicitly computed by $\mathsf{DLOGTIME}$-uniform $\mathcal{C}$-circuits of size $\mathsf{poly}(n)$, every algorithm $A(1^N)$ running in space $O(\log N)$ fails to solve $\mathcal{C}$-`Implicit`-δ-`Heavy-Avoid` on every large enough input length N. We employ this assumption to show that $\mathsf{PSPACE} \subseteq \mathsf{BP}\text{-}\mathcal{C}$. For this, we recall the notion of *instance checkers*. Let $L \subseteq \{0,1\}^*$ be a language, and let $\{C_n^{(-)}(x,z)\}_{n\in\mathbb{N}}$ be a family of probabilistic oracle circuits. We say that C is an *instance checker* for L if for every input $x \in \{0,1\}^*$:

1) $\mathbf{Pr}_z[C_{|x|}^L(x,z) = L(x)] = 1$, and
2) for every oracle $\mathcal{O}$, $\mathbf{Pr}_z[C_{|x|}^{\mathcal{O}}(x,z) \notin \{L(x), \bot\}] \leq 1/2^n$.

We will rely on an appropriate PSPACE-complete language $L^\star$ that admits highly efficient instance checkers computable in any nice circuit class. This is a consequence of a result from [26], as explained in Appendix B of the full version.

We then consider a candidate algorithm $A(1^N)$ that computes as follows. On input 1^N, define tt_N to be the truth table of $L^\star$ on n-bit inputs; we simply output tt_N. It is possible to show that A computes in space $O(\log N)$ after an appropriate scaling of parameters, which we we omit here for simplicity. Therefore, A fails to solve $\mathcal{C}$-`Implicit`-δ-`Heavy-Avoid` on every large enough input length N. This means that for every large enough N, the probability of tt_N under the distribution $G_N(\mathcal{U}_{\mathsf{poly}(n)})$ from above is at least $\delta = 1/(\log N)^{O(1)} = 1/\mathsf{poly}(n)$.

To explain how we compute $L^\star$ on an input $x \in \{0,1\}^n$, assume for simplicity that the oracle instance checker circuit (call it IC) only queries its oracle on input length n. We sample $v := n^{O(1)}$ strings $z_1, \ldots, z_v \in \{0,1\}^{\mathsf{poly}(n)}$ uniformly and independently at random, and for each string z_i, we define an oracle $\mathcal{O}_i$ whose truth table is the string $G_N(z_i) \in \{0,1\}^N$. We run IC in parallel and obtain $b_i := \mathsf{IC}_n^{\mathcal{O}_i}(x)$ for each $i \in [v]$. We output 1 if at least one bit among $b_1, \ldots, b_v$ is 1, and 0 otherwise.

Next, we argue that A computes $L^\star$ with high probability. Let tt_N denote the truth table of $L^\star$ on input length n. By our choice of v, with high probability the string tt_N appears among the strings $G_N(z_1), \ldots, G_N(z_v)$, meaning that one of the oracles $\mathcal{O}_i$ computes $L^\star$ on inputs of length n. Consequently, in this case, if $L^\star(x) = 1$ then at least one bit $b_i = 1$, and the procedure outputs 1. On the other hand, if $L^\star(x) = 0$, then by a union bound over the internal randomness of IC, with high probability every bit $b_i \in \{0, \bot\}$. In this case, the procedure outputs 0. This establishes the correctness of A. Using the efficiency of the instance checker and that $\mathcal{C}$ is a nice circuit class, it is also possible to upper the circuit complexity of A and to analyze the uniformity of the corresponding circuits. This implies that $L^\star \in \mathsf{BP}\text{-}\mathcal{C}$. Since $L^\star$ is PSPACE-complete under $\mathsf{DLOGTIME}$-uniform projection reductions, we get that $\mathsf{PSPACE} \subseteq \mathsf{BP}\text{-}\mathcal{C}$, as desired.

We now briefly comment on the additional ideas needed for the proofs of the other items in Theorem II.1. The proof of Item (*ii*) requires a different approach, since instance checkers for $\mathsf{EXP}^{\mathsf{NP}}$-complete languages are not known. We provide two different proofs in this case. In more detail, the result for $\mathsf{EXP}^{\mathsf{NP}}$ can be obtained using a win-win argument and a reduction to Item (*iii*), or through the use of *selectors* for $\mathsf{EXP}^{\mathsf{NP}}$-complete languages [32]. These two approaches provide different extensions of the result, which we discuss in detail in Section 3.3 of the full version. On the other hand, the proof of Item (*iv*) relies on a randomized depth-efficient version of the search-to-decision reduction for SAT based on the Valiant-Vazirani Isolation Lemma [33], as well as the equivalence between the polynomial hierarchy and $\mathsf{DLOGTIME}$-uniform constant-depth circuits of exponential size [34].

b) Sketch of the Proof of Theorem II.4: First, we discuss the proof of Item (*i*) in the case where $\mathcal{C}$ = "general Boolean circuits". Consider a map $G_N \colon \{0,1\}^{\mathsf{poly}(n)} \to \{0,1\}^N$ that is implicitly computable in time $\mathsf{poly}(n)$. We consider two cases, based on whether $\mathsf{EXP} = \mathsf{BPP}$.

If $\mathsf{EXP} \nsubseteq \mathsf{BPP}$, then by Theorem II.1 Item (*i*) (with circuit class $\mathcal{C}$ = "general Boolean circuits") the heavy avoid problem over G_N can be solved in deterministic polynomial time (i.e., in time $\mathsf{poly}(N)$) on infinitely many input lengths. Note that the correctness of the procedure obtained from Theorem II.1 Item (*i*) relies on the existence of instance checkers for EXP-complete languages.

In the remaining case, assume that $\mathsf{EXP} \subseteq \mathsf{BPP}$. Let $B(j)$ be the following deterministic machine with input $j \in \{0,1\}^n$: It first goes over all choices of $x \in \{0,1\}^{\mathsf{poly}(n)}$ and computes $G_N(x)$, then calculates the probability of each string in $\{0,1\}^N$ produced in this way, and finally outputs the j-th bit of the lexicographic first string y such that $\mathbf{Pr}[G_N(\mathcal{U}_{\mathsf{poly}(n)}) = y] \leq \delta$. Note that B runs in time exponential in n, its input length. Therefore, it defines a language $L_B \in \mathsf{EXP}$. By the assumption, $L_B \in \mathsf{BPP}$. Consequently, we can compute $y \in \{0,1\}^N$ from 1^N in pseudodeterministic time $\mathsf{poly}(N)$. Note that in this case the algorithm succeeds on every input length.[12]

More generally, to obtain the result claimed in Item (*i*) for a nice circuit class $\mathcal{C}$, we use a similar approach but consider whether $\mathsf{PSPACE} \nsubseteq \mathsf{BP}\text{-}\mathcal{C}$ instead (see Section 4.1 of the full version).

[12]We remark that using a more involved construction that employs the instance checker as a subroutine, one can simultaneously consider both cases to describe a single explicit algorithm that succeeds infinitely often. We omit the details. For a similar situation where a non-constructive win-win argument can be turned into an explicit algorithm, see [35, Section 3.4].

In contrast, the proof of Theorem II.4 Item (*iii*) relies on ideas from randomized time-bounded Kolmogorov complexity. More specifically, we consider the randomized Levin complexity of a string $y \in \{0,1\}^n$ [27], denoted $\mathsf{rKt}(y)$. Roughly speaking, $\mathsf{rKt}(y)$ measures the minimum description length of a time-bounded machine that outputs y with high probability. Our key idea is that, by the coding theorem of [29], if a string y can be sampled in polynomial time with probability at least δ, then $\mathsf{rKt}(y) = O(\log 1/\delta)$. Consequently, to avoid the set of heavy strings produced by a polynomial-time samplable distribution $\mathcal{D}$, it is sufficient to construct a string z such that $\mathsf{rKt}(z) \geq C \cdot \log n$, where C is large enough. In order to implement this idea, we employ a related sub-exponential time pseudodeterministic construction of strings of large rKt complexity from [36].

Finally, the proof of Theorem II.4 Item (*ii*) is obtained via a translation to Item (*iii*), using a simple "prefix" reduction described in Section 4.3 of the full version.

c) Sketch of the Proof of Theorem II.6: Using existing results [37], in order to derandomize prBPP it is sufficient to describe an algorithm that, given an input circuit $D\colon \{0,1\}^M \to \{0,1\}$ of size $O(M)$ with the promise that $\mathbf{Pr}_y[D(y) = 1] \geq 1/2$, runs in deterministic time $\mathsf{poly}(M)$ and outputs a positive input of D. To achieve this, we will rely on a novel application of the Chen–Tell generator [2] (with the improved parameters from [31]). In more detail, given a function $f\colon \{0,1\}^n \to \{0,1\}^{T(n)}$ computed by logspace uniform circuits of size $T(n)$ and depth $d(n)$, and a parameter $M(n)$ such that $c \cdot \log T \leq M \leq T^{1/c}$ (for a constant c), [2], [31] provides algorithms HSG_f and Recon_f depending on f such that:

- The algorithm $\mathsf{HSG}_f(x)$ runs in deterministic T^c time and outputs a set of M-bit strings.
- Given $x \in \{0,1\}^n$ and $i \in [T]$ as inputs, and oracle access to a candidate distinguisher $D\colon \{0,1\}^M \to \{0,1\}$, $\mathsf{Recon}_f^D(x,i)$ runs in randomized $(dnM)^c$ time. If D is dense and avoids $\mathsf{HSG}_f(x)$, then with probability $\geq 1 - 2^{-M}$, $\mathsf{Recon}_f^D(x,i)$ outputs the i-th bit of $f(x)$.

We consider an appropriate function $f'\colon \{0,1\}^{\widetilde{O}(M)} \to \{0,1\}^N$, where $N = M^{C_1}$ for a large enough constant C_1. We view the input of f' as the description of an arbitrary circuit $D\colon \{0,1\}^M \to \{0,1\}$ of size $O(M)$. In this construction, the parameter $T = M^{C_2}$ for a large enough constant $C_2 > C_1$, while $d = M^{o(1)} = N^{o(1)}$. Moreover, f' will be computed by a logspace-uniform family of circuits. We then show that $\mathsf{HSG}_{f'}(D)$ hits D if D is a dense circuit. Note that the generator runs in time $\mathsf{poly}(T) = \mathsf{poly}(M)$ by our choice of parameters.

The function f' makes use of the algorithm $\mathcal{A}$ that solves the `Implicit-`δ`-Heavy-Avoid` problem on instances $G\colon \{0,1\}^{N^\epsilon} \to \{0,1\}^N$ that are implicitly computable in N^ϵ size. In more detail, we let $f'(D) = \mathcal{A}(C_D)$, where C_D is an implicit (non-uniform) sampler of size N^ϵ for a map $G_D\colon \{0,1\}^{N^\epsilon} \to \{0,1\}^N$ described next.

First, we make a simplifying assumption: The sampler G_D has access to the code of a machine $M_{f'}$ that serves as a logspace-uniform description of a circuit family that computes f'. (Observe that this is self-referential, since we have defined $f'(D) = \mathcal{A}(C_D)$ above, while we will also use f' to define C_D. We will handle this issue later.)

The sampler G_D stores D as advice. This is possible because D is of size M, and if C_1 is large enough then $M \ll N^\epsilon$. The implicit sampler $C_D(r,i)$ for G_D then uses its random input string r of length N^ϵ and $i \in [\log N]$ to compute $\mathsf{Recon}_{f'}^D(D,i,r)$, where we have made explicit the random string r used by $\mathsf{Recon}_{f'}^D$. Since $d = M^{o(1)}$ and C_1 is large enough, we get that $\mathsf{Recon}_{f'}^D(D,i,r)$ can be computed in time $(d \cdot M^{1+o(1)} \cdot M)^c \leq M^{c+o(1)} \leq N^\epsilon$. This completes the definition of $f'(D)$ and of $\mathsf{HSG}_{f'}(D)$. We note that to establish the size, depth, and logspace-uniformity of the sequence of circuits computing f' we can rely on the fact that f' only needs to produce the *code* of C_D.[13]

Next, we argue that $\mathsf{HSG}_{f'}(D)$ hits any dense circuit D. Assume this is not the case. Then, since D avoids the generator, $\mathsf{Recon}_f^D(D,i)$ outputs the i-th bit of $f'(D)$ with probability at least $1 - 2^{-M}$. Consequently, by a union bound over $i \in [N]$, it follows that the string $\mathcal{A}(C_D) = f'(D) \in \{0,1\}^N$ is output by $\mathsf{Recon}_f^D(D,\cdot)$ with probability $1 - o(1)$. In other words, the string $f'(D)$ is sampled with high probability by the sampler G_D encoded by C_D. On the other hand, since $f'(D) = \mathcal{A}(C_D)$ and $\mathcal{A}$ solves the heavy avoid problem for G_D, we get that the string $f'(D)$ has probability $o(1)$ under G_D. This contradiction implies that $\mathsf{HSG}_{f'}(D)$ indeed hits D.

It remains to explain how to fix the self-referential nature of the definition of G_D via the implicit sampler C_D, which depends on f' (and which in turn depends on C_D). In more detail, the construction is self-referential due to the use of the routine $\mathsf{Recon}_{f'}^D$, which depends on f'. To patch the argument, we combine the following key points:

- There is a deterministic algorithm that, given the Turing machine $M_{f'}$ that prints the circuit for f' in logspace, outputs the description of $\mathsf{Recon}_{f'}$ in $\mathsf{poly}(|\langle M_{f'} \rangle|)$ time.
- We can combine constantly many samplers into a single sampler that produces the convex combination of the corresponding distributions. A string with weight $o(1)$ under the new distribution must have weight $o(1)$ under each original sampler.

Therefore, we can change the description of G_D so that it interprets a small prefix of its random input string as the description of a Turing machine M_f that prints a circuit of the expected size using logarithmic uniformity, then use the first bullet above to produce the procedure Recon_f corresponding to f. Notice that with this change the sampler G_D no longer depends on f'. Moreover, since f' is encoded by some finite machine $M_{f'}$, using the second bullet the argument described above to reach a contradiction and establish the correctness of

[13]We make a brief comment about the novelty of this argument. In order to define the "hard" function f', here we make use of the *reconstruction procedure* of the generator. This is different from an application of this generator in [31], where the code of the *hitting set procedure* plays a key role in the definition of the hard function.

the hitting set generator still holds: When D avoids $\mathsf{HSG}_{f'}(D)$ the modified sampler G_D outputs the string $f'(D) = \mathcal{A}(C_D)$ with constant probability, while as a solution to the heavy avoid problem for G_D this string has probability $o(1)$. This completes the sketch of the argument.[14]

d) Sketch of the Proof of Theorem II.7: Since this is a more sophisticated construction, we only provide a brief sketch of the idea. As alluded to above, the argument combines the two instance-wise hardness-randomness tradeoffs introduced by Chen and Tell [2] and by Liu and Pass [21], respectively.

We employ a win-win analysis based on whether the assumed algorithm for $\texttt{Implicit-}\delta\texttt{-Heavy-Avoid}$ (call it Avoid) is "*leakage resilient*" hard. In more detail, let $f\colon \{0,1\}^n \to \{0,1\}^T$ be a function, A be a randomized algorithm, and $x \in \{0,1\}^n$ be an input of f. We say that $f(x)$ is *ℓ-leakage resilient hard* against A if for every "leakage string" $\mathsf{leak} \in \{0,1\}^\ell$, there is some $i \in [T]$ such that $\mathbf{Pr}[A(x, \mathsf{leak}, i) = f(x)_i] \le 2/3$, where the probability is taken over the internal randomness of A. Liu and Pass [21] showed that leakage resilient hardness can be used for derandomization.

We can now explain the main idea behind the win-win analysis. If Avoid is leakage resilient hard, we use the hardness-randomness tradeoffs in [21]. If this is not the case, we show that Avoid can actually be implemented by a low-depth circuit. We can then use the hardness-randomness tradeoffs in [2], which requires the hard function to be computed by a low-depth circuit family.

Implementing this plan turns out to require a delicate construction and the notion of infinitely-often* correctness appearing in the statement of Theorem II.7. We refer to Section 5.2 of the full version for more details.

IV. Related Work

Our work relates to several recent lines of research in algorithms and complexity theory.

a) Algorithmic Characterizations of Uniform Lower Bounds: The algorithmic method of Williams [38], which derives $\mathcal{C}$-circuit lower bounds for NEXP or $\mathsf{EXP}^{\mathsf{NP}}$ from non-trivial algorithms for Satisfiability or $\mathsf{GAP\text{-}SAT}$ for $\mathcal{C}$ circuits, has been successful in showing several new circuit lower bounds [9], [39], [40], [41], [42], [10]. However, in settings where *non-uniform* lower bounds are unknown, it is unclear how to use such methods to at least give uniform randomized lower bounds. A step towards such methods is to give *algorithmic characterizations* of uniform lower bounds, which show that certain algorithmic results are both necessary and sufficient for lower bounds. An algorithmic approach to the NEXP vs BPP problem is given in [4], but this does not seem to give a characterization. An algorithmic characterization of $\mathsf{EXP} \neq \mathsf{BPP}$ follows from [15], but this characterization does not extend to frontier lower bound questions such as $\mathsf{EXP}^{\mathsf{NP}} \not\subseteq \mathsf{BP\text{-}TC}^0$ and $\mathsf{EXP} \not\subseteq \mathsf{BP\text{-}ACC}^0$. Theorem II.1 gives generic characterizations that apply to these and other frontier questions – this showcases the benefits of considering the Heavy Avoid problem rather than the previously studied Gap-SAT or Circuit Acceptance Probability Problem (CAPP). Theorem II.1 also gives characterizations of uniform lower bounds for NP, where no algorithmic characterizations at all where known before. An algorithmic approach to uniform lower bounds for NP is given in [43], but the method there does not seem to extend to randomized lower bounds, and moreover does not give unconditional algorithmic characterizations.

b) Range Avoidance: There have been several recent works on the Range Avoidance problem [44], [13], [14], [45], [46], [47], [48], [49], [50], which is a total search problem where we are given a circuit C from n bits to m bits, $m > n$, and need to output an m-bit string that is not in the range of C.[15] The Range Avoidance problem is tightly connected to proving non-uniform lower bounds (see, e.g., [52], [13], [14], [54], [55], [50]). Heavy Avoid can be thought of as an easier version of Range Avoidance, where we are asked to output some m-bit string that does not have *many* pre-images, rather than a string that has no pre-images at all. Indeed, for δ that is inverse polynomial, Heavy Avoid with parameter δ is a BPP search problem, which is unknown for Range Avoidance and would have new circuit lower bound consequences if it were the case. We refer to Appendix A of the full version for reductions from both Heavy Avoid and Heavy Find to Range Avoidance. One of our motivations for defining and studying Heavy Avoid is that algorithms for this easier problem might give a way to exploit uniformity in the lower bound.

c) Quantified Derandomization: In the quantified derandomization setting [56], [57], we are interested in the possibility of derandomizing algorithms with overwhelming success probability, e.g., derandomization of randomized algorithms for a decision problem that err on at most $S(n)$ random bit sequences, for some $S(n)$ that is sub-exponential or even just slightly super-polynomial. Note that this derandomization setting is quite specialized, since bounded-error randomized algorithms are in general allowed to err on an inverse polynomial fraction of all random bit sequences. A naive way to derandomize such algorithms for decision problems is to run the randomized algorithm on the lexicographically first $2S(n)+1$ random bit sequences, and take the majority answer. The question of when it is possible to do better has been studied extensively [56], [58], [59], [60], [57]. Our Corollary II.3 identifies an interesting phenomenon for quantified derandomization of BPP search problems, which has not been studied before. In this setting, the previously mentioned "naive" method to derandomize doesn't work, as verifying a candidate solution itself involves the use of randomness. We show that for any $\delta = o(1)$ there is a natural search problem, i.e., Heavy Avoid with parameter δ, such that the problem can be solved by a randomized algorithm which

[14]We note that this argument is non-black-box. The code of a machine $M_{f'}$ that describes a uniform circuit family for f' is needed to instantiate the Chen-Tell generator. In the aforementioned construction, this means that black-box access to the algorithm $\mathcal{A}$ is not enough.

[15]The problem is closely related to the dual weak pigeonhole principle, which has been widely investigated in logic and bounded arithmetic (see [51], [52], [53], [50] and references therein).

errs on at most $1/\delta = \omega(1)$ random bit sequences, but any efficient randomized algorithm which errs on at most $O(1)$ random bit sequences would imply $\mathsf{EXP} \neq \mathsf{BPP}$! Thus even slightly beating the performance of a known algorithm in a quantified derandomization sense would imply a breakthrough lower bound.

d) Pseudodeterministic Algorithms: The notion of *pseudodeterminism* was introduced in the influential work of [3]. A pseudodeterministic algorithm for a search problem is a randomized algorithm that outputs some fixed solution to the search problem with high probability. There has been a lot of recent work on pseudodeterminism in various settings, as well as connections to complexity theory (see, e.g., [61], [35], [62], [63], [64], [65], [66], [36], [67], [68]). In general, one might hope to show that every BPP search problem can also be solved pseudodeterministically, but this is not known, though there are important special cases such as finding an N-bit prime for which efficient pseudodeterministic algorithms are known infinitely often [31]. Theorem II.4 gives unconditional pseudodeterministic algorithms for Heavy Avoid that can be implemented in low depth when the circuit sampler for Heavy Avoid is itself low depth. Moreover, improving our strongest pseudodeterministic algorithms would have immediate consequences for the long-standing open problem of showing a hierarchy for randomized time (Proposition II.5). Note that unlike the problem of finding an N-bit prime, Heavy Avoid is a fairly powerful BPP search problem, as evidenced by Theorems II.6 and II.7.

e) Minimal Assumptions for Derandomization: The standard "hardness vs. randomness" approach towards $\mathsf{prBPP} = \mathsf{prP}$ requires lower bounds against non-uniform circuits [6], [7], [69]; another line of work employs uniform lower bounds such as $\mathsf{EXP} \neq \mathsf{BPP}$ to obtain heuristic (i.e., average-case) derandomization of BPP [15], [70], [71], [72]. A long-standing open problem is whether circuit lower bounds such as $\mathsf{EXP} \not\subseteq \mathsf{SIZE}[\mathsf{poly}]$ are indeed *necessary* for derandomization (see, e.g., [73] and [71]). Recently, Chen and Tell [2] proposed an *instance-wise* hardness-randomness tradeoff and showed that *almost-all-inputs* hardness suffices for worst-case derandomization. The Chen-Tell result has already sparked a new line of research on instance-wise hardness-randomness tradeoffs and minimal assumptions for derandomization [23], [21], [72], [24], [74] (see also the survey [75]). As demonstrated in Theorems II.6 and II.7, these instance-wise hardness-randomness tradeoffs can be used as *proof techniques* to connect Heavy Avoid to the problem of derandomizing prRP or prBPP. Moreover, as shown in Section 5.4 of the full version of this paper, the *almost-all-inputs* hardness assumptions used in [2] are closely connected to Heavy Avoid. Given the rich interplay between Heavy Avoid and derandomization, we believe that investigating the complexity of Heavy Avoid is likely to shed further light on the minimal assumptions required for derandomization.

Acknowledgment

We thank the anonymous reviewers for their comments and suggestions regarding the presentation.

References

[1] Z. Lu, I. C. Oliveira, H. Ren, and R. Santhanam, "On the complexity of avoiding heavy elements," *Electron. Colloquium Comput. Complex.*, vol. TR24-115, 2024.

[2] L. Chen and R. Tell, "Hardness vs randomness, revised: Uniform, non-black-box, and instance-wise," in *Symposium on Foundations of Computer Science* (FOCS), 2021, pp. 125–136.

[3] E. Gat and S. Goldwasser, "Probabilistic search algorithms with unique answers and their cryptographic applications," *Electron. Colloquium Comput. Complex.*, vol. TR11-136, 2011.

[4] R. Williams, "Towards NEXP versus BPP?" in *International Computer Science Symposium in Russia* (CSR), 2013, pp. 174–182.

[5] ——, "Some estimated likelihoods for computational complexity," in *Computing and Software Science - State of the Art and Perspectives*. Springer, 2019, pp. 9–26.

[6] N. Nisan and A. Wigderson, "Hardness vs randomness," *J. Comput. Syst. Sci.*, vol. 49, no. 2, pp. 149–167, 1994.

[7] R. Impagliazzo and A. Wigderson, "P = BPP if E requires exponential circuits: Derandomizing the XOR lemma," in *Symposium on Theory of Computing* (STOC). ACM, 1997, pp. 220–229.

[8] L. M. Adleman, "Two theorems on random polynomial time," in *Symposium on Foundations of Computer Science* (FOCS), 1978, pp. 75–83.

[9] R. Williams, "Nonuniform ACC circuit lower bounds," *J. ACM*, vol. 61, no. 1, pp. 2:1–2:32, 2014.

[10] L. Chen, X. Lyu, and R. Williams, "Almost-everywhere circuit lower bounds from non-trivial derandomization," in *Symposium on Foundations of Computer Science* (FOCS), 2020, pp. 1–12.

[11] R. Impagliazzo, V. Kabanets, and A. Wigderson, "In search of an easy witness: exponential time vs. probabilistic polynomial time," *J. Comput. Syst. Sci.*, vol. 65, no. 4, pp. 672–694, 2002.

[12] R. Williams, "Natural proofs versus derandomization," *SIAM J. Comput.*, vol. 45, no. 2, pp. 497–529, 2016.

[13] O. Korten, "The hardest explicit construction," in *Symposium on Foundations of Computer Science* (FOCS), 2021, pp. 433–444.

[14] H. Ren, R. Santhanam, and Z. Wang, "On the range avoidance problem for circuits," in *Symposium on Foundations of Computer Science* (FOCS), 2022, pp. 640–650.

[15] R. Impagliazzo and A. Wigderson, "Randomness vs time: Derandomization under a uniform assumption," *J. Comput. Syst. Sci.*, vol. 63, no. 4, pp. 672–688, 2001.

[16] M. Karpinski and R. Verbeek, "On the Monte Carlo space constructible functions and seperation results for probabilistic complexity classes," *Inf. Comput.*, vol. 75, no. 2, pp. 178–189, 1987.

[17] B. Barak, "A probabilistic-time hierarchy theorem for "slightly non-uniform" algorithms," in *International Workshop on Randomization and Approximation Techniques* (RANDOM), 2002, pp. 194–208.

[18] L. Fortnow and R. Santhanam, "Hierarchy theorems for probabilistic polynomial time," in *Symposium on Foundations of Computer Science* (FOCS), 2004, pp. 316–324.

[19] L. Fortnow, R. Santhanam, and L. Trevisan, "Hierarchies for semantic classes," in *Symposium on Theory of Computing* (STOC), 2005, pp. 348–355.

[20] D. van Melkebeek and K. Pervyshev, "A generic time hierarchy for semantic models with one bit of advice," in *Conference on Computational Complexity* (CCC), 2006, pp. 129–144.

[21] Y. Liu and R. Pass, "Leakage-resilient hardness vs randomness," in *Computational Complexity Conference* (CCC), 2023, pp. 32:1–32:20.

[22] L. A. Levin, "Universal sequential search problems," *Problemy peredachi informatsii*, vol. 9, no. 3, pp. 115–116, 1973.

[23] Y. Liu and R. Pass, "Characterizing derandomization through hardness of Levin-Kolmogorov complexity," in *Computational Complexity Conference* (CCC), 2022, pp. 35:1–35:17.

[24] L. Chen, R. Tell, and R. Williams, "Derandomization vs refutation: A unified framework for characterizing derandomization," in *Symposium on Foundations of Computer Science* (FOCS), 2023, pp. 1008–1047.

[25] M. Sudan, L. Trevisan, and S. P. Vadhan, "Pseudorandom generators without the XOR lemma," *J. Comput. Syst. Sci.*, vol. 62, no. 2, pp. 236–266, 2001.

[26] L. Chen, "New lower bounds and derandomization for ACC, and a derandomization-centric view on the algorithmic method," in *Innovations in Theoretical Computer Science Conference* (ITCS), 2023, pp. 34:1–34:15.

[27] I. C. Oliveira, "Randomness and intractability in Kolmogorov complexity," in *International Colloquium on Automata, Languages, and Programming* (ICALP), 2019, pp. 32:1–32:14.

[28] Z. Lu and I. C. Oliveira, "Theory and applications of probabilistic Kolmogorov complexity," *Bull. EATCS*, vol. 137, 2022.

[29] ——, "An efficient coding theorem via probabilistic representations and its applications," in *International Colloquium on Automata, Languages, and Programming* (ICALP), 2021, pp. 94:1–94:20.

[30] Z. Lu, I. C. Oliveira, and M. Zimand, "Optimal coding theorems in time-bounded Kolmogorov complexity," in *International Colloquium on Automata, Languages, and Programming* (ICALP), 2022, pp. 92:1–92:14.

[31] L. Chen, Z. Lu, I. C. Oliveira, H. Ren, and R. Santhanam, "Polynomial-time pseudodeterministic construction of primes," in *Symposium on Foundations of Computer Science* (FOCS), 2023, pp. 1261–1270.

[32] S. Hirahara, "Identifying an honest $\mathrm{EXP}^{\mathrm{NP}}$ oracle among many," in *Computational Complexity Conference* (CCC), 2015, pp. 244–263.

[33] L. G. Valiant and V. V. Vazirani, "NP is as easy as detecting unique solutions," *Theor. Comput. Sci.*, vol. 47, no. 3, pp. 85–93, 1986.

[34] D. A. M. Barrington, N. Immerman, and H. Straubing, "On uniformity within NC^1," *J. Comput. Syst. Sci.*, vol. 41, no. 3, pp. 274–306, 1990.

[35] I. C. Oliveira and R. Santhanam, "Pseudodeterministic constructions in subexponential time," in *Symposium on Theory of Computing* (STOC), 2017, pp. 665–677.

[36] Z. Lu, I. C. Oliveira, and R. Santhanam, "Pseudodeterministic algorithms and the structure of probabilistic time," in *ACM Symposium on Theory of Computing* (STOC), 2021, pp. 303–316.

[37] H. Buhrman and L. Fortnow, "One-sided versus two-sided error in probabilistic computation," in *Symposium on Theoretical Aspects of Computer Science* (STACS), 1999, pp. 100–109.

[38] R. Williams, "Improving exhaustive search implies superpolynomial lower bounds," *SIAM J. Comput.*, vol. 42, no. 3, pp. 1218–1244, 2013.

[39] ——, "New algorithms and lower bounds for circuits with linear threshold gates," *Theory Comput.*, vol. 14, no. 1, pp. 1–25, 2018.

[40] ——, "Limits on representing Boolean functions by linear combinations of simple functions: Thresholds, ReLUs, and low-degree polynomials," in *Computational Complexity Conference* (CCC), 2018, pp. 6:1–6:24.

[41] C. D. Murray and R. R. Williams, "Circuit lower bounds for nondeterministic quasi-polytime from a new easy witness lemma," *SIAM J. Comput.*, vol. 49, no. 5, 2020.

[42] L. Chen and R. Williams, "Stronger connections between circuit analysis and circuit lower bounds, via PCPs of proximity," in *Computational Complexity Conference* (CCC), 2019, pp. 19:1–19:43.

[43] R. Santhanam, "An algorithmic approach to uniform lower bounds," in *Computational Complexity Conference* (CCC), 2023, pp. 35:1–35:26.

[44] R. Kleinberg, O. Korten, D. Mitropolsky, and C. H. Papadimitriou, "Total functions in the polynomial hierarchy," in *Innovations in Theoretical Computer Science Conference* (ITCS), 2021, pp. 44:1–44:18.

[45] V. Guruswami, X. Lyu, and X. Wang, "Range avoidance for low-depth circuits and connections to pseudorandomness," in *International Workshop on Randomization and Approximation Techniques* (RANDOM), 2022, pp. 20:1–20:21.

[46] Y. Chen, Y. Huang, J. Li, and H. Ren, "Range avoidance, remote point, and hard partial truth table via satisfying-pairs algorithms," in *Symposium on Theory of Computing* (STOC), 2023, pp. 1058–1066.

[47] R. Ilango, J. Li, and R. R. Williams, "Indistinguishability obfuscation, range avoidance, and bounded arithmetic," in *Symposium on Theory of Computing* (STOC), 2023, pp. 1076–1089.

[48] K. Gajulapalli, A. Golovnev, S. Nagargoje, and S. Saraogi, "Range avoidance for constant depth circuits: Hardness and algorithms," in *International Workshop on Randomization and Approximation Techniques* (RANDOM), 2023, pp. 65:1–65:18.

[49] Y. Chen and J. Li, "Hardness of range avoidance and remote point for restricted circuits via cryptography," in *Symposium on Theory of Computing* (STOC), 2024, pp. 620–629.

[50] J. Krajícek, "Proof complexity generators," Monograph (In Progress), 2024.

[51] J. Krajíček, *Bounded Arithmetic, Propositional Logic, and Complexity Theory*, ser. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1995.

[52] E. Jeřábek, "Dual weak pigeonhole principle, Boolean complexity, and derandomization," *Ann. Pure Appl. Log.*, vol. 129, no. 1-3, pp. 1–37, 2004.

[53] E. Jeřábek, "Weak pigeonhole principle and randomized computation," Ph.D. dissertation, Charles University in Prague, 2005.

[54] L. Chen, S. Hirahara, and H. Ren, "Symmetric exponential time requires near-maximum circuit size," in *Symposium on Theory of Computing* (STOC), 2024, pp. 1990–1999.

[55] Z. Li, "Symmetric exponential time requires near-maximum circuit size: Simplified, truly uniform," in *Symposium on Theory of Computing* (STOC), 2024, pp. 2000–2007.

[56] O. Goldreich and A. Wigderson, "On derandomizing algorithms that err extremely rarely," in *Symposium on Theory of Computing* (STOC), 2014, pp. 109–118.

[57] R. Tell, "Quantified derandomization: How to find water in the ocean," *Found. Trends Theor. Comput. Sci.*, vol. 15, no. 1, pp. 1–125, 2022.

[58] ——, "Improved bounds for quantified derandomization of constant-depth circuits and polynomials," in *Computational Complexity Conference* (CCC), 2017, pp. 13:1–13:48.

[59] ——, "Quantified derandomization of linear threshold circuits," in *Symposium on Theory of Computing* (STOC), 2018, pp. 855–865.

[60] L. Chen and R. Tell, "Bootstrapping results for threshold circuits "just beyond" known lower bounds," in *Symposium on Theory of Computing* (STOC), 2019, pp. 34–41.

[61] O. Goldreich, S. Goldwasser, and D. Ron, "On the possibilities and limitations of pseudodeterministic algorithms," in *Innovations in Theoretical Computer Science* (ITCS), 2013, pp. 127–138.

[62] P. Dixon, A. Pavan, and N. V. Vinodchandran, "On pseudodeterministic approximation algorithms," in *Symposium on Mathematical Foundations of Computer Science* (MFCS), 2018, pp. 61:1–61:11.

[63] I. C. Oliveira and R. Santhanam, "Pseudo-derandomizing learning and approximation," in *International Conference on Randomization and Computation* (RANDOM), 2018, pp. 55:1–55:19.

[64] O. Grossman and Y. P. Liu, "Reproducibility and pseudo-determinism in Log-Space," in *Symposium on Discrete Algorithms* (SODA), 2019, pp. 606–620.

[65] P. Dixon, A. Pavan, and N. V. Vinodchandran, "Complete problems for multi-pseudodeterministic computations," in *Innovations in Theoretical Computer Science* (ITCS), 2021.

[66] S. Goldwasser, R. Impagliazzo, T. Pitassi, and R. Santhanam, "On the pseudo-deterministic query complexity of NP search problems," in *Computational Complexity Conference* (CCC), 2021, pp. 36:1–36:22.

[67] P. Dixon, A. Pavan, J. V. Woude, and N. V. Vinodchandran, "Pseudodeterminism: promises and lowerbounds," in *Symposium on Theory of Computing* (STOC), 2022, pp. 1552–1565.

[68] A. Chattopadhyay, Y. Dahiya, and M. Mahajan, "Query complexity of search problems," in *Symposium on Mathematical Foundations of Computer Science* (MFCS), vol. 272, 2023, pp. 34:1–34:15.

[69] C. Umans, "Pseudo-random generators for all hardnesses," *J. Comput. Syst. Sci.*, vol. 67, no. 2, pp. 419–440, 2003.

[70] L. Trevisan and S. P. Vadhan, "Pseudorandomness and average-case complexity via uniform reductions," *Comput. Complex.*, vol. 16, no. 4, pp. 331–364, 2007.

[71] L. Chen, R. D. Rothblum, R. Tell, and E. Yogev, "On exponential-time hypotheses, derandomization, and circuit lower bounds," *J. ACM*, vol. 70, no. 4, pp. 25:1–25:62, 2023.

[72] L. Chen, R. D. Rothblum, and R. Tell, "Unstructured hardness to average-case randomness," in *Symposium on Foundations of Computer Science* (FOCS), 2022, pp. 429–437.

[73] O. Goldreich, "In a world of P=BPP," in *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*, ser. Lecture Notes in Computer Science. Springer, 2011, vol. 6650, pp. 191–232.

[74] D. van Melkebeek and N. M. Sdroievski, "Instance-wise hardness versus randomness tradeoffs for Arthur-Merlin protocols," in *Computational Complexity Conference* (CCC), 2023, pp. 17:1–17:36.

[75] L. Chen and R. Tell, "Guest column: New ways of studying the $\mathsf{BPP} = \mathsf{P}$ conjecture," *SIGACT News*, vol. 54, no. 2, pp. 44–69, 2023.