Nondeterministic Circuit Lower Bounds from Mildly Derandomizing Arthur-Merlin Games

Barış Aydınlıoğlu
Department of Computer Sciences
University of Wisconsin
Madison, WI 53706, USA
baris@cs.wisc.edu

Dieter van Melkebeek
Department of Computer Sciences
University of Wisconsin
Madison, WI 53706, USA
dieter@cs.wisc.edu

***Abstract*—Hardness against nondeterministic circuits is known to suffice for derandomizing Arthur-Merlin games. We show a result in the other direction – that hardness against nondeterministic circuits is *necessary* for derandomizing Arthur-Merlin games. In fact, we obtain an equivalence for a mild notion of derandomization: Arthur-Merlin games can be simulated in $\Sigma_2\mathrm{SUBEXP}$ (the subexponential version of $\Sigma_2\mathrm{P}$) with subpolynomial advice on infinitely many input lengths if and only if $\Sigma_2\mathrm{E}$ (the linear-exponential version of $\Sigma_2\mathrm{P}$) requires nondeterministic circuits of superpolynomial size on infinitely many input lengths.**

Our equivalence result represents a full analogue of a similar result by Impagliazzo et al. in the deterministic setting: Randomized polynomial-time decision procedures can be simulated in $\mathrm{NSUBEXP}$ (the subexponential version of NP) with subpolynomial advice on infinitely many input lengths if and only if NE (the linear-exponential version of NP) requires deterministic circuits of superpolynomial size on infinitely many input lengths.

A key ingredient in our proofs is improved Karp-Lipton style collapse results for nondeterministic circuits. The following are two instantiations that may be of independent interest: Assuming that Arthur-Merlin games can be derandomized in $\Sigma_2\mathrm{P}$, we show that (i) $\mathrm{PSPACE} \subseteq \mathrm{NP/poly}$ implies $\mathrm{PSPACE} \subseteq \Sigma_2\mathrm{P}$, and (ii) $\mathrm{coNP} \subseteq \mathrm{NP/poly}$ implies $\mathrm{PH} \subseteq \mathrm{P}^{\Sigma_2\mathrm{P}}$.

***Keywords*-Derandomization, Circuit Lower Bounds, Arthur-Merlin Games**

I. Introduction

The power of randomness constitutes a central topic in complexity theory. In the context of randomized decision procedures the question is whether the class BPP, or its promise version prBPP, can be simulated deterministically without much overhead – in subexponential or maybe even polynomial time. Similarly, in the context of randomized verification procedures one seeks for efficient nondeterministic simulations of Arthur-Merlin games: the class AM, or its promise version prAM.

A major development in the area are hardness versus randomness tradeoffs [18], [3], [11], which state that either nonuniformity speeds up computations significantly or else nontrivial derandomization is possible. More precisely, these results show how to use a language in some complexity class $\mathcal{C}$ that is *assumed* to require "large" circuits, to construct a pseudorandom generator (PRG) with "small" seed length that is computable in $\mathcal{C}$. The PRG transforms its seed into a longer string, say of length s, such that the average behavior of any circuit C of size s is almost the same when the input to C is provided from the uniform distribution or from the output distribution of the PRG on a uniform seed. We say that the PRG fools the circuit C. If $\mathcal{C}$ requires large circuits of a certain type τ, then the resulting PRG fools circuits of type τ [16], and can be used to derandomize randomized processes that can be modeled by small circuits of type τ. See the table below for some examples from the above papers and [19], where $\mathrm{E} \doteq \mathrm{DTIME}(2^{O(n)})$ and $\mathrm{NE} \doteq \mathrm{NTIME}(2^{O(n)})$ and $\tau = \mathrm{d/n}$ denotes deterministic/nondeterministic circuits.

		randomized	
τ	class $\mathcal{C}$	class	derandomization
d	E	prBPP	$\mathrm{DTIME}(t)$
n	$\mathrm{NE} \cap \mathrm{coNE}$	prAM	$\mathrm{NTIME}(t)$

Once we have such a PRG, the derandomization is obtained by cycling over all seeds and simulating the randomized algorithm on the output of the PRG for each seed. Stronger circuit lower bounds for $\mathcal{C}$ imply smaller seed lengths for the generator, yielding more efficient derandomizations. At the "low end", superpolynomial circuit lower bounds yield subpolynomial seed length and derandomizations that run in subexponential time t. At the "high end", linear-exponential circuit lower bounds yield logarithmic seed length and derandomizations that run in polynomial time t.

As the circuit lower bounds seem plausible, even at the high end, the hardness versus randomness tradeoffs have fueled the conjecture that prBPP can be fully derandomized to P, and prAM to NP. However, even the low-end hardness conditions remain open to date. This raises the question whether there are means of derandomizing that do not need any hardness assumptions. In other words, whether derandomization *implies* hardness.

In recent years we have seen a number of results for $\mathrm{(pr)BPP}$ showing that derandomization implies hardness of some sort, although typically not strong enough so as to imply back the derandomization (e.g., [10], [12], [15]). One exceptional example is a low-end equivalence for prBPP in [10]: prBPP can be simulated in nondeterministic subexponential time (NSUBEXP) with subpolynomial advice for infinitely many input lengths if and only if nondeterministic linear-exponential time (NE) requires deterministic circuits of superpolynomial size for infinitely many input lengths. Regardless of the equivalence, the result shows that derandomizing prBPP requires advancing the frontiers in deterministic circuit lower bounds – it is unknown whether superpolynomial size circuits are required for NE, or even for E^{NP} for that matter.

In contrast to the class prBPP, derandomization-to-hardness connections for prAM have not gathered much attention thus far. The only result in this direction is a "hybrid" connection that shows an implication from derandomizing prAM to *deterministic* circuit lower bounds [2], whereas the hardness-to-derandomization implication for prAM involves *nondeterministic* circuits. In particular, what kind of nondeterministic circuit lower bounds, if any, are implied by derandomizing prAM is an open question.

Our results. In this paper we take up this question and obtain an equivalence of hardness and derandomization for prAM. Specifically, we show that a mild derandomization of prAM with small advice implies nondeterministic circuit lower bounds, which in turn imply back the same derandomization. By a "mild" derandomization of prAM we mean a simulation of prAM in $\Sigma_2\mathrm{SUBEXP} \doteq \cap_{\epsilon>0}\Sigma_2\mathrm{TIME}(2^{n^\epsilon})$. The standard notion of derandomization for prAM refers to simulations in $\mathrm{NTIME}(t)$, which is trivially included in $\Sigma_2\mathrm{TIME}(t)$. Hence the term "mild."

Theorem 1 (Equivalence for Arthur-Merlin games). *The following are equivalent:*

- *For every* $\epsilon > 0$*,* prAM *can be simulated in* $\Sigma_2\mathrm{TIME}(2^{n^\epsilon})/n^\epsilon$ *for infinitely many input lengths.*
- $\Sigma_2\mathrm{E}$ *contains a language that requires nondeterministic circuits of superpolynomial size for infinitely many input lengths.*

Theorem 1 represents an analogue of the equivalence established by Impagliazzo et al. [10] for prBPP. Their equivalence involves simulations of prBPP in $\mathrm{NTIME}(t)$ with small advice, whereas the standard notion of derandomization for prBPP refers to simulations in $\mathrm{DTIME}(t)$. Recall that prAM can be simulated in $\Pi_2\mathrm{P}$, but simulations in $\Sigma_2\mathrm{P}$ are open, even though plausible hardness assumptions imply simulations in NP. Gutfreund et al. [9] suggest an approach to unconditionally prove that prAM can be simulated in $\Sigma_2\mathrm{SUBEXP}$. Theorem 1 implies that such a result would yield new circuit lower bounds as it is open whether $\Sigma_2\mathrm{E}$ requires superpolynomial-size nondeterministic circuits. In fact, as far as we know, it is even open whether $\mathrm{E}^{\Sigma_2\mathrm{P}}$ requires such circuits. Thus, the situation for Arthur-Merlin games is similar to the one for decision procedures, except "one level up" in the exponential-time hierarchy.

A key step in the proof of Theorem 1 shows that mild derandomizations of prAM imply improved Karp-Lipton style collapse results for nondeterministic circuits. The following are two instantiations that may be of independent interest.

Theorem 2 (High-end collapse results). *Suppose* prAM *can be simulated in* $\Sigma_2\mathrm{P}$*. Then*

(i) $\mathrm{PSPACE} \subseteq \mathrm{NP}/\mathrm{poly} \implies \mathrm{PSPACE} \subseteq \Sigma_2\mathrm{P}$*, and*
(ii) $\mathrm{coNP} \subseteq \mathrm{NP}/\mathrm{poly} \implies \mathrm{PH} \subseteq \mathrm{P}^{\Sigma_2\mathrm{P}}$.

Karp and Lipton [14] showed that if $\mathrm{NP} \subseteq \mathrm{P}/\mathrm{poly}$ then $\mathrm{PH} \subseteq \Sigma_2\mathrm{P}$. Yap's adaptation [20] of the Karp-Lipton result gives that if $\mathrm{coNP} \subseteq \mathrm{NP}/\mathrm{poly}$ then $\mathrm{PH} \subseteq \Sigma_3\mathrm{P}$. Modulo the derandomization assumption, the second item of Theorem 2 improves Yap's result by "half a level". Another variant of the Karp-Lipton argument (attributed to Meyer) states that if $\mathrm{PSPACE} \subseteq \mathrm{P}/\mathrm{poly}$ then $\mathrm{PSPACE} \subseteq \Sigma_2\mathrm{P}$. Relativizing the latter result yields the strongest collapse consequence of $\mathrm{PSPACE} \subseteq \mathrm{NP}/\mathrm{poly}$ known unconditionally, namely $\mathrm{PSPACE} \subseteq \Sigma_3\mathrm{P}$. The first item of Theorem 2 improves this collapse by one level, modulo the derandomization assumption. We refer to Section II-D for related work on more refined collapses.

We use a low-end version of the first collapse result in Theorem 2 to establish the forward direction of Theorem 1. The proof relies on interactive proofs for PSPACE, and as such does not relativize. If we use the second collapse result instead of the first one, we obtain a relativizing proof of a weaker result, namely that simulations of prAM in $\Sigma_2\mathrm{SUBEXP}$ imply that $\mathrm{E}^{\Sigma_2\mathrm{P}}$ contains a language that requires nondeterministic circuits of superpolynomial size infinitely often.

Organization. In Section II we sketch the ideas behind our results and discuss the relationship with earlier work. In Section III we set up our notation for the formal development. In Section IV we present the collapse results and in Section V the equivalence result.

II. Outline of the Arguments and Related Work

We now outline the proofs of Theorem 1 and Theorem 2. We focus on the most novel part of the proof of Theorem 1, which is the direction from derandomization to hardness, as well as the weaker but relativizing variant mentioned in the introduction. Both can be cast as instantiations of a general framework that also captures the related result of [10]. The framework relies on Kannan's theorem [13] that some level of the polynomial-time hierarchy cannot be decided by circuits of fixed polynomial size, and critically relies on collapse results in order to bring down the required level of the polynomial-time hierarchy.

We first explain the collapse results we need and then present the framework. For ease of exposition, in this overview we use the assumption that prAM can be simulated in $\Sigma_2\mathrm{P}$, yielding the high-end collapse results of Theorem

2, although for the proof of Theorem 1 it suffices to have the weaker assumption that prAM can be simulated in $\cap_{\epsilon>0}\mathrm{i.o.}-\Sigma_2\mathrm{TIME}(2^{n^\epsilon})/n^\epsilon$.

A. First high-end collapse result

Our first collapse result is an adaptation to the nondeterministic setting of the classical result that if PSPACE has polynomial-size deterministic circuits then $\mathrm{PSPACE} \subseteq \mathrm{MA}$ [17]. Let us first recall the proof of that classical result.

Assuming that PSPACE has polynomial-size deterministic circuits, we want to compute some PSPACE-complete language L in MA. The proof hinges on the existence of an interactive proof system for L in which the honest prover's responses are computable in PSPACE. By assumption, there is a polynomial-size deterministic circuit D_{prover} that encodes the honest prover's strategy; i.e., given a transcript of a message history, the circuit computes the next bit the honest prover sends. Now the MA-protocol for L is as follows: Merlin sends a polynomial-size circuit D' to Arthur, who then carries out the interactive protocol for L by himself, evaluating the circuit D' to determine the prover's responses. If the input is in L, Merlin can send the circuit D_{prover}, which makes Arthur accept with high probability. If the input is not in L, the soundness property of the interactive proof system guarantees that no deterministic circuit can make Arthur accept with significant probability. This proves that $L \in \mathrm{MA}$.

Now let us turn to our setting and try to achieve a similar collapse under the assumption that PSPACE has *nondeterministic* circuits of polynomial size. Since Arthur may need Merlin's help in evaluating nondeterministic circuits, we allow for one more round of interaction between Arthur and Merlin, and aim for a collapse to MAM=AM rather than MA. By assumption, there exists a polynomial-size nondeterministic circuit C implementing the honest prover's strategy; i.e., given a transcript of the message history and a bit b, C accepts if the honest prover's next message bit is b, and rejects otherwise. Now consider the following attempt at a protocol for the PSPACE-complete language L: Merlin sends a polynomial-size nondeterministic circuit C', purported to encode the strategy of the honest prover. Upon receiving the circuit C', Arthur reveals his coin flips ρ to Merlin. Merlin then provides the certificates for the circuit C' that allow Arthur to construct and verify every bit of the transcript of the interactive proof corresponding to the coin flips ρ. Finally, Arthur accepts if the resulting transcript is accepting.

This protocol is complete but not necessarily sound. Indeed, Merlin can send a nondeterministic circuit C' that has accepting and rejecting computation paths on every input, which allows Merlin to adapt his strategy to the coin flips ρ in whatever way he wants, by revealing accepting computation paths only if he wishes to. We somehow need to force Merlin to commit to a fixed strategy in advance.

In order to do so, we use our derandomization assumption and aim for a collapse to $\Sigma_2\mathrm{P}$ instead of AM. First, in an existential phase we "guess" a nondeterministic circuit C' supposed to implement the honest prover's strategy for L. We provide C' as additional input to the AM-protocol, and require Merlin to convince Arthur using the specific circuit C' provided. Moreover, in *parallel* to the AM-protocol, we make sure C' commits Merlin to a fixed strategy. More precisely, in a universal phase we check that on every partial transcript for at least one of the choices of the bit b, C' rejects on all computation paths. This fixes the soundness problem while maintaining completeness.

Note that the above procedure can be implemented within $\Sigma_2\mathrm{P}$: We use two alternations outside the AM-protocol, where the second alternation for checking the circuit is executed in parallel to the protocol. Since by our derandomization assumption the AM-protocol can be simulated in $\Sigma_2\mathrm{P}$, a collapse of PSPACE to $\Sigma_2\mathrm{P}$ follows.

B. Second high-end collapse result

To outline the proof of our second collapse result, let us first recall the proof of the classical result by Karp and Lipton for the case of NP and deterministic circuits. Assuming that satisfiability (SAT) has polynomial-size circuits, we consider any $\Pi_2\mathrm{P}$-predicate of the form $(\forall u)(\exists v)\varphi(u,v)$, where φ is a Boolean formula, and translate it into an equivalent $\Sigma_2\mathrm{P}$-predicate.

One way to construct the $\Sigma_2\mathrm{P}$-predicate goes as follows. Use an existential quantifier to "guess" a deterministic circuit D, verify that D correctly decides SAT, and then use D to transform the final existential phase of the original $\Pi_2\mathrm{P}$-predicate into a deterministic one, effectively eliminating one quantifier alternation. Hence, we obtain an equivalent predicate that reads as

$$(\exists D)\left[\texttt{correct}(D) \wedge (\forall u) D(\text{``}(\exists v)\varphi(u,v)\text{''}) = 1\right]. \quad (1)$$

Exploiting the self-reducibility of SAT, $\texttt{correct}(D)$ can be expressed as a coNP-predicate. This way (1) becomes a $\Sigma_2\mathrm{P}$-predicate.

Let us now turn to our setting and try to achieve the same collapse under the assumption that $\overline{\mathrm{SAT}}$ has *nondeterministic* circuits of polynomial size. Mimicking the above proof, we use an existential quantifier to guess a nondeterministic circuit C, check its correctness for $\overline{\mathrm{SAT}}$, and then feed the final existential phase of the $\Pi_2\mathrm{P}$-predicate into C. The latter transforms the final existential phase into an equivalent universal phase, which can be merged with the initial universal phase of the original $\Pi_2\mathrm{P}$-predicate. This complementation gives us a new equivalent predicate of the form

$$(\exists C)\left[\texttt{correct}(C) \wedge (\forall u) C(\text{``}(\exists v)\varphi(u,v)\text{''}) = 0\right]. \quad (2)$$

In fact, it suffices that the predicate $\texttt{correct}(C)$ checks the *completeness* of C for $\overline{\mathrm{SAT}}$ (that C accepts every unsatisfiable formula) without explicitly checking the *soundness* of C for $\overline{\mathrm{SAT}}$ (that C only accepts unsatisfiable formulas). However, while soundness can be tested in coNP, completeness seems to require $\Pi_2\mathrm{P}$. This turns (2) into a $\Sigma_3\mathrm{P}$-predicate rather than a $\Sigma_2\mathrm{P}$-predicate. Note that obtaining an equivalent $\Sigma_3\mathrm{P}$-predicate is trivial since we started from a $\Pi_2\mathrm{P}$-predicate. If we start from a Π_3^{P}-predicate instead, an analogous transformation yields an equivalent $\Sigma_3\mathrm{P}$-predicate, implying a collapse of the polynomial-time hierarchy to the third level (this is Yap's theorem [20]). If we could check for completeness in $\Sigma_2\mathrm{P}$, then the collapse would deepen to the second level and we would be done, but we do not know how to do this, even under our derandomization assumption. What we *can* do under our assumption, is to construct a correct nondeterministic circuit for $\overline{\mathrm{SAT}}$ "half a level" up from $\Sigma_2\mathrm{P}$, namely in $\mathrm{P}^{\Sigma_2\mathrm{P}}$ (just like we could if we knew how to check completeness in $\Sigma_2\mathrm{P}$). This collapses the polynomial-time hierarchy down to $\mathrm{P}^{\Sigma_2\mathrm{P}}$, giving our second collapse result.

The particular problem in prAM that we assume can be derandomized is the following approximate lower bound problem: Given a circuit C and an integer a, decide whether C accepts at least a inputs or noticeably less than a, say, less than $a(1-\epsilon)$ where $\epsilon = 1/\operatorname{poly}(n)$. Goldwasser and Sipser [8] showed that this problem lies in prAM, even when C is nondeterministic.

The key difference our derandomization assumption makes is the added ability to guarantee, within $\Sigma_2\mathrm{P}$, that a nondeterministic circuit C is "almost" complete, i.e., that C accepts at least a $(1-\epsilon)$-fraction of all unsatisfiable formulas, and even more generally, that a nondeterministic circuit C accepts at least a $(1-\epsilon)$-fraction of all unsatisfiable formulas that are rejected by some other given nondeterministic circuit. This enables us to construct in $\mathrm{P}^{\Sigma_2\mathrm{P}}$, out of a sound nondeterministic circuit C_i for $\overline{\mathrm{SAT}}$, another sound nondeterministic circuit C_{i+1} for $\overline{\mathrm{SAT}}$ that misses at most half as many unsatisfiable formulas as C_i does. Starting from the trivial sound circuit C_0 that rejects everything, this process yields a sound and complete nondeterministic circuit for $\overline{\mathrm{SAT}}$ within n iterations.

To explain the role of the derandomization hypothesis in more detail, we first sketch how to find C_1 because that case is easier. In constructing C_1 we make oracle queries of the form: Is there a sound nondeterministic circuit of size $s(n)$ for $\overline{\mathrm{SAT}}$ that accepts a inputs. These queries can be decided approximately by a $\Sigma_2\mathrm{P}$-oracle because of our derandomization assumption and because soundness can be checked in coNP. Assuming $\overline{\mathrm{SAT}}$ has nondeterministic circuits of size $s(n)$, this enables us to approximate the number $\bar{a}$ of unsatisfiable formulas of length n through a binary search using a $\Sigma_2\mathrm{P}$-oracle. Moreover, by self-reduction we get a sound circuit C_1 that accepts at least $(1-\epsilon)\bar{a}$ inputs, with the factor $(1-\epsilon)$ being due to the gap between the yes and no instances of the approximate lower bound problem. Setting $\epsilon = 1/2$, we thus find a sound circuit C_1 for $\overline{\mathrm{SAT}}$ that accepts at least half of all unsatisfiable formulas of length n.

To construct C_2 we want to employ a similar strategy as in the construction of C_1, namely to find a sound circuit $\tilde{C}_1$ for $\overline{\mathrm{SAT}}$ that seems to accept as many inputs as possible, and then set $C_2 = C_1 \vee \tilde{C}_1$. The difference, however, is that this time we want to maximize not over all inputs, but just over those inputs that C_1 rejects. This causes a problem because the set of inputs that C_1 rejects is in coNP, whereas the approximate lower bound problem allows us to estimate the size of NP sets only.

We overcome this obstacle by using the complementation idea again. By assumption, $\overline{\mathrm{SAT}}$ has small nondeterministic circuits not only at input length n, but also at larger input lengths. In particular, there is a nondeterministic circuit C' of size $s(n')$ for $\overline{\mathrm{SAT}}$ at input length n', where n' is large enough that we can express the computation of an n-input size-$s(n)$ nondeterministic circuit – in particular C_1 – with a Boolean formula of length n'. If we can get a hold of such a circuit C', then we can express the coNP-set $\{x \in \{0,1\}^n : C_1 \text{ rejects } x\}$ alternately as the NP-set $\{x \in \{0,1\}^n : C' \text{ accepts } \phi_{C_1}(x,\cdot)\}$, where $\phi_{C_1}(x,y)$ is a Boolean formula of size n' that expresses that y is a valid accepting computation of C_1 on input x. Since the latter set is in NP, it can be provided as input to the approximate lower bound problem. Of course, getting a hold of a circuit C' for $\overline{\mathrm{SAT}}$ at length n' is the very problem we are trying to solve – only harder since $n' > n$. We observe, however, that we do not need to explicitly check the completeness of C' for $\overline{\mathrm{SAT}}$; it suffices to check the soundness of C' for $\overline{\mathrm{SAT}}$. Since the latter can be done in coNP, we can guess and check the circuit C' in $\Sigma_2\mathrm{P}$.

To recapitulate, we want to find a sound nondeterministic circuit $\tilde{C}_1$ for $\overline{\mathrm{SAT}}$ that misses at most half of the unsatisfiable formulas that C_1 misses. We accomplish this by first encoding the computation of C_1 on a generic input x as a Boolean formula $\phi_{C_1}(x,y)$ of length n', such that $\phi_{C_1}(x_0,\cdot)$ is satisfiable for a particular x_0 iff C_1 accepts x_0. Then we make oracle queries that ask: Is there a nondeterministic circuit C' of size $s(n')$ on n' inputs, and a nondeterministic circuit $\tilde{C}_1$ of size $s(n)$ on n inputs, such that (i) the set $\{x \in \{0,1\}^n : \tilde{C}_1 \text{ accepts } x \text{ and } C' \text{ accepts } \phi_{C_1}(x,\cdot)\ \}$ is of size at least a, and (ii) C' and $\tilde{C}_1$ are both sound for $\overline{\mathrm{SAT}}$. By our derandomization assumption that $\mathrm{prAM} \subseteq \Sigma_2\mathrm{P}$, these queries can be made to a $\Sigma_2\mathrm{P}$-oracle, which allows us to construct $\tilde{C}_1$ in $\mathrm{P}^{\Sigma_2\mathrm{P}}$. We then set $C_2 = C_1 \vee \tilde{C}_1$.

As a side remark we point out that, although we do not explicitly require the circuit C' to be complete for $\overline{\mathrm{SAT}}$, the maximization of a forces C' to be complete (on the relevant instances). This is how we can avoid checking completeness explicitly (which seems to require the power of $\Pi_2\mathrm{P}$) although we rely on it.

Having found C_2, we then iterate to get a third nondeterministic circuit C_3 that misses at most half as many unsatisfiable formulas as C_2 does, and so on until we reach perfect completeness. This way we construct in $\mathrm{P}^{\Sigma_2\mathrm{P}}$ a nondeterministic circuit of size $O(n \cdot s(n))$ for $\overline{\mathrm{SAT}}$ at length n. The collapse of the polynomial-time hierarchy to $\mathrm{P}^{\Sigma_2\mathrm{P}}$ follows.

C. The lower bound framework

In both our main result and the result of Impagliazzo et al. [10] mentioned in the introduction, the proof of the forward direction – from derandomization to hardness – can be cast as an instantiation of a generic framework. We now describe that framework.

Our goal is to show that, under some derandomization assumption, some class $\mathcal{C}$ does not have polynomial-size circuits of type τ, where τ could be deterministic or nondeterministic. The proof goes by contradiction and consists of the following ingredients. Assume that $\mathcal{C}$ has τ-circuits of polynomial size.

1) Collapsing the polynomial-time hierarchy. Use the hypothesis that $\mathcal{C}$ has τ-circuits of polynomial size to show that the polynomial-time hierarchy can be simulated in some randomized class $\mathcal{R}$.
2) Derandomization. Use the derandomization assumption to show that all of $\mathcal{R}$ reduces to some fixed $K \in \mathcal{C}$ under mapping reductions computable by deterministic circuits of some fixed polynomial size, for infinitely many input lengths.

The hypothesis that $\mathcal{C}$ has τ-circuits of polynomial size implies that K has τ-circuits of size n^c for some constant c. By combining the above two ingredients, we conclude that all of the polynomial-time hierarchy can be decided by τ-circuits of size less than n^d for some fixed constant d and infinitely many input lengths. This contradicts Kannan's resultthat for any fixed polynomial n^d, there exists a language in some level of the polynomial-time hierarchy that requires τ-circuits of size at least n^d for all but finitely many input lengths [13].

The following instantiations of the above framework yield the derandomization-to-hardness results from [10], our nonrelativizing argument, and our weaker but relativizing argument.

	$\mathcal{C}$	τ	$\mathcal{R}$
[10]	NE	d	MA
nonrel. arg.	$\Sigma_2\mathrm{E}$	n	prM(AM||coNP)
rel. arg.	$\mathrm{E}^{\Sigma_2\mathrm{P}}$	n	$\mathrm{P}^{\mathrm{prM(AM\|\|coNP)}}$

The classes $\mathcal{R}$ listed for our results are technical constructs that are implicit in the proof of Theorem 2, and are explicitly defined in Section III.

D. Related work

Using Kannan's argument to get circuit lower bounds from a derandomization assumption for prAM was carried out in [2]. The same paper also presents an alternate and simpler proof that does not use Kannan's argument, but uses the power of prAM to directly diagonalize against deterministic circuits.

The general technique of using a prAM-oracle to iteratively construct a sound circuit with rapidly increasing completeness, appears in the work of Chakaravarthy and Roy [5]. Using this technique they show that PH collapses to $\mathrm{P}^{\mathrm{prAM}}$, under the classical Karp-Lipton assumption that $\mathrm{NP} \subseteq \mathrm{P}/\operatorname{poly}$.

Be it for diagonalization as in [2], or for finding a circuit as in [5], [2] and our work, the use of a prAM-oracle can be viewed as finding a witness $\tilde{y}$ that approximately maximizes a "quality measure" f defined on the set of all strings. For diagonalization purposes this measure would be the number of circuits that a given string y eliminates when viewed as the characteristic string of a function. For finding a circuit for $\overline{\mathrm{SAT}}$ at length n, y is viewed as a circuit and $f(y)$ measures the number of unsatisfiable formulas that y accepts provided that y is sound.

A related work is that of Goldreich [7], who uses a prBPP-oracle to construct a "targeted canonical generator." That work can also be viewed as approximately maximizing a quality measure f, where $f(y)$ may be defined recursively as the average quality of the extensions of y, i.e., $f(y) = \frac{1}{2}(f(y0) + f(y1))$. The difference between the works mentioned in the previous paragraph and [7] is that in the latter work f can be additively approximated using a prBPP-oracle, whereas in the former works it is multiplicatively approximated using a prAM-oracle.

Regarding our high-end collapse result involving coNP and $\mathrm{NP}/\operatorname{poly}$, at a more refined level of granularity the strongest unconditional collapse consequence of the condition $\mathrm{coNP} \subseteq \mathrm{NP}/\operatorname{poly}$ is that PH collapses to $\mathrm{S}_2\mathrm{P}^{\mathrm{NP}}$ [4], a class that contains $\mathrm{P}^{\Sigma_2\mathrm{P}}$ but is not known to equal it. Similarly, the strongest unconditional collapse consequence of $\mathrm{PSPACE} \subseteq \mathrm{NP}/\operatorname{poly}$ is that $\mathrm{PSPACE} \subseteq \mathrm{S}_2\mathrm{P}^{\mathrm{NP}}$.

III. Notation and Conventions

In this section we introduce our notation and conventions, including the notion of an augmented Arthur-Merlin protocol, which is a technical construct that naturally arises in our collapse arguments. Most of our notation is standard (see, e.g., [1]), except that in the remainder of this paper the term "circuit" always refers to a Boolean *nondeterministic* circuit, unless stated otherwise.

Circuits. A (nondeterministic Boolean) *circuit* C consists of AND and OR gates of fan-in 2, NOT gates of fan-in 1, input gates of fan-in 0, and additionally, choice gates of fan-in 0. We say that the circuit accepts input x, or $C(x) = 1$ in short, if there is some assignment of Boolean values to the choice gates that makes the circuit evaluate to 1; otherwise we say that C rejects x, or $C(x) = 0$ in short. We measure the *size* of a circuit by the number of its connections. A circuit of size s can be described by a binary string of length $O(s \log s)$.

Promise problems and languages. A *promise problem* Π is a pair of disjoint sets (Π_Y, Π_N) of strings over the binary alphabet $\{0,1\}$. A language L is a promise problem of the form $(L, \overline{L})$, where $\overline{L} \doteq \{x \in \{0,1\}^* \,:\, x \notin L\}$. A promise problem $\Pi' = (\Pi'_Y, \Pi'_N)$ is said to *agree with* a promise problem $\Pi = (\Pi_Y, \Pi_N)$ if $\Pi_Y \subseteq \Pi'_Y$ and $\Pi_N \subseteq \Pi'_N$. For two classes of promise problems $\mathcal{C}$ and $\mathcal{C}'$, we write $\mathcal{C} \subseteq \mathcal{C}'$ if for every $\Pi \in \mathcal{C}$ there exists $\Pi' \in \mathcal{C}'$ such that Π' agrees with Π. We say that Π reduces to Π' if there exists an oracle Turing machine M such that $M^{L'}$ agrees with Π for *every* language L' that agrees with Π'. In particular, a language L is in $\mathrm{P}^{\Pi'}$ if there exists a polynomial-time oracle Turing machine M such that $M^{L'}$ decides L for *every* language L' that agrees with Π'. For more on promise problems, see the survey [6].

SAT represents the language of satisfiable Boolean formulas. By prAM we denote the class of promise problems Π for which there exists a constant c and a language $L \in \mathrm{P}$ such that for every input x

$$\begin{aligned} x \in \Pi_Y \quad &\Rightarrow \quad \Pr_y[(\exists z)\langle x, y, z\rangle \in L] \geq 2/3, \text{ and} \\ x \in \Pi_N \quad &\Rightarrow \quad \Pr_y[(\exists z)\langle x, y, z\rangle \in L] \leq 1/3, \end{aligned}$$

where n denotes the length of x, the variables y and z range over $\{0,1\}^{n^c}$, and the probabilities are with respect to the uniform distribution. AM denotes those problems in prAM that are languages. Underlying each problem in prAM there is a protocol between a randomized polynomial-time verifier (Arthur) and an all-powerful prover (Merlin); we refer to these protocols as Arthur-Merlin protocols or Arthur-Merlin games.

In our proofs the following technical augmentation of Arthur-Merlin protocols arises naturally. For lack of a better name, we refer to them as "augmented" Arthur-Merlin protocols.

Definition 1 (Augmented Arthur-Merlin protocol). *The class* $\mathrm{prM(AM||coNP)}$ *consists of all promise problems* Π *for which there exists a constant* c*, a promise problem* $\Gamma \in \mathrm{prAM}$*, and a language* $V \in \mathrm{coNP}$ *such that*

$$\begin{aligned} x \in \Pi_Y \quad &\Rightarrow \quad (\exists y)\left(\langle x, y\rangle \in \Gamma_Y \ \wedge\ \langle x, y\rangle \in V\right), \text{ and} \\ x \in \Pi_N \quad &\Rightarrow \quad (\forall y)\left(\langle x, y\rangle \in \Gamma_N \ \vee\ \langle x, y\rangle \notin V\right), \end{aligned}$$

where n *denotes the length of* x*, and* y *ranges over* $\{0,1\}^{n^c}$*.* $\mathrm{M(AM||coNP)}$ *denotes those problems in* $\mathrm{prM(AM||coNP)}$ *that are languages.*

Similar to the class prAM, underlying each problem in $\mathrm{prM(AM||coNP)}$ there is a protocol between an all-powerful prover, Merlin, and – in this case – two verifiers, who cannot communicate with each other. One verifier is the usual randomized polynomial-time Arthur from the prAM-problem Γ; the other one is the coNP-verifier V. Merlin goes first and sends a common message to both verifiers. At this point, V has to make a decision to accept/reject, whereas Arthur can interact with Merlin as in the Arthur-Merlin protocol Γ before making a decision. The input is accepted by the protocol iff both verifiers accept.

IV. Collapse Results

In this section we establish our collapse result (Theorem 2), which uses the assumption that prAM can be mildly derandomized. In fact, we prove an unconditional collapse result involving the class of augmented Arthur-Merlin protocols introduced in Definition 1, from which Theorem 2 follows under the derandomization assumption. We first establish a collapse result assuming PSPACE has nondeterministic circuits of polynomial size (corresponding to the first part in Theorem 2) and then do the same for coNP instead of PSPACE (corresponding to the second part in Theorem 2).

A. *Collapse result for* PSPACE

The proof of the following theorem uses interactive proofs for PSPACE and as such does not relativize.

Theorem 3. *If* $\mathrm{PSPACE} \subseteq \mathrm{NP/poly}$ *then* $\mathrm{PSPACE} \subseteq \mathrm{M(AM||coNP)}$.

Proof: Let L be in PSPACE, fix an interactive proof system for L, and consider the language L_{prover} consisting of all tuples $\langle x, y, b\rangle$ such that y is a prefix of the transcript of an interaction of the verifier with the honest prover on input x, and the next bit in the transcript is sent by the prover and equals b. Without loss of generality we can assume that L_{prover} is paddable such that given a random string ρ for the verifier, we can construct the entire transcript with the honest prover on an input $x \in \{0,1\}^n$ by making queries to L_{prover} of a *single* length $\tau(n) = \mathrm{poly}(n)$.

By the assumption that $\mathrm{PSPACE} \subseteq \mathrm{NP/poly}$, L_{prover} can be decided by some polynomial-size nondeterministic circuit C_{prover}. Now consider the following augmented Arthur-Merlin protocol for deciding L on $x \in \{0,1\}^n$. Merlin sends to both verifiers a polynomial-size nondeterministic circuit C', purported to compute L_{prover} at length $m = \tau(n)$. The coNP-verifier V checks that C' is "single-valued", i.e., for all possible queries to the circuit C', if for some query $\langle x_0, y_0, b_0\rangle$ the circuit C' accepts then C' rejects the complementary query $\langle x_0, y_0, \neg b_0\rangle$. Note that this check is indeed in coNP.

Arthur picks a random string ρ of the appropriate length (at most $\tau(n)$) and sends it to Merlin. Merlin sends the transcript for the interactive protocol for L on input x corresponding to the coin flips ρ. Merlin also sends the certificates for C' that purportedly produce that transcript. Arthur accepts iff C' produces the transcript when given those certificates, and the transcript is accepting.

To argue completeness, consider $x \in L$. Then Merlin can just send $C' = C_{\text{prover}}$. That circuit passes the coNP-verifier V and also passes Arthur's verification with high probability.

For the soundness, consider $x \notin L$, and suppose that Merlin sends a circuit C' that passes the coNP-verifier. This means that C' is single-valued, and corresponds to a fixed prover strategy. Then Arthur rejects with high probability by the soundness of the original interactive proof system for L. ■

The proof of the first part of Theorem 2 follows immediately from Theorem 3.

Proof of part (i) of Theorem 2: If prAM can be simulated in Σ_2P then so can prM(AM||coNP). This follows because replacing Γ_Y in Definition 1 by a Σ_2P-predicate and Γ_N by its complement, turns Π_Y into a Σ_2P-predicate and Π_N into its complement. If in addition PSPACE $\subseteq$ NP/poly, Theorem 3 implies that PSPACE $\subseteq$ M(AM||coNP) $\subseteq \Sigma_2$P. ■

B. Collapse result for coNP

We proceed with a relativizable proof of the following unconditional collapse result assuming coNP has nondeterministic circuits of polynomial size.

Theorem 4. *If* coNP $\subseteq$ NP/poly *then* $\Sigma_3\mathrm{P} \subseteq \mathrm{P}^{\mathrm{prM(AM||coNP)}}$.

Note that the second part of Theorem 2 follows immediately from Theorem 4 under the mild derandomization assumption for prAM, in a relativizable way.

Proof of part (ii) of Theorem 2: As we argued in the proof of part (i), if prAM can be simulated in Σ_2P then so can prM(AM||coNP). If in addition coNP $\subseteq$ NP/poly, Yap's theorem [20] and Theorem 4 imply that PH $\subseteq \Sigma_3\mathrm{P} \subseteq \mathrm{P}^{\mathrm{prM(AM||coNP)}} \subseteq \mathrm{P}^{\Sigma_2\mathrm{P}}$. ■

We now argue Theorem 4. Assume that $\overline{\mathrm{SAT}}$ has nondeterministic circuits of size $s(n)$, where s is some polynomial. Following the outline of Section II-B, with the aid of a prM(AM||coNP)-oracle, we construct a circuit of size $O(n \cdot s(n))$ that correctly decides $\overline{\mathrm{SAT}}$ on all instances of size n. The circuit is obtained as the end of a sequence of sound circuits with rapidly improving completeness, starting from the trivial circuit that rejects everything.

To measure the improvement in each step, we consider the following function $f : \{0,1\}^* \times \{0,1\}^* \to \mathbb{N}$, which takes as arguments the current circuit C in the sequence, and a candidate circuit $\tilde{C}$ to improve the completeness of C in the next step, while maintaining soundness.

$$f(C, \tilde{C}) = \begin{cases} |C^{-1}(0) \cap \tilde{C}^{-1}(1)| & \text{if } \tilde{C} \text{ is sound for } \overline{\mathrm{SAT}} \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

We map circuits $\tilde{C}$ that are not sound to zero because their use would violate the soundness of the sequence. If $\tilde{C}$ is sound, f counts the number of instances of $\overline{\mathrm{SAT}}$ that are missed by C but caught by $\tilde{C}$.

For a given circuit C that is sound but not complete, our goal is to find a circuit $\tilde{C}$ that approximately maximizes $f(C, \tilde{C})$. For this task we only need access to an approximation of f to within a constant multiplicative factor. In general, for a function $f : \{0,1\}^* \times \{0,1\}^* \to \mathbb{N}$, approximating f to within a multiplicative factor is captured by the following promise problem $A_f = (Y_f, N_f)$, which additionally takes an integer a in binary and a positive integer $1/\epsilon$ in unary.

$$\begin{array}{rcl} Y_f & = & \{\langle x, y, a, \epsilon\rangle : f(x,y) \geq a\} \\ N_f & = & \{\langle x, y, a, \epsilon\rangle : f(x,y) < (1-\epsilon)a\}. \end{array} \tag{4}$$

The crux of our argument is the following lemma.

Lemma 1. *Let f be the function defined by (3), and A_f the promise problem given by (4). If* coNP $\subseteq$ NP/poly *then* $A_f \in$ prM(AM||coNP).

Proof: We follow the outline from Section II-B, but cast the resulting algorithm for A_f in terms of an augmented Arthur-Merlin protocol with coNP-verifier V on input $(C, \tilde{C})$.

Since V can check whether $\tilde{C}$ is sound for $\overline{\mathrm{SAT}}$, if suffices to construct an augmented Arthur-Merlin protocol for A_g, where g is the simplification of f defined by $g(C, \tilde{C}) \doteq |C^{-1}(0) \cap \tilde{C}^{-1}(1)|$.

Goldwasser and Sipser [8] showed that for every predicate $L \in$ NP and function $h : \{0,1\}^* \times \{0,1\}^* \to \mathbb{N}$ with

$$h(u,v) = |\{w \in \{0,1\}^* : \langle u, v, w\rangle \in L\}|, \tag{5}$$

the promise problem A_h is in prAM. Note that the function g is of the form (5), except that the underlying predicate $C(w) = 0 \wedge \tilde{C}(w) = 1$ is the difference of two NP sets rather than just an NP set. We remedy this issue by invoking the hypothesis coNP $\subseteq$ NP/poly as follows.

Let C and $\tilde{C}$ have n inputs and be of size at most s. By the Cook-Levin Theorem, we can construct in time poly(s) a Boolean formula $\phi_C(x,y)$ of size n' such that for all $x_0 \in \{0,1\}^n$, $C(x_0) = 0$ iff $\phi_C(x_0, \cdot) \in \overline{\mathrm{SAT}}$. Let C' denote a circuit that takes inputs of size n', and let L denote the predicate that on input $\langle u, v, w\rangle$ with $u = \langle C, C'\rangle$ and $v = \tilde{C}$, decides whether $C'(\phi_C(w, \cdot)) = 1 \wedge \tilde{C}(w) = 1$. Note that $L \in$ NP, so $A_h \in$ prAM, where h is defined by (5). Whenever C' is sound for $\overline{\mathrm{SAT}}$ on instances of size n', then $\langle\langle C, C'\rangle, \tilde{C}, w\rangle \in L$ only if $C(w) = 0 \wedge \tilde{C}(w) = 1$, which implies that $h(\langle C, C'\rangle, \tilde{C}) \leq g(C, \tilde{C})$. Moreover, if C' correctly decides $\overline{\mathrm{SAT}}$ at length n', then $\langle\langle C, C'\rangle, \tilde{C}, w\rangle \in L$ iff $C(w) = 0 \wedge \tilde{C}(w) = 1$, and $h(\langle C, C'\rangle, \tilde{C}) = g(C, \tilde{C})$. By the hypothesis that coNP $\subseteq$ NP/poly, there exists a circuit C' of size poly(n') that computes $\overline{\mathrm{SAT}}$ on instances of length n'. This suggests the following augmented Arthur-Merlin protocol for A_g on input $\langle C, \tilde{C}, a, \epsilon\rangle$.

If $a \leq 0$ the protocol trivially accepts, as g is nonnegative. Otherwise, Merlin sends as his initial message a circuit C' of size poly(n') on n'-inputs, purported to be a circuit for $\overline{\mathrm{SAT}}$ at length n'. The coNP-verifier checks that C' is sound with respect to $\overline{\mathrm{SAT}}$, and Arthur engages in a protocol with Merlin for the promise problem A_h on input $\langle\langle C, C'\rangle, \tilde{C}, a, \epsilon\rangle$.

To argue completeness of the protocol for A_g, suppose that $g(C, \tilde{C}) \geq a > 0$. In order to make both verifiers accept, Merlin sends as his initial message a polynomial-size circuit C' for $\overline{\mathrm{SAT}}$ at length n', which exists by the hypothesis that $\mathrm{coNP} \subseteq \mathrm{NP}/\mathrm{poly}$. Since C' is sound for $\overline{\mathrm{SAT}}$, the coNP-verifier accepts. As for the other verifier (Arthur) since C' is a correct circuit for $\overline{\mathrm{SAT}}$, we have that $h(\langle C, C'\rangle, \tilde{C}) = g(C, \tilde{C}) \geq a$. The completeness of the Arthur-Merlin protocol for A_h then guarantees that Arthur can be convinced.

To argue soundness, suppose that $g(C, \tilde{C}) < a(1-\epsilon)$. First, if the coNP-verifier accepts, then Merlin must have sent a sound circuit C' for $\overline{\mathrm{SAT}}$ in the first round. Whenever C' is sound for $\overline{\mathrm{SAT}}$ at length n' then $h(\langle C, C'\rangle, \tilde{C}) \leq g(C, \tilde{C})$, so $h(\langle C, C'\rangle, \tilde{C}) < a(1-\epsilon)$. By the soundness of the Arthur-Merlin protocol for A_h, this means that Arthur rejects with high probability. Thus, whenever the coNP-verifier accepts, Arthur rejects with high probability. This completes the proof. ■

Lemma 1 allows us to efficiently improve the completeness of a sound but incomplete circuit C for $\overline{\mathrm{SAT}}$ when given oracle access to a language that agrees with A_f by finding a circuit $\tilde{C}$ that approximately maximizes $f(C, \tilde{C})$, and outputting $C \vee \tilde{C}$. The approximate maximization can be done using the following generic lemma.

Lemma 2. *Let* $f : \{0,1\}^* \times \{0,1\}^* \to \mathbb{N}$ *be such that* $f(x,y) \leq 2^{|x|^c}$ *for some constant* c. *If* $A_f \in \mathrm{prM}(\mathrm{AM}||\mathrm{coNP})$, *where* A_f *denotes the promise problem defined by (4), then there exists a promise problem* $\Pi \in \mathrm{prM}(\mathrm{AM}||\mathrm{coNP})$ *such that the following holds for any language* L *that agrees with* Π. *On input* $x \in \{0,1\}^*$, *a nonnegative integer* m *in unary, and a positive integer* $1/\tilde{\epsilon}$ *in unary, we can find, in deterministic polynomial time with oracle access to* L, *a value* $\tilde{y} \in \{0,1\}^m$ *such that*

$$f(x, \tilde{y}) \geq (1-\tilde{\epsilon}) \cdot \max_{y \in \{0,1\}^m} f(x,y).$$

Lemma 2 holds more generally when $\mathrm{prM}(\mathrm{AM}||\mathrm{coNP})$ is replaced by any reasonable complexity class $\mathcal{C}$ that is closed under the existential operator,such as Σ_k^{P} for any $k \geq 1$. Chakaravarthy and Roy [5] implicitly use it with $\mathcal{C} = \mathrm{prAM}$.

Proof of Lemma 2: We run a prefix search for $\tilde{y}$. In order to do so, we make use of the auxiliary function $g(x,y) \doteq \max_{yy' \in \{0,1\}^m} f(x, yy')$, where yy' denotes the concatenation of y and y', and we suppress the unary argument m for simplicity of notation. The fact that $A_f \in \mathrm{prM}(\mathrm{AM}||\mathrm{coNP})$ implies that $\Pi \doteq A_g \in \mathrm{prM}(\mathrm{AM}||\mathrm{coNP})$. This is because on input $\langle x, y, a, \epsilon\rangle$, the augmented Arthur-Merlin protocol for A_g can have Merlin first guess $y' \in \{0,1\}^{m-|y|}$ and then run the augmented Arthur-Merlin protocol for A_f on input $\langle x, yy', a, \epsilon\rangle$. Let L denote a language that agrees with the promise problem Π.

In a first phase we find an approximation $\tilde{a}$ to $a^* \doteq \max_{y \in \{0,1\}^m} f(x,y)$. To do so, we make use of the predicate $P(a) \doteq \langle x, \lambda, a, \epsilon\rangle \in L$, where λ denotes the empty string, ϵ will be set later, and the rest of the parameters are the inputs given in the statement of the lemma. Note that $P(0)$ holds because f is nonnegative. At the other end, $P(b+1)$ fails by definition. We run a binary search for an integer value $\tilde{a} \in [0, b]$ such that (i) $P(\tilde{a})$ holds and (ii) $P(\tilde{a}+1)$ fails. This guarantees that $(1-\epsilon)\tilde{a} \leq a^* \leq \tilde{a}$, where the first inequality follows from (i) and the fact that $P(\tilde{a})$ implies that $g(x, \lambda) \geq (1-\epsilon)\tilde{a}$, and the second inequality follows from (ii) and the fact that $\neg P(\tilde{a}+1)$ implies that $g(x, \lambda) < \tilde{a}+1$. This concludes the first phase.

In a second phase we run the actual prefix search for $\tilde{y}$. We maintain the invariant that

$$g(x, \tilde{y}_{1\ldots i}) \geq \tilde{a}_i, \tag{6}$$

for $0 \leq i \leq m$, where $\tilde{y}_{1\ldots i}$ denotes the prefix of length i of $\tilde{y}$, and the values $\tilde{a}_i$ are chosen not too much smaller than $\tilde{a}$. More specially, we set $\tilde{a}_0 = (1-\epsilon)\tilde{a}$, and for $i = 0, \ldots, m-1$ we extend the prefix of $\tilde{y}$ of length i to length $i+1$ as follows. By (6) we know that for at least one choice of $\tilde{y}_{i+1} \in \{0,1\}$,

$$\langle x, \tilde{y}_{1\ldots i+1}, \tilde{a}_i, \epsilon\rangle \in L, \tag{7}$$

and for any choice of $\tilde{y}_{i+1} \in \{0,1\}$ satisfying (7), $g(x, \tilde{y}_{1\ldots i+1}) \geq (1-\epsilon)\tilde{a}_i$. Thus, we pick $\tilde{y}_{i+1}$ as the least value in $\{0,1\}$ for which (7) holds, and set $\tilde{a}_{i+1} = (1-\epsilon)\tilde{a}_i$.

In the end, we obtain $\tilde{y} \in \{0,1\}^m$ satisfying

$$f(x, \tilde{y}) \geq \tilde{a}_m = (1-\epsilon)^m \tilde{a}_0 \geq (1-\epsilon)^{m+1}\tilde{a} \geq (1-\epsilon)^{m+1} a^*,$$

which is at least $(1-\tilde{\epsilon})a^*$ provided we set $\epsilon = \tilde{\epsilon}/(m+1)$.

Since both phases run in polynomial time with oracle access to L, the result follows. ■

Starting from the trivial sound circuit C_0 that rejects all inputs, we iteratively apply the improvement step based on Lemma 1 and Lemma 2 with $\tilde{\epsilon} = 1/2$. After no more than n iterations this yields a circuit of polynomial size that decides $\overline{\mathrm{SAT}}$ on inputs of size n. We have proved the following theorem.

Theorem 5. *Suppose that* $\mathrm{coNP} \subseteq \mathrm{NP}/\mathrm{poly}$. *There exists a promise problem* $\Pi \in \mathrm{prM}(\mathrm{AM}||\mathrm{coNP})$ *such that the following holds for any language* L *that agrees with* Π. *Given* n, *we can construct a polynomial-size nondeterministic circuit for* $\overline{\mathrm{SAT}}$ *at length* n *in deterministic polynomial time with oracle access to* L.

With Theorem 5 in hand, the nondeterministic variant of the Karp-Lipton argument yields the collapse stated in Theorem 4.

Proof of Theorem 4: Let K denote the $\Sigma_3\mathrm{P}$-complete language consisting of all Boolean formulas $\varphi(x,y,z)$ on three sets of variables x, y, z such that $(\exists x)(\forall y)\, \varphi(x, y, \cdot) \in$

SAT. We show that under the assumptions of the theorem, $K \in \mathrm{P}^{\mathrm{prM(AM||coNP)}}$.

Consider the related language K' consisting of all pairs $\langle \varphi, C \rangle$, where φ is a Boolean formula as above and C is a circuit such that $(\exists x)(\forall y)\, C(\varphi(x, y, \cdot)) = 0$. As the condition $C(\varphi(x, y, \cdot)) = 0$ can be decided in quasilinear time on a co-nondeterministic machine, $K' \in \Sigma_2\mathrm{P} \subseteq \mathrm{M(AM||coNP)}$. Moreover, if C is a circuit that correctly decides $\overline{\mathrm{SAT}}$ on inputs of the appropriate size, then $\varphi \in K$ iff $\langle \varphi, C \rangle \in K'$.

In order to decide L on an input φ of size n, we first run the algorithm from Theorem 5 on input n to obtain a circuit C of polynomial size for $\overline{\mathrm{SAT}}$ on inputs of the required size, and then check whether $\langle \varphi, C \rangle \in K'$. Note that any language L that agrees with the promise problem $\Pi \in \mathrm{prM(AM||coNP)}$ from Theorem 5 suffices as oracle for the construction; the circuit C we construct may depend on the choice of L, but the final membership decision to K does not. The theorem follows. ■

V. Equivalence Result

In this section we establish our hardness-derandomization equivalence for Arthur-Merlin games (Theorem 1). We first argue the derandomization-to-hardness direction for $\Sigma_2\mathrm{E}$, as well as a weaker but relativizing claim for $\mathrm{E}^{\Sigma_2\mathrm{P}}$. We finish with the hardness-to-derandomization direction for $\Sigma_2\mathrm{E}$.

A. From derandomization to hardness

We use the lower bound framework introduced in Section II-C. We assume that $\mathcal{C} \subseteq \mathrm{NP/poly}$, where $\mathcal{C} = \Sigma_2\mathrm{E}$ or $\mathcal{C} = \mathrm{E}^{\Sigma_2\mathrm{P}}$, and derive a contradiction with Kannan's result that the polynomial-time hierarchy does not have (nondeterministic) circuits of fixed polynomial size. The derivation entails two key ingredients: (i) collapsing the polynomial-time hierarchy to some randomized class $\mathcal{R}$, and (ii) derandomizing $\mathcal{R}$ assuming that $\mathrm{prAM} \subseteq \cap_{\epsilon>0}\mathrm{i.o.}-\Sigma_2\mathrm{TIME}(2^{n^\epsilon})/n^\epsilon$. We developed (i) in Section IV. We now discuss (ii).

The classes $\mathcal{R}$ we consider involve augmented Arthur-Merlin protocols as introduced in Definition 1. More precisely, we consider $\mathcal{R} = \mathrm{prM(AM||coNP)}$ and $\mathcal{R} = \mathrm{P}^{\mathrm{prM(AM||coNP)}}$. Under the stronger derandomization assumption that $\mathrm{prAM} \subseteq \Sigma_2\mathrm{SUBEXP} \doteq \cap_{\epsilon>0}\Sigma_2\mathrm{TIME}(2^{n^\epsilon})$, those classes $\mathcal{R}$ can trivially be simulated in $\Sigma_2\mathrm{SUBEXP}$ or $\mathrm{SUBEXP}^{\Sigma_2\mathrm{P}}$, respectively. To carry over that argument to the i.o.-setting with small advice, we need to make sure that the derandomization of $\mathcal{R}$ on inputs of length n only makes use of the derandomization of prAM on one of the infinitely many good input lengths m where the latter simulation is guaranteed to work. The following lemma shows how to do that for infinitely many input lengths n by exploiting the paddability of prAM and using an additional short advice string to point to a nearby good length m.

Lemma 3 (Derandomization Lemma). *Suppose that* $\mathrm{prAM} \subseteq \cap_{\epsilon>0}\mathrm{i.o.}-\Sigma_2\mathrm{TIME}(2^{n^\epsilon})/n^\epsilon$. *Then*

(i) $\mathrm{prM(AM||coNP)} \subseteq \cap_{\epsilon>0}\mathrm{i.o.}-\Sigma_2\mathrm{TIME}(2^{n^\epsilon})/n^\epsilon$, *and*

(ii) $\mathrm{P}^{\mathrm{prM(AM||coNP)}} \subseteq \cap_{\epsilon>0}\mathrm{i.o.}-\mathrm{DTIME}^{\Sigma_2\mathrm{P}}(2^{n^\epsilon})/n^\epsilon$.

Proof: Part (i): Let $\mathcal{C}$ denote $\cap_{\epsilon>0}\mathrm{i.o.}-\Sigma_2\mathrm{TIME}(2^{n^\epsilon})/n^\epsilon$, and let $\Pi \in \mathrm{prM(AM||coNP)}$. Let Γ denote the prAM-problem underlying Π, and V the coNP-verifier, as in Definition 1. By assumption, there is a language $Q \in \mathcal{C}$ that agrees with Γ. If we replace the predicate Γ by Q in the definition of Π, we obtain the language

$$R = \{x : (\exists y \in \{0,1\}^{n^c})\,(\langle x, y \rangle \in Q \ \wedge\ \langle x, y \rangle \in V)\}. \tag{8}$$

Observe that R agrees with Π. It remains to show that $R \in \mathcal{C}$.

We can assume without loss of generality that Q is paddable; by this we mean (a) $\langle x, y \rangle \in Q$ iff $\langle x, y, 0^{pad} \rangle \in Q$ for all $pad \in \mathbb{N}$, and (b) if $\langle x, y \rangle$ is of length m then there is a setting of pad such that $\langle x, y, 0^{pad} \rangle$ is of length m' for all $m' \geq m$.

Fix any $\epsilon > 0$. We want to exhibit a language $R_\epsilon \in \Sigma_2\mathrm{TIME}(2^{n^\epsilon})/n^\epsilon$ that agrees with R on infinitely many lengths n. By assumption, for every $\delta > 0$ there is a language $Q_\delta \in \Sigma_2\mathrm{TIME}(2^{m^\delta})/m^\delta$ and an infinite set of lengths M_δ such that Q_δ agrees with Q on all lengths in M_δ. We use Q_δ for a sufficiently small value of $\delta > 0$ to construct P_ϵ as follows.

Let $\ell(n)$ denote the maximum length of the (unpadded) queries issued to the language Q when deciding the language R on inputs x of length n. Suppose there exists a length $m \in M_\delta$ in the range $\ell(n) \leq m \leq \ell(n+1)$. Then we pick such a length m, give R_ϵ at length n as advice the value of m as well as the advice for Q_δ at length m, and let R_ϵ at length n be defined by (8) but with each query "$\langle x, y \rangle \in Q$" replaced by the equivalent query to Q_δ padded to length m, i.e., by "$\langle x, y, 0^{pad} \rangle \in Q_\delta$", where pad is such that $|\langle x, y, 0^{pad} \rangle| = m$. Note that in this case R_ϵ agrees with R at length n. If there is no length $m \in M_\delta$ in the range $\ell(n) \leq m \leq \ell(n+1)$, we define R_ϵ in the same way but with m set to $\ell(n)$. In this case there is no guarantee that R_ϵ and R agree at length n.

Since the intervals $[\ell(n), \ell(n+1)]$ cover all but finitely many lengths m, and Q_δ agrees with Q for infinitely many lengths m, R_ϵ agrees with R on infinitely many lengths n.

All that remains is the complexity analysis of R_ϵ. The queries to Q_δ are padded up to length no more than $\ell(n+1)$, which is polynomially bounded in n. It follows that those queries can be decided in $\Sigma_2\mathrm{TIME}(2^{n^{c\delta}})/n^{c\delta}$ for some fixed constant c. The advice for R_ϵ is of length at most $\log(\ell(n+1)) + n^{c\delta}$. Thus, if we set $\delta = \epsilon/(c+1)$ we have that $R_\epsilon \in \Sigma_2\mathrm{TIME}(2^{n^\epsilon})/n^\epsilon$. This completes part (i).

Part (ii): Let $L \in \mathrm{DTIME}^{\Pi}(n^c)$ where $\Pi \in \mathrm{prM(AM||coNP)}$. By padding all queries of the base machine up to an appropriate length m depending on n as in part (i), and providing the base machine with the advice we gave R_ϵ in part (i), we obtain that

$$L \in \cap_{\epsilon>0}\mathrm{i.o.}-\mathrm{DTIME}^{\Sigma_2\mathrm{TIME}(2^{n^\epsilon})}(n^d)/n^\epsilon,$$

where d is a constant depending on c. Since $\Sigma_2\mathrm{TIME}(2^{n^\epsilon}) \subseteq \mathrm{DTIME}^{\Sigma_2\mathrm{P}}(2^{n^\epsilon})$, the conclusion follows. ■

What the framework needs of the derandomizations of $\mathcal{R}$, is that they reduce to some fixed language $K \in \mathcal{C}$ under mapping reductions computable by deterministic circuits of fixed polynomial size. The following lemma establishes that property for the derandomizations provided by Lemma 3.

Lemma 4 (Reducibility Lemma). *For each of the following pairs $(\mathcal{C}',\mathcal{C})$, $\mathcal{C}$ contains a language K such that every $L \in \mathcal{C}'$ reduces to K under a mapping reduction that is computable by deterministic circuits of linear size.*

$\mathcal{C}'$	$\mathcal{C}$
$\cap_{\epsilon>0}\mathrm{i.o.}-\Sigma_2\mathrm{TIME}(2^{n^\epsilon})/n^\epsilon$	$\Sigma_2\mathrm{E}$
$\cap_{\epsilon>0}\mathrm{i.o.}-\mathrm{DTIME}^{\Sigma_2\mathrm{P}}(2^{n^\epsilon})/n^\epsilon$	$\mathrm{E}^{\Sigma_2\mathrm{P}}$

Proof: Part(i): Let $K = \{\langle M, y\rangle :$ M accepts y in time $2^{|y|}$ on a machine of type $\tau\}$, where τ is the type that realizes the class $\Sigma_2\mathrm{TIME}$. Note that $K \in \Sigma_2\mathrm{TIME}(2^{O(n)})$.

Let $L \in \mathcal{C}'$. For any fixed $\epsilon > 0$, let $L_\epsilon \in \Sigma_2\mathrm{TIME}(2^{n^\epsilon})/n^\epsilon$ agree with L infinitely often. More precisely, let $L'_\epsilon \in \Sigma_2\mathrm{TIME}(2^{n^\epsilon})$ and $a : \mathbb{N} \to \{0,1\}^*$ with $|a(n)| \leq n^\epsilon$ be such that for infinitely many lengths n and all $x \in \{0,1\}^n$, $x \in L$ iff $\langle x, a(n)\rangle \in L'_\epsilon$.

Now consider the following circuit C: On input $x \in \{0,1\}^n$, C outputs $\langle M_\epsilon, y\rangle$ where $y = \langle x, a(n)\rangle$ and M_ϵ is the machine that computes L'_ϵ. The output of C, as well as its size, is $O(1) + n + n^\epsilon = O(n)$ for $\epsilon \leq 1$. On those infinitely many lengths n for which L_ϵ agrees with L, we have for every $x \in \{0,1\}^n$, $x \in L \iff x \in L_\epsilon \iff \langle x, a(n)\rangle \in L'_\epsilon \iff C(x) \in K$.

Part (ii): The above argument applies verbatim when all occurrences of $\Sigma_2\mathrm{TIME}$ are replaced with $\mathrm{DTIME}^{\Sigma_2\mathrm{P}}$. ■

The framework derives a contradiction with the following nondeterministic version of a classical result of Kannan's.

Lemma 5 (implicit in [13]). *For every constant $d > 0$ there exists a language in $\mathrm{P}^{\Sigma_3\mathrm{P}}$ that requires nondeterministic circuits of size n^d for all but finitely many input lengths n.*

We include a proof for completeness.

Proof: Let $s(n) = n^d$. For all but finitely many n, $s(n)$ is no more than the maximum circuit complexity at length n. Therefore, there exists a characteristic sequence of length 2^n that cannot be computed by circuits of size less than $s(n)$. Moreover, the length ℓ of the shortest prefix σ such that no circuit of size less than $s(n)$ agrees with σ satisfies $\ell = O(s(n)\log s(n))$. This follows because circuits of size $s(n)$ can be described by strings of length $O(s(n)\log s(n))$ and distinct prefixes need distinct circuits.

The lexicographically least such prefix σ, say σ^*, can be found through a binary search, by using a $\Sigma_3\mathrm{P}$-oracle, in time $\mathrm{poly}(s(n))$. To see this, given a size-s circuit C and a length-ℓ string σ, consider the task of deciding whether the circuit C does not agree with σ. This task can be performed with an NP-oracle in time $\mathrm{poly}(s(n))$, and hence in $\Pi_2\mathrm{TIME}(\mathrm{poly}(s(n))$. To run the binary search, we need to answer queries as to whether a given string can be extended to a string σ such that for all circuits C of size less than $s(n)$, C does not agree with σ. As these queries can be decided in $\Sigma_3\mathrm{TIME}(\mathrm{poly}(s(n)))$ and $s(n)$ is polynomial, we can construct σ^* in polynomial time with oracle access to $\Sigma_3\mathrm{P}$.

Once found, σ^* is viewed as a string σ' of length 2^n that is all zeroes beyond the first ℓ bits (and that equals σ^* in its first ℓ bits). It follows that σ' is the characteristic string of a language L that cannot be decided by a circuit of size less than $s(n)$ at length n. On input x of length n, whether $x \in L$ can be decided according to the xth bit of σ'. Thus, $L \in \mathrm{P}^{\Sigma_3\mathrm{P}}$. ■

We now have all the ingredients to instantiate the lower bound framework described in Section II-C and obtain the following derandomization-to-hardness results for Arthur-Merlin games.

Theorem 6. (i) $\Sigma_2\mathrm{E} \not\subseteq \mathrm{NP/poly}$ *if* $\mathrm{prAM} \subseteq \cap_{\epsilon>0}\mathrm{i.o.}-\Sigma_2\mathrm{TIME}(2^{n^\epsilon})/n^\epsilon$.
(ii) *Relative to any oracle,* $\mathrm{E}^{\Sigma_2\mathrm{P}} \not\subseteq \mathrm{NP/poly}$ *if* $\mathrm{prAM} \subseteq \cap_{\epsilon>0}\mathrm{i.o.}-\Sigma_2\mathrm{TIME}(2^{n^\epsilon})/n^\epsilon$.

Proof: For $\mathcal{C} \in \{\Sigma_2\mathrm{E}, \mathrm{E}^{\Sigma_2\mathrm{P}}\}$, $\mathcal{R}$ the corresponding left-hand side class in the Derandomization Lemma (Lemma 3), and $\mathcal{C}'$ the corresponding class in the Reducibility Lemma (Lemma 4), the following derivation proves both parts by contradiction.

$$\begin{array}{lll} & \mathcal{C} \subseteq \mathrm{NP/poly} & \\ \Longrightarrow & \mathrm{PH} \subseteq \mathcal{R} & \text{(by the collapse results)} \\ \Longrightarrow & \mathrm{PH} \subseteq \mathcal{C}' & \text{(by Lemma 3)} \\ \Longrightarrow & \mathrm{PH} \subseteq \mathrm{i.o.}-\mathrm{NSIZE}(n^d) & \text{(by Lemma 4)} \\ \Longrightarrow & \quad\text{contradiction} & \text{(by Lemma 5),} \end{array}$$

where d is some constant and $\mathrm{NSIZE}(n^d)$ denotes the class of languages with nondeterministic circuits of size n^d.

The relativization claim in (ii) follows because all steps in that part relativize (whereas the collapse argument for (i) involves a nonrelativizing step). ■

Note that part (i) of Theorem 6 yields the forward direction of Theorem 1.

B. From hardness to derandomization

Finally, we argue the direction from hardness to derandomization in Theorem 1. This direction follows in a straightforward way from the known hardness versus randomness tradeoffs. First, in order to derandomize prAM it suffices to construct pseudorandom generators that fool nondeterministic circuits.

Lemma 6 (implicit in [16]). *If there is a pseudorandom generator G that on seed length $\sigma(n)$ is computable in $\Sigma_2\mathrm{TIME}(2^{O(\sigma(n))})$ with advice of length $\sigma(n)$, and fools nondeterministic circuits for infinitely many n, then* prAM *can infinitely often be simulated in $\Sigma_2\mathrm{TIME}(2^{O(\sigma(n^*))}n^*)$ with advice of length $\sigma(n^*)$.*

The pseudorandom generators needed in Lemma 6 follow from the given hardness assumption by the known hardness versus randomness tradeoffs for prAM.

Lemma 7 (implicit in [19]). *There exists a positive constant c such that the following holds for any constructible function $\ell : \mathbb{N} \to \mathbb{N}$. If there is a language in $\Sigma_2\mathrm{E}$ that requires nondeterministic circuits of size n^c at length $\ell(n)$ for infinitely many n, then there exists a pseudorandom generator G that has constructible seed length $\sigma(n) = O(\ell^2(n)/\log n)$, is computable in $\Sigma_2\mathrm{TIME}(2^{O(\sigma(n))})$ on seed length $\sigma(n)$ with advice of length $\sigma(n)$, and fools nondeterministic circuits at infinitely many lengths n.*

Given the hardness hypothesis, for any $\epsilon > 0$, we can pick $\ell(n)$ in Lemma 7 to be n^δ for a small enough δ and get a pseudorandom generator G with seed length $\sigma(n) < n^\epsilon$ that is computable in $\Sigma_2\mathrm{TIME}(2^{n^\epsilon})/n^\epsilon$ and fools nondeterministic circuits for infinitely many n. Using G in Lemma 6 yields the required derandomization of prAM. This establishes the backward direction of Theorem 1.

Acknowledgements. Research supported by NSF grant 1017597.

References

[1] S. Arora and B. Barak, *Computational Complexity: A Modern Approach.* Cambridge University Press, 2009.

[2] B. Aydınlıoğlu, D. Gutfreund, J. M. Hitchcock, and A. Kawachi, “Derandomizing Arthur-Merlin games and approximate counting implies exponential-size lower bounds,” *Computational Complexity*, vol. 20, no. 2, pp. 329–366, 2011.

[3] L. Babai, L. Fortnow, N. Nisan, and A. Wigderson, “BPP has subexponential time simulations unless EXPTIME has publishable proofs,” *Computational Complexity*, vol. 3, pp. 307–318, 1993.

[4] J. Cai, V. T. Chakaravarthy, L. A. Hemaspaandra, and M. Ogihara, “Competing provers yield improved Karp-Lipton collapse results,” *Information and Computation*, vol. 198, no. 1, pp. 1–23, 2005.

[5] V. T. Chakaravarthy and S. Roy, “Finding irrefutable certificates for $\mathrm{S}_2^{\mathrm{P}}$ via Arthur and Merlin,” *STACS*, pp. 157–168, 2008.

[6] O. Goldreich, “On promise problems: A survey,” in *Essays in Memory of Shimon Even*, pp. 254–290, 2006.

[7] O. Goldreich, “Two comments on targeted canonical derandomizers,” ECCC Technical Report TR11-047, 2011.

[8] S. Goldwasser and M. Sipser, “Private coins versus public coins in interactive proof systems,” *STOC*, pp. 59–68, 1986.

[9] D. Gutfreund, R. Shaltiel, and A. Ta-Shma, “Uniform hardness versus randomness tradeoffs for Arthur-Merlin games,” *Computational Complexity*, vol. 12, no. 3-4, pp. 85–130, 2003.

[10] R. Impagliazzo, V. Kabanets, and A. Wigderson, “In search of an easy witness: exponential time vs. probabilistic polynomial time,” *Journal of Computer and System Sciences*, vol. 65, no. 4, pp. 672–694, 2002.

[11] R. Impagliazzo and A. Wigderson, “P = BPP if E requires exponential circuits: Derandomizing the XOR lemma,” *STOC*, pp. 220–229, 1997.

[12] V. Kabanets and R. Impagliazzo, “Derandomizing polynomial identity tests means proving circuit lower bounds,” *Computational Complexity*, vol. 13, no. 1/2, pp. 1–46, 2004.

[13] R. Kannan, “Circuit-size lower bounds and nonreducibility to sparse sets,” *Information and Control*, vol. 55, no. 1, pp. 40–56, 1982.

[14] R. M. Karp and R. J. Lipton, “Turing machines that take advice,” *L’Enseignement Mathématique*, vol. 28, no. 2, pp. 191–209, 1982.

[15] J. Kinne, D. van Melkebeek, and R. Shaltiel, “Pseudorandom generators, typically-correct derandomization, and circuit lower bounds,” *Computational Complexity*, vol. 21, no. 1, pp. 3–61, 2012.

[16] A. Klivans and D. van Melkebeek, “Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses,” *SIAM Journal on Computing*, vol. 31, no. 5, pp. 1501–1526, 2002.

[17] C. Lund, L. Fortnow, H. J. Karloff, and N. Nisan, “Algebraic methods for interactive proof systems,” *Journal of the ACM*, vol. 39, no. 4, pp. 859–868, 1992.

[18] N. Nisan and A. Wigderson, “Hardness vs. randomness,” *Journal of Computer and System Sciences*, vol. 49, no. 2, pp. 149–167, 1994.

[19] R. Shaltiel and C. Umans, “Pseudorandomness for approximate counting and sampling,” *Computational Complexity*, vol. 15, no. 4, pp. 298–341, 2006.

[20] C. Yap, “Some consequences of non-uniform conditions on uniform classes,” *Theoretical Computer Science*, vol. 26, pp. 287–300, 1983.