NONDETERMINISTIC CIRCUIT LOWER BOUNDS FROM MILDLY DERANDOMIZING ARTHUR-MERLIN GAMES

Barış Aydinlioğlu and Dieter van Melkebeek

March 24, 2014

Abstract.
In several settings derandomization is known to follow from circuit lower bounds that themselves are equivalent to the existence of pseudorandom generators. This leaves open the question whether derandomization implies the circuit lower bounds that are known to imply it, i.e., whether the ability to derandomize in any way implies the ability to do so in the canonical way through pseudorandom generators.

For the setting of decision problems, Impagliazzo, Kabanets, and Wigderson (2002) implicitly showed the following equivalence: Randomized polynomial-time decision procedures for promise problems can be simulated in NSUBEXP (the subexponential version of NP) with subpolynomial advice on infinitely many input lengths if *and only if* NEXP $\not\subseteq$ P/poly. We establish a full analogue in the setting of verification procedures: Arthur-Merlin games for promise problems can be simulated in Σ_2SUBEXP (the subexponential version of Σ_2P) with subpolynomial advice on infinitely many input lengths *if and only if* Σ_2EXP $\not\subseteq$ NP/poly.

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 Σ_2P, we show that (i) PSPACE $\subseteq$ NP/poly implies PSPACE $\subseteq \Sigma_2$P, and (ii) coNP $\subseteq$ NP/poly implies PH $\subseteq$ $\mathsf{P}^{\Sigma_2\mathsf{P}}$.

Of possible independent interest is a generic framework that we provide, which captures several results in the literature of the form "derandom-

ization implies circuit lower bounds." In particular, our framework enables a unified view of our result, the one by Impagliazzo et al. (2002) mentioned above, as well as the result by Kabanets and Impagliazzo (2004) that derandomizing polynomial identity testing yields arithmetic circuit lower bounds.

Keywords. circuit lower bounds, nondeterministic circuits, Arthur-Merlin games, derandomization

Subject classification. 68Q15, 68Q17

1. 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 computed deterministically without much overhead – in subexponential or maybe even polynomial time. Similarly, in the context of randomized verification procedures one seeks for efficient nondeterministic computations of Arthur-Merlin games: the class AM, or its promise version prAM.

A major development in the area are hardness versus randomness tradeoffs (Babai *et al.* 1993; Impagliazzo & Wigderson 1997; Nisan & Wigderson 1994; Yao 1982), 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) computable in $\mathcal{C}$ with "small" seed length. 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 , and can be used to derandomize any procedure that can be modeled by small circuits of type τ (Klivans & van Melkebeek 2002). See the table below for some examples from the above papers and Shaltiel & Umans (2006), where $\mathsf{E} \doteq \mathsf{DTIME}(2^{O(n)})$, $\mathsf{NE} \doteq \mathsf{NTIME}(2^{O(n)})$,

and $\tau \in \{\mathrm{d}, \mathrm{n}\}$ denotes deterministic (d) or nondeterministic (n) circuits.

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

Once we have such a PRG, the derandomization is obtained by cycling over all seeds and simulating the randomized procedure on the output of the PRG for each seed and then taking a majority vote of the simulation outcomes. 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 hence derandomizations that run in subexponential time t. At the "high end," linear-exponential circuit lower bounds yield logarithmic seed length and hence 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 of the above hardness conditions, or whether derandomization is *equivalent* to hardness.

Proving an equivalence would establish a "canonical form" of derandomization, namely through PRGs. It would show that if we can individually derandomize each procedure of the type considered, then we can derandomize them "all at once" – we do not need to know the particulars of a procedure in order to derandomize it. On the other hand, proving a non-equivalence would establish that the hardness-based PRG approach to derandomization is incomplete, i.e., that there are better avenues to derandomization. In fact, since the existence of PRGs is known to imply the hardness conditions that yield them, this would render incomplete *any* PRG-based approach – using hardness or not. Thus, another way of phrasing the question is whether or not the ability to derandomize implies the ability to do so through PRGs.

In recent years we have seen a number of results for (pr)BPP showing that derandomization implies hardness of some sort, although typically not strong enough so as to imply back the derandomization (e.g., Impagliazzo *et al.* (2002); Kabanets & Impagliazzo (2004); Kinne *et al.* (2012)). One exceptional example is an equivalence for a mild notion of derandomization for prBPP, implicit in Impagliazzo *et al.* (2002). They show that $\mathsf{NE} \not\subseteq \mathsf{P}/\mathrm{poly}$ *if and only if* every prBPP-problem can be decided in $\mathsf{NTIME}(2^{n^\epsilon})$ with advice of length n^ϵ for infinitely many input lengths, for arbitrarily small constant $\epsilon > 0$. Note that this notion of derandomization is indeed mild compared to what is conjectured to hold (namely a derandomization of prBPP in P): it uses nondeterminism, a subexponential amount of time, a subpolynomial amount of nonuniformity, and it is only required to work infinitely often. Mild though it may be, it is a nontrivial derandomization nonetheless, and one that is not known to hold. If it can be done at all, the Impagliazzo *et al.* (2002) result shows that it can be done in a canonical way – through PRGs.

In contrast to the class prBPP, not much is known regarding derandomization-to-hardness connections for Arthur-Merlin games. The only result in this direction is a "hybrid" connection that shows an implication from derandomizing prAM to *deterministic* circuit lower bounds (Aydınlıoğlu *et al.* 2011), 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 last question and obtain an analogue of the equivalence of hardness and derandomization from Impagliazzo *et al.* (2002) for the class prAM instead of prBPP.

Theorem 1.1 (Equivalence for Arthur-Merlin games).
The following are equivalent:

- *Every* prAM*-problem can be decided in* $\Sigma_2\mathsf{TIME}(2^{n^\epsilon})/n^\epsilon$ *for infinitely many input lengths, for every constant* $\epsilon > 0$.

- $\Sigma_2\mathsf{E} \not\subseteq \mathsf{NP}/\mathrm{poly}$.

Recall that prAM can be simulated in $\Pi_2\mathsf{P}$ (Babai & Moran 1988). Although plausible hardness assumptions imply simulations in NP, it is open whether a simulation in $\Sigma_2\mathsf{TIME}(t)$ is possible for subexponential t – even with subpolynomial advice, and even if the simulation is only required to succeed infinitely often. In this sense, Theorem 1.1 is a full analogue of the equivalence result in Impagliazzo *et al.* (2002) for prBPP. In both their equivalence and ours the derandomizations are mild in the same way: compared to the standard notion they use extra time, extra advice, and an extra level of nondeterminism (where each "extra" is quantified identically in both results), and they are only required to work infinitely often.

A further analogy follows from the fact that both prBPP and prAM have complete problems related to approximating the fraction of inputs that a given circuit accepts. For deterministic circuits, distinguishing between the case where the fraction is at least 2/3 and at most 1/3 is complete for prBPP. The corresponding problem for nondeterministic circuits is complete for prAM. If follows that the classes prBPP and prAM in the statements of Impagliazzo *et al.* (2002) and Theorem 1.1 can be replaced by their respective complete problems.

Theorem 1.1 implies that mildly derandomizing Arthur-Merlin games, or solving the above promise problem for nondeterministic circuits in "better than $\Sigma_2\mathsf{E}$", would yield new nondeterministic circuit lower bounds, namely that $\Sigma_2\mathsf{E}$ requires nondeterministic circuits of superpolynomial size. Unconditionally, it is known that the class one level up from $\Sigma_2\mathsf{E}$ in the exponential-time hierarchy, namely $\Sigma_3\mathsf{E}$, has this hardness; it is open whether $\mathsf{E}^{\Sigma_2\mathsf{P}}$ does. This situation mimics the one for deterministic circuits "one level down", continuing the parallel with Impagliazzo *et al.* (2002).

Perhaps more interestingly, Theorem 1.1 implies that mildly derandomizing prAM *in any way* implies that the same derandomization can be done *canonically*.

Corollary 1.2. *If* prAM *can be derandomized as in Theorem 1.1, then there exist pseudorandom generators that yield the same de-*

randomization.

A key step in the proof of Theorem 1.1 shows that 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 1.3 (High-end collapse results). *Suppose that we can decide* prAM *in* $\Sigma_2\mathsf{P}$. *Then*

(i) $\mathsf{PSPACE} \subseteq \mathsf{NP}/\operatorname{poly} \implies \mathsf{PSPACE} \subseteq \Sigma_2\mathsf{P}$, *and*

(ii) $\mathsf{coNP} \subseteq \mathsf{NP}/\operatorname{poly} \implies \mathsf{PH} \subseteq \mathsf{P}^{\Sigma_2\mathsf{P}}$.

Karp and Lipton (Karp & Lipton 1982) showed that if $\mathsf{NP} \subseteq \mathsf{P}/\operatorname{poly}$ then $\mathsf{PH} \subseteq \Sigma_2\mathsf{P}$. Yap's adaptation (Yap 1983) of the Karp-Lipton result gives that if $\mathsf{coNP} \subseteq \mathsf{NP}/\operatorname{poly}$ then $\mathsf{PH} \subseteq \Sigma_3\mathsf{P}$. Modulo the derandomization assumption, the second item of Theorem 1.3 improves Yap's result by "half a level". Another variant of the Karp-Lipton argument (attributed to Meyer in Karp & Lipton (1982)) states that if $\mathsf{PSPACE} \subseteq \mathsf{P}/\operatorname{poly}$ then $\mathsf{PSPACE} \subseteq \Sigma_2\mathsf{P}$. Relativizing the latter result yields the strongest collapse consequence of $\mathsf{PSPACE} \subseteq \mathsf{NP}/\operatorname{poly}$ known unconditionally, namely $\mathsf{PSPACE} \subseteq \Sigma_3\mathsf{P}$. The first item of Theorem 1.3 improves this collapse by one level, modulo the derandomization assumption. We refer to Section 2.4 for related work on more refined collapses.

We use a low-end version of the first collapse result in Theorem 1.3 to establish the forward direction of Theorem 1.1. 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 the same derandomization assumption implies $\mathsf{E}^{\Sigma_2\mathsf{P}} \not\subseteq \mathsf{NP}/\operatorname{poly}$.

A lower bound framework. Once we have established our collapse results, we develop the derandomization-to-hardness part of Theorem 1.1 as an instantiation of a generic framework that we formulate in this paper. Our framework provides a unified view of our result along with several well-known conditional lower bound results, in particular the one from Impagliazzo *et al.* (2002)

mentioned above, as well as the one from Kabanets & Impagliazzo (2004) that derandomizing polynomial identity testing implies arithmetic circuit lower bounds for NE.

Organization. In Section 2 we sketch the ideas behind our results, set up the lower bound framework, and discuss the relationship with earlier work. In Section 3 we introduce our notation for the formal development. In Section 4 we present the collapse results, and in Section 5 the main result.

2. Outline of the Arguments and Related Work

We now outline the proofs of Theorem 1.1 and Theorem 1.3. We focus on the most novel part of the proof of Theorem 1.1, 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 several known results of this ilk. The framework is based on Kannan's theorem (Kannan 1982) 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 Σ_2P, yielding the high-end collapse results of Theorem 1.3, although for the proof of Theorem 1.1 it suffices to have the weaker assumption that prAM can be simulated in $\cap_{\epsilon>0}$i.o.-Σ_2TIME$(2^{n^\epsilon})/n^\epsilon$.

2.1. 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 PSPACE $\subseteq$ MA (Lund *et al.* 1992). 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 (Shamir 1992). By assumption, there is a polynomial-size deterministic circuit D_{prover} that encodes the honest prover's strategy; i.e., given the 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 $L \in \mathsf{MA}$.

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 $\mathsf{MAM} = \mathsf{AM}$ (Babai & Moran 1988) rather than MA. By assumption, there exists a polynomial-size nondeterministic circuit C_{prover} implementing the honest prover's strategy; i.e., given the transcript of a message history and a bit b, C_{prover} 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, it could be that the nondeterministic circuit C' sent by Merlin has accepting and rejecting computation paths on every input, which would allow 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 $\Sigma_2\mathsf{P}$ 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 $\Sigma_2\mathsf{P}$: 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 $\Sigma_2\mathsf{P}$, a collapse of PSPACE to $\Sigma_2\mathsf{P}$ follows.

2.2. 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 $\Pi_2\mathsf{P}$-predicate of the form

$$(\forall u)(\exists v)\varphi(u,v), \tag{2.1}$$

where φ is a Boolean formula, and translate it into an equivalent $\Sigma_2\mathsf{P}$-predicate.

One way to construct the $\Sigma_2\mathsf{P}$-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 inner existential phase of the original $\Pi_2\mathsf{P}$-predicate into a deterministic one, effectively eliminating one quantifier alternation. Hence, we obtain an equivalent predicate that reads as

$$(\exists D)\left[\texttt{correct}(D) \wedge (\forall u) D(\text{“}(\exists v)\varphi(u,v)\text{”}) = 1\right]. \tag{2.2}$$

Exploiting the self-reducibility of SAT, $\texttt{correct}(D)$ can be expressed as a coNP-predicate. This way ((2.2)) becomes a $\Sigma_2\mathsf{P}$-predicate.

Let us now turn to our setting and try to achieve the same collapse under the assumption that $\overline{\text{SAT}}$ has *nondeterministic* circuits of polynomial size, by attempting to transform a $\Pi_2\mathsf{P}$-predicate of the form ((2.1)) into an equivalent $\Sigma_2\mathsf{P}$-predicate. Mimicking the above proof, we use an existential quantifier to guess a nondeterministic circuit C, check its correctness for $\overline{\text{SAT}}$, and then feed the existential phase of the $\Pi_2\mathsf{P}$-predicate into C. The latter transforms the existential phase into an equivalent universal phase, which can be merged with the initial universal phase of the original $\Pi_2\mathsf{P}$-predicate. This complementation gives us a new equivalent predicate of the form

$$(2.3) \qquad (\exists C)\,[\texttt{correct}(C) \wedge (\forall u)(\forall w) C(\text{“}(\exists v)\varphi(u,v)\text{”}, w) = 0]\,,$$

where $C(y,w)$ denotes the deterministic output of C on input y and nondeterministic guess bits w.

In fact, it suffices that the predicate $\texttt{correct}(C)$ checks the *completeness* of C for $\overline{\text{SAT}}$ (that C accepts every unsatisfiable formula) without explicitly checking the *soundness* of C for $\overline{\text{SAT}}$ (that C only accepts unsatisfiable formulas). However, while soundness can be tested in coNP, completeness seems to require $\Pi_2\mathsf{P}$. This turns ((2.3)) into a $\Sigma_3\mathsf{P}$-predicate rather than a $\Sigma_2\mathsf{P}$-predicate. Note that obtaining an equivalent $\Sigma_3\mathsf{P}$-predicate is trivial since we started from a $\Pi_2\mathsf{P}$-predicate. If we start from a Π_3^{P}-predicate instead, an analogous transformation yields an equivalent $\Sigma_3\mathsf{P}$-predicate, implying a collapse of the polynomial-time hierarchy to the third level (this is Yap's theorem (Yap 1983)). If we could check for completeness in $\Sigma_2\mathsf{P}$, 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 $\overline{\text{SAT}}$ "half a level" up from $\Sigma_2\mathsf{P}$, namely in $\mathsf{P}^{\Sigma_2\mathsf{P}}$ (just like we could if we knew how to check completeness in $\Sigma_2\mathsf{P}$). This collapses the polynomial-time hierarchy down to $\mathsf{P}^{\Sigma_2\mathsf{P}}$, 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 & Sipser (1986) 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 $\Sigma_2\mathsf{P}$, 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 $\mathsf{P}^{\Sigma_2\mathsf{P}}$, 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 $\Sigma_2\mathsf{P}$-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 $\Sigma_2\mathsf{P}$-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, $\overline{\text{SAT}}$ 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 $\overline{\text{SAT}}$ 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 $\overline{\text{SAT}}$ 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 $\overline{\text{SAT}}$; it suffices to check the soundness of C' for $\overline{\text{SAT}}$. Since the latter can be done in coNP, we can guess and check the circuit C' in $\Sigma_2\text{P}$.

To recapitulate, we want to find a sound nondeterministic circuit $\tilde{C}_1$ for $\overline{\text{SAT}}$ 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 $\phi_{C_1}(x,\cdot)$ is satisfiable for a particular x iff C_1 accepts x. 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 $\overline{\text{SAT}}$. By our derandomization assumption that $\text{prAM} \subseteq \Sigma_2\text{P}$, these queries can be made to a $\Sigma_2\text{P}$-oracle, which allows us to construct $\tilde{C}_1$ in $\text{P}^{\Sigma_2\text{P}}$. 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 $\overline{\text{SAT}}$, 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 $\Pi_2\mathsf{P}$) 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 $\mathsf{P}^{\Sigma_2\mathsf{P}}$ 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 $\mathsf{P}^{\Sigma_2\mathsf{P}}$ follows.

2.3. The lower bound framework. In both our main equivalence for prAM and the one for prBPP due to Impagliazzo *et al.* (2002), the proof of the forward direction – from derandomization to hardness – can be cast as an instantiation of a generic framework that we provide in this paper. This framework enables a unified view of several results in the literature that obtain circuit lower bounds assuming the existence of efficient algorithms of some sort – in particular, assuming nontrivial derandomizations. We first present the framework and then describe some instantiations.

Our goal is to show, under some derandomization assumption, that some class $\mathcal{C}$ does not have type-τ circuits of size $O(s(n))$. In a first reading the reader can take $\mathcal{C} = \mathsf{NE}$, $s(n) = n^k$ for some fixed constant k, and τ as deterministic; more generally the argument also applies to other linear-exponential classes $\mathcal{C}$, resource bounds $s(n)$, and to nondeterministic and arithmetic[1] circuits.

The proof goes by contradiction and consists of the following ingredients. Suppose that the lower bound fails to hold, i.e., that $\mathcal{C}$ has τ-circuits of size $O(s)$.

(i) *Collapse.* Use the hypothesis that $\mathcal{C}$ has τ-circuits of size $O(s)$ to show that the entire polynomial time hierarchy PH, and more generally $\mathsf{DTIME}^{\mathsf{PH}}(\mathrm{poly}(s))$, can be decided in some randomized class $\mathcal{R}$.

(ii) *Simulation.* Use the derandomization assumption to show that every language in $\mathcal{R}$ can be decided in $\mathcal{C}$ with linear

[1] We say that a class $\mathcal{C}$ has arithmetic circuits of size $s(n)$ if for every language $L \in \mathcal{C}$, the multilinear extension of L over $\mathbb{Z}$ is computable by arithmetic circuits of size $s(n)$. The multilinear extension of L over $\mathbb{Z}$ at length n is the unique n-variate polynomial over $\mathbb{Z}$ that has degree at most one in each variable and agrees with the characteristic function of L on $\{0,1\}^n$.

advice for infinitely many input lengths.

(iii) *Kannan's theorem.* Kannan (1982) showed that for any resource bound s, the class $\mathsf{DTIME}^{\Sigma_3\mathsf{P}}(O(s \log s))$ contains a language that for all but finitely many input lengths cannot be decided by τ-circuits of size $O(s)$.[2]

By combining the first two ingredients, we conclude that every language in $\mathsf{DTIME}^{\mathsf{PH}}(\mathrm{poly}(s))$ is computable by τ-circuits of size $O(s)$ on infinitely many input lengths. This contradicts the third item and concludes the proof. We summarize with notation:

$$
\begin{aligned}
\mathcal{C} \subseteq & \tau\mathsf{SIZE}(O(s(n))) \\
& \Longrightarrow \mathsf{DTIME}^{\mathsf{PH}}(\mathrm{poly}(s(n))) \subseteq \mathcal{R} \\
& \qquad \text{(by collapse)} \\
& \Longrightarrow \mathsf{DTIME}^{\mathsf{PH}}(\mathrm{poly}(s(n))) \subseteq \mathsf{i.o.}\text{-}\mathcal{C}/n \\
& \qquad \text{(by simulation)} \\
& \Longrightarrow \mathsf{DTIME}^{\mathsf{PH}}(\mathrm{poly}(s(n))) \subseteq \mathsf{i.o.}\text{-}\tau\mathsf{SIZE}(O(s(n)) \\
& \qquad \text{(by the hypothesis)} \\
& \Longrightarrow \text{contradiction} \\
& \qquad \text{(by Kannan's Theorem)}
\end{aligned}
$$

In executing the above framework, a key part of the effort lies in identifying an appropriate randomized class $\mathcal{R}$ that meets two competing requirements: On one hand, $\mathcal{R}$ must be powerful enough to simulate the entire polynomial-time hierarchy assuming $\mathcal{C}$ has small circuits. On the other hand, under the derandomization hypothesis $\mathcal{R}$ must allow infinitely-often simulations in $\mathcal{C}$ with linear advice. The two constraints are at odds with each other and satisfying both of them amounts to a balancing act of sorts, but is manageable in a number of settings.

Instantiations. The above generic framework captures several results from the literature of the form "derandomization implies

[2] Kannan's result is originally stated for deterministic circuits only, but is straightforward to generalize.

circuit lower bounds." The first two lines in Figure 2.1 list the instantiations corresponding to the results mentioned in the introduction. The last two lines in Figure 2.1 represent our results. We now expand on those instantiations.

$\mathcal{C}$	τ	$\mathcal{R}$	derandomization hypothesis
NE	d	MA	$\mathsf{prBPP} \subseteq \cap_{\epsilon>0}\mathsf{i.o.\text{-}NTIME}(2^{n^\epsilon})/n^\epsilon$
NE	a	$\mathsf{MA}_{\mathrm{PIT}}$	$\mathrm{PIT} \in \cap_{\epsilon>0}\mathsf{i.o.\text{-}NTIME}(2^{n^\epsilon})/n^\epsilon$
Σ_2E	n	M(AM||coNP)	$\mathsf{prAM} \subseteq \cap_{\epsilon>0}\mathsf{i.o.\text{-}}\Sigma_2\mathsf{TIME}(2^{n^\epsilon})/n^\epsilon$
$\mathsf{E}^{\Sigma_2\mathsf{P}}$	n	$\mathsf{P}^{\mathsf{prM(AM\|\|coNP)}}$	$\mathsf{prAM} \subseteq \cap_{\epsilon>0}\mathsf{i.o.\text{-}}\Sigma_2\mathsf{TIME}(2^{n^\epsilon})/n^\epsilon$

Figure 2.1: Instantiations of the framework to show that a class $\mathcal{C}$ requires superpolynomial circuits of type τ assuming a derandomization hypothesis and using an intermediate randomized class $\mathcal{R}$.

- The first line in Figure 2.1 is implicit in Impagliazzo *et al.* (2002). It derives the lower bound $\mathsf{NE} \not\subseteq \mathsf{SIZE}(\mathrm{poly}(n))$ under the hypothesis that prBPP can be mildly derandomized. To obtain the result from the framework, we set $\mathcal{C} = \mathsf{NE}$, $\tau = \mathrm{d}$, and $s(n) = n^k$ for an arbitrary constant k.

  - *Collapse.* Since $\mathsf{NE} \subseteq \mathsf{SIZE}(n^k)$ implies that $\mathsf{P}^{\#\mathsf{P}} \subseteq \mathsf{SIZE}(\mathrm{poly}(n))$, we can use the Karp-Lipton style collapse $\mathsf{P}^{\#\mathsf{P}} \subseteq \mathsf{SIZE}(\mathrm{poly}(n)) \implies \mathsf{PH} \subseteq \mathsf{MA}$ (Lund *et al.* 1992).

  - *Simulation.* Under the derandomization hypothesis, the class MA allows simulations in NE with linear advice infinitely often. This follows because by the hypothesis Arthur can be simulated on infinitely many input lengths, and a linear amount of advice suffices to specify one of those good input lengths if one exists nearby.

  Thus, we set $\mathcal{R}$ as the class MA of languages decidable by Merlin-Arthur games, and conclude by contradiction that $\mathsf{NE} \not\subseteq \mathsf{SIZE}(n^k)$. Since k is an arbitrary constant, this shows

that $\mathsf{NE} \not\subseteq \mathsf{SIZE}(\mathrm{poly}(n)) \doteq \cup_k \mathsf{SIZE}(n^k)$ as the condition $\mathsf{NE} \subseteq \mathsf{SIZE}(\mathrm{poly}(n))$ is equivalent to $(\exists k)\, \mathsf{NE} \subseteq \mathsf{SIZE}(n^k)$.[3]

- The second line in Figure 2.1 captures the sequel to Impagliazzo *et al.* (2002), namely Kabanets & Impagliazzo (2004), which states that if polynomial identity testing (i.e., the language PIT of pairs of arithmetic circuits over $\mathbb{Z}$ that are functionally equivalent) can be derandomized, then the same lower bound as in Impagliazzo *et al.* (2002) follows – except for arithmetic circuits rather than Boolean circuits: $\mathsf{NE} \not\subseteq \mathsf{ASIZE}(\mathrm{poly}(n))$.[4] To obtain this result from the framework, similar to the argument above for Impagliazzo *et al.* (2002), we set $\mathcal{C} = \mathsf{NE}$, $\tau = \mathrm{a}$, and $s(n) = n^k$ for an arbitrary constant k.

  - *Collapse.* We use the fact that $\mathsf{NE} \subseteq \mathsf{ASIZE}(n^k)$ implies that $\mathsf{P}^{\#\mathsf{P}} \subseteq \mathsf{ASIZE}(\mathrm{poly}(n))$, and then apply the collapse $\mathsf{P}^{\#\mathsf{P}} \subseteq \mathsf{ASIZE}(\mathrm{poly}(n)) \implies \mathsf{PH} \subseteq \mathsf{MA}_{\mathrm{PIT}}$, where $\mathsf{MA}_{\mathrm{PIT}}$ represents the variant of Merlin-Arthur games in which Arthur uses his randomness solely to decide a single instance of PIT.
  - *Simulation.* Along the same lines as the first instantiations, under the derandomization hypothesis, the PIT query can alternately be decided by a computation in NE with small advice infinitely often, which allows us to simulate $\mathsf{MA}_{\mathrm{PIT}}$ in $\mathsf{i.o.\text{-}NE}/n$.

  Thus, we set $\mathcal{R} = \mathsf{MA}_{\mathrm{PIT}}$, and proceed in the same way as for the Impagliazzo *et al.* (2002) instantiation.

- Our main result (Theorem 1.1) is obtained via our first collapse involving PSPACE, and our weaker but relativizing result via our second collapse involving coNP. The simulations

[3] This follows from a padding argument and the existence of a complete language for NE under linear reductions.

[4] The original lower bound in Kabanets & Impagliazzo (2004) is slightly stronger, but is more complicated to state. Our framework also captures this result; we opt for a simpler statement here that facilitates comparison.

follow in a straightforward way. The classes $\mathcal{R}$ used are technical augmentations of Arthur-Merlin protocols; they are implicit in the proof of Theorem 1.3, and are explicitly defined in Section 3. We develop the arguments in Section 5.1.

As a side remark we point out that the first two lines in Figure 2.1 represent proofs that are simpler than the original ones; in addition they scale better and yield somewhat stronger results. For example, the original proof of the first line uses the collapse $\mathsf{NE} \subseteq \mathsf{SIZE}(n^k) \implies \mathsf{NE} \subseteq \mathsf{MA}$ (Impagliazzo *et al.* 2002), which relies on the nondeterministic time hierarchy theorem, a hardness-based PRG construction, as well as multi-prover interactive proofs. The framework derives the same conclusion with the simpler collapse technology involving #P mentioned above. In addition to being simpler, the latter collapse scales better: It holds in general that $\mathsf{P}^{\#\mathsf{P}} \subseteq \mathsf{SIZE}(s'(n)) \implies \mathsf{DTIME}^{\mathsf{PH}}(s'(n)) \subseteq \mathsf{MATIME}(s'(s'(n)))$, where $s'(n)$ denotes $\mathrm{poly}(s(n))$. Under the derandomization hypothesis $\mathsf{prBPP} \subseteq \mathsf{NTIME}(t'(n)))$, the scaled collapse yields the lower bound $\mathsf{NE} \not\subseteq \mathsf{SIZE}(O(s(n)))$, for any resource bounds $t(n)$ and $s(n)$ satisfying $t'(s'(s'(n)) = 2^{O(n)}$. In fact, the same conclusion holds for $\mathsf{NE} \cap \mathsf{coNE}$ rather than NE, as the collapse can be strengthened to $\mathsf{MATIME}(\cdot) \cap \mathsf{coMATIME}(\cdot)$. A similar analysis is worked out in Aaronson & van Melkebeek (2011); we refer to that paper for more details.

2.4. Related work. In the setting of decision problems, Goldreich (2011a,b) showed that the standard notion of derandomization for prBPP (deterministic simulations without advice that work for all but finitely many inputs) is equivalent to the existence of so-called "targeted PRGs," which are PRGs that have access to the input or even to the circuit that models the randomized computation on the given input. For standard PRGs (which are oblivious to the input and only depend on the running time of the randomized computation) he showed an equivalence between their existence and derandomizations of prBPP in an average-case setting: deterministic simulations without advice that may err on some inputs but such that generating erroneous inputs is computationally difficult. In both cases the equivalence with circuit lower bounds

remains open.

In the setting of Arthur-Merlin games, Gutfreund *et al.* (2003) suggest an approach to prove that AM can be simulated in the class Σ_2SUBEXP. Theorem 1.1 implies that if the approach works for prAM then it would yield new circuit lower bounds.

Using Kannan's argument to get circuit lower bounds from a derandomization assumption for prAM was carried out in Aydınlıoğlu *et al.* (2011). 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 technique of using a prAM-oracle to iteratively construct a sound circuit with rapidly increasing completeness, appears in Chakaravarthy & Roy (2011). Using this technique they show that PH collapses to $\mathsf{P}^{\mathsf{prAM}}$, under the classical Karp-Lipton assumption that NP $\subseteq$ P/ poly.

Be it for diagonalization as in Aydınlıoğlu *et al.* (2011), or for finding a circuit as in Aydınlıoğlu *et al.* (2011); Chakaravarthy & Roy (2011) 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 $\overline{\text{SAT}}$ 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.

In Goldreich (2011a,b) a prBPP-oracle is used to construct targeted PRGs. The constructions also involve approximately maximizing a quality measure f; in this case $f(y)$ 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 Goldreich's is that in the latter work f can be additively approximated using a prBPP-oracle, whereas in the former f is multiplicatively approximated using a prAM-oracle.

Regarding our high-end collapse result involving the classes coNP and NP/ poly, at a more refined level of granularity the strongest unconditional collapse consequence of coNP $\subseteq$ NP/ poly is that PH collapses to $\mathsf{S}_2\mathsf{P}^{\mathsf{NP}}$ (Cai *et al.* 2005), a class that con-

tains $\mathsf{P}^{\Sigma_2\mathsf{P}}$ but is not known to equal it. Similarly, the strongest unconditional collapse consequence of $\mathsf{PSPACE} \subseteq \mathsf{NP}/\,\mathrm{poly}$ is that $\mathsf{PSPACE} \subseteq \mathsf{S}_2\mathsf{P}^{\mathsf{NP}}$. As a consequence, it is known that the linear-exponential analog of $\mathsf{S}_2\mathsf{P}^{\mathsf{NP}}$ requires nondeterministic circuits of superpolynomial size (Cai *et al.* 2005).

The alternate proof of the result of Kabanets & Impagliazzo (2004) yielded by our framework in Figure 2.1 was observed earlier in Aaronson & van Melkebeek (2011). To facilitate comparison with other results, Figure 2.1 lists a weaker version of the original Kabanets & Impagliazzo (2004) result; Aaronson & van Melkebeek (2011) gives a refined analysis that yields the original result.

3. 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., Arora & Barak (2009)), except that in the remainder of this paper the term "circuit" always refers to a Boolean *nondeterministic* circuit, unless stated otherwise.

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 $\Pi_N \subseteq \Pi'_N$. To *decide* a promise problem Π is to correctly determine, for all inputs $x \in \Pi_Y \cup \Pi_N$, which of Π_Y or Π_N contains x. 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 $\mathsf{P}^{\Pi'}$ 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 Goldreich (2006).

Arthur-Merlin protocols and augmentations. prAM represents the class of promise problems Π for which there exists a constant c and a language $L \in \mathsf{P}$ such that for every input x

(3.1)

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

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

$$\begin{array}{rrcl} \textit{[completeness]} & x \in \Pi_Y & \Rightarrow & (\exists y)\,(\langle x, y\rangle \in \Gamma_Y \ \wedge\ \langle x, y\rangle \in V)\,, \\ \textit{[soundness]} & x \in \Pi_N & \Rightarrow & (\forall y)\,(\langle x, y\rangle \in \Gamma_N \ \vee\ \langle x, y\rangle \notin V)\,, \end{array}$$

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

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

word "verifier" connotes restricted computational power, it may be helpful to think of Henry as having private access to a second all-powerful prover who competes with Merlin by providing a certificate that is to serve as a counter-certificate to Merlin's initial message.)

We point out that our use of the symbol "||" in $\mathsf{prM}(\mathsf{AM}||\mathsf{coNP})$ can be viewed as a binary operator $\wedge$ on (classes of) promise problems. For two given promise problems $\Pi = (\Pi_Y, \Pi_n)$ and $\Pi' = (\Pi'_Y, \Pi'_N)$, one can define $\Pi \wedge \Pi'$ as the promise problem $(\Pi_Y \cap \Pi'_Y, \Pi_N \cup \Pi'_N)$. One can also extend the definition of known unary operators from (classes of) languages to (classes of) promise problems in the natural way: $\exists$ capturing nondeterminism, $\forall$ capturing conondeterminism, and BP capturing randomness with bounded error (Schöning 1989). Using that notation, we can write the class $\mathsf{prM}(\mathsf{AM}||\mathsf{coNP})$ as $\exists\,(\mathrm{BP}\,\exists\ \wedge\ \forall)\,\mathsf{P}$.

The following basic fact equates a derandomization assumption on Arthur-Merlin protocols to the same one on augmented Arthur-Merlin protocols.

PROPOSITION 3.3. $\mathsf{prAM} \subseteq \Sigma_2\mathsf{P}$ *if and only if* $\mathsf{prM}(\mathsf{AM}||\mathsf{coNP}) \subseteq \Sigma_2\mathsf{P}$.

PROOF. The forward direction follows because replacing Γ_Y in Definition 3.2 by a $\Sigma_2\mathsf{P}$-predicate and Γ_N by its complement, turns Π_Y into $\Sigma_2\mathsf{P}$-predicate and Γ_N into its complement. The backward direction is trivial as $\mathsf{prAM} \subseteq \mathsf{prM}(\mathsf{AM}||\mathsf{coNP})$. □

Proposition 3.3 generalizes to weaker derandomizations; in particular its analogue for mild derandomizations is critical in establishing Theorem 1.1 (and is the content of Lemma 5.1).

Nondeterministic 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 $O(s \log s)$.

4. Collapse Results

In this section we establish our collapse result (Theorem 1.3), which uses a derandomization assumption for prAM. In fact, we prove an unconditional collapse result involving the class of augmented Arthur-Merlin protocols introduced in Definition 3.2, from which Theorem 1.3 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 1.3) and then do the same for coNP instead of PSPACE (corresponding to the second part in Theorem 1.3).

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

THEOREM 4.1.
If $\mathsf{PSPACE} \subseteq \mathsf{NP}/\operatorname{poly}$ *then* $\mathsf{PSPACE} \subseteq \mathsf{M}(\mathsf{AM}||\mathsf{coNP})$.

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 $\ell(n) = \operatorname{poly}(n)$.

By the assumption that $\mathsf{PSPACE} \subseteq \mathsf{NP}/\operatorname{poly}$, 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 nondeterministic circuit C' of polynomial size, purported to compute L_{prover} at length $\ell(n)$. The coNP-verifier V checks that C' is nowhere "ambiguous", i.e., V checks that 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 $\ell(n)$) 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 Henry 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 Henry. This means that C' is nowhere ambiguous, 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 1.3 follows immediately from Theorem 4.1:

Proof (of part (i) of Theorem 1.3). By Proposition 3.3, if prAM can be simulated in Σ_2P then so can prM(AM||coNP). If in addition PSPACE ⊆ NP/poly, Theorem 4.1 implies that PSPACE ⊆ M(AM||coNP) ⊆ Σ_2P. □

4.2. 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 4.2. *If* coNP ⊆ NP/poly *then* Σ_3P ⊆ $\mathsf{P}^{\mathsf{prM(AM||coNP)}}$.

The second part of Theorem 1.3 follows immediately from Theorem 4.2 under its derandomization assumption for prAM, in a relativizable way.

Proof (of part (ii) of Theorem 1.3). By Proposition 3.3, if prAM can be simulated in the class Σ_2P then so can prM(AM||coNP). If in addition coNP ⊆ NP/poly, Yap's theorem (Yap 1983) and Theorem 4.2 imply that PH ⊆ Σ_3P ⊆ $\mathsf{P}^{\mathsf{prM(AM||coNP)}}$ ⊆ $\mathsf{P}^{\Sigma_2\mathsf{P}}$. □

We now argue Theorem 4.2. Assume that $\overline{\text{SAT}}$ has nondeterministic circuits of size $s(n)$, where s is some polynomial. Following the outline of Section 2.2, with the aid of a $\mathsf{prM}(\mathsf{AM}||\mathsf{coNP})$-oracle, we construct a circuit of size $O(n \cdot s(n))$ that correctly decides $\overline{\text{SAT}}$ 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. Recall that a circuit with n inputs is sound/complete for a set $S \subseteq \{0,1\}^n$ if it accepts only/all inputs in S.

To measure the improvement in each step, we consider the following function $\mu : \{0,1\}^* \times \{0,1\}^* \to \mathbb{N}$, 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.

$$\mu(C, \tilde{C}) = \begin{cases} |C^{-1}(0) \cap \tilde{C}^{-1}(1)| & \text{if } \tilde{C} \text{ is sound for } \overline{\text{SAT}} \\ 0 & \text{otherwise.} \end{cases} \tag{4.3}$$

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, μ counts the number of instances of $\overline{\text{SAT}}$ 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 $\mu(C, \tilde{C})$. For this task we only need access to an approximation of μ to within a constant multiplicative factor. This motivates the following definition.

Definition 4.4. *For a function $f : \{0,1\}^* \times \{0,1\}^* \to \mathbb{N}$, A_f denotes the promise problem (Y_f, N_f) such that*

$$\begin{array}{rcl} Y_f & = & \{\langle x, y, a, \epsilon \rangle \,:\, f(x,y) \geq a\} \\ N_f & = & \{\langle x, y, a, \epsilon \rangle \,:\, f(x,y) < (1-\epsilon)a\}, \end{array} \tag{4.5}$$

where a is a nonnegative integer in binary and $1/\epsilon$ is a positive integer in unary.

The crux of our argument is the following lemma.

LEMMA 4.6. *Let μ be the function defined by* ((4.3)), *and A_μ the promise problem given by* ((4.5)). *If* coNP $\subseteq$ NP/ poly *then* $A_\mu \in$ prM(AM||coNP).

PROOF. We follow the outline from Section 2.2, but cast the resulting algorithm for A_μ in terms of an augmented Arthur-Merlin protocol on input $\langle C, \tilde{C}, a, \epsilon\rangle$.

Since the coNP-verifier Henry can check whether $\tilde{C}$ is sound for $\overline{\text{SAT}}$, if suffices to construct an augmented Arthur-Merlin protocol for A_ν, where ν is the relaxation of μ defined by

$$\nu(C, \tilde{C}) \doteq |C^{-1}(0) \cap \tilde{C}^{-1}(1)|.$$

Goldwasser & Sipser (1986) showed that for every predicate $L \in$ NP and function

(4.7)

$$f : \{0,1\}^* \times \{0,1\}^* \to \mathbb{N} : (u, v) \mapsto |\{w \in \{0,1\}^* : \langle u, v, w\rangle \in L\}|,$$

the promise problem A_f is decidable by an Arthur-Merlin protocol. Note that the function ν is of the form ((4.7)), except that the underlying predicate $L \doteq \{\langle C, \tilde{C}, w\rangle : C(w) = 0 \wedge \tilde{C}(w) = 1\}$ syntactically looks like the difference of two NP languages rather than a mere NP language. More precisely, $L = L_0 \cap L_1$, where

$$L_0 \doteq \{\langle C, v, w\rangle : C(w) = 0\} \text{ and } L_1 \doteq \{\langle u, \tilde{C}, w\rangle : \tilde{C}(w) = 1\}.$$

Note that $L_0 \in$ coNP and $L_1 \in$ NP. To remedy this issue, we invoke the hypothesis coNP $\subseteq$ NP/ poly and use the added power afforded by Henry in an augmented protocol as follows.

Let 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 $\phi_C(w, z)$ of size n' such that for all $w \in \{0,1\}^n$ and $z \in \{0,1\}^{n'}$, $\phi_C(w, z)$ evaluates to true iff z represents an accepting computation of C on input w. In other words, for all $w \in \{0,1\}^n$, $C(w) = 0$ iff $\phi_C(w) \in \overline{\text{SAT}}$. Now define $L' = L'_0 \cap L_1$, where

$$L'_0 \doteq \{\langle\langle C, C'\rangle, v, w\rangle : C'(\phi_C(w)) = 1\}.$$

Intuitively, L'_0 is L_0 "according to" the circuit C', for any given C'. Indeed, let $L_0^{C'}$ denote the restriction of L'_0 by fixing its parameter

C', i.e., the inputs $\langle C, \tilde{C}, w\rangle$ such that $\langle\langle C, C'\rangle, \tilde{C}, w\rangle \in L'_0$. We have that if C' computes $\overline{\text{SAT}}$ correctly, then $L_0^{C'}$ is identical to L_0 on the relevant inputs, i.e., an input $\langle C, v, w\rangle$ is in $L_0^{C'}$ iff it is in L_0. More generally, whenever C' is *sound* for $\overline{\text{SAT}}$ we have that an input $\langle C, v, w\rangle$ is in $L_0^{C'}$ iff it is in L_0. Observe that L'_0 is an NP-predicate, and therefore so is L'.

Now suppose $\mathsf{coNP} \subseteq \mathsf{NP}/\,\text{poly}$. Then there is a circuit $C'_{\overline{\text{SAT}}}$ of size $\text{poly}(n')$ for $\overline{\text{SAT}}$ on inputs of length n'. This suggests the following augmented Arthur-Merlin protocol for A_ν on input $\langle C, \tilde{C}, a, \epsilon\rangle$.

Merlin sends as his initial message a circuit C' of size $\text{poly}(n')$ on n' inputs, purported to be a circuit for $\overline{\text{SAT}}$ at length n'. Henry checks that C' is sound for $\overline{\text{SAT}}$,[5] and Arthur engages in a protocol with Merlin for the promise problem $A_{\nu'}$, where ν' is the variant of ν defined by the underlying predicate L' instead of L. Since L' is an NP-predicate, ν' exactly matches the Goldwasser-Sipser format ((4.7)), and hence an Arthur-Merlin protocol for $A_{\nu'}$ exists.

To argue the correctness of the protocol for A_ν, let $L_0^{C'}$ denote, as before, the restriction of L'_0 by fixing its parameter C', and let $L^{C'}$ and ν' denote the analogous notions for L' and ν', respectively.

We begin with the completeness of the protocol. Suppose that $\nu(C, \tilde{C}) \geq a$. In order to make both verifiers accept, Merlin can send as his initial message a polynomial-size circuit $C'_{\overline{\text{SAT}}}$ for $\overline{\text{SAT}}$ at length n', which exists by the hypothesis that $\mathsf{coNP} \subseteq \mathsf{NP}/\,\text{poly}$. Since $C'_{\overline{\text{SAT}}}$ is sound for $\overline{\text{SAT}}$, Henry accepts. As for Arthur, since $C'_{\overline{\text{SAT}}}$ correctly computes $\overline{\text{SAT}}$ at length n', we have that $L_0^{C'_{\overline{\text{SAT}}}}$ is identical to L_0 on the relevant instances: an input $\langle C, v, w\rangle$ is in $L_0^{C'_{\overline{\text{SAT}}}}$ iff it is in L_0. Therefore, $L^{C'_{\overline{\text{SAT}}}}$ is identical to L on the relevant instances: an input $\langle C, \tilde{C}, w\rangle$ is in $L^{C'_{\overline{\text{SAT}}}}$ iff it is in L. It follows that $\nu^{C'}(C, \tilde{C}) = \nu(C, \tilde{C}) \geq a$. The completeness of the Arthur-Merlin protocol for $A_{\nu'}$ then guarantees that Arthur can be convinced with high probability.

To argue soundness, suppose that $\nu(C, \tilde{C}) < a(1 - \epsilon)$. First, if Henry accepts, then Merlin must have sent a sound circuit C'

[5] Formally, referring to Definition 3.2, we set $V \doteq \{\langle u, C'\rangle : (\forall v)[C'(v) \text{ accepts } \Rightarrow v \in \overline{\text{SAT}}]\}$.

for $\overline{\text{SAT}}$ in the first round. In that case $L_0^{C'} \subseteq L_0$, and hence $\nu^{C'}(C, \tilde{C}) \leq \nu(C, \tilde{C}) < a(1-\epsilon)$. By the soundness of the Arthur-Merlin protocol for $A_{\nu'}$, this means that Arthur rejects with high probability. Thus, whenever Henry accepts, Arthur rejects with high probability. This completes the proof. □

Lemma 4.6 allows us to efficiently improve the completeness of a sound but incomplete circuit C for $\overline{\text{SAT}}$ when given oracle access to a language that agrees with A_μ by finding a circuit $\tilde{C}$ that approximately maximizes $\mu(C, \tilde{C})$, and outputting $C \vee \tilde{C}$. The approximate maximization can be done using the following generic lemma.

LEMMA 4.8. *Let $f : \{0,1\}^* \times \{0,1\}^* \to \mathbb{N}$ be such that $f(x,y) \leq 2^{|x|^c}$ for some constant c, for all inputs x, y. If $A_f \in \mathsf{prM}(\mathsf{AM}||\mathsf{coNP})$, where A_f denotes the promise problem defined by ((4.5)), then there exists a promise problem $\Pi \in \mathsf{prM}(\mathsf{AM}||\mathsf{coNP})$ such that the following holds for any language L that agrees with Π. On input a binary string x, and $1/\epsilon$ in unary, we can find, in deterministic polynomial time with oracle access to L, a value $\tilde{y} \in \{0,1\}^m$ such that*

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

PROOF (of Lemma 4.8). 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'. Observe that the assumption $A_f \in \mathsf{prM}(\mathsf{AM}||\mathsf{coNP})$ implies that $\Pi \doteq A_g \in \mathsf{prM}(\mathsf{AM}||\mathsf{coNP})$. 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 $\Pi \doteq A_g$ and let L denote any language that agrees with the promise problem Π.

We run the search for $\tilde{y}$ in two phases. In the first phase we find an approximation $\tilde{a}$ to $a^* \doteq \max_{y \in \{0,1\}^m} f(x,y)$ satisfying

$$(1-\eta)\tilde{a} \leq a^* \leq \tilde{a} \tag{4.9}$$

for some value η that depends on ϵ and will be set later. To do so, we make use of the predicate

$$P(a) \doteq (a \leq 2^{|x|^c}) \wedge (\langle x, \lambda, a, \eta \rangle \in L), \tag{4.10}$$

where λ denotes the empty string, η is as just above, 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, $P(2^{|x|^c}+1)$ fails by definition. We run a binary search for an integer value $\tilde{a} \in [0, 2^{|x|^c}]$ such that $P(\tilde{a})$ holds and $P(\tilde{a}+1)$ fails. Observe that any such $\tilde{a}$ is guaranteed to satisfy ((4.9)). Indeed, if $P(\tilde{a})$ holds, then by ((4.10)) we have $\langle x, \lambda, \tilde{a}, \eta \rangle \in L$, and since L agrees with A_g, by ((4.5)) we in turn have $\langle x, \lambda, \tilde{a}, \eta \rangle \notin N_g$, i.e., $(1-\eta)\tilde{a} \leq g(x, \lambda) = a^*$. This argues the first half of ((4.9)). The second half follows along similar lines: If $P(\tilde{a}+1)$ fails, then by ((4.10)) and by the assumption on the range of f we have $g(x, \lambda) < \tilde{a}+1$. This concludes the first phase.

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

$$g(x, \tilde{y}_{1\ldots i}) \geq \tilde{a}_i, \tag{4.11}$$

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 $\tilde{a}_0 = (1-\eta)\tilde{a}$, and for $i = 0, \ldots, m-1$ we extend the prefix of $\tilde{y}$ of length i to length $i+1$ as follows. By ((4.11)) we know that for at least one choice of $\tilde{y}_{i+1} \in \{0,1\}$ we must have $g(x, \tilde{y}_{1\ldots i+1}) \geq \tilde{a}_i$. Since L agrees with A_g, using ((4.5)) this means that

$$\langle x, \tilde{y}_{1\ldots i+1}, \tilde{a}_i, \eta \rangle \in L. \tag{4.12}$$

We further know that for any choice of $\tilde{y}_{i+1} \in \{0,1\}$ satisfying ((4.12)), we must have $g(x, \tilde{y}_{1\ldots i+1}) \geq (1-\eta)\tilde{a}_i$. This follows again from ((4.5)) and because L agrees with A_g. Therefore, we may pick $\tilde{y}_{i+1}$ as any value in $\{0,1\}$ for which ((4.12)) holds, and set $\tilde{a}_{i+1} = (1-\eta)\tilde{a}_i$.

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

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

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

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 4.6 and Lemma 4.8 with $\epsilon = 1/2$. After no more than n iterations this yields a circuit of polynomial size that decides $\overline{\text{SAT}}$ on inputs of size n. We have proved the following theorem.

THEOREM 4.13. *Suppose that* $\mathsf{coNP} \subseteq \mathsf{NP}/\operatorname{poly}$. *There exists a promise problem* $\Pi \in \mathsf{prM}(\mathsf{AM}||\mathsf{coNP})$ *such that the following holds for any language* L *that agrees with* Π. *Given* n, *we can construct a polynomial-size nondeterministic circuit for* $\overline{\text{SAT}}$ *at length* n *in deterministic polynomial time with oracle access to* L.

With Theorem 4.13 in hand, the nondeterministic variant of the Karp-Lipton argument yields the collapse stated in Theorem 4.2.

PROOF (of Theorem 4.2). Let K denote the $\Sigma_3\mathsf{P}$-complete language consisting of all Boolean formulas $\varphi(x,y,z)$ on three sets of variables x,y,z such that $(\exists x)(\forall y)\,\varphi(x,y,\cdot) \in \text{SAT}$. We show that under the assumptions of the theorem, $K \in \mathsf{P}^{\mathsf{prM}(\mathsf{AM}||\mathsf{coNP})}$.

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 coNP, $K' \in \Sigma_2\mathsf{P} \subseteq \mathsf{M}(\mathsf{AM}||\mathsf{coNP})$. Moreover, if C is a circuit that correctly decides $\overline{\text{SAT}}$ 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 4.13 on input n to obtain a circuit C of polynomial size for $\overline{\text{SAT}}$ 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 $\Pi \in \mathsf{prM}(\mathsf{AM}||\mathsf{coNP})$ from Theorem 4.13 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. □

5. Equivalence Result

In this section we establish our hardness-derandomization equivalence for Arthur-Merlin games (Theorem 1.1). We first argue the derandomization-to-hardness direction for $\Sigma_2\mathsf{E}$, as well as a weaker but relativizing claim for $\mathsf{E}^{\Sigma_2\mathsf{P}}$. We finish with the hardness-to-derandomization direction for $\Sigma_2\mathsf{E}$.

5.1. From derandomization to hardness. We use the lower bound framework introduced in Section 2.3. We assume that $\mathcal{C} \subseteq \mathsf{NP}/\operatorname{poly}$, where $\mathcal{C} = \Sigma_2\mathsf{E}$ or $\mathcal{C} = \mathsf{E}^{\Sigma_2\mathsf{P}}$, and derive a contradiction with Kannan's result that the polynomial-time hierarchy does not have (nondeterministic) circuits of size n^k for any fixed constant k. The derivation entails two key ingredients: (i) collapsing the polynomial-time hierarchy to some randomized class $\mathcal{R}$, and (ii) simulating $\mathcal{R}$ assuming that $\mathsf{prAM} \subseteq \cap_{\epsilon>0}\mathsf{i.o.}\text{-}\Sigma_2\mathsf{TIME}(2^{n^\epsilon})/n^\epsilon$. We developed (i) in Section 4. We now discuss (ii).

The classes $\mathcal{R}$ we consider involve augmented Arthur-Merlin protocols as introduced in Definition 3.2 of Section 3. More precisely, we consider $\mathcal{R} = \mathsf{prM}(\mathsf{AM}||\mathsf{coNP})$ and $\mathcal{R} = \mathsf{P}^{\mathsf{prM}(\mathsf{AM}||\mathsf{coNP})}$. Under the stronger derandomization assumption that prAM is in $\Sigma_2\mathsf{SUBEXP} \doteq \cap_{\epsilon>0}\Sigma_2\mathsf{TIME}(2^{n^\epsilon})$, those classes $\mathcal{R}$ can trivially be simulated in $\Sigma_2\mathsf{SUBEXP}$ or $\mathsf{SUBEXP}^{\Sigma_2\mathsf{P}}$, respectively. To carry over that argument to the i.o.-setting with small advice, we need to make sure that the simulation 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 derandomization 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 5.1 (Simulation Lemma).
If $\mathsf{prAM} \subseteq \cap_{\epsilon>0}\mathsf{i.o.}\text{-}\Sigma_2\mathsf{TIME}(2^{n^\epsilon})/n^\epsilon$ *then*

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

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

PROOF. Part (i): Let $\mathcal{C}$ denote $\cap_{\epsilon>0}$i.o.-Σ_2TIME$(2^{n^\epsilon})/n^\epsilon$, and let $\Pi \in$ prM(AM||coNP). We exhibit a language $R \in \mathcal{C}$ that agrees with Π under the assumption of the lemma. Per Definition 3.2, underlying Π there is a prAM-problem Γ and a coNP-language V . 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

$$R = \{x : (\exists y \in \{0,1\}^{n^c})\,(\langle x, y\rangle \in Q \ \wedge\ \langle x, y\rangle \in V)\}. \tag{5.2}$$

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 for all $m' \geq m$ there is a setting of pad such that $\langle x, y, 0^{pad}\rangle$ is of length m'.

Fix any $\epsilon > 0$. We want to exhibit $R_\epsilon \in \Sigma_2$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 $Q_\delta \in \Sigma_2$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 R_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 ((5.2)) 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 Q_δ agrees with Q for infinitely many lengths m, and the intervals $[\ell(n), \ell(n+1)]$ cover all but finitely many lengths m, it follows from the construction of R_ϵ that 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 $\Sigma_2\mathsf{TIME}(2^{n^{c\delta}})/n^{c\delta}$ 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 $R_\epsilon \in \Sigma_2\mathsf{TIME}(2^{n^\epsilon})/n^\epsilon$. This completes part (i).

Part (ii): Let $L \in \mathsf{DTIME}^{\Pi}(n^c)$ where $\Pi \in \mathsf{prM}(\mathsf{AM}||\mathsf{coNP})$. Fix any $\epsilon > 0$. We want to exhibit a language $L_\epsilon \in \mathsf{DTIME}^{\Sigma_2\mathsf{P}}(2^{n^\epsilon})/n^\epsilon$ that agrees with L on infinitely many lengths n. By assumption and by part (i), for every $\delta > 0$ there is a language $R_\delta \in \Sigma_2\mathsf{TIME}(2^{m^\delta})/m^\delta$ and an infinite set of lengths M_δ such that R_δ agrees with Π on all lengths in M_δ. Similar to part (i), we use R_δ to construct L_ϵ, as follows.

Let D be the oracle machine deciding L when given as oracle any language that agrees with Π. We give to D as oracle R_δ, and alter D so that all its queries are padded to the same length m, and supply the value for m as advice to D – just as we did in part (i). It follows that

$$L_\epsilon \in \text{i.o.-}\mathsf{DTIME}^{\Sigma_2\mathsf{TIME}(2^{m^\delta})/m^\delta}(n^d)/d\log n, \tag{5.3}$$

where d is a constant depending on c, and $m = n^d$. We now simplify ((5.3)). First, the oracle machine D underlying L_ϵ can simulate its oracle queries with an oracle for $\Sigma_2\mathsf{TIME}(2^{m^\delta})$ instead, provided that the advice of length m^δ needed in the original oracle is given as additional advice to D. Next, we set $\delta = \epsilon/(d+1)$, thereby turning ((5.3)) into

$$L_\epsilon \in \text{i.o.-}\mathsf{DTIME}^{\Sigma_2\mathsf{TIME}(2^{n^\epsilon})}(n^d)/n^\epsilon. \tag{5.4}$$

Finally, since $\mathsf{DTIME}^{\Sigma_2\mathsf{TIME}(s(n))}(r(n)) \subseteq \mathsf{DTIME}^{\Sigma_2\mathsf{P}}(s(r(n)))$ for nicely-behaved functions r and s (such as those in ((5.4))), the conclusion follows. □

For our main result, and more generally for superpolynomial lower bounds, the framework exploits the fact that linear-exponential classes such as E, NE, etc., have polynomial-size circuits if and only if they have *fixed*-polynomial-size circuits; this follows because those classes contain complete languages under linear-time reductions. The next lemma formalizes this fact in a way that will be handy later.

LEMMA 5.5. *Let $\mathcal{C} \in \{\Sigma_2\mathsf{E}, \mathsf{E}^{\Sigma_2\mathsf{P}}\}$. Suppose $\mathcal{C} \subseteq \mathsf{NP}/\operatorname{poly}$. Then every language in $\mathcal{C}/n$ has nondeterministic circuits of size n^d for all but finitely many input lengths, where d is a fixed constant.*

We include a proof for completeness.

PROOF. Let K be a complete language for $\mathcal{C}$ under linear-time mapping reductions. By hypothesis, there exists a family $C_0, C_1, \ldots$ of nondeterministic circuits such that C_ℓ decides K on inputs of length ℓ, and the size of C_ℓ is at most ℓ^c for some constant c.

By definition, for any language $L \in \mathcal{C}/n$, there exists $L' \in \mathcal{C}$ and a sequence $a_0, a_1, \ldots$ of strings with $|a_n| \leq n$ such that for any string x of length n, $x \in L \Leftrightarrow \langle x, a_{|x|}\rangle \in L'$. Since $L' \in \mathcal{C}$, there exists a linear-time mapping f such that for any input z, $z \in L' \Leftrightarrow f(z) \in K$. The following process then decides L on input x: Compute $y \doteq f(\langle x, a_{|x|}\rangle)$, and run $C_{|y|}$ on input y.

Since $|y| = |f(\langle x, a_n\rangle)| = O(n)$, by incorporating all C_ℓ for ℓ up to $O(n)$, the above process can be implemented by a nondeterministic circuit of size $O(n^{c+1})$, which is no more than n^d for $d > c$ and sufficiently large n. □

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

LEMMA 5.6 (implicit in Kannan 1982). *For every constant $d > 0$ there exists a language in $\mathsf{P}^{\Sigma_3\mathsf{P}}$ that requires nondeterministic circuits of size n^d for all but finitely many input lengths n.*

We include a proof for completeness.

PROOF. Let $s(n) = n^d$. Since any (nondeterministic) circuit of size $s(n)$ can be described by a string of length $O(s(n) \log s(n)) = o(2^n)$, for all but finitely many n there are more strings of length 2^n than there are Boolean functions computable by circuits of size $s(n)$. Thus there exists a characteristic sequence of length 2^n that cannot be computed by circuits of size less than $s(n)$. In fact, for the same reason, there exists a length-$\ell(n)$ prefix σ, where $\ell(n) = O(s(n) \log s(n)) = o(2^n)$, such that no size-$s(n)$ circuit agrees with σ, i.e., no such circuit can compute a characteristic sequence that extends σ.

The lexicographically least such prefix σ, say σ^*, can be found through a binary search, by using a $\Sigma_3\mathsf{P}$-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 $\Pi_2\mathsf{TIME}(\mathrm{poly}(s(n))$. 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 $\Sigma_3\mathsf{TIME}(\mathrm{poly}(s(n)))$, it follows by a padding argument that we can construct σ^* in time $\mathrm{poly}(s(n))$ with oracle access to $\Sigma_3\mathsf{P}$.

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, $L \in \mathsf{DTIME}^{\Sigma_3\mathsf{P}}(\mathrm{poly}(s))$. Since $s(n)$ is polynomial, the claim follows. □

We now have all the ingredients to instantiate the lower bound framework described in Section 2.3 and obtain the following derandomization-to-hardness results for Arthur-Merlin games.

THEOREM 5.7. *(i) If* $\mathsf{prAM} \subseteq \cap_{\epsilon>0}\mathsf{i.o.}\text{-}\Sigma_2\mathsf{TIME}(2^{n^\epsilon})/n^\epsilon$ *then* $\Sigma_2\mathsf{E} \not\subseteq \mathsf{NP}/\mathrm{poly}$.

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

PROOF. The derivation below proves both parts by contradiction. For part (i) we set $\mathcal{C} = \Sigma_2\mathsf{E}$ and $\mathcal{R} = \mathsf{M}(\mathsf{AM}||\mathsf{coNP})$, and for part (ii) we set $\mathcal{C} = \mathsf{E}^{\Sigma_2\mathsf{P}}$ and $\mathcal{R} = \mathsf{P}^{\mathsf{prM}(\mathsf{AM}||\mathsf{coNP})}$.

The first line for part (i) follows from Theorem 4.1; this is because the hypothesis $\Sigma_2\mathsf{E} \subseteq \mathsf{NP}/\mathrm{poly}$ implies that $\Sigma_2\mathsf{EXP} \subseteq \mathsf{NP}/\mathrm{poly}$ by padding, and $\mathsf{PSPACE} \subseteq \Sigma_2\mathsf{EXP}$. The first line for part (ii) follows from Theorem 4.2 as $\mathsf{coNP} \subseteq \mathsf{E}^{\Sigma_2 P}$.

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

$$\begin{aligned}
\mathcal{C} \subseteq & \mathsf{NP}/\operatorname{poly} \\
& \Longrightarrow \mathsf{PH} \subseteq \mathcal{R} \\
& \qquad \text{(by the collapse results from Section 4)} \\
& \Longrightarrow \mathsf{PH} \subseteq \text{i.o.-}\mathcal{C}/n \\
& \qquad \text{(by Lemma 5.1)} \\
& \Longrightarrow \mathsf{PH} \subseteq \text{i.o.-}\mathsf{NSIZE}(n^d) \\
& \qquad \text{(by Lemma 5.5)} \\
& \Longrightarrow \text{contradiction} \\
& \qquad \text{(by Lemma 5.6),}
\end{aligned}$$

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

□

Note that part (i) of Theorem 5.7 yields the forward direction of Theorem 1.1.

5.2. From hardness to derandomization. Finally, we argue the direction from hardness to derandomization in Theorem 1.1. This direction follows in a straightforward way from the known hardness versus randomness tradeoffs, and is implicit in Klivans & van Melkebeek (2002); Shaltiel & Umans (2006).

THEOREM 5.8. *Let s and σ denote monotone constructible functions from $\mathbb{N}$ to $\mathbb{N}$. If*

$$\Sigma_2\mathsf{E} \not\subseteq \mathsf{NSIZE}(s(n)),$$

then there exists a pseudorandom generator that yields the derandomization

$$\mathsf{prAM} \subseteq \text{i.o.-}\Sigma_2\mathsf{TIME}(2^{\sigma(n^{O(1)})})/\sigma(n^{O(1)}),$$

provided that $\sigma(n) \geq (s^{-1}(n^c))^2/\log n$, where $c > 0$ is a universal constant and $s^{-1}(m)$ denotes $\min\{n \in \mathbb{N} : s(n) \geq m\}$.

Given the hardness hypothesis, for any $\epsilon > 0$, we can pick $s(n)$ in Theorem 5.8 to be n^k for a large enough constant k and get a PRG that yields the simulation of a polynomial-time Arthur-Merlin protocol in $\Sigma_2\mathsf{TIME}(2^{n^\epsilon})/n^\epsilon$ for infinitely many input lengths n. This establishes the backward direction of Theorem 1.1.

Proof (of Theorem 5.8). The proof follows by combining two lemmas that are implicit in the literature. We state the lemma's and briefly indicate how they are obtained.

First, in order to derandomize prAM it suffices to construct pseudorandom generators that fool nondeterministic circuits.

Lemma 5.9 (implicit in Klivans & van Melkebeek 2002). *If there is a pseudorandom generator G that on seed length $\sigma(n)$ is computable in $\Sigma_2\mathsf{TIME}(2^{O(\sigma(n))})$ with advice of length $\sigma(n)$, and fools nondeterministic circuits of size n for infinitely many n, then prAM can infinitely often be simulated in $\Sigma_2\mathsf{TIME}(2^{O(\sigma(n^{O(1)}))}n^{O(1)})$ with advice of length $\sigma(n^{O(1)})$.*

A pseudorandom generator G being computable in a class $\mathcal{C}$ means that the language $L_G \doteq \{\langle s, i, b\rangle \ : \ \text{the } i\text{th bit of } G(s) \text{ equals } b\}$ belongs to $\mathcal{C}$.

Referring to ((3.1)) in the definition of Arthur-Merlin games, the simulation in Lemma 5.9 on input x follows from using G to generate the random bit string y of length $n^{O(1)}$ needed by Arthur, and then performing a trivial derandomization of the resulting pseudorandom process. More precisely, we guess half of the seeds s of length $\sigma(n^{O(1)})$ and do the following for each guessed s: Guess y, check that $y = G(s)$, guess z, and check that $\langle x, y, z\rangle \in L$. The correctness of the simulation follows from the fact that G fools nondeterministic circuits. The computability of G implies that the simulation runs in $\Sigma_2\mathsf{TIME}(2^{O(\sigma(n^{O(1)}))}n^{O(1)})$ with advice of length $\sigma(n^{O(1)})$.

The pseudorandom generators needed in Lemma 5.9 follow from the given hardness assumption by the known hardness versus randomness tradeoffs for prAM.

LEMMA 5.10 (implicit in Shaltiel & Umans 2006). *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 $\Sigma_2\mathsf{E}$ 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 $\Sigma_2\mathsf{TIME}(2^{O(\sigma(n))})$ on seed length $\sigma(n)$ with advice of length $\sigma(n)$, and fools nondeterministic circuits of size n at infinitely many lengths n.*

The generator G from Lemma 5.10 is built out of a language $H \in \Sigma_2\mathsf{E}$ that follows from the hardness hypothesis. More specifically, the ith bit of $G(s)$ is defined as the membership to H of a string u that is computable in polynomial time from s and the index i. This implies that the language

$$L'_G \doteq \{\langle s, i\rangle \,:\, \text{the } i\text{th bit of } G(s) \text{ equals } 1\}$$

is in $\Sigma_2\mathsf{TIME}(2^{O(\ell(n))}n^{O(1)})$. Using as advice the number of pairs $\langle s', i'\rangle$ of the same length as $\langle s, i\rangle$ that are in L'_G, we can check that the ith bit of $G(s)$ is 0 by guessing that many pairs $\langle s', i'\rangle$ of that length different from $\langle s, i\rangle$ and verifying that each is in L'_G. Noting that the hardness condition of Lemma 5.10 implies that $\ell(n) = \Omega(\log n)$ and $\ell^2(n)/\log n = \Omega(\log n)$, it follows that G is computable in $\Sigma_2\mathsf{TIME}(2^{O(\sigma(n))})$ on seed length $\sigma(n)$ with advice of length $\sigma(n)$ for some function $\sigma(n) = O(\ell^2(n)/\log n)$.

Using G from Lemma 5.10 in Lemma 5.9 yields the required derandomization of prAM. □

6. Concluding Remarks

Both our main result (Theorem 1.1) and the corresponding result for prBPP in Impagliazzo *et al.* (2002) establish an equivalence of derandomization and PRGs at just one point within the derandomization spectrum. A natural question to ask is whether these equivalences can be extended to other regions of the derandomization spectrum: simulations of $\mathsf{prBPP}/\mathsf{prAM}$ using less time, less advice, or with one fewer alternation – if they can be done at all, then can they be done through PRGs? What about simulations

that succeed on all-but-finitely-many input lengths rather than infinitely many?

Acknowledgements

We would like to thank the anonymous referees for their constructive suggestions. The research was partially supported by NSF grants CCF-1017597 and CCF-1319822. An extended abstract of this paper appears in the proceedings of CCC 2012 (Aydınlıoğlu & van Melkebeek 2012).

References

Scott Aaronson & Dieter van Melkebeek (2011). On circuit lower bounds from derandomization. *Theory of Computing* **7**, 177–184.

Sanjeev Arora & Boaz Barak (2009). *Computational Complexity: A Modern Approach*. Cambridge University Press.

Barış Aydinlioğlu, Dan Gutfreund, John M. Hitchcock & Akinori Kawachi (2011). Derandomizing Arthur-Merlin Games and Approximate Counting Implies Exponential-Size Lower Bounds. *Computational Complexity* **20**(2), 329–366.

Barış Aydinlioğlu & Dieter van Melkebeek (2012). Nondeterministic Circuit Lower Bounds from Mildly Derandomizing Arthur-Merlin Games. In *Proceedings of the IEEE Conference on Computational Complexity*, 269–279.

László Babai, Lance Fortnow, Noam Nisan & Avi Wigderson (1993). BPP Has Subexponential Time Simulations Unless EXPTIME has Publishable Proofs. *Computational Complexity* **3**, 307–318.

László Babai & Shlomo Moran (1988). Arthur-Merlin games: a randomized proof system, and a hierarchy of complexity classes. *Journal of Computer and System Sciences* **36**(2), 254–276.

Jin-Yi Cai, Venkatesan T. Chakaravarthy, Lane A. Hemaspaandra & Mitsunori Ogihara (2005). Competing provers yield improved Karp-Lipton collapse results. *Information and Computation* **198**(1), 1–23.

Venkatesan T. Chakaravarthy & Sambuddha Roy (2011). Arthur and Merlin as oracles. *Computational Complexity* **20**(3), 505–558.

Oded Goldreich (2006). On Promise Problems: A Survey. In *Essays in Memory of Shimon Even*, 254–290.

Oded Goldreich (2011a). In a World of P=BPP. In *Studies in Complexity and Cryptography*, 191–232.

Oded Goldreich (2011b). Two Comments on Targeted Canonical Derandomizers. Technical Report TR11-047, Electronic Colloquium on Computational Complexity (ECCC).

Shafi Goldwasser & Michael Sipser (1986). Private Coins versus Public Coins in Interactive Proof Systems. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, 59–68.

Dan Gutfreund, Ronen Shaltiel & Amnon Ta-Shma (2003). Uniform hardness versus randomness tradeoffs for Arthur-Merlin games. *Computational Complexity* **12**(3-4), 85–130.

Russell Impagliazzo, Valentine Kabanets & Avi Wigderson (2002). In search of an easy witness: exponential time vs. probabilistic polynomial time. *Journal of Computer and System Sciences* **65**(4), 672–694.

Russell Impagliazzo & Avi Wigderson (1997). P = BPP if E Requires Exponential Circuits: Derandomizing the XOR Lemma. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, 220–229.

Valentine Kabanets & Russell Impagliazzo (2004). Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity* **13**(1/2), 1–46.

Ravi Kannan (1982). Circuit-size lower bounds and nonreducibility to sparse sets. *Information and Control* **55**(1), 40–56.

Richard M. Karp & Richard J. Lipton (1982). Turing machines that take advice. *L'Enseignement Mathématique* **28**(2), 191–209.

Jeff Kinne, Dieter van Melkebeek & Ronen Shaltiel (2012). Pseudorandom Generators, Typically-Correct Derandomization, and Circuit Lower Bounds. *Computational Complexity* **21**(1), 3–61.

Adam Klivans & Dieter van Melkebeek (2002). Graph Nonisomorphism Has Subexponential Size Proofs Unless the Polynomial-Time Hierarchy Collapses. *SIAM Journal on Computing* **31**(5), 1501–1526.

Carsten Lund, Lance Fortnow, Howard J. Karloff & Noam Nisan (1992). Algebraic Methods for Interactive Proof Systems. *Journal of the ACM* **39**(4), 859–868.

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

Uwe Schöning (1989). Probabilistic complexity classes and lowness. *Journal of Computer and System Sciences* **39**(1), 84–100.

Ronen Shaltiel & Christopher Umans (2006). Pseudorandomness for Approximate Counting and Sampling. *Computational Complexity* **15**(4), 298–341.

Adi Shamir (1992). IP = PSPACE. *Journal of the ACM* **39**(4), 869–877.

Andrew Chi-Chih Yao (1982). Theory and Applications of Trapdoor Functions (Extended Abstract). In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS)*, 80–91.

Chee-Keng Yap (1983). Some Consequences of Non-Uniform Conditions on Uniform Classes. *Theoretical Computer Science* **26**, 287–300.

Manuscript received 1 September 2013

Barış Aydinlioğlu
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