Polylogarithmic-round Interactive Proofs for coNP Collapses the Exponential Hierarchy

Alan L. Selman*
Department of Computer Science and Engineering
University at Buffalo, Buffalo, NY 14260

Samik Sengupta†
Department of Computer Science and Engineering
University at Buffalo, Buffalo, NY 14260

January 13, 2004

Abstract

It is known [BHZ87] that if every language in coNP has a constant-round interactive proof system, then the polynomial hierarchy collapses. On the other hand, Lund *et al*. [LFKN92] have shown that $\#\mathrm{SAT}$, the $\#\mathrm{P}$-complete function that outputs the number of satisfying assignments of a Boolean formula, can be computed by a linear-round interactive protocol. As a consequence, the coNP-complete set $\overline{\mathrm{SAT}}$ has a proof system with linear rounds of interaction.

We show that if every set in coNP has a polylogarithmic-round interactive protocol then the exponential hierarchy collapses to the third level. In order to prove this, we obtain an exponential version of Yap's result [Yap83], and improve upon an exponential version of the Karp-Lipton theorem [KL80], obtained first by Buhrman and Homer [BH92].

1 Introduction

Bábai [Báb85] and Bábai and Moran [BM88] introduced *Arthur-Merlin Games* to study the power of randomization in interaction. Soon afterward, Goldwasser and Sipser [GS89] showed that these classes are equivalent in power to *Interactive Proof Systems*, introduced by Goldwasser *et al*. [GMR85]. Study of interactive proof systems and Arthur-Merlin classes has been exceedingly successful [ZH86, BHZ87, ZF87, LFKN92, Sha92], eventually leading to the discovery of Probabilistically Checkable Proofs [BOGKW88, LFKN92, Sha92, BFL81, BFLS91, FGL$^+$91, AS92, ALM$^+$92].

Interactive proof systems are successfully placed relative to traditional complexity classes. In particular, it is known that for any constant k, $\mathrm{IP}[k] \subseteq \Pi_2^p$ [BM88], and $\mathrm{IP}[\mathrm{poly}] = \mathrm{PSPACE}$ [Sha92]. However, the relationship between coNP and interactive proof systems is not entirely clear. On the one hand, Boppana, Håstad and Zachos [BHZ87] proved that if every set in coNP has a constant-round interactive proof system, then the polynomial-time hierarchy collapses below the second level. On the other hand, the best interactive protocol for any language in coNP comes from the result of Lund *et al*. [LFKN92], who show that $\#\mathrm{SAT}$, a problem hard for the entire polynomial-time hierarchy [Tod91], is accepted by an interactive proof system with $O(n)$ rounds of interaction on an input of length n. Can every set in coNP be accepted by an interactive

*Research partially supported by NSF grant CCR-0307077. Email: selman@cse.buffalo.edu
†Email: samik@cse.buffalo.edu

proof system with more that constant but sublinear number of rounds? Answering this question has been the motivation for this paper.

We show in this paper that coNP cannot have a polylogarithmic-round interactive proof system unless the exponential hierarchy collapses to the third level, i.e., $\mathrm{NEXP}^{\Sigma_2^p} = \mathrm{coNEXP}^{\Sigma_2^p}$. Three principal steps lead to the proof of our main result, and these results are of independent interest. Note that although we use Arthur-Merlin protocols to obtain our results, our main theorem holds for interactive proof systems as well due to the result of Goldwasser and Sipser [GS89], who showed that an interactive proof system with m rounds can be simulated by an $2m + 4$-move Arthur-Merlin protocol.

- An Arthur-Merlin protocol with m moves where both Arthur and Merlin exchange at most $l(n)$ bits at every move can be simulated by a two-move AM protocol where Arthur moves first and uses $O(l(n)^{m-1})$ random bits (Theorem 3.1).
- If L is accepted by a two-move AM protocol where Arthur uses 2^{polylog} random bits, then L belongs to the advice class $\mathrm{NP}/2^{\mathrm{polylog}}$ (Lemma 3.3).
- If coNP $\subseteq \mathrm{NP}/2^{\mathrm{polylog}}$ (equivalently NP $\subseteq$ coNP$/2^{\mathrm{polylog}}$) then

$$\mathrm{NEXP}^{\Sigma_2^p} = \mathrm{coNEXP}^{\Sigma_2^p} = \mathrm{S}_2^{exp} \circ \mathrm{P}^{\mathrm{NP}} \text{ (Theorem 4.1).}$$

In addition to these results, we improve upon a result of Buhrman and Homer [BH92], showing that if NP has 2^{polylog}-size family of circuits, then $\mathrm{NEXP}^{\mathrm{NP}} = \mathrm{coNEXP}^{\mathrm{NP}} = \mathrm{S}_2^{\mathrm{EXP}}$.

We note that Goldreich and Håstad [GH98] and Goldreich, Vadhan, and Wigderson [GVW01] have studied Arthur-Merlin games and interactive proof systems with bounded message complexity. Their results are incomparable to our Theorem 3.1. In particular, our simulation of an m-move Arthur-Merlin protocol by a 2-move Arthur-Merlin protocol assumes that l, the number of message bits used in every move, is polynomial in the length of the input, and we obtain that the number of random bits used by Arthur in the 2-move protocol is not exponential in l. This is crucial in order to get Theorem 4.1.

2 Preliminaries

For definitions of standard complexity classes, we refer the reader to Homer and Selman [HS01]. The exponential hierarchy is defined as follows:

$$\mathrm{EXP} = \Sigma_0^{exp}, \mathrm{NEXP} = \Sigma_1^{exp}, \mathrm{NEXP}^{\mathrm{NP}} = \Sigma_2^{exp},$$

and in general, for $k \geq 0$,

$$\Sigma_{k+1}^{exp} = \mathrm{NEXP}^{\Sigma_k^p}.$$

For every $k \geq 0$,

$$\Pi_k^{exp} = \{L \mid \overline{L} \in \Sigma_k^{exp}\}.$$

We define $\mathrm{polylog} = \bigcup_{k>0} \log^k n$ and $2^{\mathrm{polylog}} = \bigcup_{k>0} 2^{\log^k n}$. Let $\mathcal{C}$ be a complexity class. A set $L \in \mathcal{C}/2^{\mathrm{polylog}}$ if there is a function $s : 1^* \mapsto \Sigma^*$, some constant $k > 0$, and a set $A \in \mathcal{C}$ such that

1. For every n, $|s(1^n)| \leq 2^{\log^k n}$, and
2. For all x, $x \in L \Leftrightarrow (x, s(1^{|x|})) \in A$. Here A is called the *witness language*.

It is easy to see that $\mathcal{D} \subseteq \mathcal{C}/2^{\text{polylog}}$ if and only if $\text{co}\mathcal{D} \subseteq \text{co}\mathcal{C}/2^{\text{polylog}}$.

Bábai [Báb85] introduced *Arthur-Merlin protocol*, a combinatorial game that is played by Arthur, a probabilistic polynomial-time machine, and Merlin, a computationally unbounded Turing machine. Arthur can use random bits, but these bits are public, i.e., Merlin can see them and move accordingly.

Given an input string x, Merlin tries to convince Arthur that x belongs to some language L. The game consists of a predetermined finite number of moves with Arthur and Merlin moving alternately. In each move Arthur (or Merlin) prints a finite string on a read-write communication tape. Arthur's moves depend on his random bits. After the last move, Arthur either accepts or does not accept x.

Definition 2.1 ([Báb85, BM88]) *For any $m > 0$, a language L is in $\text{AM}[m]$ (respectively $\text{MA}[m]$) if for every string x of length n*

- *The game consists of m moves*
- *Arthur (resp., Merlin) moves first*
- *After the last move, Arthur behaves deterministically to either accept or not accept the input string*
- *If $x \in L$, then there exists a sequence of moves by Merlin that leads to the acceptance of x by Arthur with probability is at least $\frac{3}{4}$*
- *if $x \notin L$ then for all possible moves of Merlin, the probability that Arthur accepts x is less than $\frac{1}{4}$.*

Bábai and Moran [BM88] showed that $\text{AM}[k]$, where $k > 1$ is some constant, is the same as $\text{AM}[2] = \text{AM}$. Note that $\text{MA}[2] = \text{MA}$, $\text{AM}[1] = \text{BPP}$, and $\text{MA}[1] = \text{M} = \text{NP}$. Bábai [Báb85] proved that $\text{MA} \subseteq \text{AM}$.

Now we need to extend modestly the notion of Arthur-Merlin protocols. In the following limited manner, we consider the possibility that Arthur is a probabilisitic Turing machine, but not necessarily polynomial-time-bounded. Let $f : \mathbb{N} \mapsto \mathbb{N}$ be some function. Then we define the class $\text{AM}(f)$ to be the class of languages accepted by protocols in which there are only two moves, Arthur moves first, and Arthur can use at most $f(n)$ random bits on an input of length n. In particular, we will have occasion to consider f to be quasipolynomial; that is, we will consider the class $\text{AM}(2^{\log^c n})$, where c is a positive constant.

We note the following standard proposition.

Proposition 2.2 *Let E be an event that occurs with probability at least $\frac{3}{4}$. Then, for any polynomial $p(\cdot)$ such that $p(n) \geq n$, there is a constant c such that within $t \stackrel{def}{=} c \times p(n)$ independent trials, E occurs for more than $\frac{t}{2}$ times with probability $(1 - \frac{1}{2^{p(n)}})$.*

We define S_2^{exp} as the exponential version of the S_2 operator defined by Russell and Sundaram [RS98] and Canetti [Can96]. A set L is in $\text{S}_2^{exp} \circ \mathcal{C}$ if there is some $k > 0$ and $A \in \mathcal{C}$ such that for every $x \in \{0,1\}^n$,

$$\begin{aligned} x \in L &\implies \exists y\, \forall z\, (x,y,z) \in A, \text{ and} \\ x \notin L &\implies \exists z\, \forall y\, (x,y,z) \notin A, \end{aligned}$$

where $|y|, |z| \leq 2^{n^k}$.

Similar to $\text{S}_2^{\text{P}} \stackrel{def}{=} \text{S}_2 \circ \text{P}$, the class $\text{S}_2^{exp} \circ \mathcal{C}$ can be thought of as a game between two provers and a verifier. Let $L \in \text{S}_2^{exp} \circ \mathcal{C}$. On any input x of length n, the *Yes-prover* attempts to show that $x \in L$, and the *No-prover* attempts to show that $x \notin L$. Both the proofs are at most exponentially long in $|x|$. If $x \in L$, then there must be a proof by the yes-prover (called a *yes-proof*) that convinces the verifier that $x \in L$ no matter what the proof provided by the no-prover (called a *no-proof*) is; symmetrically, if $x \notin L$, then there must exist some no-proof such that the verifier rejects x irrespective of the yes-proof. For every input x,

there is a yes-prover and a no-prover such that exactly one of them is correct. The verifier has the ability of the class $\mathcal{C}$; for example, if $\mathcal{C} = \mathrm{P}$, then the verifier is a deterministic polynomial-time Turing machine, and if $\mathcal{C} = \mathrm{P}^{\mathrm{NP}}$, then the verifier is a polynomial-time oracle Turing machine with SAT as the oracle. It is also easy to see that if $\mathcal{C}$ is closed under complement, then $\mathrm{S}_2^{exp} \circ \mathcal{C}$ is also closed under complement.

We concentrate on the classes $\mathrm{S}_2^{\mathrm{EXP}} \stackrel{def}{=} \mathrm{S}_2^{exp} \circ \mathrm{P}$ and $\mathrm{S}_2^{exp} \circ \mathrm{P}^{\mathrm{NP}}$. The proofs of Russell and Sundaram can be easily modified to show the following.

Proposition 2.3

1. $\mathrm{S}_2^{\mathrm{EXP}} \subseteq \mathrm{NEXP}^{\mathrm{NP}} \cap \mathrm{coNEXP}^{\mathrm{NP}}$.
2. $\mathrm{NEXP}^{\mathrm{NP}} \cup \mathrm{coNEXP}^{\mathrm{NP}} \subseteq \mathrm{S}_2^{exp} \circ \mathrm{P}^{\mathrm{NP}} \subseteq \mathrm{NEXP}^{\Sigma_2^{\mathrm{P}}} \cap \mathrm{coNEXP}^{\Sigma_2^{\mathrm{P}}}$.

Proof We give a short proof of the second inclusion of item (2). Note that since $\mathrm{S}_2^{exp} \circ \mathrm{P}^{\mathrm{NP}}$ is closed under complement, it suffices to show that $\mathrm{S}_2^{exp} \circ \mathrm{P}^{\mathrm{NP}}$ is a subset of $\mathrm{NEXP}^{\Sigma_2^{\mathrm{P}}}$. Let $L \in \mathrm{S}_2^{exp} \circ \mathrm{P}^{\mathrm{NP}}$; therefore, $\exists k > 0, L' \in \mathrm{P}^{\mathrm{NP}}$ such that

$$\begin{aligned} x \in L &\implies \exists y\, \forall z\, (x,y,z) \in L', \text{ and} \\ x \notin L &\implies \exists z\, \forall y\, (x,y,z) \notin L', \end{aligned}$$

where $|y|, |z| \leq 2^{|x|^k}$. We define the language

$$A = \{(x, y, 0^{2^{|x|^k}}) \,\big|\, \exists z (x,y,z) \notin L'\}.$$

A is in Σ_2^p. We define a NEXP machine N that decides L with A as an oracle. On input x, N guesses $y, |y| \leq 2^{|x|^k}$, and accepts x if and only if $(x, y, 0^{2^{|x|^k}}) \notin A$. If $x \in L$, then for the correctly guessed y, $(x,y,z) \in L'$ for every z; therefore, N accepts x. On the other hand, if $x \notin L$, then there is a z such that for every y, $(x,y,z) \notin L'$, and therefore, $(x, y, 0^{2^{|x|^k}}) \in A$ and N rejects x. This completes the proof. □

3 Arthur-Merlin Games with Polylogarithmic Moves

In this section, we show that languages accepted by an Arthur-Merlin protocol with m moves where Arthur and Merlin exchange at most $l(n)$ bits in every move (on an input of length n) can be accepted by a two-move Arthur-Merlin protocol where Arthur moves first and uses $O(l(n)^{m-1})$ random bits. This implies that if coNP has a polylogarithmic-move Arthur-Merlin protocol, then coNP can be accepted by a two-move Arthur-Merlin protocol where Arthur uses 2^{polylog} random bits. As a consequence, using Lemma 3.3, we obtain that coNP can be solved by nondeterministic polynomial-time machines with 2^{polylog}-length advice.

Theorem 3.1 *Let $m > 2$, and let $L \in \mathrm{AM}[m]$ where on an input of length n, Arthur and Merlin exchange messages of length at most $l(n)$ in every move. Then there is a constant k_m such that $L \in \mathrm{AM}(k_m l(n)^{m-1})$.*

Proof We show this by induction on m. We assume that in any move during the interaction, Arthur and Merlin exchange at most $l(n)$ bits. We will show the following stronger result. For $m > 2$, there is some constant k_m such that

1. $\mathrm{AM}[m] \subseteq \mathrm{AM}(k_m l(n)^{m-1})$

2. $\mathrm{MA}[m] \subseteq \mathrm{AM}(k_m l(n)^{m-1})$

Our proof is reminiscent of the proof of $\mathrm{MA} \subseteq \mathrm{AM}$. In order to help the reader understand the proof of Theorem 3.1, we digress to recall that proof. Let $L \in \mathrm{MA}$. There is $L' \in \mathrm{P}$ so that for any $w \in \{0,1\}^n$,

$$\begin{aligned} w \in L &\implies \exists v \ \Pr_z[(w,v,z) \in L'] \geq \frac{3}{4}, \text{ and} \\ w \notin L &\implies \forall v \ \Pr_z[(w,v,z) \in L'] \leq \frac{1}{4}, \end{aligned}$$

where $|v|, |z| \leq l(n)$. By Proposition 2.2 there exists some constant $c > 0$ such that if Arthur repeats its computation $cl(n)$ times and takes the majority vote, then it will get an error probability of at most $\frac{1}{2^{l(n)+3}}$. Therefore,

$$\begin{aligned} w \in L &\implies \exists v \Pr_z[(w,v,z) \in L'] \geq 1 - \frac{1}{2^{l(n)+3}}, \text{ and} \\ w \notin L &\implies \forall v \Pr_z[(w,v,z) \in L'] \leq \frac{1}{2^{l(n)+3}}, \end{aligned}$$

where $|z| \leq cl(n)^2$. Now we want to switch quantifiers, i.e., Arthur should move first. When w is in L, there is at least one v for which $1 - \frac{1}{2^{l(n)+3}}$ fraction of z result in $(w,v,z) \in L'$. Therefore, for $1 - \frac{1}{2^{l(n)+3}}$ fraction of z, the same v will result in acceptance of w. On the other hand, the maximum number of strings $v, |v| \leq l(n)$, provided by Merlin is $2^{l(n)+1}$. Therefore, when $w \notin L$, the fraction of z for which there exists some v where $(w,v,z) \in L'$ is at most $\frac{1}{2^{l(n)+3}} \times 2^{l(n)+1} \leq \frac{1}{4}$. Hence, we have the following:

$$\begin{aligned} w \in L &\implies \Pr_z[\exists v \ (w,v,z) \in L'] \geq 1 - \frac{1}{2^{l(n)+3}} \geq \frac{3}{4}, \text{ and} \\ w \notin L &\implies \Pr_z[\exists v \ (w,v,z) \in L'] \leq \frac{1}{4}, \end{aligned}$$

where $|z| \leq cl(n)^2$. Therefore, $\mathrm{MA} \subseteq \mathrm{AM}$. Observe that the total number of random bits required by Arthur is at most $cl(n)^2$.

Now we prove the base cases, i.e., $m = 3$. Let $L \in \mathrm{AMA}$. Then there is $L' \in \mathrm{MA}$ such that for every x of length n

$$\begin{aligned} x \in L &\implies \Pr_y[(x,y) \in L'] \geq \frac{3}{4}, \text{ and} \\ x \notin L &\implies \Pr_y[(x,y) \in L'] \leq \frac{1}{4}, \end{aligned}$$

where $|y| \leq l(n)$. We reduce the error probability to $\frac{1}{8}$ by increasing the length of y to $kl(n)$, where k is some constant. By the above proof of $\mathrm{MA} \subseteq \mathrm{AM}$, we know that $L' \in \mathrm{AM}$ for a random z of length at most $cl(n)^2$. Therefore, there is a set $B \in \mathrm{P}$ such that

$$\begin{aligned} w \in L' &\implies \Pr_z[\exists v \ (w,v,z) \in B] \geq \frac{3}{4}, \text{ and} \\ w \notin L' &\implies \Pr_z[\exists v \ (w,v,z) \in B] \leq \frac{1}{4}. \end{aligned}$$

Again, we reduce the error probability to $\frac{1}{8}$ by increasing the length of z to $k \times cl(n)^2$. This implies the following:

$$\begin{aligned} x \in L &\implies \Pr_y \Pr_z[\exists v \ (x,y,v,z) \in B] \geq \frac{3}{4}, \text{ and} \\ x \notin L &\implies \Pr_y \Pr_z[\exists v \ (x,y,v,z) \in B] \leq \frac{1}{4}, \end{aligned}$$

where $|y| \leq kl(n)$, and $|z| \leq kcl(n)^2$. This bound works since for both y and z the error probability is at most $\frac{1}{8}$, and therefore, when considered as one move (i.e. (y,z) is Arthur's random bit string) the error probability is at most $\frac{1}{8}+\frac{1}{8}=\frac{1}{4}$. Therefore, the total number of random bits required by Arthur is at most $kl(n)+kcl(n)^2$.

Similarly, let $L \in \mathrm{MAM}$. There is $L' \in \mathrm{AM}$ such that

$$\begin{aligned} x \in L &\implies \exists y\ (x,y) \in L', \text{ and} \\ x \notin L &\implies \forall y\ (x,y) \notin L', \end{aligned}$$

where $|y| \leq l(n)$. We amplify the success probability of the AM protocol for L' to $1-\frac{1}{2^{l(n)+3}}$; so, Arthur requires at most $cl(n)^2$ random bits, and there is an NP set L'' such that the following holds:

$$\begin{aligned} x \in L &\implies \exists y\ \Pr_z[(x,y,z) \in L''] \geq 1-\frac{1}{2^{l(n)+3}}, \text{ and} \\ x \notin L &\implies \forall y\ \Pr_z[(x,y,z) \in L''] \leq \frac{1}{2^{l(n)+3}}. \end{aligned}$$

Here $|y| \leq l(n)$ and $|z| \leq cl(n)^2$. Again, we switch the quantifiers as we have done before for the proof of $\mathrm{MA} \subseteq \mathrm{AM}$. The critical counting is that the fraction of z for which there is some y such that $(x,y,z) \in L''$ is at most $\sum_{y \in \Sigma^{\leq l(n)}} \frac{1}{2^{l(n)+3}} \leq \frac{1}{4}$. Therefore, we have

$$\begin{aligned} x \in L &\implies \Pr_z[\exists y\ (x,y,z) \in L''] \geq 1-\frac{1}{2^{l(n)+3}}, \text{ and} \\ x \notin L &\implies \Pr_z[\exists y\ (x,y,z) \in L''] \leq \frac{1}{4}, \end{aligned}$$

with $|z| \leq cl(n)^2$. Note that since $L'' \in \mathrm{NP}$, this shows that L is in AM. Take $k_3 = 2kc$; we have shown that $\mathrm{MAM} \subseteq \mathrm{AM}(k_3 l(n)^2)$ and $\mathrm{AMA} \subseteq \mathrm{AM}(k_3 l(n)^2)$ as well. Therefore, our base case is proved.

Now, let $m > 3$, and $L \in \mathrm{AM}[m]$, i.e., Arthur moves first. Then, similar to the case of $m=3$, we have that there is a constant k and $L' \in \mathrm{MA}[m-1]$ such that

$$\begin{aligned} x \in L &\implies \Pr_y[(x,y) \in L'] \geq \frac{3}{4}, \text{ and} \\ x \notin L &\implies \Pr_y[(x,y) \in L'] \leq \frac{1}{4}, \end{aligned}$$

where $|y| \leq l(n)$. We reduce the error probability of this protocol to $\frac{1}{8}$ by increasing the length of y to $kl(n)$. Since $L' \in \mathrm{MA}[m-1] \subseteq \mathrm{AM}(k_{m-1} \times l(n)^{m-2})$, there is a set $B \in \mathrm{P}$ such that for every x of length n

$$\begin{aligned} x \in L &\implies \Pr_y\ \Pr_z[\exists v\ (x,y,v,z) \in B] \geq \frac{3}{4}, \text{ and} \\ x \notin L &\implies \Pr_y\ \Pr_z[\exists v\ (x,y,v,z) \in B] \leq \frac{1}{4}, \end{aligned}$$

where $|y| \leq kl(n)$ and $|z| \leq kk_{m-1}l(n)^{m-2}$. Note that z is also increased k-fold so that the error probability of L' is reduced to $\frac{1}{8}$. Therefore, we have $L \in \mathrm{AM}(kl(n)+kk_{m-1}l(n)^{m-2})$.

On the other hand, let us assume that Merlin moves first. Then, there is a set L' in $\mathrm{AM}[m-1]$, and therefore, in $\mathrm{AM}(k_{m-1}l(n)^{m-2})$, such that for every string x of length n,

$$\begin{aligned} x \in L &\implies \exists y\ (x,y) \in L', \text{ and} \\ x \notin L &\implies \forall y\ (x,y) \notin L', \end{aligned}$$

where $|y| \leq l(n)$. Using the same technique described above to show that $\mathrm{MA} \subseteq \mathrm{AM}$, we need to amplify the success probability of the Arthur-Merlin protocol for L' to $1 - \frac{1}{2^{l(n)+3}}$. Therefore, there is a set $L'' \in \mathrm{NP}$ and some $c > 0$ such that Arthur has to repeat the protocol $cl(n)$ times to obtain the following:

$$\begin{aligned} x \in L \quad &\Longrightarrow \quad \exists y \ \Pr_z[(x,y,z) \in L''] \geq 1 - \frac{1}{2^{l(n)+3}}, \text{ and} \\ x \notin L \quad &\Longrightarrow \quad \forall y \ \Pr_z[(x,y,z) \in L''] \leq \frac{1}{2^{l(n)+3}}, \end{aligned}$$

where $|z| \leq cl(n) \times k_{m-1} l(n)^{m-2} = k_{m-1} cl(n)^{m-1}$. Switching quantifiers as before, we obtain

$$\begin{aligned} x \in L \quad &\Longrightarrow \quad \Pr_z[\exists y \ (x,y,z) \in L''] \geq 1 - \frac{1}{2^{l(n)+3}} \geq \frac{3}{4}, \text{ and} \\ x \notin L \quad &\Longrightarrow \quad \Pr_z[\exists y \ (x,y,z) \in L''] \leq \sum_{y \in \Sigma^{\leq l(n)}} \frac{1}{2^{l(n)+3}} \leq \frac{1}{4}. \end{aligned}$$

Note that Arthur now needs at most $cl(n) \times k_{m-1} l(n)^{m-2} = k_{m-1} cl(n)^{m-1}$ random bits. Also observe that $L'' \in \mathrm{NP}$; therefore, the above equations define an Arthur-Merlin protocol. By taking $k_m = k k_{m-1} c$, we prove the induction step:

$$\mathrm{AM}[m] \subseteq \mathrm{AM}(k_m l(n)^{m-1}),$$

and

$$\mathrm{MA}[m] \subseteq \mathrm{AM}(k_m l(n)^{m-1}).$$

This completes the proof. □

Corollary 3.2 *For any $k > 0$, there is a $c > 0$ such that*

$$\mathrm{AM}[\log^k n] \subseteq \mathrm{AM}(2^{\log^c n}).$$

Proof Assume that Arthur and Merlin exchange at most $l(n)$ bits during every move of communication. From Theorem 3.1, we obtain that there is a constant k' such that $\mathrm{AM}[\log^k n] \subseteq \mathrm{AM}(k' l(n)^{\log^k n - 1})$. Taking an appropriate c such that $2^{\log^c n} \geq k' l(n)^{\log^k n - 1}$, we obtain the result.

□

Lemma 3.3 $\mathrm{AM}(2^{\mathrm{polylog}}) \subseteq \mathrm{NP}/2^{\mathrm{polylog}}$.

Proof Let $L \in \mathrm{AM}(2^{\log^k n})$. Consider any input x of length n. There is a constant k and a polynomial-time predicate R such that:

$$\begin{aligned} x \in L \quad &\Longrightarrow \quad \Pr_y[\exists z \ R(x,y,z)] \geq \frac{3}{4} \\ x \notin L \quad &\Longrightarrow \quad \Pr_y[\exists z \ R(x,y,z)] \leq \frac{1}{4} \end{aligned}$$

where $|y|, |z| \leq 2^{\log^k n}$. Note that by repeating the above protocol $c_1 n$ times for some constant c_1, we can reduce the probability of error to $\frac{1}{2^{n+1}}$. Therefore, for every $x \in \{0,1\}^n$, we get

$$\begin{aligned} x \in L \quad &\Longrightarrow \quad \Pr_y[\exists z \ R(x,y,z)] \geq 1 - \frac{1}{2^{n+1}} \\ x \notin L \quad &\Longrightarrow \quad \Pr_y[\exists z \ R(x,y,z)] \leq \frac{1}{2^{n+1}} \end{aligned}$$

where $|y| \leq c_1 n \times 2^{\log^k n} = 2^{\log^k n + \log c_1 + \log n} \leq 2^{\log^c n}$ for some appropriate $c > k$. There are at most 2^n many strings of length n, and for every y the error probability is at most $\frac{1}{2^{n+1}}$. Therefore any random y will be correct on every input string with probability at least $1 - (2^n \times \frac{1}{2^{n+1}}) > 0$. Hence there must be some $\hat{y}, |\hat{y}| \leq 2^{\log^c n}$ such that the following holds for every x of length n:

$$\begin{aligned} x \in L &\implies \exists z\ R(x, \hat{y}, z) \\ x \notin L &\implies \forall z\ \neg R(x, \hat{y}, z) \end{aligned}$$

This shows that $L \in \mathrm{NP}/2^{\log^c n}$. □

Corollary 3.4 *For any constant $k > 0$, there is a constant $c > 0$ such that*

$$\mathrm{coNP} \subseteq \mathrm{AM}[\log^k n] \implies \mathrm{coNP} \subseteq \mathrm{NP}/2^{\log^c n}.$$

Proof This follows directly from Corollary 3.2 and Lemma 3.3. □

4 Quasipolynomial advice for NP

In this section, we study the consequences of the existence of quasipolynomial length (i.e., 2^{polylog}-length) advice for NP. This question was first studied by Buhrman and Homer [BH92]. They showed that if NP has 2^{polylog}-size family of circuits, then the exponential hierarchy collapses to the second level (i.e. $\mathrm{NEXP}^{\mathrm{NP}} = \mathrm{coNEXP}^{\mathrm{NP}}$). In Theorem 4.4, we improve this collapse to $\mathrm{S}_2^{\mathrm{EXP}}$. In Theorem 4.1 we obtain an exponential version of Yap's theorem [Yap83]. We prove that if NP is contained in $\mathrm{coNP}/2^{\mathrm{polylog}}$, then the exponential hierarchy collapses to $\mathrm{S}_2^{exp} \circ \mathrm{P}^{\mathrm{NP}}$. We use this theorem to obtain the main result of this paper, which is Theorem 4.2.

We note that Cai *et al*. [CCHO03] improved Yap's theorem . They use self-reducibility of a language in NP^A (for any set A) to show that $\mathrm{NP} \subseteq \mathrm{coNP/poly} \implies \mathrm{PH} = \mathrm{S}_2 \circ \mathrm{P}^{\mathrm{NP}}$. Theorem 4.1 is somewhat similar in form to the result of Cai *et al*. However, we use a completely different technique from theirs. Furthermore, in Theorem 4.5 below, we will use our technique to give an independent (and hopefully easier) proof of their result.

Theorem 4.1 $\mathrm{NP} \subseteq \mathrm{coNP}/2^{\mathrm{polylog}} \implies \mathrm{NEXP}^{\Sigma_2^{\mathrm{P}}} = \mathrm{coNEXP}^{\Sigma_2^{\mathrm{P}}} = \mathrm{S}_2^{exp} \circ \mathrm{P}^{\mathrm{NP}}$.

Proof Since $\mathrm{S}_2^{exp} \circ \mathrm{P}^{\mathrm{NP}}$ is closed under complement, it suffices to show under the hypothesis that $\mathrm{NEXP}^{\Sigma_2^{\mathrm{P}}} = \mathrm{S}_2^{exp} \circ \mathrm{P}^{\mathrm{NP}}$. Let $L \in \mathrm{NEXP}^{\Sigma_2^{\mathrm{P}}}$ via some exponential-time nondeterministic oracle machine N that has some Σ_2^p language A as an oracle. For any input $x \in \{0,1\}^n$, M runs in 2^{n^k} time. Therefore, any query that N makes to A is also of length 2^{n^k}, and the number of queries is also bounded by 2^{n^k}.

The yes-proof consists of an accepting computation $\mathcal{P}$ of N on x, with queries and their answers. Note that for any query q, $q \in A \Leftrightarrow \exists y_q\ \phi_{q,y_q} \notin \mathrm{SAT}$. When $|q| = 2^{n^k}$, let $|\phi_{q,y_q}|$ be denoted by m (some exponential in n). By our assumption, SAT is in $\mathrm{coNP}/2^{\mathrm{polylog}}$; let us assume that the advice for strings of length m is w, where $|w| = 2^{\mathrm{polylog}(m)} = 2^{n^c}$ for some constant c, and let $B \in \mathrm{coNP}$ be the witness language.

For every query q that is answered "Yes" on the path $\mathcal{P}$, the yes-prover also provides y and ϕ_{q,y_q}. Note that the total length of the yes-proof is bounded by some exponential in n. Also, since the verifier is a P^{NP} machine, it can find out (making one oracle query) whether $\phi_{q,y_q} \in \mathrm{SAT}$. Obviously, if some ϕ_{q,y_q} provided

with an "Yes" query is satisfiable, the verifier rejects x. In the following, therefore, we show how the verifier can verify whether "No" queries are answered correctly on the path $\mathcal{P}$.

The no-proof consists of the advice w for strings in SAT of length m. Note that there is a set $C \in$ coNP such that for any query q,

$$\begin{aligned} q \notin A &\Leftrightarrow \forall y_q\ \phi_{q,y_q} \in \mathrm{SAT} \\ &\Leftrightarrow \forall y_q\ (\phi_{q,y_q}, w) \in B \\ &\Leftrightarrow (q, w) \in C \end{aligned}$$

where $C = \{(q,w) \,\big|\, \forall y_q\ (\phi_{q,y_q}, w) \in B\}$. Therefore, if $q \notin A$, and w is the correct advice string, the verifier can make a query to the NP oracle and determine whether $(q,w) \in C$. If this happens for every query q answered "No" on $\mathcal{P}$ where w is supplied by the no-prover, the verifier accepts x. This does not automatically imply that w is the correct advice string; however, by the definition of the S_2^{exp} operator, we can always assume that one of the provers is correct, and therefore, when they agree (in this case, that $x \in L$) the consensus decision must be correct.

Otherwise, there must be some q such that $(q,w) \notin C$. This may result from the fact that the query is answered incorrectly on $\mathcal{P}$, or it may happen that the advice string is wrong. (As outlined before, the case when both the proofs are wrong need not be considered.) In this case, by making prefix search using the NP oracle, the verifier can construct y_q and from that, it can obtain ϕ_{q,y_q}. The verifier accepts x if and only if its NP oracle says that $\phi_{q,y_q} \in$ SAT.

If $q \notin A$, it must hold that for every y_q, $\phi_{q,y_q} \in$ SAT. Therefore, in particular, for the y_q that is obtained by the prefix search, $\phi_{q,y_q} \in$ SAT as well. On the other hand, if w is the correct advice string, and $q \in A$, then the prefix search will produce the correct y_q for which $\phi_{q,y_q} \notin$ SAT; therefore, the yes-prover is lying, and $x \notin L$. This completes the proof. □

Now we prove our main theorem.

Theorem 4.2 *For every constant k, if* coNP $\subseteq \mathrm{AM}[\log^k n]$*, then* $\mathrm{NEXP}^{\Sigma_2^{\mathrm{P}}} = \mathrm{coNEXP}^{\Sigma_2^{\mathrm{P}}} = \mathrm{S}_2^{exp} \circ \mathrm{P}^{\mathrm{NP}}$.

Proof If every language in coNP has an Arthur-Merlin proof system with $\log^k n$ moves for any $k > 0$, then by Corollary 3.4, we obtain that coNP $\subseteq \mathrm{NP}/2^{\log^c n}$ for some constant $c > 0$. This is equivalent to saying that NP $\subseteq$ coNP$/2^{\log^c n}$. From Theorem 4.1, we get the consequence that $\mathrm{NEXP}^{\Sigma_2^{\mathrm{P}}} = \mathrm{coNEXP}^{\Sigma_2^{\mathrm{P}}} = \mathrm{S}_2^{exp} \circ \mathrm{P}^{\mathrm{NP}}$. This completes the proof. □

We can prove a version of Theorem 4.2 for $(\log n)^{\log\log n}$-round interactive proof for $\overline{\mathrm{SAT}}$. Let NEEXP be the set of languages that can be decided by a nondeterministic Turing machine that takes at most $2^{2^{n^k}}$ time on an input of length n, and let $\mathrm{coNEEXP} = \{L \,\big|\, \overline{L} \in \mathrm{NEEXP}\}$.

Theorem 4.3 *If* $\overline{\mathrm{SAT}} \in \mathrm{AM}[(\log n)^{\log\log n}]$*, then* $\mathrm{NEEXP}^{\Sigma_2^{\mathrm{P}}} = \mathrm{coNEEXP}^{\Sigma_2^{\mathrm{P}}}$.

Proof The proof is similar to the proof of Theorem 4.2. We first observe that $\overline{\mathrm{SAT}} \in \mathrm{AM}[(\log n)^{\log\log n}]$ implies that $\overline{\mathrm{SAT}} \in \mathrm{NP}/2^{(\log n)^{O(\log\log n)}}$. As a consequence, we obtain $\mathrm{NEEXP}^{\Sigma_2^{\mathrm{P}}} = \mathrm{coNEEXP}^{\Sigma_2^{\mathrm{P}}}$. The crucial point to note is that if $m = 2^{2^{n^k}}$, then $2^{(\log m)^{O(\log\log m)}} = 2^{2^{n^c}}$ for some $c > k$. □

In the following theorem, we improve the result of Buhrman and Homer [BH92, Theorem 1], who showed under the same hypothesis that $\mathrm{NEXP}^{\mathrm{NP}} = \mathrm{coNEXP}^{\mathrm{NP}}$.

Theorem 4.4 *If* NP *has* 2^{polylog}*-size family of circuits, then* $\mathrm{NEXP}^{\mathrm{NP}} = \mathrm{coNEXP}^{\mathrm{NP}} = \mathrm{S}_2^{\mathrm{EXP}}$.

Proof Since $\mathrm{S}_2^{\mathrm{EXP}} \subseteq \mathrm{NEXP}^{\mathrm{NP}} \cap \mathrm{coNEXP}^{\mathrm{NP}}$ (Proposition 2.3), it suffices to show that $\mathrm{NEXP}^{\mathrm{NP}} = \mathrm{S}_2^{\mathrm{EXP}}$.

Let $L \in \mathrm{NEXP}^{\mathrm{NP}}$ be accepted by a nondeterministic machine N with SAT as an oracle. There is some $k > 0$ such that N runs in time 2^{n^k} on any input of length n. Therefore, the formulas queried by N on any input of length n are of size $m \leq 2^{n^k}$, and therefore, have circuit size $2^{\mathrm{polylog}(m)} = 2^{n^c}$ for some c.

On an input $x, |x| = n$, the yes-proof consists of the following:

- Accepting path $\mathcal{P}$ of N on x, including the queries made on that path and their answers
- One satisfying assignment for every query ϕ that is answered "yes"

The verifier can verify whether the assignments provided with each query that is answered "yes" is satisfying, and will reject x if any of them is not satisfying. Therefore, in the following, we consider the queries that are answered "no" on $\mathcal{P}$.

We can assume that any circuit for SAT outputs not only 1 or 0 indicating whether the input formula is satisfiable or not, but also outputs a satisfying assignment when it claims that the input formula is satisfiable. This can be done by a polynomial blow-up in the size of the circuit, and therefore, the circuit still remains of the size 2^{polylog}. The no-proof is such a circuit C for SAT at length m.

Given C, the verifier inputs each query ϕ that is answered "no" on $\mathcal{P}$. If C outputs 0 on all these formulas, indicating that they are unsatisfiable, then the verifier accepts x. On the other hand, if C outputs a satisfying assignment for some ϕ that is answered "no" on $\mathcal{P}$, then the verifier rejects x.

If $x \in L$, then there must be an accepting path of N on x, and the yes-prover can provide this path with the queries and their correct answers. No circuit (correct or otherwise) can output a satisfying assignment on any formula that is answered "no" correctly on this path. On the other hand, if $x \notin L$, the path provided by the yes-prover must be wrong. There are two possible scenarios in this case. First, some query that is answered "yes" on this path is unsatisfiable and is not satisfied by the assignment that is provided by the yes-prover, in which case the verifier rejects x. Otherwise, some formula that is answered "no" is unsatisfiable, and a correct circuit for SAT (provided by the no-prover) can output a satisfying assignment for that formula. In this case, the verifier rejects x as well. This completes the proof. □

Now we improve Yap's theorem.

Theorem 4.5 *If* $\mathrm{NP} \subseteq \mathrm{coNP/poly}$*, then* $\mathrm{PH} = \mathrm{S}_2 \circ \mathrm{P}^{\mathrm{NP}}$.

Proof Since $\mathrm{S}_2 \circ \mathrm{P}^{\mathrm{NP}}$ is closed under complement, it suffices to show under the hypothesis that $\mathrm{NP}^{\Sigma_2^{\mathrm{P}}} = \mathrm{S}_2 \circ \mathrm{P}^{\mathrm{NP}}$. Let $L \in \mathrm{NP}^{\Sigma_2^{\mathrm{P}}}$ via some polynomial-time nondeterministic oracle machine N that has some Σ_2^p language A as an oracle. For any input $x \in \{0,1\}^n$, M runs in n^k time. Therefore, any query that N makes to A is also of length n^k, and the number of queries is also bounded by n^k.

The yes-proof consists of an accepting computation $\mathcal{P}$ of N on x, with queries and their answers. Note that for any query q, $q \in A \Leftrightarrow \exists y_q \ \phi_{q,y_q} \notin \mathrm{SAT}$. When $|q| = n^k$, let $|\phi_{q,y_q}|$ be denoted by m (some polynomial in n). By our assumption, SAT is in coNP/poly; let us assume that the advice for strings of length m is w, where $|w| = n^c$ for some constant c, and let $B \in \mathrm{coNP}$ be the witness language.

For every query q that is answered "Yes" on the path $\mathcal{P}$, the yes-prover also provides y and ϕ_{q,y_q}. Also, since the verifier is a P^{NP} machine, it can find out (making one oracle query) whether $\phi_{q,y_q} \in \mathrm{SAT}$. Obviously, if some ϕ_{q,y_q} provided with an "Yes" query is satisfiable, the verifier rejects x. In the following, therefore, we show how the verifier can verify whether "No" queries are answered correctly on the path $\mathcal{P}$.

The no-proof consists of the advice w for strings in SAT of length m. Note that there is a set $C \in \mathrm{coNP}$ such that for any query q,

$$\begin{aligned} q \notin A &\Leftrightarrow \forall y_q \ \phi_{q,y_q} \in \mathrm{SAT} \\ &\Leftrightarrow \forall y_q \ (\phi_{q,y_q}, w) \in B \\ &\Leftrightarrow (q, w) \in C \end{aligned}$$

where $C = \{(q, w) \,|\, \forall y_q \ (\phi_{q,y_q}, w) \in B\}$. Therefore, if $q \notin A$, and w is the correct advice string, the verifier can make a query to the NP oracle and determine whether $(q, w) \in C$. If this happens for every query q answered "No" on $\mathcal{P}$ where w is supplied by the no-prover, the verifier accepts x. This does not automatically imply that w is the correct advice string; however, by the definition of the S_2 operator, we can always assume that one of the provers is correct, and therefore, when they agree (in this case, that $x \in L$) the consensus decision must be correct.

Otherwise, there must be some q such that $(q, w) \notin C$. This may result from the fact that the query is answered incorrectly on $\mathcal{P}$, or it may happen that the advice string is wrong. (As outlined before, the case when both the proofs are wrong need not be considered.) In this case, by making prefix search using the NP oracle, the verifier can construct y_q and from that, it can obtain ϕ_{q,y_q}. The verifier accepts x if and only if its NP oracle says that $\phi_{q,y_q} \in \mathrm{SAT}$.

If $q \notin A$, it must hold that for every y_q, $\phi_{q,y_q} \in \mathrm{SAT}$. Therefore, in particular, for the y_q that is obtained by the prefix search, $\phi_{q,y_q} \in \mathrm{SAT}$ as well. On the other hand, if w is the correct advice string, and $q \in A$, then the prefix search will produce the correct y_q for which $\phi_{q,y_q} \notin \mathrm{SAT}$; therefore, the yes-prover is lying, and $x \notin L$. □

4.1 Interactive Proof Systems

Let $\mathrm{IP}[g(n)]$ denote an interactive proof system with $g(n)$ rounds in the Goldwasser, Micali and Rackoff [GMR85] formalization. Goldwasser and Sipser [GS89] proved that $\mathrm{IP}[g(n)] \subseteq \mathrm{AM}[2g(n) + 4]$ as long as $g(n)$ is bounded by a polynomial. Thus, if $L \in \mathrm{IP}[\log^k n]$, then $L \in \mathrm{AM}[\log^{k+1} n]$. So the following corollary follows immediately from Theorem 4.2.

Corollary 4.6 *If every set in* coNP *has a polylogarithmic-round interactive proof system, then* $\mathrm{NEXP}^{\Sigma_2^{\mathrm{P}}} = \mathrm{coNEXP}^{\Sigma_2^{\mathrm{P}}} = \mathrm{S}_2^{exp} \circ \mathrm{P}^{\mathrm{NP}}$.

5 Conclusions

We have shown that if coNP has polylogarithmic-round interactive proofs then the exponential hierarchy collapses to the third level. Similarly, the double exponential hierarchy collapses if $\overline{\mathrm{SAT}}$ has a $(\log n)^{\log \log n}$-round interactive protocol. An obvious extension would be to obtain consequences of $\overline{\mathrm{SAT}}$ having n^{ϵ}-round interactive proof systems for some $\epsilon < 1$.

One longstanding open problem in this area is to show that if SAT has polynomial-sized circuits, then PH collapses to AM. Since coNP $\subseteq$ AM implies that PH collapses to AM, it suffices to show under this hypothesis that coNP is included in AM. Moreover, Arvind *et al*. [AKSS95] have shown that if SAT has a polynomial-size family of circuits, then $\mathrm{MA} = \mathrm{AM}$. Since $\mathrm{MA} \subseteq \mathrm{S}_2^{\mathrm{P}}$, this would improve the best-known version of Karp-Lipton theorem [KL80] (by Sengupta, reported in Cai [Cai01]).

It is not known whether AM is properly included in $\mathrm{AM}[\mathrm{polylog}]$, but Aiello, Goldwasser and Håstad [AGH90] have shown that AM is properly included in AM[$\mathrm{polylog}$] in a relativized world.

6 Acknowledgements

The authors thank D. Sivakumar for suggesting the question that we address in this paper.

References

[AGH90] W. Aiello, S. Goldwasser, and J. Hastad. On the power of interaction. *Combinatorica*, 10(1):3–25, 1990.

[AKSS95] V. Arvind, J. Köbler, U. Schöning, and R. Schuler. If NP has polynomial-size circuits, then MA = AM. *Theoretical Computer Science*, 137(2):279–282, 1995.

[ALM+92] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and hardness of approximation problems. In *Proceedings of the 33rd Annual IEEE Symposium on Foundations of Computer Science*, pages 14–22. IEEE Computer Society Press, 1992.

[AS92] S. Arora and S. Safra. Approximating clique is NP-complete. In *Proceedings of the 33rd Annual IEEE Symposium on Foundations on Computer Science*, pages 2–13. IEEE Computer Society Press, 1992.

[Báb85] L. Bábai. Trading group theory for randomness. In *Proceedings of the 17th Symposium on Theory of Computing*, pages 421–429. ACM Press, 1985.

[BFL81] L. Bábai, L. Fortnow, and C. Lund. Nondeterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1(1):3–40, 1981.

[BFLS91] L. Bábai, L. Fortnow, L. Levin, and M. Szegedy. Checking computations in polylogarithmic time. In *Proceedings of the 23rd Annual ACM Symopsium on Theory of Computing*, pages 21–31, 1991.

[BH92] H. Buhrman and S. Homer. Superpolynomial circuits, almost sparse oracles, and the exponential hierarchy. In *Foundations of Software Technology and Theoretical Computer Science, 12th Conference, New Delhi, India, December 18-20, 1992, Proceedings*, volume 652 of *Lecture Notes in Computer Science*, pages 116–127. Springer-Verlag, 1992.

[BHZ87] R. B. Boppana, J. Håstad, and S. Zachos. Does co-NP have short interactive proofs? *Information Processing Letters*, 25(2):127–132, 1987.

[BM88] L. Bábai and S. Moran. Arthur-merlin games : a randomized proof system, and a hierarchy of complexity classes. *Journal of Computer and System Sciences*, 36:254–276, 1988.

[BOGKW88] M. Ben-Or, S. Goldwasser, J. Kilian, and A. Wigderson. Multiprover interactive proofs: How to remove the intractability assumptions. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, pages 113–131, 1988.

[Cai01] J-Y. Cai. $\mathrm{S}_2^{\mathrm{P}} \subseteq \mathrm{ZPP}^{\mathrm{NP}}$. In *Proceedings of the 42nd IEEE Conference on Foundations of Computer Science*, pages 620–629, 2001.

[Can96] R. Canetti. On BPP and the polynomial-time hierarchy. *Information Processing Letters*, pages 237–241, 1996.

[CCHO03] J-Y. Cai, V. Chakaravarthy, L. Hemaspaandra, and M. Ogihara. Competing provers yield improved karp-lipton collapse results. In *Proceedings 20th Symposium on Theoretical Aspects of Computer Science*, pages 535–546, 2003.

[FGL+91] U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy. Approximating clique is almost NP-complete. In *Proceedings 32nd Symposium on Foundations of Computer Science*, pages 2–12. IEEE Computer Society Press, 1991.

[GH98] O. Goldreich and J. Hastad. On the complexity of interactive proofs with bounded communication. *Information Processing Letters*, 67(4):205–214, 1998.

[GMR85] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proofs. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, pages 291–304, 1985.

[GS89] S. Goldwasser and M. Sipser. Private coins versus public coins in interactive proof systems. In S. Micali, editor, *Randomness and Computation*, volume 5 of *Advances in Computing Research*, pages 73–90. JAI Press, 1989.

[GVW01] O. Goldreich, S. Vadhan, and A. Wigderson. On interactive proofs with laconic provers. In *Proceedings of the 28th International Colloquium on Automata, Languages, and Programming*, volume 2076 of *Lecture Notes in Computer Science*, pages 334–345. Springer Verlag, 2001.

[HS01] S. Homer and A. Selman. *Computability and Complexity Theory*. Springer-Verlag, 2001.

[KL80] R. Karp and R. Lipton. Some connections between nonuniform and uniform complexity classes. In *Proceedings of the 12th ACM Symposium on Theory of Computing*, pages 302–309, 1980.

[LFKN92] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic methods for interactive proof systems. *Journal of the Association of Computing Machines*, 39(4):859–868, 1992.

[RS98] A. Russell and R. Sundaram. Symmetric alternation captures BPP. *Journal of Computational Complexity*, 7(2):152–162, 1998.

[Sha92] A. Shamir. IP = PSPACE. *Journal of the Association of Computing Machines*, 39(4):869–877, 1992.

[Tod91] S. Toda. PP is as hard as the polynomial time hierarchy. *SIAM Journal on Computing*, 20:865–877, 1991.

[Yap83] C. Yap. Some consequences of non-uniform conditions on uniform classes. *Theoretical Computer Science*, 26(3):287–300, 1983.

[ZF87] S. Zachos and M. Fürer. Probabilistic quantifiers vs distrustful adversaries. In *Foundations of Software Technology and Theoretical Computer Science, 1987, Proceedings*, volume 287 of *Lecture Notes in Computer Science*, pages 449–455. Springer-Verlag, 1987.

[ZH86] S. Zachos and H. Heller. A decisive characterization of BPP. *Information and Control*, 69:125–135, 1986.