Parallel Repetition of Zero-Knowledge Proofs and the Possibility of Basing Cryptography on NP-Hardness

Rafael Pass
Cornell University
rafael@cs.cornell.edu

January 29, 2007

Abstract

Two long-standing open problems exist on the fringe of Complexity Theory and Cryptography:

1. Does there exist a reduction from an **NP**-Complete Problem to a one-way function?
2. Do parallelized versions of classical constant-round zero-knowledge proofs for **NP** conceal every "hard" bit of the witness to the statement proved?

We show that, unless the Polynomial-Hierarchy collapses, *black-box reductions* cannot be used to provide positive answers to *both* questions.

1 Introduction

Consider the following important and outstanding open questions on the fringe of Complexity Theory and Cryptography:

1. Does there exist a reduction from an **NP**-Complete Problem to a one-way function (OWF)?

2. Do parallelized versions of classical constant-round zero-knowledge proofs for **NP** (such as [26, 9]) conceal every "hard" bit of the witness to the statement proved?

The first question originates in the seminal paper by Diffie and Hellman [11], and is very related to the fundamental and well-studied question of worst-case to average-case reductions. The second question is motivated by the celebrated notion of zero-knowledge proofs [24] and dates back to the Ph.D. thesis of Feige [14]. In this paper we show a somewhat surprising connection between these seemingly unrelated problems. Roughly speaking, our main results shows that, unless the Polynomial-Hierarchy collapses, *black-box reductions* cannot be used to provide positive answers to *both* questions. In other words, with respect to black-box reductions the answer to either question 1 or question 2 is "no" (unless the Polynomial-Hierarchy collapses).

Before further discussing our results, let us start by providing some background on the above questions, as well as a brief overview of known related results.

1.1 Background on the Problems

All previous works concerning both of the above questions exclusively consider *black-box* reductions. Thus, in the remainder of this section we refrain from explicitly stating this.

1.1.1 OWFs Based on NP-Hardness

The notion of a one-way function is at the heart of modern cryptography. It is the most basic cryptographic primitive, and its existence is implied by most other cryptographic notions. Furthermore, by the results of [29, 22], this basic notion implies the existence of private-key encryption schemes. A very fundamental open question is whether the existence of one-way functions can be based on some other "weaker" assumption, such as a worst-case assumption of the type $\mathbf{NP} \not\subset \mathbf{BPP}$. Results indicating both positive and negative answers to the above question have appeared in the literature.

Negative Results One possible approach for ruling out reductions from an **NP**-complete problem to a OWF, is to rule out the possibility of worst-case to average case reductions for **NP**-complete problems. Fortnow and Feigenbaum show that the existence of *non-adaptive random-self reducible* reductions for **NP**-complete languages

implies the collapse of the Polynomial Hierarchy (PH). Impagliazzo [30] rules out the existence of *relativizing* worst-case to average-case reductions for **NP**-complete problems. Bogdanov and Trevisan [8] extend the result of Fortnow and Feigenbaum to show that the existence of a *non-adaptive* worst-case to average-case reduction for an **NP**-complete problem also implies the collapse of the hierarchy. As a corollary, they obtain that the existence of a non-adaptive reduction from an NP-complete problem to a OWF implies the collapse of the hierarchy. A recent results by Akavia et al. [3] obtains a stronger collapse of the PH, by directly considering reductions to a OWF (and thus taking advantage of the additional one-wayness property of the average-case problem at hand).

Another approach for providing a negative answer to question 1 is to rule out the existence of reductions from worst-case problems to functions that are *one-way on the worst-case*. As pointed out in [8], this approach alone is not sufficient to provide a negative answer for general one-way functions, since indeed it is easy to construct a worst-case one-way function based on the assumption that $\mathbf{NP} \not\subset \mathbf{BPP}$.[1] Nevertheless, this approach has been successfully used to rule out the existence of reductions to *specific* types of one-way functions. More than two decades ago, Brassard [6] observes that the existence of a *deterministic* reduction from an **NP**-complete problem to a worst-case one-way *permutation* implies that $\mathbf{coNP} \subseteq \mathbf{NP}$. Very recently, Akavia et al [3] obtain the first impossibility result for arbitrary *probabilistic and adaptive* (black-box) reductions, showing that the existence of reductions from an **NP**-complete problem to a *regular* one-way function with *efficiently recognizable range* (and more generally, so called size-verifiable one-way functions, i.e., one-way function for which there exists an AM proof showing the size of any pre-image set) implies the collapse of the PH. The result of [3] also rules out the existence of reductions from an **NP**-complete problem to functions that are one-way on the worst-case.

Positive results On the positive side, Ajtai [1] shows how to construct a one-way function based on the worst-case hardness of a specific problem (which is in $\mathbf{NP} \cap \mathbf{coNP}$). Other constructions of this type have been presented by e.g. Goldreich-Goldwasser-Halevi [19], Micciancio [27], Micciancio-Regev [28], Ajtai-Dwork [2] and Regev [34]. All the worst-case problems that are used for these constructions are however in $\mathbf{NP} \cap \mathbf{coNP}$ and are therefore unlikely to be **NP**-complete.

Summary To sum up, known negative results concerning question 1, either rule out restricted types of reductions (i.e., *non-adaptive* reductions), or rule out reductions to restricted types of one-way functions (e.g. regular one-way functions with an efficiently recognizable range). Thus, given our current understanding it is still

[1] Consider the function $f(\phi, x) = (\phi, \phi(x))$, where $\phi(x)$ denotes the evaluation of the 3-SAT formula ϕ on the assignment x. Clearly f is efficiently computable. Furthermore, assuming that $\mathbf{NP} \not\subset \mathbf{BPP}$, it holds that f is worst-case hard to invert.

conceivable that there exists an *adaptive* reduction from an **NP**-complete problem to a *one-to-one* OWF!

1.1.2 Parallel Repetition of Zero-Knowledge Proofs

Zero-knowledge proofs, introduced by Goldwasser, Micali and Rackoff [24], are interactive proofs that have the paradoxical property of not revealing any information beyond the validity of the assertion. Very soon after their conception, Goldreich, Micali and Wigderson (GMW) [26] show the existence of a constant-round zero-knowledge *public-coin* (i.e., Arthur-Merlin (AM)) proof system for **NP**, based on the existence of one-way functions. Unfortunately, the constant-round protocol of GMW (as well as a subsequent protocol by Blum [9]) only has constant soundness error. A natural question that arises is whether the *"parallelized"* version of the GMW protocol (i.e., the protocol obtained by parallel repetition of the GMW protocol), which clearly have "small" soundness error, remains zero-knowledge.[2] Goldreich and Krawczyk [20] show that this protocol, and any other constant-round AM proof for **NP** with negligible soundness error cannot be *black-box zero-knowledge*, i.e., the zero-knowledge property cannot be demonstrated by simply using the verifier as a black-box. However, to date it is still unknown whether the parallelized version of the GMW protocol (or any other constant-round AM proof system for **NP** e.g., [9]) "leaks" some "useful" knowledge. In particular, an outstanding open question is whether this protocol, or any other constant-round AM proof system for **NP**, with negligible soundness error, hides all "hard" bits of the witness to the statement proved — it is very conceivable that the parallelized GMW protocol is not zero-knowledge, but still hides all hard bits of the witness![3] As with question 1, results indicating both positive and negative answers to the above question have appeared in the literature.

Positive Results Feige and Shamir [15] define the notion of *Witness Indistinguishability* (WI). Roughly speaking a proof system is WI if it does not reveal (in a computational sense) what witness the prover is using to provide the proof. Interestingly, Feige and Shamir [15] show that the parallelized version of the GMW (and Blum) protocol is WI, providing hope of the possibility of proving that this protocol also hides all hard bits of the witness.

A related and seemingly stronger property, called *Strong Witness Indistinguishability* (sWI) was introduced by Goldreich [17, 18]. Roughly speaking, a proof system

[2] Sequential repetition of the GMW protocol does indeed result in a zero-knowledge proof with negligible soundness error; albeit at the cost of no longer being a constant-round protocol.

[3] A "heuristics" for constructing efficient digital signatures due to Fiat and Shamir [16] relies on the following paradigm. Take a 3-round AM proof with negligible soundness, and replace the verifier challenge with the "hash" of the first prover message, and the message that is supposed to be signed. If the resulting scheme indeed is a digital signature, then it cannot be zero-knowledge (since it clearly reveals something, namely the signature). However, it might still be the case that the signature does not reveal any predicate of the secret key of the signer.

is sWI if proofs of computationally indistinguishable statements are computationally indistinguishable. It is unknown whether the parallelized version of the GMW proof system, or any other constant-round AM proof system for **NP**, with negligible soundness error, is sWI.[4]

Feige and Shamir [15] also define the notion of *Witness Hiding*. Roughly speaking a proof system is witness hiding if it does not reveal the *whole* witness of the statement proved. Although Feige and Shamir obtain witness hiding AM proofs for "hard" languages, it is unknown whether the parallelized GMW protocol (or any other constant-round AM proof for **NP**) is witness hiding.

Pass [32] considers interactive proofs that can be simulated in *quasi-polynomial* time. As is noted in [32], such proof systems have the property of hiding all predicates of the witness that are hard for quasi-polynomial time (as opposed to polynomial-time). Interestingly, such proof systems are significantly simpler to construct than traditional zero-knowledge proofs, and indeed a parallelized version of the GMW proof system *does* satisfy this property (see [12, 32]).

Negative Results As already mentioned, Goldreich and Krawczyk [20] show that constant-round AM proof systems with negligible soundness error, that are black-box zero-knowledge, only exists for languages in $\mathcal{BPP}$. Goldreich and Krawczyk [20] also present certain (artificial) zero-knowledge proof systems that provably reveal the *whole* witnesss of the statement proved, when executed in parallel.

Summary Again, to sum up, previous results regarding question 2 can be divided into two categories: (1) (black-box) impossibility results for AM proofs satisfying *stronger* properties than "bit-hiding", and (2) results showing that parallelized versions of classical zero-knowledge proofs satisfy *weaker* or incomparable notions.

1.2 Our Results

In our treatment we tackle both the above questions in their full generality, i.e., we do not put any restrictions on type of reductions used (more then it being black-box), nor any restrictions on the one-way function. However, rather than addressing each of them separately, we show a connection between them.

As already mentioned, our main result show that unless the PH collapses, black-box reductions cannot be used to provide positive answers to both question 1 and question 2. A bit more precisely (but still informally), we stipulate question 1 as follows.

1. Does there exist a function $f : \{0,1\}^* \to \{0,1\}^*$ and a black-box reduction $\mathcal{R}_1$ from deciding the language **SAT** to inverting f (on the average), i.e., $\mathcal{R}_1^A$

[4] We warn the reader that in the first edition of [17] it was erroneously claimed that the the parallelized version of the GMW protocol indeed satisfied sWI (in a black-box way). We refer the reader to the second edition for further reading [18].

decides **SAT** (with inverse polynomial advantage) given any machine A that inverts f with polynomial advantage.

We generalize question 2 as follows (since we are proving a lower bound this generalization only makes our result stronger).

2. Does there exists a (constant-round) Arthur-Merlin proof system with perfect completeness $(P_{\mathsf{BitHid}}, V_{\mathsf{BitHid}})$ for **SAT**, which is based on the existence of OWFs and has negligible soundness error, and a black-box reduction $\mathcal{R}_2$ for showing that $(P_{\mathsf{BitHid}}, V_{\mathsf{BitHid}})$ conceals every "hard" bit of the witness w to the statement x proved by P_{BitHid}, i.e., $\mathcal{R}_2^A$ (either inverts f with inverse polynomial probability or) predicts bit i of the witness w with inverse polynomial advantage, given any machine A that predicts bit i *with probability 1* after hearing a proof of x.

Note that the parallelized versions of classical zero-knowledge proof systems for **NP** (such as [26] and [9]) indeed are AM proof systems which are based on the existence of OWFs, and have negligible soundness error. Thus, a negative answer to question 2 above, in particular, implies the impossibility of using black-box reductions to demonstrate a bit-hiding property of the parallelized versions of these classical zero-knowledge proof systems.

Our main result states that the if the answer to both the above questions is "yes", then **SAT** $\in$ **coAM** which by a result of Boppana, Håstad and Zachos [5] implies the collapse of the PH (i.e., unless the PH collapses, the answer to at least one of the questions is no).

We also show that the same result holds if replacing Question 2 with the following question regarding *strongly Witness Indistinguishable proofs*:

3. Does there exists a constant-round Arthur-Merlin proof system $(P_{\mathsf{sWI}}, V_{\mathsf{sWI}})$ for SAT, which is based on the existence of OWFs and has negligible soundness error, and a black-box reduction $\mathcal{R}_3$ for showing that $(P_{\mathsf{sWI}}, V_{\mathsf{sWI}})$ is strongly Witness Indistinguishable?[5]

1.3 A Note on Non Black-Box Reductions

Note that it is possible that *non black-box* reductions can be successful where black-box reductions fail. In particular, in a recent result by Barak [4], a constant-round public-coin zero-knowledge *argument* (i.e., with only computational soundness) for **NP** is shown. The striking feature of the protocol of Barak is that the zero-knowledge property is demonstrated using a non black-box simulator. We mention, however, that currently *no* non black-box reductions are known for interactive *proof* systems (as opposed to arguments), or for worst-case to average case reductions.

[5] Again, we warn the reader that a positive answer to this question was erroneously claimed in the first edition of [17]. See the second edition [18] for more details.

1.4 Our Techniques

As observed by Akavia et al [3], it is easy to show that the existence of a reduction from **SAT** to a one-way permutation, implies that **coSAT** $\in$ **AM**: Assume the existence of a reduction $\mathcal{R}$ such that $\mathcal{R}^A$ decides **SAT**, for every machine A that inverts the one-way permutation f. The **AM** proof for **coSAT** would simply let the verifier start by picking a random tape for the reduction $\mathcal{R}$, which is sent to the prover, and then the prover is supposed to provide the answers to all queries made by the reduction on this particular random tape. The verifier accepts if (1) the reduction rejects the instance given the answers provided by the prover, and (2) the prover's answers are consistent with f and the queries made by $\mathcal{R}$. Since f is a one-way permutation, the following properties hold: a) there is only one "correct" answer to the queries of $\mathcal{R}$, and b) the validity of the answer can be efficiently checked. This means that the verifier can efficiently verify that the prover provides *exactly* the same answers as an oracle for inverting the one-way permutation would have.

As pointed out by Akavia et al, if we instead consider a general one-way function (instead of a one-way permutation), the above approach cannot be directly used; the reason being that some queries to the "inversion oracle" might not have any answers at all (which cannot be efficiently checked), or can have many answers. Akavia et al. show how to modify the above approach to work also for reductions to *regular*[6] one-way functions with an *efficiently recognizable* range (or more generally, so called size-verifiable one-way functions, i.e., one-way function for which there exists an AM proof showing the size of any pre-image set). On a very high-level (and very oversimplified), this is obtained by a) using universal hashing [13] to "force" the prover to only have a single possible answer (see e.g [35, 21]), and b) using the efficiently recognizable range condition, in order to let the verifier check when there are no answers to some oracle query.

In our treatment we instead start of with a *general* one-way function, which we transform into an *interactive* "object", i.e., a *game*, that roughly speaking has the same "characteristics" as a one-way permutation — there is only one valid answer to each query asked in the game, and the validity of the answer can be efficiently checked. The construction of this game relies on the existence of a bit-hiding (or strongly witness indistinguishable) AM proof for **NP**, with a black-box reduction. On a very high-level, the construction can be described as follows:

1. We use the OWF to generate a hard instance of a promise-**NP**$\cap$**coNP** problem – this instance is the query.

2. We then rely on the bit-hiding AM proof, to prove that the instance generated indeed is in the promise.

Oversimplifying, the game results in the generation of a "hard" query for which

[6] A function is said to be regular if every element in the range of the function has the same number of inverses.

1. the answer is unique, and

2. there exists a "certificate" (namely the NP-witness) to each valid answer.

Once an appropriate game has been constructed, we then apply the above-mentioned approach to show that, *assuming the existence of a black-box bit-hiding AM proof for* **NP** *which is based on the existence of OWFs*, the existence of a black-box reduction from **SAT** to a OWF implies that **SAT** $\in$ **coAM**. This last step, however, reveals certain technical complication arising from the interactive nature of our game. To cirumvent these issues we inherently rely on the *constant-round*, and *negligible soundness error* properties of the bit-hiding proof system.

Techniques employed One crucial ingredient in the construction of our game is the result of Håstad, Impagliazzo, Luby and Levin [29] providing a black-box reduction from a OWF to a pseudorandom generator. We furthermore employ techniques from the black-box lower-bound for zero-knowledge proofs of Goldreich and Krawczyk [20], the commitment scheme of Naor [31] and the construction of a hard-on-the average promise-**NP** $\cap$ **coNP** problem from a OWF, of Pass and Shelat [33].

1.5 Overview

In Section 2 and 3 we provide some notation and preliminaries. Section 4 contains formal definitions of the problems considered. Section 5 contains a formal statement of our main results and a proof outline. The remainder of the proof is found in Section 6.

2 Notation

We employ the following general notation.

Integer and String representation. We denote by N the set of natural numbers: 0, 1, 2, ... Unless otherwise specified, a natural number is presented in its binary expansion (with no *leading* 0s) whenever given as an input to an algorithm. If $n \in N$, we denote by 1^n the unary expansion of n (i.e., the concatenation of n 1's). Given a string x, we let $x|_i$ denote the i'th bit of x. Given two strings a, b, we let $a||b$ denote the concatenation of a and b.

Probabilistic notation. We employ the following probabilistic notation from [25]. We focus on probability distributions $X : S \to R^+$ over finite sets S.

Probabilistic assignments. If D is a probability distribution and p a predicate, then "$x \stackrel{R}{\leftarrow} D$" denotes the elementary procedure consisting of choosing an element x at random according to D and returning x. according to D until $p(x)$ is true and then returning x.

Probabilistic experiments. Let p be a predicate and $D_1, D_2, \ldots$ probability distributions, then the notation $\Pr[x_1 \stackrel{R}{\leftarrow} D_1;\ x_2 \stackrel{R}{\leftarrow} D_2;\ \ldots : p(x_1, x_2, \ldots)]$ denotes the probability that $p(x_1, x_2, \ldots)$ will be true after the ordered execution of the probabilistic assignments $x_1 \stackrel{R}{\leftarrow} D_1;\ x_2 \stackrel{R}{\leftarrow} D_1;\ \ldots$

New probability distributions. If D_1, D_2, ... are probability distributions, the notation $\{x \stackrel{R}{\leftarrow} D_1; y \stackrel{R}{\leftarrow} D_2; \cdots : (x, y, \cdots)\}$ denotes the new probability distribution over $\{(x, y, \cdots)\}$ generated by the ordered execution of the probabilistic assignments $x \stackrel{R}{\leftarrow} D_1$, $y \stackrel{R}{\leftarrow} D_2, \cdots$.

Probability ensembles. A *probability ensemble* is a vector of random variables $X = \{X_n\}_{n \in N}$. We will consider ensembles of the form $X = \{X_n\}_{n \in N}$ where X_n ranges over strings of length $p(n)$, for some fixed, positive polynomial p.

In order to simplify notation, we sometimes abuse of notation and employ the following "short-cut": Given a probability distribution X, we let X denote the random variable obtained by selecting $x \leftarrow X$ and outputting x.

Algorithms. We employ the following notation for algorithms.

Deterministic algorithms. By a deterministic algorithm we mean a Turing machine. We only consider *finite* algorithms, i.e., machines that have some fixed upper-bound on their running-time (and thus always halt).

Probabilistic algorithms. By a probabilistic algorithms we mean a Turing machine that receives an auxiliary random tape as input. We let the notation "$M(x)$" denote the probability distribution over the outputs of M on input x where each bit of the random tape r is selected at random and independently(note that this is a well-defined probability distribution since we only consider algorithms with finite running-time.)

Interactive Algorithms. We assume familiarity with the basic notions of an *Interactive Turing Machine* [24] (ITM for brevity) and a *protocol*. (Briefly, a protocol is pair of ITMs computing in turns. In each turn, called a round, only one ITM is active. A round ends with the active machine either halting —in which case the protocol halts— or by sending a message m to the other machine, which becomes active with m as a special input. By an interactive algorithm we mean a (probabilistic) interactive Turing Machine.

In this paper we only consider protocols (A, B) where both algorithms A, B receive the *same* string as input; this input string will be denoted the *common input* of A and B.

Given a pair of interactive algorithms (A, B), we let $\langle A, B \rangle(x)$ denote the probability distribution over the outputs of B after interacting with P on the common input x.

Oracle algorithms. An oracle algorithm is an machine that gets oracle access to another machine. Given a probabilistic oracle algorithm M and a probabilistic algorithm A, we let $M^A(x)$ denote the probability distribution over the outputs of the oracle algorithm M on input x, when given oracle access to A.

We will also consider oracle algorithms that get access to *deterministic* interactive algorithms. Given a probabilistic oracle algorihm M, and a *deterministic* interactive algorithm A, we let $M^A(x)$ denote the probability distribution over the outputs of the algorithm M on input x, when given oracle access to the "next-messages" function of A (i.e., the function that on input the messages $(m_1, .., m_l)$ outputs the next messages sent by A on common input x and receiving the messages $(m_1, .., m_l)$; note that this is well defined since we only consider deterministic oracles.)

Efficient Computation We say that an algorithm is *efficient* if its running-time is polynomially bounded. A function is said to be *efficiently computable* if it is computable by a (deterministic) polynomial-time algorithm.

Negligible functions. The term "negligible" is used for denoting functions that are asymptotically smaller than the inverse of any fixed polynomial. More precisely, a function $\nu(\cdot)$ from non-negative integers to reals is called *negligible* if for every constant $c > 0$ and all sufficiently large n, it holds that $\nu(n) < n^{-c}$.

3 Definitions and Preliminaries

In this section we provide some standard definitions and preliminaries.

3.1 Black-box reductions

We start by defining the notion of a black-box reduction.

Definition 1 (Black-box reduction) *A* black-box reduction *is a probabilistic polynomial-time oracle algorithm.*

Remark: For simplicity we only consider *uniform* probabilistic reductions. We mention, however, that an analog of our impossibility results also holds with respect to non-uniform probabilistic reductions if we assume that $\mathbf{SAT} \notin \mathbf{coAM}/poly$ (instead of assuming that $\mathbf{SAT} \notin \mathbf{coAM}$).

3.2 Witness Relations, SAT, and Deciding Languages

We recall the definition of a witness relation for an **NP** language [17].

Definition 2 (Witness relation) *A* witness relation *for a language $L \in \mathbf{NP}$ is a binary relation R_L that is polynomially bounded, polynomial time recognizable and characterizes L by*

$$L = \{x : \exists w \, s.t. \, (x, w) \in R_L\}$$

We say that w is a witness for the membership $x \in L$ if $(x, w) \in R_L$.

Let **SAT** denote the language of all satisfiable formulas, i.e.,

$$\mathbf{SAT} = \{\phi \quad | \quad SAT(\phi) = 1\}$$

where $SAT(\phi) = 1$ if and only if ϕ is satisfiable. Let $R_{\mathbf{SAT}}$ denote some canonical witness relation for the language **SAT**, such that $(\phi, w) \in R_{\mathbf{SAT}}$ if and only if the number of variables is ϕ is $|w|$, and the assignment resulting from letting the i'th variable in ϕ be set to w_i, is satisfying ϕ.

We recall what it means to probabilistically decide a language (with two-sided error).

Definition 3 (Deciding a language) *We say that a probabilistic algorithm A* decides *the language L* with probability *p* on instances of lenght *n if for every $x \in \{0,1\}^n$ the following two conditions hold:*

1. *If $x \in L$, $\Pr[A(x) = 1] \geq p$*
2. *If $x \notin L$, $\Pr[A(x) = 0] \geq p$*

3.3 Inverting functions and One-wayness

We recall the standard definition of what it means to invert a function.

Definition 4 (Inverting functions) *Let $f : \{0,1\}^* \to \{0,1\}^*$ be a function, and let A be a probabilistic algorithm. Then A is said to* invert *the function f* with probability *p* on instances of lenght *n if*

$$\Pr\left[x \leftarrow \{0,1\}^n : A(f(x)) \in f^{-1}(f(x))\right] \geq p$$

Intuitively, a function f is one-way if 1) it is efficiently computable, and 2) efficient algorithm can only invert f with negligible probability.

3.4 Hard Bits

A bit is said to be *computationally hard* if it cannot be *predicted* by a probabilistic polynomial-time algorithm significantly better than a random guess [7]. We here focus on the prediction of bits of witnesses for **NP** languages.

Definition 5 (Predicting a bit) *Let $\mathcal{D}$ be a probability distribution over $\{0,1\}^* \times \{0,1\}^*$, and A be a probabilistic algorithm. Then A is said to* predict *bit i* on $\mathcal{D}$ with probability *p if*

$$\Pr\left[(x,w) \stackrel{R}{\leftarrow} \mathcal{D} : A(x) = w|_i\right] \geq p$$

3.5 Computational Indistinguishability

Our proof will rely on cryptographic proof techniques, and in particular the notion of *computational indistinguishability* [22]. Intuitively, two distributions are computationally indistinguishable if no polynomial-time algorithm can distinguish them.

Definition 6 (Distinguishers) *An probabilistic algorithm D (called the distinguisher) is said to distinguish the probability distributions A, B with probability p if*

$$\left|\Pr\left[a \leftarrow A : D(a) = 1\right] - \Pr\left[a \leftarrow B : D(b) = 1\right]\right| \geq p$$

3.6 Pseudorandom Generators

Intuitively, a string is said to be pseudorandom if it is computationally indistinguishable from a truly random string [7, 36]. A pseudorandom generator is an efficiently computable function that on input a "short" random string, outputs a longer pseudorandom string. We will rely on the following fundamental theorem by Håstad, Impagliazzo, Luby and Levin [29] showing that the existence of one-way functions implies the existence of pseudorandom generators. In fact, the result of [29] shows the existence of *black-box* construction of a pseudorandom generator from a one-way function.

Theorem 1 ([29]) *Let $f : \{0,1\}^* \to \{0,1\}^*$ be a function. Then, there exists a polynomial-time oracle algorithm $\mathbf{g}$, a black-box reduction $\mathcal{R}$, and polynomials $p(\cdot)$, $t(\cdot)$, such that the following holds. Consider any $n \in N$, and any probabilistic algorithm A that distinguishes the following distributions with probability s*

- $\left\{r \stackrel{R}{\leftarrow} \{0,1\}^{t(n)} \quad : \quad \mathbf{g}^f(r)\right\}$
- $\left\{r \stackrel{R}{\leftarrow} \{0,1\}^{3t(n)} \quad : \quad r\right\}$

Then, $\mathcal{R}^A$ inverts f with probability $p(s, \frac{1}{n})$ on instances of lenght n.

3.7 Interactive Proof Systems and Arthur-Merlin Games

The notions of *Interactive Proof Systems* and *Arthur-Merlin Games* were respectively and independently introduced by Goldwasser, Micali and Rackoff [24], and Babai and Moran [7]. Arthur-Merlin (AM) games/proof are interactive proofs where the Verifier does not use any private randomness.

Definition 7 (Interactive Proof System [24]) *A pair of probabilisitic interactive algorithms (P, V) is called an* interactive proof system *for a language L if V is polynomial-time and the following two conditions hold with respect to some negligible functions $\nu_c(\cdot)$, $\nu_s(\cdot)$:*

- Completeness: *For every $x \in L$ and every $n \in N$*

$$\Pr\Big[\langle P, V\rangle(1^n, x) = 1\Big] \geq 1 - \nu_c(|x|)$$

- Soundness: *For every $x \notin L$, every $n \in N$ and every interactive algorithm P^**

$$\Pr\Big[(\langle P^*, V\rangle(1^n, x) = 0\Big] \geq 1 - \nu_s(|x|)$$

We say that an interactive proof has perfect completeness if $\nu_c = 0$.

Remark: Note that in the above definition there are no requirements on the input 1^n. Looking forward, this input (usually called the security parameter) is used to specify the level of some additional *computational* security property (such a bit-hiding or strong witness indistinguishability). Whenever we consider interactive proofs without such "secrecy" properties, we simply omit this additional parameter.

The definition of interactive proofs can also be generalized to consider arbitrary completeness and soundness bounds $c(\cdot)$ and $s(\cdot)$ (by exchanging the right-hand sides of the completeness condition, and soundness condition respectively to $c(|x|)$, and $s(|x|)$.

We proceed to define Arthur-Merlin Proof systems.

Definition 8 (Arthur-Merlin Proof System [7]) *Let (P, V) be an interactive proof system for the language L. Then (P, V) is said to be an* Arthur-Merlin (AM) proof system *if the following conditions hold:*

1. *(P, V) is a public-coin protocol, i.e., the messages that V sends to P are disjoint subsets (of fixed length) of its random tape.*
2. *(P, V) is a constant-round protocol.*

Let **AM** denote the class of languages having Arthur-Merlin proof systems, and let **coAM** denote the class of languages L such that the complement of L (denoted $\bar{L}$) has an Arthur-Merlin proof systems.

Remarks:

1. We have preferred to use the name Arthur-Merlin *proof systems* instead of Arthur-Merlin *games* as it was originally denoted in [7] in order to emphasize the fact that such proof systems are special cases of interactive proof systems.

2. Note that in our definition of Arthur-Merlin proof systems we require that the proof system is constant-round. We remark that although this was not part of the original definition by Babai and Moran, it has today become the standard way of defining the class **AM**.[7]

4 Formal Definitions of the Questions

We provide formal definitions of the questions mentioned in the introduction.

4.1 Reductions from SAT to a OWF

The existence of a black-box reduction for providing a positive answer to Question 1 is stated in the Assumption below.

Assumption 1 (Reduction from SAT to a OWF) *There exists some efficiently computable function* $\mathbf{f} : \{0,1\}^* \to \{0,1\}^*$*, a black-box reduction* $\mathcal{R}_1$*, and polynomials* $t_1(\cdot), p_1(\cdot)$*, such that the following holds. Consider any* $n \in N$*, and any probabilistic algorithm* A *that inverts* $\mathbf{f}$ *with probability* s *on instances of lenght* $t_1(n)$*. Then,* $\mathcal{R}_1^A$ *decides* **SAT** *with probability* $\frac{1}{2} + p_1(s, \frac{1}{n})$ *on instances of lenght* n*.*

4.2 Reductions for Bit-hiding AM proofs

We formally state the assumption that there exist a black-box reduction for providing a positive answer to Question 2. Recall that Question 2 considers AM proof systems that are based on the existence of OWFs. To make Assumption 2 as weak as possible we do not require that the AM proof makes black-box use of this function; rather we consider a class of AM proofs which is specified by the OWF. Furthermore, to make the assumption slighty weaker, it will be sufficient for us to consider an AM proof for **SAT** that simply hides the *first* bit of the witness for the witness relation $R_{\mathbf{SAT}}$.

Assumption 2 (Bit-hiding AM proofs from OWFs) *There exists a class of AM proof systems with perfect completeness* $\left\{(P^f_{\mathsf{BitHid}}, V^f_{\mathsf{BitHid}})\right\}_{f \in \{0,1\}^* \to \{0,1\}^*}$ *for* **SAT***, a*

[7] Goldwasser and Sipser [23] show that any proof system were the verifier uses private randomness can be transformed into a interactive proof system were the verifier only uses public randomness. Thus, in order to distinguish the class **AM** of languages having the Arthur-Merlin proof systems, from the class **IP** of languages having (general) interactive proof systems, the constant-round restriction was put on the class **AM**.

black-box reduction $\mathcal{R}_2$, and a polynomial $p_2(\cdot)$, such that the following holds. Consider any $n, l \in N$, any probability distribution $\mathcal{D}_n$ over $R_{\mathbf{SAT}} \cap \{0,1\}^l \times \{0,1\}^*$, any efficiently computable function $f : \{0,1\}^* \to \{0,1\}^*$, and any determinstic *interactive algorithm A such that*

$$\Pr\left[(x,w) \xleftarrow{R} \mathcal{D}_n : \langle P^f_{\mathsf{BitHid}}, A\rangle(1^n, x) = w|_1\right] = 1$$

Then, either $\mathcal{R}_2^A$ inverts f with probability $\frac{1}{p_2(n,l)}$ on instances of lenght n or $\mathcal{R}_2^A$ predicts bit 1 on $\mathcal{D}_n$ with probability $\frac{1}{2} + \frac{1}{p_2(n,l)}$.

Remarks:

1. Note that Assumption 2 only guarantees the existence of an AM proof with a very "weak" bit-hiding property. In particular, we only require a reduction from an algorithm that predicts the first bit with probability 1 after hearing a proof, to an algorithm that predicts it with only with a *noticeable* advantage (i.e, with only an inverse polynomial advantage over a random guess), without hearing the proof.

2. We note that the perfect completeness condition in Assumption 2 can be relaxed at the price of instead assuming that the reduction works for any algorithm that predicts the first bit with probability 1-$\nu(|x|)$ for some negligible function ν. For clarity and simplicity of exposition, we have chosen not to pursue this path.

4.3 Reductions for Strong-WI AM Proofs

We also consider an alternative assumption to Assumption 2. Assumption 3, below, stipulates the existence of a AM proof for **NP** that satisfies a weak form of *strong Witness Indistinguishability* (sWI) which is provable using a black-box reduction. (Recall that roughly speaking, a proof system is sWI if proofs of computationally indistinguishable statements are computationally indistinguishable.)

Assumption 3 (Strong-WI AM proofs from OWFs) *There exists a class of AM proof systems with perfect completeness $\left\{(P^f_{\mathsf{sWI}}, V^f_{\mathsf{sWI}})\right\}_{f\in\{0,1\}^*\to\{0,1\}^*}$ for* **SAT**, *a black-box reduction $\mathcal{R}_3$, and a polynomial $p_3(\cdot)$, such that the following holds. Consider any $n, l \in N$, any probability two distributions $\mathcal{D}_n^1$, $\mathcal{D}_n^2$ over $R_{\mathbf{SAT}} \cap \{0,1\}^l \times \{0,1\}^*$, any efficiently computable function $f : \{0,1\}^* \to \{0,1\}^*$, and any* determinstic *interactive algorithm A such that*

$$\Pr\left[b \xleftarrow{R} \{1,2\}; (x,w) \xleftarrow{R} \mathcal{D}_n^b : \langle P^f_{\mathsf{sWI}}, A\rangle(1^n, x) = b\right] = 1$$

Then, either $\mathcal{R}_3^A$ inverts f with probability $\frac{1}{p_3(n,l)}$ on instances of lenght n, or

$$\Pr\left[b \xleftarrow{R} \{1,2\}; (x,w) \xleftarrow{R} \mathcal{D}_n^b : \mathcal{R}_3^A(x) = b\right] \geq \frac{1}{2} + \frac{1}{p_3(n,l)}$$

Remark: Note again that Assumption 3 only guarantees the existence of an AM proof with a very "weak" sWI property. In particular, the reduction needs only to work for algorithms that predict what distribution the instance came from with probability 1.

5 Main Theorem and Proof Outline

Our main result shows the following:

Theorem 2 *If Assumptions 1 and 2 (or Assumption 1 and 3) are true then* $\mathbf{SAT} \in \mathbf{coAM}$.

By applying the result of Boppana, Håstad and Zachos [5] we thus obtain that the validity of Assumption 1 and 2 (or Assumption 1 and 3) implies the collapse of the Polynomial Hierarchy.

5.1 Proof Outline

Recall that we wish to prove that if Assumption 1 and 2 (or Assumption 1 and 3) hold, then $\mathbf{SAT} \in \mathbf{coAM}$. Towards this goal, we proceed in several steps.

1. We start by assuming that Assumption 2 (or 3) holds. Given any efficiently computable function f, we next define an *interactive* game G^f. The construction of this game heavily relies on Assumption 2 (or 3) and Theorem 1 (i.e., the construction of a PRG from any one-way function of [29]). We furthermore show a black-box reduction $\mathcal{R}_G$ from inverting the function f to "winning" in the game G^f —in other words, consider any player A that wins in the game G^f; then $\mathcal{R}_G^A$ inverts the function f (with inverse polynomial probability). On a very high-level, the game G^f will consist of an interaction between a challenge-generator CG and the player A. Roughly speaking, CG and A will exchange messages, in order to finally generate an instance that (with overwhelming probability) will be in the promise of a promise-$\mathbf{NP} \cap \mathbf{coNP}$ problem. The player A is then supposed to decide the generated instance, and is considered to win, when it is able to do so. We hint that in the process of generating this instance, CG employs the the bit-hiding (or sWI) AM proof (from Assumption 2 or 3) in order to prove to the player A that an instance in the promise has been generated. The bit-hiding (or sWI) property of the AM proof is used to guarantee that this proof does not significantly help A to decide the generated instance.

2. We next assume that also Assumption 1 holds. This concludes that there exists some particular efficiently computable function $\mathbf{f}$, and a reduction $\mathcal{Z}$ from deciding **SAT** to winning in the game $G^{\mathbf{f}}$.

3. In the final step we show that for any language L and any efficiently computable function f, the existence of a black-box reduction $\mathcal{Z}$ from deciding the language L to winning in the game G^f, implies that $L \in \mathbf{coAM}$. We here rely on the fact that with overwhelming probability there exist a witness (for containment or non-containment) for the "instances" generated in the interactive game G^f. The main technical challenge here is to show that (with high-probability) the reduction $\mathcal{Z}$ will not be able to generate instances that are not in the promise of our promise-$\mathbf{NP} \cap \mathbf{coNP}$ problem. Intuitively, this should follow from the soundness of the AM proof used in the game. However, since the reduction might actually rewind and restart the oracle it has access to, "regular" soundness of the AM proof is not sufficient. We overcome this problem by relying on the the public-coin, constant-round, and "unconditional" soundness[8] of the AM proof, and using techniques similar to those of Goldreich and Krawczyk [20].

By combining the above three steps we have thus shown that Assumption 1 and 2 (or 3) implies that $\mathbf{SAT} \in \mathbf{coAM}$.

6 Proof of Main Theorem

6.1 Step 1 - Defining the Game G^f.

Given an efficiently computable function f, we define a game G^f. The game G^f consist of an interaction (i.e., an exchange of messages) between a challenge generator CG, and a player A. On a high-level, this interaction proceeds in two stages. The first stage (called the *preamble phase*) consist of an interaction after which the player A (with high-probability) will receive two strings q_0, q_1, one of which will be in the support of a pseudorandom generator (PRG), while the other one is not. In the second stage (called the *Guess Phase*), the task of the player A is to decide which of the strings is in the support of the PRG. Looking ahead, the idea behind such a game is that there exists an **NP** certificate (namely the seed to the PRG) for the validity of the answer of the player.

We remark that the idea of using a pair of strings, one of which is in the range of a PRG, originates from Naor's commitment scheme [31]. Such a construction was also recently used by Pass and Shelat [33] in order to reduce a OWF to a promise-$\mathbf{NP} \cap \mathbf{coNP}$ problem that is hard on the average, and where instances in the promise can be efficiently generated (with overwhelming probability): a pair q_0, q_1 is said to be a YES-instance if q_0 is in the range of the PRG and q_1 is not, and a NO-instance if q_1 is in the range of the PRG but q_0 is not. Note that both YES and NO instances are easy to generate with overwhelming probability, by picking one random string in the domain of the PRG and applying the PRG, and one "long" random string; with

[8] We here emphasize that we need to rely on a interactive *proof* system, and not a so called interactive *argument* which only guarantees "computational" soundness.

overwhelming probability the long random string will not be in the range of the PRG. Furthermore, it holds that randomly generated YES-instances are *indistinguishable* from randomly generated NO-instances, which means that the language is hard to decide (on the promise).

In our application we need to make sure that even a malicious challenge generator CG will not be able to generate instances that are not in the promise of the above promise problem (except with negligible probability). Towards this goal, we therefore augment the above generation procedure in the following ways.

1. Just as in Naor [31], a "coin-tossing" is used to guarantee that the probability that both strings q_0 and q_1 are in the range of the PRG, is negligible.

2. CG is furthermore required to provide a proof that *at least* one of q_0 and q_1 is in the range of the PRG. In order to guarantee that it is still hard to distinguish YES-instances from NO-instance also after receiving such a proof, we here rely on the AM proof $(P^f_{\mathsf{BitHid}}, V^f_{\mathsf{BitHid}})$ from Assumption 2 (or $(P^f_{\mathsf{sWI}}, V^f_{\mathsf{sWI}})$ from Assumption 3).[9]

Defining G^f. More formally, given an efficiently computable function f, and an AM proof system with perfect completeness (P_G, V_G) for **SAT**, game G^f is defined as follows for player A. Let g denote the function $g(x) = \mathbf{g}^f(x)$, where $\mathbf{g}$ is the oracle algorithm guaranteed by Theorem 1 (i.e., let g denote the generator obtained by applying the construction of [29] to the function the function f). Let encode denote the *efficiently computable* function that on input two n-bit strings q_0, q_1 outputs a **SAT**-formula x that is satifiable if and only if the pair (q_0, q_1) is "well-formed", i.e., there exist a bit b' and string $s' \in \{0,1\}^{n/3}$ such that $q_{b'} = g(s'), q_{1-b'} = g(s') \oplus r$. The exact formula x, output, is chosen such that $(x, b'||s') \in R_{\mathbf{SAT}}$ if and only if $b' \in \{0,1\}, s' \in \{0,1\}^{n/3}$ and $q_{b'} = g(s'), q_{1-b'} = g(s') \oplus r$ (in other words, x is chosen such that $b'||s$ will be a witness for x if and only $q_{b'} = g(s'), q_{1-b'} = g(s') \oplus r$.)

Let $t(\cdot)$ be the polynomial specified in Theorem 1. G^f consists of an interaction between the (unbounded) challenge generator CG and the player A on common input 1^n. CG proceeds as follows on input 1^n:

Preamble Phase

1. CG waits to receive a string $r \in \{0,1\}^{3t(n)}$ from A.
2. Let $b \xleftarrow{R} \{0,1\}, s \xleftarrow{R} \{0,1\}^{t(n)}$, $q_b = g(s), q_{1-b} = g(s) \oplus r$.
3. Let $x = \mathsf{encode}(q_0, q_1)$.
4. CG sends x to A.

[9] At first sight it might seem that the "coin-tossing" is unnecessary since the generating party could simply give a proof that *exactly* one of q_0 and q_1 is in the range of the PRG. Note, however, that this statement is not in **NP** and can therefore not be proven in a constant number of rounds.

5. CG uses the AM proof system (P_G, V_G) to provide an interactive proof to A that $x \in \mathbf{SAT}$, i.e., CG executes $P_G(1^n, x)$ with A.

Guess Phase

1. After the preamble phase has finished CG waits to receive a "guess" $\tilde{b}$ from A.
2. CG outputs win if $\tilde{b} = b$. Otherwise it outputs loose.

We say that a player A *wins* in the game G^f on input 1^n, if CG outputs win with probability 1 when interacting with A, on common input 1^n.

Proposition 1 *Suppose that Assumption 2 is true, and that the proof system (P_G, V_G) in the definition of the game G^f is $(P_{\mathsf{BitHid}}, V_{\mathsf{BitHid}})$. Then, there exists a polynomial $p(\cdot)$ and a black-box reduction $\mathcal{R}_G$ such that the following holds. Consider any n, and any deterministic interactive algorithm A that wins in the game G^f on input 1^n. Then, $\mathcal{R}_G^A$ inverts f with probability $\frac{1}{p(n)}$ on instances of lenght n.*

Proof: The proposition follows using "standard" cryptographic proof techniques. We show that there exists a reduction $\mathcal{R}$, and a polynomial $p(\cdot)$ that satisfies the conditions of the proposition.

Consider any n, and any deterministic interactive algorithm A that wins in the game G^f on input 1^n. We show how to use A in a black-box way to invert f with probability $p(n)$ on instances of lenght n. Consider the distribution $\mathcal{D}_n$ defined as follows:

1. Let $r \leftarrow A(1^n)$ (i.e., let r be the first output of A).
2. Let $b \stackrel{R}{\leftarrow} \{0,1\}, s \stackrel{R}{\leftarrow} \{0,1\}^{t(n)}$, $q_b = g(s), q_{1-b} = g(s) \oplus r$.
3. Output $x = \mathsf{encode}(q_0, q_1)$, $w = b||s$.

We first claim that with probability 1, A is able to predict the first bit of the witness w (i.e., b) after interacting with $P_{\mathsf{BitHid}}(1^n, x)$, given a random sample $(x, w) \stackrel{R}{\leftarrow} \mathcal{D}_n$., i.e.,

Claim 1

$$\Pr\left[(x,w) \stackrel{R}{\leftarrow} \mathcal{D}_n : \langle P^f_{\mathsf{BitHid}}, A\rangle(1^n, x) = w|_1\right] = 1$$

Proof: The claim directly follows from the fact that the distribution $\mathcal{D}_n$ together with P_{BitHid} perfectly emulate G^f on input 1^n (note that we here rely on the fact that A is a deterministic machine). ■

By assumption 2, it thus holds that $\mathcal{R}_2^A$ either inverts f with probability $\frac{1}{p_2(n,|x|)}$ on instances of lenght n, or predicts the bit 1 on $\mathcal{D}_n$ with probability $\frac{1}{2} + \frac{1}{p_2(n,|x|)}$. Since

$|x|$ is polynomially related to n, it holds that there exists some polynomial $p'_2(\cdot)$ such that $\mathcal{R}_2^A$ either inverts f with probability $\frac{1}{p'_2(n)}$ on instances of lenght n, or predicts the bit 1 on $\mathcal{D}_n$ with probability $\frac{1}{2} + \frac{1}{p'_2(n)}$. In other words, either $\mathcal{R}_2^A$ inverts f with probability $\frac{1}{p'_2(n)}$ on instances of lenght n, or it distinguishes the following two distributions with probability $\frac{1}{p'_2(n)}$:

- $\left\{ s \stackrel{R}{\leftarrow} \{0,1\}^{t(n)}, r \leftarrow A(1^n) : g(s), r \oplus g(s) \right\}$
- $\left\{ s \stackrel{R}{\leftarrow} \{0,1\}^{t(n)}, r \leftarrow A(1^n) : r \oplus g(s), g(s) \right\}$

Note that if replacing the pseudorandom string with a truly random sequence in the above distributions, the two resulting distributions would be identical, i.e., the following distributions are identical:

- $\left\{ \sigma \stackrel{R}{\leftarrow} \{0,1\}^{3t(n)}, r \leftarrow A(1^n) : \sigma, r \oplus \sigma \right\}$
- $\left\{ \sigma \stackrel{R}{\leftarrow} \{0,1\}^{3t(n)}, r \leftarrow A(1^n) : r \oplus \sigma, \sigma \right\}$

It follows using a hybrid argument [22] that $\mathcal{R}_2^A$ either inverts f (with inverse polynomial probability) on instances of length n or distinguishes (with inverse polynomial probability) between either the following distributions

- $\left\{ s \stackrel{R}{\leftarrow} \{0,1\}^{t(n)}, r \leftarrow A(1^n) : g(s), r \oplus g(s) \right\}$
- $\left\{ \sigma \stackrel{R}{\leftarrow} \{0,1\}^{3t(n)}, r \leftarrow A(1^n) : \sigma, r \oplus \sigma \right\}$

or the following ones

- $\left\{ s \stackrel{R}{\leftarrow} \{0,1\}^{t(n)}, r \leftarrow A(1^n) : r \oplus g(s), g(s) \right\}$
- $\left\{ \sigma \stackrel{R}{\leftarrow} \{0,1\}^{3t(n)}, r \leftarrow A(1^n) : r \oplus \sigma, \sigma \right\}$

Thus, R_2^A can be turned, in a black-box way, into an algorithm that either inverts f on instances of lenght n, or distinguishes the output of $\mathbf{g}$ on a random input of lenght $t(n)$, from a truly random string of lenght $3t(n)$. Finally, by combining the above reduction with the reduction from Theorem 1, we have shown the existence of a black-box reduction $\mathcal{R}_G$, and a polynomial $p(\cdot)$ such that R^A inverts f with probability $\frac{1}{p(n)}$ on instances of lenght n, for any A and n such that A that wins in G^f on input 1^n. This concludes the proof of the proposition. ■

Proposition 2 *Suppose that Assumption 3 is true, and that the proof system (P_G, V_G) in the definition of the game G^f is $(P_{\mathsf{sWI}}, V_{\mathsf{sWI}})$. Then, there exists a polynomial $p(\cdot)$ and a black-box reduction $\mathcal{R}_G$ such that the following holds. Consider any n, and any deterministic interactive algorithm A that wins in the game G^f on input 1^n. Then $\mathcal{R}_G^A$ inverts f with probability $\frac{1}{p(n)}$ on instances of lenght n.*

Proof: We show that there exists a reduction $\mathcal{R}$, and a polynomial $p(\cdot)$ that satisfies the conditions of the proposition. Consider any n, and any deterministic interactive algorithm A that wins in the game G^f on input 1^n. Again, we show how to use A in a black-box way to invert f with probability $p(n)$ on instances of lenght n.

Consider the distributions $\mathcal{D}_n^1$, $\mathcal{D}_n^2$, where $\mathcal{D}_n^b$ is defined exactly as $\mathcal{D}_n$ in the proof of 1 except that the bit b defined in $\mathcal{D}_n$ is no longer randomly chosen, but fixed. More precisely, $\mathcal{D}_n^b$ is defined as follows:

1. Let $r \leftarrow A(1^n)$ (i.e., let r be the first output of A).
2. Let $s \stackrel{R}{\leftarrow} \{0,1\}^{t(n)}$, $q_b = g(s), q_{1-b} = g(s) \oplus r$.
3. Output $x = \mathsf{encode}(q_0, q_1)$, $w = b||s$.

It follows using the same argument as in the proof of Proposition 1 that A distinguishes the distributions $\mathcal{D}_n^1, \mathcal{D}_n^2$ with probability 1, i.e.,

$$\Pr\left[b \stackrel{R}{\leftarrow} \{1,2\}; (x,w) \stackrel{R}{\leftarrow} \mathcal{D}_n^b : \langle P_{\mathsf{sWI}}^f, A\rangle(1^n, x) = b\right] = 1$$

By assumption 3, it thus holds that $\mathcal{R}_3^A$ either inverts f with inverse polynomial probability on instances of lenght n, or distinguishes, with inverse polynomial probability, the following distributions:

- $\left\{s \stackrel{R}{\leftarrow} \{0,1\}^{t(n)}, r \leftarrow A(1^n) : g(s), r \oplus g(s)\right\}_{n\in N}$
- $\left\{s \stackrel{R}{\leftarrow} \{0,1\}^{t(n)}, r \leftarrow A(1^n) : r \oplus g(s), g(s)\right\}_{n\in N}$

We now conclude using exactly the same proof as in Proposition 1, that there exists a reduction from inverting f with inverse polynomial probability to winning in G^f. ■

Remarks:

1. Note that in the proofs of Propositions 1 and 2, we only rely on the perfect completeness, public-coin and bit-hiding (or sWI) properties of the AM proof system. The constant-round, unconditional soundness properties (and again the public-coin property) will be important in the second step of the proof of the main theorem.
2. Also note that if the player A honestly follows its instructions, then except with negligible probability it holds that q_0 and q_1 have the property that one of them is in the support of g while the other is not. This property will also be crucial in the second step of the proof (in fact, looking ahead, we will need a slighty stronger form of this property).
3. Finally, note that the "preamble phase" in the game is not necessarily efficiently computable. This follows from the fact that the AM proof system (P_G, V_G) does not necessarily have an efficient prover strategy.

6.2 Step 2 - Using Both Assumptions 1 and 2 (or 3)

By combining Proposition 1 and Proposition 2 with Assumption 1 we directly get:

Proposition 3 *Suppose that Assumption 1 and 2 (or Assumption 1 and 3) are true, and that the proof system (P_G, V_G) in the definition of the game G^f is $(P_{\mathsf{BitHid}}, V_{\mathsf{BitHid}})$ (or $(P_{\mathsf{sWI}}, V_{\mathsf{sWI}})$). Let $\mathbf{f}$ denote the efficiently computable function specified in Assumption 1. Then, there exists polynomial $p'(\cdot)$, $t'(\cdot)$ and a black-box reduction $\mathcal{Z}$ such that the following holds. Consider any $n \in N$, and any deterministic interactive algorithm A that wins in the game $G^{\mathbf{f}}$ on input 1^n. Then $\mathcal{Z}^A$ decides* **SAT** *with probability $\frac{1}{p'(n)}$ on instances of lenght 1^n.*

Proof: Suppose that Assumption 1 and 2 (or Assumption 1 and 3) are true. Let the $t_1(\cdot), p_1(\cdot)$ be the polynomials specified in Assumption 1. Recall that by Proposition 1 (or 2), there exists a polynomial $p(\cdot)$ and a black-box reduction $\mathcal{R}_G$ such that for every deterministic interactive algorithm A that wins in the game $G^{\mathbf{f}}$ (where (P_G, V_G) is instantiated with $(P_{\mathsf{BitHid}}, V_{\mathsf{BitHid}})$ or $(P_{\mathsf{sWI}}, V_{\mathsf{sWI}})$) on input $1^{t_1(n)}$, it holds that $\mathcal{R}_G^A$ inverts $\mathbf{f}$ with probability $\frac{1}{p(t_1(n))}$ on instances of lenght $1^{t_1(n)}$. By Assumption 1 it thus holds that $\mathcal{R}_1^{\mathcal{R}_G^A}$ decides **SAT** with probability

$$\frac{1}{2} + p_1\Big(\frac{1}{p(t_1(n))}, \frac{1}{n}\Big)$$

on instances of lenght n. Since both $\mathcal{R}_1$ and $\mathcal{R}_G$ are polynomial-time, their composition is so as well. Furthermore, since, $p(\cdot), p_1(\cdot), t_1(\cdot)$ all are polynomials their composition is so all well; the propostion follows. ■

6.3 Step 3 - Showing that SAT $\in$ coAM

We show the following proposition which combined with Proposition 3 yields the main theorem:

Proposition 4 *Let L be a language. Suppose that there exist an efficiently computable function $f : \{0,1\}^* \to \{0,1\}^*$, polynomials $p(\cdot)$, $t(\cdot)$ and black-box reduction $\mathcal{Z}$ such that for every $n \in N$, and every deterministic interactive algorithm A that wins in the game G^f on input $1^{t(n)}$, it holds that $\mathcal{Z}^A$ decides the language L with probability $\frac{1}{2} + \frac{1}{p(n)}$ on instances of lenght 1^n. Then, $L \in$* **coAM**.

Proof: Suppose the hypothesis of the claim is true. Let $m = m(n)$ denote the running time of $\mathcal{Z}$ on inputs of length n. Let $\mathcal{H}_n$ denote the set of m-wise independent hashfunctions $h : \{0,1\}^{\ell(n)} \to \{0,1\}^{\ell(n)}$ where ℓ denotes the total communication complexity of the game G^f on input $1^{t(n)}$.

We define a *specific* (exponential-time) deterministic interactive algorithm $\tilde{A}_n$ which satisfies the following properties: (1) for almost every $h \in \mathcal{H}_n$, $\tilde{A}_n(h)$ wins in the game G^f on input $1^{t(n)}$, and (2) for a random $h \in \mathcal{H}_h$, $\mathcal{Z}$ will (almost) never be

able to convince $\tilde{A}_n(h)$ of any *false* statements, *even* if $\mathcal{Z}$ rewinds or restarts $\tilde{A}_n(h)$ (i.e., even if $\mathcal{Z}$ has oracle access to $\tilde{A}_n(h)$). We then use $\tilde{A}_n$ in order to construct an AM-proof for the complement of L. In this construction, both the above properties are crucial.

Given an $h \in \mathcal{H}_n$, consider the (exponential-time) interactive algorithm $\tilde{A}_n(h)$ defined as follows:

1. When activated on input $1^{t(n)}$, $\tilde{A}_n$ externally sends the first l bits of $h(0)$, where l denotes the length of the first message in G^f on input $1^{t(n)}$. If activated on any other input, $\tilde{A}_n$ instead halts.

2. When requested to send an l bit long random challenge as part of the AM proof (P_G, V_G), $\tilde{A}_n$ applies h to the history of all messages received in the game, to obtain the string c, and externally sends the first l bits of c. (The reason for using the hashfunction h is to guarantee that $\tilde{A}_n$ will (almost) never accept false statements, even if $\tilde{A}_n$ is rewound or restarted.)

3. If the AM proof was accepting, and the bit b is uniquely defined, $\tilde{A}_n$ decides (in exponential-time) what b is and outputs this value. If the AM proof is rejecting $\tilde{A}_n$ outputs rejected-proof. If the bit b is not uniquely defined (this happens if both q_0 and q_1 are in the support of g, or if none of them is so) then $\tilde{A}_n$ outputs fail.

We start by noting that for almost every $h \in \mathcal{H}_n$. $\tilde{A}_n(h)$ wins in the game G^f on input $1^{t(n)}$.

Claim 2 *There exists some negligible function $\nu(\cdot)$ such that for every $n \in N$ it holds that for a fraction $1 - \nu(n)$ of functions $h \in \mathcal{H}_n$, $\tilde{A}_n(h)$ wins in game G^f on input $1^{t(n)}$.*

Proof: Consider a random $h \in \mathcal{H}_n$. If follows using the same counting argument as in Naor [31] that except for a fraction 2^{-l} of all strings $r \in \{0,1\}^{3l}$, it holds that there do not exist strings $s_1, s_2 \in \{0,1\}^l$ such that $g(s_1) \oplus r = g(s_2)$ (since $|g(s_1)| = |g(s_2)| = 3l$). Since the string $r = h(0)$ sent by $\tilde{A}_n$ is truly random if h is a random function in $\mathcal{H}_n$, we conclude that except with negligible probability over the choice of h, we end up in a situation where the bit b is uniquely defined. This in turn means that, whenever such a "good" h is picked, $\tilde{A}_n$ will correctly determine the bit b, *unless the AM proof is rejecting*. However, by the perfect completeness of (P_G, V_G), and from the fact that the honest strategy V_G puts non-zero probability on all possible verifier messages (this follows from the public-coin property of V_G), it holds that the latter event never occurs, i.e., $\tilde{A}_n(h)$ wins in G^f whenever a good h is picked.[10] ■

[10]Note that if we do not make the assumption that (P_G, V_G) has perfect completeness, we can only conclude that $\tilde{A}_n(h)$ guesses the bit b with probability $1 - \mu(n)$ for some negligible function $\mu(\cdot)$. To be able to complete the proof, we thus must instead assume that the bit-hiding (or sWI) reduction also works for such players.

Let $\bar{L}$ denote the complement language of L. We show that the protocol $(P_{\bar{L}}, V_{\bar{L}})$ defined below is a 2-round AM proof for $\bar{L}$. (The proof system $(P_{\bar{L}}, V_{\bar{L}})$ "only" has completeness and soundness bound $\frac{1}{2} + \frac{1}{p'(\cdot)}$ where $p'(\cdot)$ is some polynomial. However, standard repetition can be used to boost both the completeness and soundness bounds.) This concludes that $\bar{L} \in AM$.

The AM proof $(P_{\bar{L}}, V_{\bar{L}})$ for $\bar{L}$: On common input a statement x, $P_{\bar{L}}$ and $V_{\bar{L}}$ proceed as follows:

1. Verifier $V_{\bar{L}}$ sends an $m = m(|x|)$-bit long random string $r_{\mathcal{Z}} \in \{0,1\}^m$ and an random m-wise independent hashfunction $h \in \mathcal{H}_n$, to $P_{\bar{L}}$.

2. Prover $P_{\bar{L}}$ proceeds as follows:

   (a) $P_{\bar{L}}$ initiates an execution of $\tilde{A}_n(h)$.

   (b) $P_{\bar{L}}$ then executes $\mathcal{Z}$ with random coins fixed to $r_{\mathcal{Z}}$, providing it with oracle access to $\tilde{A}_n(h)$.

   (c) $P_{\bar{L}}$ then sends the following to $V_{\bar{L}}$:

      i. The answers $a_1, .., a_m$ of $\tilde{A}_n$ to all queries made by $\mathcal{Z}$.

      ii. For all answers that are part of the Guessing Phase, that are not the string fail or rejected-proof, $P_{\bar{L}}$ also provides a "certificate" of the validity of the answer (i.e., the guess b); the certificate being a string s such that $(x, b||s) \in R_{\mathbf{SAT}}$, where x is the statement proved in the preamble phase preceeding the guess. Note that if the answer is neither of the strings fail or rejected-proof, then by the definition of $\tilde{A}_n$ such a certificate always exists.

3. Verifier $V_{\bar{L}}$ proceeds as follows:

   (a) It checks that $\mathcal{Z}$ answers 0, when executed on random tape r_Z and feed the answers $a_1, ..., a_m$ (received from $P_{\bar{L}}$).

   (b) It checks the "correctness" of the answers $a_1, ..., a_m$. This is done as follows:

      i. For answers a_i that are part of the Preamble Phase, $V_{\bar{L}}$ checks that they have been generated by running $\tilde{A}_n(h)$, in the execution with $\mathcal{Z}$ (with random tape fixed to r_Z and receiving the answers $a_1, ..., a_m$), i.e., that they have been generated by applying h to the questions sent by $\mathcal{Z}$.

      ii. For answers a_i that are part of the Guessing Phase, $V_{\bar{L}}$ proceeds as follows:

         A. It checks that the answer a_i is not fail.

         B. If the answer a_i is rejected-proof, it checks that $\tilde{A}_n$ rejected the AM proof that preceded the guess.

C. If the answer a_i is a bit b, it check that 1) the proof preceeding the guess was accepting, and 2) that the certificate s provided by $P_{\bar{L}}$ is OK, i.e., $(x, b||s) \in R_{\mathbf{SAT}}$, where x is the statement proved in the preamble phase preceeding the guess.

(c) If all checks succeed, $V_{\bar{L}}$ accepts, and rejects otherwise.

We start by noting that since $\mathcal{Z}$ is probabilistic polynomial-time, $V_{\bar{L}}$ will be so as well. The proposition now follows from the following two claims.

Claim 3 *There exists a polynomial $p_c(\cdot)$ such that $(P_{\bar{L}}, V_{\bar{L}})$ has completeness bound $\frac{1}{2} + \frac{1}{p_c(\cdot)}$.*

Proof: Recall that by Claim 2, for all but a negligible fraction of $h \in \mathcal{H}_n$, $\tilde{A}_n(h)$ wins in game G^f on input $1^{t(n)}$. By our assumptions on $\mathcal{Z}$, it thus holds that there exists a polynomial p', such that for all but a negligible fraction of $h \in \mathcal{H}_n$, $\mathcal{Z}^{\tilde{A}_n}(h)$ outputs 0 with probability (at least) $\frac{1}{2} + \frac{1}{p'(|x|)}$ on input $x \in \bar{L} \cap \{0,1\}^n$. That is, there exists a polynomial p'' such that $\mathcal{Z}^{\tilde{A}_n}(h)$ outputs 0 with probability (at least) $\frac{1}{2} + \frac{1}{p''(|x|)}$ on input $x \in \bar{L} \cap \{0,1\}^n$ given a random $h \in \mathcal{H}_n$.

Since $V_{\bar{L}}$ "truthfully" picks the "random tape" for $\mathcal{Z}$ and a random hashfunction $h \in \mathcal{H}_n$ for $\tilde{A}_n$, it follows that $P_{\bar{L}}$'s emulation of $\mathcal{Z}$ and $\tilde{A}_n$, with the random tape and hashfunction supplied by $V_{\bar{L}}$ results in $\mathcal{Z}$ outputting 0 with probability (at least) $\frac{1}{2} + \frac{1}{p''(|x|)}$ on input $x \in \bar{L} \cap \{0,1\}^n$. It thus holds that the probability that $V_{\bar{L}}$ accepts a proof of true statement is $\frac{1}{2} + \frac{1}{p''(|x|)}$ minus the probability that that the answers $a_1, .., a_m$ provided by $P_{\bar{L}}$ are "rejected" in Step 3 (b). Recall that by the definition of $\tilde{A}_n$, a certificate to its answers always exists unless it outputs rejected-proof or fail. Furthermore $\tilde{A}_n$ only outputs rejected-proof if the AM proof in the preamble phase was rejecting, which means that in this case $V_{\bar{L}}$ will always accepts. We conclude that $V_{\bar{L}}$ only rejects the proof in Step 3 (b) if one of the answers $a_1, .., a_m$ is fail. However, by the same counting argument as in Claim 2, it follows that this happens only with negligible probability. (Note that we here again rely on the fact that $\tilde{A}_n$ generates its first messages r in the game as $h(0)$, which is a truly random string is h is randomly chosen in $\mathcal{H}_n$.) It follows that there exist a polynomial p_c such that $V_{\bar{L}}$ accepts proofs of valid statements x with probability at least $\frac{1}{2} + \frac{1}{p_c(|x|)}$. ■

Claim 4 *There exists a polynomial $p_s(\cdot)$ such that $(P_{\bar{L}}, V_{\bar{L}})$ has soundness bound $\frac{1}{2} + \frac{1}{p_s(\cdot)}$.*

Proof: Let $\mathsf{fail}_1(\phi)$ denote the probability that $\mathcal{Z}(\phi)$ with oracle access to $\tilde{A}_n(h)$, where h in randomly chosen in $\mathcal{H}_n$, succeeds in providing an accepting proof to $\tilde{A}_n(h)$ of a statement x for which there exists strings s_1, s_0 so that $(x, 0||s_0) \in R_{\mathbf{SAT}}$ and $(x, 1||s_1) \in R_{\mathbf{SAT}}$.

Let $\mathsf{fail}_2(\phi)$ denote the probability that $\mathcal{Z}(\phi)$ with oracle access to $\tilde{A_n}(h)$, where h in randomly chosen in $\mathcal{H}_n$, succeeds in providing an accepting proof to $\tilde{A_n}(h)$ of a statement x such that $x \notin \mathbf{SAT}$.

We start by showing that if assuming that $\mathsf{fail}_1(\cdot)$ and $\mathsf{fail}_2(\cdot)$ are identically 0, then there exists some polynomial $p'(\cdot)$ such that $V_{\bar{L}}$ rejects the proof of any statement $\phi \notin \bar{L}$ with probability $\frac{1}{2} + \frac{1}{p'(|\phi|)}$. Note that under the (unjustified) assumption that $\mathsf{fail}_1(\cdot)$ and $\mathsf{fail}_2(\cdot)$ are identically 0, it holds that any malicious prover P^* must *honestly* provide the same answers as $\tilde{A_n}(h)$ would, or else $V_{\bar{L}}$ will reject the proof based on the checks in Step 3 (b). This follows since:

1. The validity answers of $\tilde{A_n}$ in the preamble phase of G^f are checked by $V_{\bar{L}}$ (since they are efficiently computable).

2. By our (unjustified) assumption, it holds that unless the proof in the preamble phase was rejecting, there is only one *unique* answer for which a certificate, that is accepted by $V_{\bar{L}}$, exists.

3. Finally, whenever the proof in the preamble phase is rejecting, there is again only one valid answer for the guess phase (namely rejected-proof).

This means that $V_{\bar{L}}$ rejects proofs of any statement $\phi \notin \bar{L}$ (at least) as often as $\mathcal{Z}^{\tilde{A_n}(h)}$ outputs 1 on input $\phi \in L$. By our assumption (on $\mathcal{Z}$) and by claim 2 it holds that there exists some polynomial $p'(\cdot)$ such that for all but a negligible negligible fraction of $h \in \mathcal{H}_n$ it holds that $\mathcal{Z}^{\tilde{A_n}(h)}$ outputs 1 on input $\phi \in L$ with probability at least $\frac{1}{2} + \frac{1}{p'(|\phi|)}$. It follows that there exists some polynomial $p''(\cdot)$ such that $\mathcal{Z}^{\tilde{A_n}(h)}$ outputs 1 on input $\phi \in L$ with probability at least $\frac{1}{2} + \frac{1}{p''(|\phi|)}$ given a random $h \in \mathcal{H}_n$.

We next show that both $\mathsf{fail}_1(\cdot)$ and $\mathsf{fail}_2(\cdot)$ are negligible functions. This concludes that there exists some polynomial p_s such that $V_{\bar{L}}$ rejects any instance $x \notin \bar{L}$ with probability at least $\frac{1}{2} + \frac{1}{p_s(|x|)}$.

Sub-Claim 1 *There exists some negligible function $\mu_1(\cdot)$ such that for every ϕ, $\mathsf{fail}_1(\phi) \leq \mu_1(|\phi|)$.*

Proof: It directly follows, using the same argument as in Claim 2, that only with negligible probability (over $h \in \mathcal{H}_h$) $\mathcal{Z}$ succeeds in proving a statement x to $\tilde{A_n}(h)$ for which there exist strings s_1, s_0 such that $(x, 0||s_0) \in R_{\mathbf{SAT}}$ and $(x, 1||s_1) \in R_{\mathbf{SAT}}$. (Again, this follows from the fact that $\tilde{A_n}(h)$ generates its first messages in the game as $h(0)$, which is a truly random string if h is randomly chosen in $\mathcal{H}_n$.) ■

Sub-Claim 2 *There exists some negligible function $\mu_2(\cdot)$ such that for every ϕ, $\mathsf{fail}_1(\phi) \leq \mu_2(|\phi|)$.*

Proof: Recall that we need to show that the probability that $\mathcal{Z}$ succeeds in convincing $\tilde{A_n}(h)$ of a false statement is negligible. Intuitively, this claim should follow from

the soundness of the AM proof (P_G, V_G). Note, however, that the reduction $\mathcal{Z}$ might "rewind", and restart $\tilde{A}_n(h)$ (since it has oracle access to $\tilde{A}_n(h)$); thus "stand-alone" soundness of the AM proof (P_G, V_G) is in fact not sufficient to argue this claim. In order to circumvent the "rewinding" problem, we rely on the fact that $\tilde{A}_n(h)$ applies the hashfunction h to the messages sent by $\mathcal{Z}$ in order to generate its challenges. (We note that this is essentially the same technique that was used by Goldreich and Krawczyk [20] in order to show the impossibility of *black-box zero-knowledge* AM proofs.[11])

Assume, for contradiction, that there exists some polynomial $p'(\cdot)$ such that for infinitely many n, there exists $\phi \in \{0,1\}^n$ such that $\mathcal{Z}(\phi)$ is able to convince $\tilde{A}_n(h)$ of a false statement with probability $\frac{1}{p'(n)}$, given a random $h \in \mathcal{H}_n$. We show how to transform $\mathcal{Z}$ into an (unbounded) cheating prover P^* that breaks the unconditional soundness of $\langle P_G, V_G \rangle$ (for infinitely many input lengths).

Consider any $\phi \in \{0,1\}^n$ such that $\mathcal{Z}(\phi)$ convinces $\tilde{A}_n(h)$ (for a random h) of a false statement with probability $\frac{1}{p'(n)}$. Let q denote the number of rounds in $\langle P_G, V_G \rangle$, and assume (without loss of generality) that P_G sends the first message in the protocol. Since $\tilde{A}_n(h)$ is a deterministic machine we can also assume without loss of generality that $\mathcal{Z}$ never asks the same query twice to its oracle.

We proceed to constructing a cheating prover P^*. P^* will (probablistically) select a statement x that it will then attempt to prove (using (P_G, V_G)). (By an averaging argument, such a P^* can be turned into a deterministic algorithm that always proves the same statement. Furthermore this new prover has the same success probability as P^*). P^* picks $i_1, i_2, .., i_q \in [1, .., m(n)]$ (note that there are only $\binom{m}{q}$ such sequences; since q is a constant this quantity is thus polynomial in m and consequently also in n). P^* then emulates $\mathcal{Z}(\phi)$ given oracle access to $\tilde{A}_n(h)$ for a randomly chosen $h \in \mathcal{H}_n$, with the following differences:

1. If the i_1'st query to $\tilde{A}_n(h)$ is of "length 1", i.e., if it consists of a first round query $(x||m_1)$, proceed as follows. Select the statement x to prove externally, and forward externally the message m_1 (recall that by the construction of G^f, this message is a first message for the protocol (P_G, V_G)). Upon receiving an answer, return the answer to $\mathcal{Z}$ as if it came from $\tilde{A}_n(h)$.

2. If furthermore the i_2'nd query is of "lenght 2", i.e., if it consists of a second round query $(x'||m_1', m_2)$ which additionally is consistent with the first round query $x||m_1$ (i.e., $x' = x$ and $m_1 = m_1'$), forward externally m_2, and return to $\mathcal{Z}$ the answer received back.

3. Continue in the same manner for all i_j, $j \leq q$.

[11] We, nevertheless, point out that whereas in the proof of [20] also a "computational" pseudo-random function can be used, in our scenario we crucially rely on the fact that the m-wise independent hashfunction is "pseudo-random" also in the presence of *unbounded* distinguishers.

We start by noting that it follows directly by the construction of P^* and by the m-wise independence property of h, that $\mathcal{Z}(\phi)$, in the emulation by P^*, succeeds in producing an accepting proof of a false statement (either to the internally emulated $\tilde{A}_n(h)$, or to the external verifier) with probability $\frac{1}{p'(n)}$ (note that we here rely on the assumption that $\mathcal{Z}$ never asks the same query twice, and that $\mathcal{Z}$'s running time is smaller than m, which in turn implies that $\mathcal{Z}$ makes at most m queries to $\tilde{A}_n$). Furthermore, since the query indexes $i_1, i_2, .., i_q$ are chosen at random, it holds that with probability $\frac{1}{\binom{m}{q}}$ (which is polynomial in n), P^* will externally forward the execution where $\mathcal{Z}$ succeeds in proving the false statement (whenever such an execution exists). We conclude that P^* produces an accepting proof of a false statement x with inverse polynomial probability in n. Since $|x|$ is polynomially related to n, we get that P^* concvinces V_G of some false statement x with inverse polynomial probability in $|x|$. Finally, since by the construction of G^f the lenght of the statements x, proved in G^f, grows with n, we conclude that there exists some polynomial $p''(\cdot)$ such that there exists infinitely many $x \notin \mathbf{SAT}$ for which there exists some machine P^* that concvinces $V_G(x)$ with probability $\frac{1}{p''(x)}$. This contradicts the soundness of the AM proof (P_G, V_G). ■

Remark: Note that the above proof crucially relies on the fact that (P_G, V_G) is *public coin*, has a *constant number of rounds*, is *unconditionally sound*, and has *negligible soundness error*. In particular, note that the unconditional soundness property is due to the fact that the cheating prover P^* must emulate $\tilde{A}_n$, which requires exponential time (in order to decide the bit b). ■

■

7 Acknowledgments

I am very grateful to Adi Akavia, Yevgenis Dodis, Shafi Goldwasser, Johan Håstad, Silvio Micali, Omer Reingold, Alon Rosen, Madhu Sudan and Vinod Vaikuntanathan for many helpful conversations and suggestions. Special thanks to Omer and Alon for critically reflecting on the proof, and to Alon for several simplifications of the presentation. Thanks a lot!

References

[1] M. Ajtai. Generating Hard Instances of Lattice Problems. In *28th STOC*, pages 99–108, 1996.

[2] M. Ajtai, C. Dwork. A Public-Key Cryptosystem with Worst-Case/Average-Case Equivalence. In *29th STOC*, pages 284–293, 1997.

[3] A. Akavia, O. Goldreich, S. Goldwasser and D. Moshkovitz. On Basing One-Way Functions on NP-Hardness. Manuscript.

[4] B. Barak. How to go Beyond the Black-Box Simulation Barrier. In *42nd FOCS*, pages 106–115, 2001.

[5] R. Boppana, J. Håstad, S. Zachos. Does co-NP Have Short Interactive Proofs? Inf. Process. Lett. 25(2), pages 127–132, 1987.

[6] G. Brassard. Relativized Cryptography. In *20th FOCS*, pages 383–391, 1979.

[7] L. Babai and S. Moran. Arthur-Merlin games: A randomized proof system, and a hierarchy of complexity classes. *JCSS*, Vol. 36, pages 254–276, 1988.

[8] A. Bogdanov and L. Trevisan. On Worst-Case to Average-Case Reductions for NP Problems In *44th FOCS*, pages 308–317, 2003

[9] M. Blum. How to prove a Theorem So No One Else Can Claim It. *Proc. of the International Congress of Mathematicians,* Berkeley, California, USA, pages 1444–1451, 1986.

[10] M. Blum, A. De Santis, S. Micali, and G. Persiano. Noninteractive zero-knowledge. *SIAM J. Computing*, 20(6):1084–1118, 1991.

[11] W. Diffie, M. Hellman. New Directions in Cryptography. IEEE Transactions on Information Theory. Issue 22 (6), pages 644–654, 1976.

[12] R. Canetti, O. Goldreich, S. Goldwasser and S. Micali. Resettable Zero-Knowledge. In *32nd STOC*, pages 235–244, 2000.

[13] L. Carter and M. Wegman. Universal Hash Functions. Journal of Computer and System Sciences, 18(2), page 143–154, 1979.

[14] U. Feige. Ph.D. thesis, Alternative Models for Zero Knowledge Interactive Proofs. Weizmann Institute of Science, 1990.

[15] U. Feige and A. Shamir. Witness Indistinguishability and Witness Hiding Protocols. In *22nd STOC*, pages 416–426, 1990.

[16] A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In *Crypto86*, Springer LNCS 263, pages 181–187, 1987

[17] O. Goldreich. *Foundations of Cryptography – Basic Tools.* Cambridge University Press, 2001. First Edition.

[18] O. Goldreich. *Foundations of Cryptography – Basic Tools.* Cambridge University Press, 2002. Second Edition.

[19] O. Goldreich., S. Goldwasser, S. Halevi. Public-Key Cryptosystems from Lattice Reduction Problems. In *Crypto 1997*, pages 112–131, 1997.

[20] O. Goldreich and H. Krawczyk. On the Composition of Zero-Knowledge Proof Systems. *SIAM Jour. on Computing*, Vol. 25(1), pages 169–192, 1996.

[21] O. Goldreich, H. Krawczyk, M. Luby. On the Existence of Pseudorandom Generators. *SIAM Jour. on Computing*, Vol 22(6), pages 1163–1175, 1993.

[22] S. Goldwasser, S. Micali. Probabilistic Encryption. JCSS 28(2), pages 270–299, 1984.

[23] S. Goldwasser, S. Sipser. Private Coins versus Public Coins in Interactive Proof Systems. In *18th STOC*, pages 59–68, 1986.

[24] S. Goldwasser, S. Micali and C. Rackoff. The Knowledge Complexity of Interactive Proof Systems. *SIAM Jour. on Computing*, Vol. 18(1), pages 186–208, 1989.

[25] S. Goldwasser, S. Micali and R.L. Rivest. A Digital Signature Scheme Secure Against Adaptive Chosen Message Attacks. *SIAM Jour. on Computing*, Vol. 17, No. 2, pages 281–308, 1988.

[26] O. Goldreich, S. Micali and A. Wigderson. Proofs that Yield Nothing But Their Validity or All Languages in NP Have Zero-Knowledge Proof Systems. *JACM*, Vol. 38(1), pp. 691–729, 1991.

[27] Micciancio. Almost Perfect Lattices, the Covering Radius Problem, and Applications to Ajtai's Connection Factor. *SIAM Jour. on Computing*, Vol. 34, No. 1, pages 118–169, 2004.

[28] D. Micciancio, O. Regev. Worst-Case to Average-Case Reductions Based on Gaussian Measures. In *45th FOCS*, pages 372–381, 2004.

[29] J. Håstad, R. Impagliazzo, L.A. Levin and M. Luby. Construction of Pseudorandom Generator from any One-Way Function. *SIAM Jour. on Computing*, Vol. 28 (4), pages 1364–1396, 1999.

[30] R. Impagliazzo. A personal view on average-case complexity. In *10th Structure in Complexity Theory Conference*, pages 134-147, 1995.

[31] M. Naor. Bit Commitment using Pseudorandomness. *Jour. of Cryptology*, Vol. 4, pages 151–158, 1991.

[32] R. Pass. Simulation in Quasi-polynomial Time and its Application to Protocol Composition. In *EuroCrypt2003*, Springer LNCS 2656, pages 160–176, 2003.

[33] R. Pass and A. Shelat. Unconditional Characterizations of Non-interactive Zero-Knowledge. In *Crypt2005*, Springer LNCS 3621, pages 118–134, 2005.

[34] O. Regev. New lattice based cryptographic constructions. In *35th STOC*, 407-416, 2003.

[35] L. Valiant, V. Vazirani. NP is as Easy as Detecting Unique Solutions. *Theorerical Computer Science*, Vol 47(3), pages 85–93, 1986.

[36] A. Yao. Theory and Applications of Trapdoor Functions. In *23th FOCS*, pages 80-91, 1982.