Lower Bounds for Non-Black-Box Zero Knowledge*

Boaz Barak† Yehuda Lindell‡ Salil Vadhan§

September 8, 2004

Abstract

We show new lower bounds and impossibility results for general (possibly *non-black-box*) zero-knowledge proofs and arguments. Our main results are that, under reasonable complexity assumptions:

1. There does not exist a two-round zero-knowledge *proof* system with perfect completeness for an **NP**-complete language.

   The previous impossibility result for two-round zero knowledge, by Goldreich and Oren (J. Cryptology, 1994) was only for the case of *auxiliary-input* zero-knowledge proofs and arguments.

2. There does not exist a constant-round zero-knowledge *strong* proof or argument of knowledge (as defined by Goldreich (2001)) for a nontrivial language.

3. There does not exist a constant-round public-coin *proof* system for a nontrivial language that is *resettable zero knowledge*. This result also extends to *bounded-resettable* zero knowledge, in which the number of resets is a priori bounded by a polynomial in the input length and prover-to-verifier communication.

   In contrast, we show that under reasonable assumptions, there does exist such a (computationally sound) *argument* system that is bounded-resettable zero knowledge.

The complexity assumptions we use are not commonly used in cryptography. However, in all cases, we show that assumptions similar to ours are necessary for the above results.

Most previously known lower bounds, such as those of Goldreich and Krawczyk (SIAM J. Computing, 1996), were only for *black-box* zero knowledge. However, a result of Barak (FOCS 2001) shows that many (or even most) of these black-box lower bounds do *not* extend to the case of general zero knowledge.

Keywords: zero knowledge, interactive proof systems, argument systems, non-black-box simulation, pseudorandom generators, randomness extractors

*An extended abstract of this paper appeared in *FOCS 2003* [BLV].

†Institute for Advanced Study, Princeton, NJ. Email: boaz@ias.edu . Supported by NSF grants DMS-0111298 and CCR-0324906. Most of this work done while studying in the Weizmann Institute of Science.

‡Department of Computer Science, Bar-Ilan University, Ramat Gan 52900, ISRAEL. E-mail: lindell@cs.biu.ac.il

§Division of Engineering & Applied Science, Harvard University, Cambridge, MA 02138. E-mail: salil@eecs.harvard.edu. URL: http://eecs.harvard.edu/~salil/. Supported by NSF grants CCR-0133096 and CCR-0205423 and a Sloan Research Fellowship.

Contents

1 Introduction **3**
- 1.1 Our Results . . . 5
  - 1.1.1 Lower Bounds (Impossibility Results). . . . 5
  - 1.1.2 Protocols. . . . 5
- 1.2 Organization . . . 6

2 Preliminaries and Definitions **6**
- 2.1 Basic Notations . . . 6
- 2.2 Interactive Proofs and Arguments . . . 7
- 2.3 Indistinguishability . . . 9
- 2.4 Zero Knowledge . . . 9

I Negative Results (Lower Bounds) **12**

3 Two-Round Zero Knowledge **12**
- 3.1 Triviality of 2-round Public-Coin ZK Proofs . . . 12
- 3.2 Limitations of 2-round Private-Coin ZK Proofs . . . 14

4 Resettable Zero Knowledge **16**

5 Strong proofs of knowledge **19**
- 5.1 Definitions . . . 19
- 5.2 Triviality . . . 23
- 5.3 Tools: Strong Pseudorandomness . . . 25
- 5.4 Extra-Strong Proofs and Arguments of Knowledge . . . 26
- 5.5 Constant-Round ZK Strong Proofs and Arguments of Knowledge . . . 29

II Protocols **33**

6 Two Round Zero Knowledge **33**
- 6.1 A Non-trivial 2-Round (Private-Coin) ZK Proof? . . . 33
- 6.2 Zero Knowledge from the Power of Nondeterminism . . . 37
- 6.3 A Two-Round Zero-Knowledge Argument for **NP**? . . . 39

7 A Constant-Round Public-Coin Bounded-Resettable ZK Argument **42**
- 7.1 Proof of Theorem 7.1 . . . 42
- 7.2 Analyzing Protocol 7.2 . . . 44

8 Zero Knowledge from a CIRCUIT SAT **Algorithm** **46**
- 8.1 The protocol and its analysis . . . 48
  - 8.1.1 Proof of Lemma 8.3 . . . 52

III Conclusions and Open Problems **55**

9 Constant-Round Public-Coin Zero-Knowledge Proofs **55**

10 Conclusions **60**

IV Appendices **66**

A Uniform Zero Knowledge **66**
A.1 Definition 66
A.2 Triviality 67
A.3 Two-round Zero Knowledge 69

List of Tables

Protocol 3.8: **MA** proof system for $\overline{L}$ 16
Protocol 4.2: A 3-round public-coin proof system 17
Protocol 4.3: Transformed 2-round public-coin proof system 18
Protocol 6.6: Interactive Proof for Disjoint Images 35
Protocol 6.11: Generic bounded auxiliary-input zero-knowledge protocol. 38
Protocol 6.14: Uniform zero-knowledge using two-round universal arguments. 41
Protocol 7.2: bounded-rZK public-coin argument 43
Protocol 8.4: Blum's basic protocol [Blu2] 49
Protocol 9.12: **AM** proof system for $\overline{L}$ 60

1 Introduction

Zero-knowledge proof systems, introduced by Goldwasser, Micali and Rackoff [GMR], are a fundamental notion in cryptography. Since their introduction, there have been a vast number of positive results for constructing them. The fundamental positive result is that every language in **NP** has a zero-knowledge proof (assuming one-way functions exist) [GMW]. However, there were also many works constructing zero-knowledge proofs that satisfy some additional properties. One important example of such a property is having a small number of rounds of interaction. Protocols with a constant number of rounds were constructed by [FS1, BCY, GK1]. Other properties that have been considered include stronger notions of zero knowledge, such as *auxiliary-input* zero knowledge [GO], *concurrent* zero knowledge [DNS] *resettable* zero knowledge [CGGM], and *universally composable* zero knowledge [Can]. These stronger notions are related to whether the protocol remains zero knowledge when executed several times sequentially (in the case of auxiliary-input zero knowledge), concurrently (in the case of concurrent zero knowledge), under a resetting attack (in the case of resettable zero knowledge), or within an arbitrary environment (in the case of universally composable zero knowledge).

Negative results. There are also a few *negative* results for zero-knowledge proofs. Goldreich and Oren [GO] showed that any zero-knowledge proof system for a nontrivial language (i.e., for a language outside **BPP**) must be interactive, and both the verifier algorithm and the prover algorithm must be probabilistic. They also showed that there does not exist a two-round *auxiliary-input* zero-knowledge proof system for a nontrivial language. The results of [GO] do not depend on any assumption and hold also for zero-knowledge *arguments*.[1] It was also shown that it is impossible to obtain *universally composable* zero-knowledge proofs for nontrivial languages (regardless of the number of rounds) [Can].[2]

Negative results for black-box zero knowledge. In addition to these results, there have been a number of negative results for *black-box* zero knowledge. Loosely speaking, a protocol is *black-box* zero-knowledge if the zero-knowledge condition is shown via a universal simulator that only utilizes black-box/oracle access to the verifier's strategy. Black-box zero knowledge is a stronger condition than auxiliary-input zero knowledge, and it is incomparable to concurrent and resettable zero knowledge. We note that until recently, all known zero-knowledge protocols were in fact black-box zero knowledge. Goldreich and Krawczyk [GK2] showed that there is no black-box zero-knowledge proof for a nontrivial language with 3 rounds. They also showed that there is no black-box zero-knowledge proof for a nontrivial language that has a constant number of rounds and is of the *public-coin* type. (A *public-coin* proof system, also known as an *Arthur–Merlin game*, is one in which the verifier's strategy in the proof merely consists of sending random strings as messages, and at the end deciding whether or not to accept by evaluating a polynomial-time predicate on the transcript of the execution.) It has also been shown that any black-box *concurrent* zero-knowledge proof must have at least $\tilde{\Omega}(\log n)$ many rounds, where n is the input length/security parameter [CKPR] (building on [KPR, Ros]). As in [GK2], the results of [CKPR] do not depend on any

[1] Loosely speaking, in an *argument system* (sometimes called a *computationally sound* proof), the soundness requirement is required to hold only against cheating prover strategies that can be implemented by an efficient algorithm. In contrast, a *proof system* is required to be statistically sound, i.e. soundness is guaranteed even against computationally unbounded provers.

[2] Loosely speaking, the reason is that the universally composable definition requires a black-box simulator that cannot rewind the adversary [CKL]. It is not hard to show that without setup assumptions, it is impossible to obtain such a simulator for nontrivial languages.

assumption and hold also for the case of arguments. Other black-box lower bounds have also been shown in [BGGL, BL, Rey].

Non-black-box zero knowledge. The problem with black-box lower bounds is that in almost all applications of zero knowledge, standard (i.e., non-black-box) zero knowledge suffices. Therefore, the question of interest is typically whether there exist standard (i.e., non-black-box) zero-knowledge protocols with certain properties, and not whether there exist black-box zero knowledge protocols with these properties. This point has become much more acute with a recent result of Barak [Bar], that proves the existence of zero-knowledge protocols that are *not black-box* zero knowledge. Specifically, [Bar] constructs a constant-round public-coin zero-knowledge argument for **NP** that remains zero knowledge when composed concurrently any fixed polynomial number of times. The results of [GK2] (and also of [CKPR]) show that such a protocol *cannot* be black-box zero knowledge.

This refutes the belief that a black-box lower bound indicates a non-black-box lower bound and means that an important research direction is to try to find *non-black-box* lower bounds for zero-knowledge proofs. In particular, several natural questions are:

1. Does there exist a (non-auxiliary-input) zero-knowledge proof or argument with 2 rounds?
2. Does there exist an auxiliary-input zero-knowledge proof or argument with 3 rounds?
3. Does there exist a constant-round *concurrent* or *resettable* zero-knowledge proof or argument?
4. Does there exist a constant-round public-coin zero-knowledge *proof* system? (The protocol of [Bar] is a constant-round public-coin *argument*.)
5. Does there exist a constant-round zero-knowledge *strong* proof of knowledge?

   Strong proofs of knowledge, defined by Goldreich [Gol2, Sec. 4.7.6], are proofs of knowledge whereby the knowledge extractor fulfills the following more stringent requirement: If a given prover convinces the honest verifier to accept with nonnegligible probability, then the knowledge extractor runs in strict probabilistic polynomial time and outputs a witness with probability that is close to 1.[3]

Recall that a 3-round, public-coin proof system for **NP** can be obtained from the well-known protocols of [GMW] (for Three-Coloring) or [Blu2] (for Hamiltonicity) via parallel repetition (to make the soundness error negligible, as required). However, it is not known whether these parallelized protocols are zero knowledge, concurrent zero knowledge, or strong proofs of knowledge. (It is easy to see that they are not resettable zero knowledge.) Thus it is even of interest to study Items 2–5 for these particular protocols. Indeed, this was the motivation for the black-box results of [GK2]. However, as mentioned above, recent results show that one cannot infer from the results of [GK2] an implication on standard (non-black-box) zero knowledge.

We remark that Item 4 has more practical significance than may seem at first glance. The reason is a relation between the Fiat-Shamir heuristic [FS2] and zero-knowledge protocols. The Fiat-Shamir heuristic is a way to transform a *constant-round, public-coin* proof or argument system into a noninteractive proof system. It is a very popular heuristic, and its security is an intriguing open question. It is known that this heuristic is completely *insecure* if it is applied to a protocol

[3] We note that [BL] used non black-box techniques in order to construct a constant-round zero-knowledge argument with a *strict* polynomial-time extractor. However, in their protocol, the extraction probability is not close to 1.

that is zero knowledge [DNRS]. Thus, the results of [Bar] imply that there exists an argument system on which the heuristic fails completely (see also [GT]). However, it is not known whether or not this heuristic is sound when applied to (statistically sound) *proofs*. Resolving Item 4 may shed light on this question.

1.1 Our Results

1.1.1 Lower Bounds (Impossibility Results).

In this work, we pursue the question of obtaining non-black-box lower bounds for zero knowledge, and give partial answers to the above questions. In particular, we show that:

1. **Two-round zero knowledge proofs:** Under a reasonable assumption, (namely, that $\mathbf{E} = \mathbf{DTIME}(2^{O(n)})$ has a function of nondeterministic circuit complexity $2^{\Omega(n)}$; see Assumption 3.2), there does *not* exist a 2-round public-coin zero-knowledge *proof* system for any nontrivial language. Under a somewhat stronger assumption and assuming $\mathbf{NP} \neq \mathbf{coNP}$, there does not exist a 2-round zero-knowledge proof system with perfect completeness for any **NP**-complete language.

   The previous impossibility result of [GO] held only for the case of *auxiliary-input* zero knowledge.

2. **Strong proofs and arguments of knowledge:** Under a reasonable assumption (namely, that there exists a pseudorandom generator or one-to-one one-way function that is secure against $2^{\Omega(n)}$-sized circuits), there does not exist a constant-round zero-knowledge strong proof or argument of knowledge for a nontrivial language.

3. **Resettable zero knowledge proofs:** There does not exist a constant-round public-coin *proof* system for a nontrivial language that is auxiliary-input *resettable* zero knowledge (or even "bounded-resettable" zero knowledge).

1.1.2 Protocols.

Unlike previous lower bounds, some of our bounds hold only for the case of (statistically sound) *proof* systems. They also use complexity assumptions, and even ones that are not common in cryptography, such as one-way functions strong against $2^{\epsilon n}$-sized (as opposed to super-polynomial or $2^{n^{\epsilon}}$-sized) circuits, and the existence of functions in **E** that are hard for nondeterministic algorithms. However, we show that such restrictions (i.e., to proofs only) and complexity assumptions are in fact inherent. Specifically, we show that:

1. **Two-round zero knowledge proofs and arguments:**

   (a) Ruling out 2-round zero-knowledge proofs requires some sort of a lower bound on nondeterministic algorithms. Roughly speaking, if **NP** could simulate superpolynomial-time deterministic algorithms, then there exists a 2-round public-coin zero-knowledge proof system for **NP**. This justifies our use of a nondeterministic hardness assumption for proving impossibility for two-round zero-knowledge proofs.

   (b) Under a form of the "Noninteractive CS Proofs" conjecture posed by Micali [Mic], there exists a 2-round public-coin zero-knowledge *argument* system for **NP**. Thus, if Micali's conjecture turns out to be true, some form of 2-round zero knowledge is possible. Viewed

differently, this shows that in order to rule out 2-round zero-knowledge arguments, one must refute Micali's conjecture.

(c) Under the (non-standard) "Knowledge-of-Exponent" assumption suggested by Damgård [Dam], there exists a (private-coin) 2-round zero-knowledge proof system for a promise problem outside of **BPP**. Hence, our negative result for **NP**-complete languages cannot be extended to all nontrivial problems without refuting this assumption.

2. **Resettable zero-knowledge arguments:** Under *standard assumptions,* our lower bound for resettable public-coin proofs does not extend to *arguments*. Specifically, we construct a constant-round public-coin *argument* for **NP** that is bounded-resettable zero knowledge. This result is interesting as a positive result in its own right as it is the first *constant-round* (public-coin) protocol that is bounded-resettable zero knowledge.

3. **Zero-knowledge strong proofs of knowledge:** Ruling out constant-round (or even 3-round public-coin) zero-knowledge strong proofs of knowledge requires some sort of an exponential lower bound on deterministic algorithms (for a problem in **NP**). Thus, an exponential lower bound is *necessary* to rule out constant-round zero-knowledge strong proofs of knowledge. Loosely speaking, we demonstrate this by showing that if Circuit Satisfiability (CSAT) can be solved in subexponential time, then the parallel Hamiltonicity proof system [Blu2] is a zero-knowledge strong proof of knowledge. Thus, proving that parallel Hamiltonicity is not zero knowledge requires proving (or assuming) that CSAT cannot be solved in subexponential time.

1.2 Organization

This paper has two main parts. The first part (Part I; Sections 3–4) contains all the *lower bounds* of this paper. That is, it contains all our impossibility results for various forms of zero-knowledge. The results are all presented under a definition of computational indistinguishability that refers to nonuniform distinguishers (as is common in the literature); a uniform treatment is presented in the appendix. The second part (Part II; Sections 6–8) contains several constructions of zero-knowledge protocols. All of these results are conditional and some of them are based on assumptions that are highly non-standard or even unlikely to be true. Thus many of these results should be interpreted not as positive results per se, but rather as complementing Part I by showing what obstacles one has to face in extending our negative results. (The main exception to the above is our protocol for bounded-resettable zero-knowledge public-coin arguments that is proven assuming only standard assumptions.) Part III (Sections 9 and 10) contains our conclusions and open questions. In particular, it contains discussions of what we consider the main open question of this area which is the existence of constant-round public-coin zero-knowledge proofs for **NP**. We present several conjectures which, if resolved, would shed light on this problem.

2 Preliminaries and Definitions

2.1 Basic Notations

For a finite set $S \subseteq \{0,1\}^*$, we write $x \leftarrow_{\mathrm{R}} S$ to say that x is distributed uniformly over the set S. We denote by U_n the uniform distribution over the set $\{0,1\}^n$. A function $\mu : \mathbb{N} \to [0,1]$ is called *negligible* if $\mu(n) = n^{-\omega(1)}$. We let $\mathrm{neg}(n)$ denote an arbitrary negligible function (i.e., when we

say that $f(n) < \text{neg}(n)$ we mean that *there exists* a negligible function $\mu(n)$ such that for every n, $f(n) < \mu(n)$). Likewise, $\text{poly}(n)$ denotes an arbitrary polynomial.

For a probabilistic algorithm A, we write $A(x;r)$ to denote the output of A on input x and coin tosses r. $A(x)$ is a random variable denoting the output of A for uniformly selected coin tosses. *PPT* refers to probabilistic algorithms (i.e. Turing machines) that run in *strict* polynomial time.

Whenever we refer to reductions and **NP**-completeness in this paper, we assume it is with respect to reductions f that are *non-shrinking*, i.e. there is a constant $\varepsilon > 0$ such that $|f(x)| \geq |x|^{\varepsilon}$ for all $x \in \{0,1\}^*$. The reason is that the security properties of zero-knowledge proofs are traditionally measured as functions of the input length (rather than a separate security parameter). We say that f is *invertible* if it is one-to-one (not necessarily onto) and the inverse is computable in polynomial time.

For a relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$, the *language associated with* R is $L_R = \{x : \exists y (x,y) \in R\}$. For a value x, we denote $R_x = \{y : (x,y) \in R\}$. R is *poly-balanced* if there is a polynomial p such that $(x,y) \in R \Rightarrow |y| \leq p(|x|)$. R is an **NP**-*relation* if it is poly-balanced and decidable in polynomial time. A relation R is **NP**-*complete* if R is an **NP**-relation and for every **NP**-relation R', there exist non-shrinking polynomial-time computable functions $f : \{0,1\}^* \to \{0,1\}^*$ and $g : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ such that $(x,w) \in R' \Rightarrow (f(x), g(x,w)) \in R$ (which implies $x \in L_{R'} \Rightarrow f(x) \in L_R$) and $x \notin L_{R'} \Rightarrow f(x) \notin L_R$. Note that this implies L_R is an **NP**-complete language in the usual sense. (This is essentially Levin's notion of **NP**-completeness [Lev].) We say R is **NP**-*complete via invertible reductions* if the above holds with f and g both being invertible. All the classic **NP**-complete problems come with natural relations that are **NP**-complete via invertible relations.

A relation R is an **MA**-*relation* if it is poly-balanced and there is a PPT A such that $(x,w) \in R \Rightarrow \Pr[A(x,w) = \texttt{accept}] \geq 2/3$ and $x \notin L_R \Rightarrow \Pr[A(x,w) = \texttt{accept}] \leq 1/3$. Note that this is more general than requiring that $R \in \mathbf{BPP}$, because when $x \in L_R$, there may exist some w's such that $1/3 < \Pr[A(x,w) = \texttt{accept}] < 2/3$. **MA** is the class of languages of the form L_R for some **MA**-relation R. We note that under reasonable (but strong) complexity assumptions, it is known that $\mathbf{MA} = \mathbf{NP}$ [IW] (in fact, under Assumption 3.2 both are equal to the class **AM** [MV]).

A *nondeterministic circuit* of input length n is a standard boolean circuit C with $n+m$ input gates, where for every $x \in \{0,1\}^n$, we define $C(x)$ to equal 1 if and only if there exists a $y \in \{0,1\}^m$ such that $C(x;y) = 1$. (We require that the partition of C's input gates into the n standard input gates and the m nondeterministic gates is explicit in the description of C.) Similarly, a Σ_2-*circuit* of input length n is a standard boolean circuit C with $n+m+m'$ input bits, where for every $x \in \{0,1\}^n$, $C(x) = 1$ iff $\exists y \in \{0,1\}^m \forall z \in \{0,1\}^{m'}$ it holds that $C(x;y;z) = 1$.

2.2 Interactive Proofs and Arguments

An *interactive protocol* (A,B) consists of two algorithms that compute the *next-message function* of the (honest) parties in the protocol. Specifically, $A(x,a,\alpha_1,\ldots,\alpha_k;r)$ denotes the next message α_{k+1} sent by party A when the common input is x, A's auxiliary input is a, A's coin tosses are r, and the messages exchanged so far are $\alpha_1,\ldots,\alpha_k$. There are two special messages, `accept` and `reject`, which immediately halt the interaction. We say that party A (resp. B) is *probabilistic polynomial time (PPT)* if its next-message function can be computed in polynomial time (in $|x| + |a| + |\alpha_1| + \cdots + |\alpha_k|$).

For an interactive protocol (A,B), we write $(A(a),B(b))(x)$ to denote the random process obtained by having A and B interact on common input x, (private) auxiliary inputs a and b to A and B, respectively (if any), and independent random coin tosses for A and B. We call (A,B)

polynomially bounded if there is a polynomial p such that for all x, a, b, the total length of all messages exchanged in $(A(a), B(b))(x)$ is at most $p(|x|)$ with probability 1. Moreover, if B^* is any interactive algorithm, then A will immediately halt and reject in $(A(a), B^*(b))(x)$ if the total length of the messages ever exceeds $p(|x|)$, and similarly for B interacting with any A^*.

The number of *rounds* in an execution of the protocol is the *total* number of messages exchanged between A and B, not including the final `accept`/`reject` message. We call the protocol (A, B) *public coin* if all of the messages sent by B are random strings independent of the history, and B's final output (typically `accept` or `reject`) is computed as a deterministic function of the transcript. (Such protocols are also sometimes known as *Arthur-Merlin games* [BM].)

Definition 2.1. An interactive protocol (P, V) is an *interactive proof system* for a language L if there is a polynomially balanced relation R such that $L = L_R$, and functions $c, s : \mathbb{N} \to [0, 1]$ such that $1 - c(n) > s(n) + 1/\text{poly}(n)$ and the following holds:

- (efficiency): (P, V) is polynomially bounded, and V is computable in probabilistic polynomial time.
- (completeness): If $x \in L$ and $w \in R_x$, then V accepts in $(P(w), V)(x)$ with probability at least $1 - c(|x|)$,
- (soundness): If $x \notin L$, then for every P^*, V accepts in $(P^*, V)(x)$ with probability at most $s(|x|)$.

We call $c(\cdot)$ the *completeness error* and $s(\cdot)$ the *soundness error*. We say that (P, V) has *negligible error* if both c and s are negligible. We say that it has *perfect completeness* if $c = 0$.

P is an *efficient prover* if $P(w)$ is computable by a probabilistic polynomial-time algorithm when $w \in R_x$. When we wish to specify the relation R, we call (P, V) an interactive proof *with respect to* R.

The purpose of the relation R is to have a meaningful notion of efficient provers. If the prover's complexity is unbounded, then without loss of generality the prover $P(w)$ can ignore the witness w, and thus we may as well set R to be $R = \{(x, \lambda) : x \in L\}$, where λ is the empty string (we call this the *trivial relation* for L). We note that the class of languages having interactive proofs with efficient provers equals **MA**. In cryptographic applications, we are typically interested in interactive proofs for languages in **NP** with respect to their natural **NP**-relation.

Definition 2.2. An *interactive argument system* (P, V) is defined in the same way as an interactive proof system, with the following modification:[4]

- The soundness condition is replaced with: For every nonuniform PPT P^* and for all sufficiently long $x \notin L$, the verifier V accepts in $(P^*, V)(x)$ with probability at most $s(|x|)$.

We note that if a language $L = L_R$ has an *efficient-prover* argument system wrt relation R, then R is an **MA**-relation and thus $L \in \mathbf{MA}$. To see this, note that the algorithm $A(x, w)$ which simulates $(P(w), V)(x)$ and outputs V's decision constitutes a witness-checking algorithm for R. (The fact that $P(x, w')$ is a PPT algorithm together with the computational soundness condition of arguments imply A rejects with high probability when $x \notin L$.)

[4]Some definitions of argument systems require that the honest prover P is an efficient prover, but we choose to decouple the two requirements and explicitly say "efficient prover" when we need it.

2.3 Indistinguishability

Recall that the formulation of the "zero knowledge" property requires that there exists a PPT simulator whose output distribution is "indistinguishable" from the verifier's view of the interaction. There are several choices for the notion of indistinguishable (which yield different variants of zero knowledge), which we recall below.

Definition 2.3 (perfect and statistical indistinguishability). Two ensembles of probability distributions $\{A_x\}_{x\in I}$ and $\{B_x\}_{x\in I}$ are *statistically indistinguishable* if there exists a negligible function μ such that for every $x \in I$, A_x and B_x have statistical difference at most $\mu(|x|)$. That is, there exists a negligible function μ such that for every function $D : \{0,1\}^* \to \{0,1\}$ and for every $x \in I$,

$$|\Pr[D(A_x) = 1] - \Pr[D(B_x) = 1]| \leq \mu(|x|).$$

$\{A_x\}_{x\in I}$ and $\{B_x\}_{x\in I}$ are *perfectly indistinguishable* if the above holds with $\mu = 0$.

Definition 2.4 ((nonuniform) computational indistinguishability). Two ensembles of probability distributions $\{A_x\}_{x\in I}$ and $\{B_x\}_{x\in I}$ are *computationally indistinguishable* if for every nonuniform PPT algorithm D, there exists a negligible function μ such that for every $x \in I$,

$$|\Pr[D(A_x) = 1] - \Pr[D(B_x) = 1]| \leq \mu(|x|).$$

Even though the above definition does not explicitly say so, we note that the it implies indistinguishability even if the distinguisher D depends (arbitrarily) on the index x. That is, for all families of circuits $\{D_x\}_{x\in I}$ such that $|D_x| \leq \text{poly}(|x|)$, there is a negligible μ such that $|\Pr[D_x(A_x) = 1] - \Pr[D_x(B_x) = 1]| \leq \mu(|x|)$ for all $x \in I$.

The standard definitions of zero knowledge in the literature typically work with nonuniform indistinguishability as above. Goldreich [Gol1] has given a uniform treatment of zero knowledge. Unfortunately, the uniform setting requires much more cumbersome definitions and notation, and so we present uniform analogues of (some of) our results in Appendix A.

2.4 Zero Knowledge

We write $\langle A(a), B(b)\rangle(x)$ to denote B's view of the interaction, i.e. a transcript $(\gamma_1, \gamma_2, \ldots, \gamma_t; r)$, where the γ_i's are all the messages exchanged and r is B's coin tosses.

There are various notions of zero knowledge, referring to how rich a class of verifier strategies are considered. The weakest is to consider only the verifier that follows the specified protocol.

Definition 2.5 (honest-verifier zero knowledge). An interactive proof system (P, V) with respect to relation R is *(perfect/statistical/computational) honest-verifier zero knowledge* if there exists a probabilistic polynomial-time *simulator* S such that the ensembles $\{\langle P(w), V\rangle(x)\}_{(x,w)\in R}$ and $\{S(x)\}_{(x,w)\in R}$ are (perfectly/statistically/computationally) indistinguishable.

We will often drop the word "computational" in reference to computational zero knowledge.

Next we consider all polynomial-time verifier strategies.

Definition 2.6 (plain zero knowledge[5]). An interactive proof system (P, V) with respect to relation R is *(perfect/statistical/computational) plain zero knowledge* if for every PPT V^*, there

[5] In the preliminary version of this paper [BLV], we referred to plain zero knowledge as "uniform zero knowledge." However, as argued in [Gol1], that terminology confuses two orthogonal issues — the uniformity of the verifier and the uniformity of the distinguisher. In the main body of this paper, we always consider distinguishers that are *nonuniform* algorithms. See Appendix A for discussion on the case of *uniform* indistinguishability.

exists a probabilistic polynomial-time *simulator* S such that the ensembles $\{\langle P(w), V^*\rangle(x)\}_{(x,w)\in R}$ and $\{S(x)\}_{(x,w)\in R}$ are (perfectly/statistically/computationally) indistinguishable.

We will often drop the word "computational" in reference to computational zero knowledge.

An equivalent formulation of computational zero knowledge is that for every PPT V^*, there exists a PPT simulator S such that for all families of circuits $\{D_{x,w}\}_{(x,w)\in R}$ such that $|D_{x,w}| \leq \text{poly}(|x|)$, we have

$$|\Pr[D_{x,w}(\langle P(w), V^*\rangle(x))] - \Pr[D_{x,w}(S(x)) = 1]| \leq \text{neg}(|x|)$$

for all $(x, w) \in R$. That is, indistinguishability holds for poly-sized circuits that even can have hardwired into them (a polynomial amount of) nonuniform advice that depends arbitrarily on the input x and witness w. We stress that this nonuniformity refers only to the distinguishers; the verifier V^* and simulator S are still restricted to *uniform* PPT algorithms. As discussed in [Gol2, Sec 4.3.3], the "right" way to incorporate nonuniformity in the verifier is through an auxiliary input given to both it and the simulator, as done below.

Definition 2.7 (auxiliary-input zero knowledge[6]). An interactive proof system (P, V) with respect to relation R is (perfect/statistical/computational) *auxiliary-input zero knowledge* if for every PPT V^* and polynomial p, there exists a PPT S such that the ensembles

$$\{\langle P(w), V^*(z)\rangle(x)\}_{(x,w)\in R, z\in\{0,1\}^{p(|x|)}} \qquad \text{and} \qquad \{S(x,z)\}_{(x,w)\in R, z\in\{0,1\}^{p(|x|)}} \tag{1}$$

are (perfectly/statistically/computationally) indistinguishable.

The auxiliary input z in the above definition models a priori information that the verifier may possess before the interaction begins. Thus auxiliary-input zero knowledge is usually necessary when zero-knowledge proofs are to be used as a subprotocol in a larger protocol, or even when composing zero-knowledge proofs with themselves. Indeed, it is known that auxiliary-input zero knowledge is closed under sequential composition [GO], but plain zero knowledge is not [GK2]. For this reason, auxiliary-input zero knowledge is the definition typically used in the literature.

We note that, effectively, the indistinguishability is required even for distinguishers that have additional a priori information beyond the auxiliary input of the verifier. This is because this information can be hardwired into the nonuniform distinguisher.

We also note that for auxiliary-input zero-knowledge, there is a universal verifier $V^*_{\text{uni}}(x, z)$ that interprets its auxiliary input z as a Boolean circuit C_z and uses C_z as its strategy (i.e. next-message function). If the zero-knowledge condition holds for V^*_{uni}, then it holds for all PPT verifier strategies V^*. However, we allow the simulator to depend not only on V^*_{uni} but also on the polynomial bound $p(n)$ on the size of circuit given to it as auxiliary input. Allowing this dependence makes our lower bounds stronger.

Definition 2.8 (black-box zero knowledge). We say that (P, V) is (perfect/statistical/computational) *black-box zero knowledge* if there exists a single oracle PPT S_{bb}, that works for every V^*, such that

[6]Our formulation of auxiliary-input zero knowledge is slightly different than, but equivalent to, the definition in the text [Gol2]. We allow V^* to run in polynomial time in the lengths of both its the input x and its auxiliary input z, but put a polynomial bound on the length of the auxiliary input. In [Gol2, Sec 4.3.3], V^* is restricted to run in time that is polynomial in just the length of the input x, and no bound is imposed on the length of the auxiliary input z (so V^* may only be able to read a prefix of z). The purpose of allowing the auxiliary input to be longer than the running time of z is to provide additional nonuniformity to the distinguisher (beyond that which the verifier has); we do this directly by allowing the distinguisher to be nonuniform in Definition 2.4.

$S(x,z)$ can be replaced in (1) by $S_{\text{bb}}^{V^*(x,z,\cdot;\cdot)}(x)$ (where $V^*(x,z,\cdot;\cdot)$ denotes the next message function of V^* with a fixed input x and auxiliary input z). That is, for every PPT V^* and polynomial p, the ensembles

$$\{\langle P(w), V^*(z)\rangle(x)\}_{(x,w)\in R, z\in\{0,1\}^{p(|x|)}} \qquad \text{and} \qquad \{S_{\text{bb}}^{V^*(z,x,\cdot)}(x)\}_{(x,w)\in R, z\in\{0,1\}^{p(|x|)}}$$

are (perfectly/statistically/computationally) indistinguishable.

All known zero-knowledge protocols (based on standard assumptions) prior to the paper [Bar] were *black-box* zero knowledge. However, in this paper, because our focus is obtaining lower bounds, we will mainly be interested in general (i.e., not necessarily black box) zero knowledge.

In all the forms of zero knowledge above, we may assume without loss of generality that the simulator for a verifier V^* always outputs *consistent transcripts*, namely transcripts $(\gamma_1, \gamma_2, \ldots, \gamma_t; r)$ such that each V^*-message γ_i is computed correctly, in the sense that $\gamma_i = V^*(x, \gamma_1, \ldots, \gamma_{i-1}; r)$. The reason is that inconsistent transcripts can easily be distinguished from the real interaction, so instead of outputting an inconsistent transcript the simulator may as well output some trivial consistent transcript (eg where all the prover messages are the empty string, and the verifier messages are computed according to V^*).

Zero knowledge and NP-completeness. As argued in [GMW], a zero-knowledge proof for an **NP**-complete languages L implies zero-knowledge proofs for all languages $L' \in \mathbf{NP}$: both parties apply the reduction from L' to L and execute the zero-knowledge proof for L. This transformation clearly preserves most complexity parameters (constant round complexity, negligible error, etc.).[7] *Plain* zero knowledge is preserved if L is **NP**-complete with respect to *invertible reductions* as defined in Section 2.1 (cf., [GMW]). Invertibility is easily achieved in the reductions to most natural **NP**-complete languages, such as HAMILTONICITY, SATISFIABILITY, and THREE-COLORING. However, invertibility is not needed for *auxiliary-input* zero knowledge. We note that, if the zero-knowledge proof for L is with respect to an **NP**-complete *relation* R (such that $L_R = L$), then the reduction of [GMW] yields a zero-knowledge proof for $L' = L_{R'}$ with respect to R' for any **NP**-relation R'. Furthermore, if the zero-knowledge proof for L has an efficient prover with respect to R, then the zero-knowledge proof for L' as an efficient prover with respect to R'.

Expected polynomial-time simulators. For simplicity, in this paper we restrict our attention to simulators that run in *strict* polynomial time. However, all of our negative results hold for expected polynomial-time simulators, and in fact even for *weak* zero knowledge, where the order of quantifiers between the simulator and distinguishing probability is swapped: for every polynomial p, there exists a *strict poly-time* simulator S such that the simulator's output cannot be distinguished from verifier's view except with advantage $1/p(n)$ for sufficiently large n.

[7] Recall that we restrict to non-shrinking reductions.

Part I
Negative Results (Lower Bounds)

In this part we present several impossibility results for non-black-box zero-knowledge. For some of these results we have matching (or almost matching) *positive results*, showing either limitations on extending the negative results, or the necessity of the computational assumptions that we use. These positive results are presented in Part II.

3 Two-Round Zero Knowledge

As mentioned in the introduction, Goldreich and Oren [GO] gave the first impossibility result for 2-round zero knowledge:

Theorem 3.1 ([GO]). *If a language L has a 2-round proof or argument system that is nonuniform* auxiliary-input *zero knowledge, then $L \in \mathbf{BPP}$.*

In this section, we present results that replace "auxiliary-input zero knowledge" with the weaker requirement of "plain zero knowledge," at the expense of relying on complexity assumptions and restricting to *proof* systems. Both of these costs are addressed in Part II of the paper (specifically, Section 6).

3.1 Triviality of 2-round Public-Coin ZK Proofs

Our first result, for *public-coin* proof systems, is based on the following assumption:

Assumption 3.2. $\mathbf{E} = \mathbf{DTIME}(2^{O(n)})$ has a function of nondeterministic circuit complexity $2^{\Omega(n)}$. That is, there exists a language L and constants $c, \epsilon > 0$ such that L is decidable in time 2^{cn}, and yet for every nondeterministic circuit family $\{C_n\}_{n\in\mathbb{N}}$ that decides L, the size of C_n is at least $2^{\epsilon n}$ for all sufficiently large n.

Assumption 3.2 can be seen as a natural strengthening of the assumption that $\mathbf{EXP} \nsubseteq \mathbf{NP}$ (by considering nondeterministic algorithms that can use not only polynomial but even subexponential time and advice). It has been used previously by works in the field of derandomization [AK, KvM, MV, SU], who showed that it implies that $\mathbf{AM} = \mathbf{NP}$. The following lemma states that Assumption 3.2 implies the existence of pseudorandom generators that fool even *nondeterministic* distinguishers.

Lemma 3.3 ([SU]). *Under Assumption 3.2, there exists a function $G = \bigcup_n G_n : \{0,1\}^{\ell(n)} \to \{0,1\}^n$ such that for every n, G_n maps inputs of length $\ell(n) = O(\log n)$ to length n in time* $\mathrm{poly}(n)$*, and for all nondeterministic circuits D of size n,*

$$|\Pr\left[D(G_n(U_{\ell(n)})) = 1\right] - \Pr\left[D(U_n) = 1\right]| \leq 1/n.$$

Note that the distinguishers above have less running time than the generator; without this, it would not be possible to fool nondeterministic distinguishers (as they could identify outputs of the generator by guessing the seed and evaluating the generator). The pseudorandom generator in Lemma 3.3 implies $\mathbf{AM} = \mathbf{NP}$ (and also $\mathbf{BPP} = \mathbf{P}$). We can now state our first theorem.

Theorem 3.4. *Under Assumption 3.2, if a language L has a 2-round public-coin proof system that is* plain *zero knowledge and has an efficient prover (or alternatively, has perfect completeness), then $L \in \mathbf{P}$.*

Proof. The idea behind the proof is to use the pseudorandom generator from Lemma 3.3 to derandomize the verifier in the 2-round proof system and obtain a verifier that uses only a logarithmic number of random bits. We then claim that a simulator for such a verifier must succeed for *all* choices of the verifier's random tape (because there are only polynomially many such choices). We then show that using such a simulator one can decide the language.

Let (P, V) be a 2-round public-coin zero-knowledge proof system for language L with an efficient prover (wrt relation R). For simplicity, we assume that the completeness and soundness errors are at most $1/3$, but the proof can be extended to any error bounds (such that $1-c(n) > s(n)+1/\mathrm{poly}(n)$). For any input x of length n, witness w, and fixed coin tosses r for the prover, the prover's response to a verifier message α can be computed by a circuit $P_{x,w,r}(\alpha)$ of polynomial size (notice that we use the assumption that P is an efficient prover here). In addition, there is a *nondeterministic* circuit $C_x(\alpha)$ of polynomial size that decides whether there exists a prover message β such that $V(x, \alpha, \beta)$ accepts. Let $t(n) = \mathrm{poly}(n)$ be an upper bound on the sizes of these two circuits. Note that the generator $G_{t(n)}$ given by Lemma 3.3 has seed length $\ell = O(\log t(n)) = O(\log n)$. Consider the following cheating verifier strategy, on an input x of length n.

Verifier $V^*(x)$:

1. Choose a seed $s \leftarrow \{0,1\}^\ell$ and send $\alpha = G_{t(n)}(s)$ (truncated to the appropriate length) to P.
2. Receive response β from P.
3. Accept iff $V(x, \alpha, \beta) = \texttt{accept}$.

By the pseudorandomness property of G, (P, V^*) is still a complete and sound proof system for L (albeit with nonnegligible error). Specifically, completeness uses the fact that the prover response to α is computed by the circuit $P_{x,w,r}$ of size at most $t(n)$ (or the fact that (P, V) has perfect completeness), and soundness uses the fact that the existence of an accepting prover response is computed by the nondeterministic circuits C_x of size at most $t(n)$.

By the zero-knowledge property of P, there exists a probabilistic polynomial-time simulator S such that for every $(x, w) \in R$, $\{S(x)\}$ and $\{\langle P(w), V^*\rangle(x)\}$ are computationally indistinguishable. We use S to construct an $\mathbf{RP}$ algorithm for L as follows.

$M(x)$: Run $S(x)$ many times to obtain transcripts $(G_{t(n)}(s_1), \beta_1; s_1), (G_{t(n)}(s_2), \beta_2; s_2), \ldots, (G_{t(n)}(s_q), \beta_q; s_q)$, where $q = n \cdot 2^\ell$. Accept if $\{s_1, \ldots, s_q\} = \{0,1\}^\ell$ and, for the majority of $s \in \{0,1\}^\ell$, there exists an i such that $s_i = s$ and $V^*(x, G_{t(n)}(s_i), \beta_i; s_i)$ accepts.

Suppose $x \in L$. Then q independent samples of $S(x)$ are computationally indistinguishable from q independent samples of $\langle P(w), V^*\rangle(x)$ (because the indistinguishability holds with respect to nonuniform distinguishers). Thus, if we replace the samples of $S(x)$ with samples of $\langle P(w), V^*\rangle(x)$, the probability that $M(x)$ accepts only changes negligibly. When we sample $(G_{t(n)}(s_i), \beta_i; s_i)$ from $\langle P, V^*\rangle(x)$, then s_i is distributed uniformly in $\{0,1\}^\ell$ and V^* accepts β_i with high probability. Thus, if we sample $q = n \cdot 2^\ell$ such transcripts, then with probability at least $1 - 2^\ell \cdot (1-2^{-\ell})^q = 1 - 2^{-\Omega(n)}$, the s_i's will cover all of $\{0,1\}^\ell$ and the majority of the transcripts will be accepting.

Suppose $x \notin L$. Then, by the soundness of V^*, for the majority of $s \in \{0,1\}^\ell$, there does not exist an accepting response β to $G_{t(n)}(s)$. Thus, $M(x)$ never accepts.

We conclude that $L \in \mathbf{RP}$. Recalling that the hypothesis also implies $\mathbf{RP} = \mathbf{P}$ [IW], we have $L \in \mathbf{P}$. □

Remarks:

1. The efficient prover condition can be relaxed to saying that for every input $x \in L$ and every fixing of the prover's coin tosses r, there is a *nondeterministic* polynomial-sized circuit $C_{x,r}$ such that $C_{x,r}(\alpha) = 1$ iff $P(x, \alpha; r)$ makes the verifier accept. Under stronger complexity assumptions (like Assumption 3.5 below), we can allow $C_{x,r}$ to be even more powerful, e.g. corresponding to a higher level of the polynomial-time hierarchy.

2. The only place we make use of the fact that computational indistinguishability is defined with respect to nonuniform distinguishers (see Definition 2.4) is to argue that q samples of $S(x)$ are indistinguishable from q samples of $\langle P(w), V^* \rangle(x)$. Thus the result can be extended to formulations of zero knowledge with respect to uniform distinguishers in which multiple-sample indistinguishability is ensured. Indeed, in Appendix A, we present an analogue of the above result for one formulation of "uniform zero knowledge" (essentially that of [Gol1]).

3. We say that a proof/argument system has a *publicly verifiable transcript* if the verifier's choice to accept is made by computing an efficient predicate to the execution's transcript. Most known proof systems possess this property and it is useful for some applications (e.g., [BL]). It can be shown that for 2-round proof systems with a publicly verifiably transcript, the verifier has nothing to gain by not sending its random tape as its first message. Thus our lower bound holds also for such systems.

4. Assumption 3.2 is not commonly used in cryptography. However, in Section 6.2, we show that some sort of lower bound on nondeterministic algorithms is necessary for this result.

5. Unlike the Goldreich–Oren theorem (Theorem 3.1), this result (and the private-coin one below) only apply to *proof systems*. In Section 6.3, we show that if a (variant of) the "Noninteractive CS Proofs" conjecture of Micali [Mic] is true, then there do exist 2-round, public-coin plain zero-knowledge arguments for **NP**. Thus, our result cannot be extended to argument systems without refuting Micali's conjecture.

6. In a concurrent work, the same derandomization technique was used to obtain a positive result, namely the first construction of 1-round witness-indistinguishable proofs for all of **NP** [BOV].

3.2 Limitations of 2-round Private-Coin ZK Proofs

For *private-coin* proof systems, we require a natural strengthening of Assumption 3.2, namely the existence of a function that is hard for circuits with two quantifiers (i.e., Σ_2-circuits).

Assumption 3.5. $\mathbf{E} = \mathbf{DTIME}(2^{O(n)})$ has a function of Σ_2-circuit complexity $2^{\Omega(n)}$.

Since the proof of Lemma 3.3 relativizes (cf., [KvM]), Assumption 3.5 implies the existence of pseudorandom generators that fool Σ_2-circuits.

Lemma 3.6. *Under Assumption 3.5, there exists a function* $G = \bigcup_n G_n : \{0,1\}^{\ell(n)} \to \{0,1\}^n$ *such that for every* n, G_n *maps inputs of length* $\ell(n) = O(\log n)$ *to length* n *in time* $\mathrm{poly}(n)$, *and for all* Σ_2*-circuits* D *of size* n,

$$\left| \Pr\left[D(G_n(U_{\ell(n)})) = 1\right] - \Pr\left[D(U_n) = 1\right] \right| \leq 1/n.$$

Under this assumption, we obtain the following negative result about 2-round zero-knowledge proofs.

Theorem 3.7. *Under Assumption 3.5, if L is a language with a 2-round (possibly private-coin) proof system with perfect completeness that is* plain *zero knowledge, then $L \in \mathbf{coNP}$.*

Note that, unlike Theorem 3.4, we only conclude $L \in \mathbf{coNP}$ rather than $L \in \mathbf{P}$. Still, this provides evidence that **NP**-complete languages do not have such proof systems. Moreover, in Section 6.1, we show that under the (non-standard) "Knowledge-of-Exponent Assumption" suggested by Damgård [Dam], there *does* exist a promise problem outside of **BPP** (but in **coNP**) with a 2-round zero-knowledge proof system. Thus, extending our impossibility result to all nontrivial problems would require disproving the KEA Assumption.

Also unlike Theorem 3.4, here we require perfect completeness and do not have an analogous result for efficient provers, nor do we have a uniform analogue of this result in the appendix.

Proof. Let (P, V) be the 2-round zero-knowledge proof system for L with perfect completeness and soundness $s(n) \leq 1 - 1/\mathrm{poly}(n)$. Analogous to the proof of Theorem 3.4, we use a pseudorandom generator $G_{t(n)}$ from Lemma 3.6 for a sufficiently large polynomial $t(n)$ to obtain the following cheating verifier strategy V^* that uses only $\ell = O(\log t(n)) = O(\log n)$ random bits:

Verifier $V^*(x)$:
1. Choose a seed $s \leftarrow \{0,1\}^\ell$, and send $\alpha = V(x; G_{t(n)}(s))$ to P.
2. Receive response β.
3. Output the view $(\alpha, \beta; s)$.

However, unlike the proof of Theorem 3.4, we can no longer argue that V^* is still a sound verifier if we use the original acceptance predicate of V with coin tosses $G_{t(n)}(s)$. (Intuitively, it may be the case that all seeds s yield distinct messages α, in which case the "private coin" aspect of the proof system is lost. More technically, the argument used in proving Theorem 3.4 fails because in the case of non public-coin proofs, the problem of deciding whether or not there exists a convincing prover response may not be in **NP**.) Thus, we instead consider the acceptance predicate with respect to a random sequence of V's coin tosses that are consistent with α. That is, for a verifier message α and prover message β, define $\mathrm{Con}_\alpha = \{r : V(x; r) = \alpha\}$ to be the set of coin tosses consistent with α, and $\mathrm{Rej}_{\alpha,\beta} = \{r \in \mathrm{Con}_\alpha : V(x, \beta; r) = \texttt{reject}\}$ to be those that reject β.

If $x \in L$, then by perfect completeness, if we set $\beta = P(x, \alpha)$, then $\mathrm{Rej}_{\alpha,\beta} = \emptyset$ with probability 1 (over P's coin tosses). If $x \notin L$, then there is a polynomial $p(n)$ such that with probability at least $1/p(n)$ over $\alpha \leftarrow_{\mathrm{R}} V(x)$, we have

$$\frac{|\mathrm{Rej}_{\alpha,\beta}|}{|\mathrm{Con}_\alpha|} \geq \frac{1}{p(n)} \tag{2}$$

for *all* prover responses β. (Otherwise, the soundness error would be $1 - \mathrm{neg}(n)$.) We now claim that this must also hold, with $p(n)$ replaced by $p'(n) = 2p(n)$, for $\alpha \leftarrow_{\mathrm{R}} V^*(x)$. This is follows from the pseudorandomness of $G_{t(n)}$ and the fact that approximate counting can be done in the polynomial hierarchy [Sip, Sto]. More precisely, there is a $\mathbf{BPP}^{\mathbf{NP}}$ algorithm $T(\alpha, \beta)$ that accepts w.h.p. if $|\mathrm{Rej}_{\alpha,\beta}|/|\mathrm{Con}_\alpha| \geq 1/p(n)$ and rejects w.h.p. if $|\mathrm{Rej}_{\alpha,\beta}|/|\mathrm{Con}_\alpha| \leq 1/2p(n)$. Using nonuniformity to eliminate the randomness, and incorporating the $\forall\beta$ quantifier, we end up with a polynomial-sized Σ_2-circuit that essentially tests Inequality (3.2). This circuit is fooled by $G_{t(n)}$ when $t(n)$ is a sufficiently large polynomial.

Public input: x	x $\downarrow$ $\boxed{M}$ $\quad$ $\boxed{A}$
Step M1: Choose $s \leftarrow_{\mathrm{R}} \{0,1\}^{\ell}$. Set $\alpha = V^*(x;s)$. Choose $r \leftarrow_{\mathrm{R}} \mathrm{Con}_{\alpha}$. **Step A1:** Run $S(x)$ for $n \cdot 2^{\ell}$ times and consider the first transcript of the form $(\alpha, \beta; s)$ for some β. If no such transcript is obtained, halt and accept. Otherwise, accept as specified below.	$\xrightarrow{s, \alpha, r}$
Accept if $\alpha = V^*(x;s)$, $V(x;r) = \alpha$, and $V(x, \alpha, \beta; r) = \texttt{reject}$.	

Protocol 3.8. MA proof system for $\overline{L}$

By the zero-knowledge property of P, there exists a probabilistic polynomial-time simulator S such that for every $x \in L$, $S(x)$ and $\langle P, V^* \rangle(x)$ are computationally indistinguishable. We use S to construct an **MA** proof system (M, A) for $\overline{L}$ as shown in Protocol 3.8.

We now analyze Protocol 3.8, beginning with completeness. Suppose $x \notin L$ and that M follows the specified strategy. We will show that A accepts with probability at least $1/\mathrm{poly}(n)$. We know that with probability at least $1/p(n)$ over s, Inequality (3.2) holds for all prover responses β. In particular, it holds for the prover response β computed in Step A1. Thus, the probability that $r \in \mathrm{Rej}_{\alpha,\beta}$ is at least $1/p(n)$. (Note that β and r are computed from α independently of each other.) Hence, with probability at least $1/p(n)^2$, we have $V(x, \alpha, \beta; r) = \texttt{reject}$ and A accepts.

For soundness, suppose that $x \in L$ and fix any s, $\alpha = V^*(x;s)$, and $r \in \mathrm{Con}_{\alpha}$. We will show that A accepts with negligible probability (over the executions of the simulator in Step A1). First we note that transcripts of the form $(\alpha, \beta; s)$ occur with probability $1/2^{\ell} \geq 1/\mathrm{poly}(n)$ in $\langle P, V^* \rangle(x)$, and thus must occur in the simulator's output with probability at least $1/2^{\ell} - \mathrm{neg}(n)$. So, with $n \cdot 2^{\ell}$ tries, A will obtain such a transcript with probability at least $1 - 2^{-\Omega(n)}$. By perfect completeness, for every α and $r \in \mathrm{Con}_{\alpha}$, $V(x, \alpha, \beta; r) = \texttt{reject}$ with probability 0 over $\beta \leftarrow_{\mathrm{R}} P(x, \alpha)$. Since this condition can be tested by a circuit $C_{x,r}(\alpha, \beta)$ of polynomial size, the simulator must produce rejecting β's with only negligible probability. Thus A accepts with negligible probability.

We conclude that $\overline{L} \in \mathbf{AM}$. Recalling that the hypothesis also implies $\mathbf{AM} = \mathbf{NP}$, we have $\overline{L} \in \mathbf{NP}$. □

4 Resettable Zero Knowledge

In this section, we consider the notion of *resettable* zero knowledge, as recently introduced by Canetti, Goldreich, Goldwasser, and Micali [CGGM]. Here one allows a (cheating) verifier V^* to "reset" the prover so that it uses the same random tape in multiple executions. This is a strengthening of the adversarial capability of the verifier, and the requirement is that even such a verifier's view can be simulated efficiently. (For a formal definition, see [CGGM].) As usual, the basic definition of *plain resettable zero knowledge* can be strengthened to *auxiliary-input resettable zero knowledge* and *black-box resettable zero knowledge*. Resettable zero knowledge is a strong requirement, and is even stronger than concurrent zero knowledge.

Public input: x We denote the lengths of the P's messages by $\ell = \text{poly}(n)$, V's messages by $m = \text{poly}(n)$, and P's coin tosses by s.	x ↓ P V
Step P1: Choose coin tosses $R \leftarrow_{\text{R}} \{0,1\}^s$. Compute $\alpha = P(x;R) \in \{0,1\}^\ell$	$\xrightarrow{\alpha}$
Step V2: $\beta \leftarrow_{\text{R}} \{0,1\}^m$	$\xleftarrow{\beta}$
Step P3: Compute $\gamma = P(x,\beta;R) \in \{0,1\}^\ell$	$\xrightarrow{\gamma}$
The verifier accepts according to $V(x,\alpha,\beta,\gamma)$.	

Protocol 4.2. A 3-round public-coin proof system

Under standard cryptographic assumptions, it is known how to construct (black-box) resettable zero-knowledge proofs for **NP** with $\tilde{O}(\log n)$ rounds [RK, CGGM, KP, PRS]. There is an almost matching lower bound, showing that $\tilde{\Omega}(\log n)$ rounds are necessary for *black-box* resettable zero-knowledge proofs or arguments [CKPR]. In this section, we obtain lower bounds that apply even to *non-black-box* simulation, and show that a nonconstant number of rounds is necessary for *public-coin* resettable zero-knowledge *proof* systems.

The central ingredient of our proof is the following lemma

Lemma 4.1. *For any constant c and polynomial $r(n) \geq 2$, if a language L has a $c \cdot r(n)$-round public-coin resettable zero-knowledge proof system (P,V), then L has an $r(n)$-round public-coin resettable zero-knowledge proof system (P',V').*

The form of resettable zero knowledge (perfect, statistical, or computational; plain, auxiliary-input, or black-box) is preserved. Moreover, if P is an efficient prover, then so is P'. In addition, there is a polynomial p (depending only on the constant c) such that if the total length of messages sent by P is at most $\ell(n)$ and (P,V) is zero knowledge against verifiers that make at most $p(\ell(n))$ resets, then (P',V') is zero knowledge (for a single execution, with no resets).

Proof. The main idea behind the proof is that the Babai–Moran [BM] speedup theorem for public-coin proof systems actually preserves *resettable* zero knowledge. We focus on the special case of converting 3-round public-coin proof system to a 2-round proof system, as this already conveys the main idea. Without loss of generality, we assume that the initial 3-round protocol is of the form in Protocol 4.2.

We can convert it to a 2-round proof system via the transformation of [BM], obtaining Protocol 4.3.

It is shown in [BM] that this transformation from Protocol 4.2 to Protocol 4.3 preserves completeness and soundness. The preservation of prover efficiency is clear by inspection. We argue that it also preserves resettable zero knowledge. The reason is that any interaction with prover P' can be simulated by a reset attack on prover P. This can be seen by inspecting the definition of prover P': Any query to P' on coin tosses R can be simulated by $t+1 = O(\ell)$ queries to P on coin tosses R.

Public input: x We set $t = O(\ell)$.	x $\downarrow$ $\boxed{P'}$ $\boxed{V'}$
Step V′1: $\beta_1, \ldots, \beta_t \leftarrow \{0,1\}^m$ **Step P′1:** Choose coin tosses $R \leftarrow \{0,1\}^s$, Compute $\alpha = P(x;R)$, $\gamma_1 = P(x,\alpha,\beta_1;R), \ldots, \gamma_t = P(x,\alpha,\beta_t;R)$	$\xleftarrow{\beta_1, \ldots, \beta_t}$ $\xrightarrow{\alpha, \gamma_1, \ldots, \gamma_t}$
V' accepts according to the majority of $V(x,\alpha,\beta_1,\gamma_1), \ldots, V(x,\alpha,\beta_t,\gamma_t)$.	

Protocol 4.3. Transformed 2-round public-coin proof system

To reduce the number of rounds in a protocol with more than three rounds involves repeated application of this idea. For a constant-round protocol, the above transformation can be repeatedly applied to the last 3 rounds, each time reducing the number of rounds by 1. For a nonconstant number of rounds, a more efficient version of this idea is employed. The above transformation is applied simultaneously to disjoint 3-round segments of the protocol, thereby reducing the number of rounds by a constant factor. In order to avoid the multiplicative factor of t in the complexity from accumulating with each segment, after each transformed segment $((\beta_1, \ldots, \beta_t), (\alpha, \gamma_1, \ldots, \gamma_t))$, the verifier (in its next message) chooses a random $i \leftarrow \{1, \ldots, t\}$ and the protocol continues based on history $(\alpha, \beta_i, \gamma_i)$. A more detailed description can be found in [BM, GVW] (the latter specifically analyzing the complexity of the transformation in terms of the length of the prover's messages). The many-round transformation preserves resettable zero knowledge for essentially the same reason as given above for the basic three-to-two round transformation. □

Combining this lemma with the impossibility of 2-round zero knowledge (Theorem 3.1 and Theorem 3.4), we obtain the main result of this section.

Theorem 4.4.

1. *If a language L has a constant-round public-coin proof system that is auxiliary-input resettable zero knowledge, then $L \in \mathbf{BPP}$.*

2. *Under Assumption 3.2, if a language L has a constant-round public-coin proof system that is plain resettable zero knowledge and has an efficient prover (alternatively, perfect completeness), then $L \in \mathbf{P}$.*

In both cases, the resettable zero-knowledge requirement can be weakened to zero knowledge against verifiers that may reset the prover at most $\ell(n)^c$ times, where $\ell(n)$ is the total length of the prover's messages and c can be any constant. We call a protocol that satisfies this property a bounded-resettable zero-knowledge *protocol.*[8]

[8]Note that this is a somewhat stronger requirement than requiring zero knowledge against n^c many resets (for any fixed constant c). This is because under our definition the number of resets may be larger than the prover's communication complexity. Both our lower bound (Theorem 4.4) and our upper bound (Theorem 7.1) use this stronger definition of bounded-resettable zero knowledge. This weakens the lower bound and strengthens the upper bound.

We remark that Lemma 4.1 and Theorem 4.4 also extend to *private-coin* proof systems in which the verifier's messages are computed in a history-independent manner, using independent coin tosses for each round. (The acceptance predicate can, however, depend on all the coin tosses.) However, Theorem 4.4 does *not* extend to arguments. That is, under standard assumptions, we construct a constant-round public-coin argument system that is *bounded*-resettable zero knowledge. Since previous such systems used both private coins and a non-constant number of rounds, this construction, which is described in Section 7, is interesting in its own right.

Following Remark 2 after the proof of Theorem 3.4, the proof of Theorem 4.4 doesn't make use of any nonuniformity in the distinguisher (other than the input x); here even the multiple-sample indistinguishability needed follows from the resettable zero knowledge property. Thus, the result extends to uniform distinguishers, and indeed we present a uniform version of it in Appendix A.

5 Strong proofs of knowledge

In this section, we present lower bounds for achieving *strong* proofs and arguments of knowledge, as defined by Goldreich [Gol2, Sec. 4.7.6]. We remark that this section is structured so that later proofs rely on ideas from earlier proofs.

5.1 Definitions

Before presenting the definition of *strong* proofs and arguments of knowledge [Gol2, Sec. 4.7.6], we informally recall the standard notion of proofs of knowledge. A proof of knowledge is an interactive proof which convinces a verifier that the prover "knows" a witness to a certain statement. This is in contrast to a regular interactive proof, where the verifier is just convinced of the validity of the statement. The concept of "knowledge" for machines is formalized by saying that if a prover can convince the verifier, then there exists an efficient procedure that can "extract" a witness from this prover (thus the prover knows a witness because it could run the extraction procedure on itself). More formally, a proof of knowledge has the property that for every machine P^* there exists a knowledge extractor K, who "extracts" witnesses from the prover P^*. The requirements on this K come in two flavors, that are equivalent for **NP**-relations:

1. K runs in expected polynomial time and obtains a witness to the statement x with probability that is negligibly close to the probability that P^* convinces V. More exactly, let $p(x, y, r)$ denote the probability that P^* convinces V upon common input x, and when P^* has auxiliary input y and random tape r. Then, K obtains a witness with probability that is negligibly close to $p(x, y, r)$. We stress that here, K runs in expected time that is independent of $p(x, y, r)$.

2. K runs in expected time that is inversely proportional to $p(x, y, r)$, but must always output a witness for x.

Informally speaking, a *strong* proof (or argument) of knowledge [Gol2] enjoys the best of both worlds with respect to the above two definitions of proofs of knowledge. That is, on the one hand, K runs in time that is independent of $p(x, y, r)$, like in the first definition. On the other hand, K always (or almost always), obtains a witness for x, like in the second definition. More specifically, if the probability $p(x, y, r)$ that P^* convinces V is greater than some negligible function $s(|x|)$, then K (running in time independent of P^*'s actual success probability) obtains a witness for x with *high* probability; the function s is called K's *(soundness) error function*. The important point is that *neither* K's running time nor its success probability are dependent on $p(x, y, r)$. For more discussion on strong proofs of knowledge, see [Gol2, Sec. 4.7.6].

We are now ready to present the formal definition:

Definition 5.1 (strong proofs of knowledge). An interactive protocol (P, V) is system of *strong proofs of knowledge* for a (poly-balanced) relation R if there are functions $c, s, \varepsilon : \mathbb{N} \to [0, 1]$ such that $1 - c(n) > s(n) + 1/\text{poly}(n)$ and $\varepsilon(n) \leq 1 - 1/\text{poly}(n)$ such that the following holds:

- (efficiency): (P, V) is polynomially bounded, and V is computable in probabilistic polynomial time.
- (nontriviality/completeness): If $(x, w) \in R$, then V accepts in $(P(w), V)(x)$ with probability at least $1 - c(|x|)$,
- (strong validity/soundness): There exists a probabilistic polynomial-time oracle machine K such that for every interactive function P^* and every $x, y, r \in \{0, 1\}^*$, machine K satisfies the following condition:

$$\begin{aligned} p(x, y, r) &\stackrel{def}{=} \Pr\left[V \text{ accepts in } (P^*(y; r), V)(x)\right] > s(|x|) \\ &\Rightarrow \quad \Pr\left[K^{P^*(x,y,\cdot;r)}(x) = w \text{ s.t. } (x, w) \in R\right] \geq 1 - \varepsilon(|x|) \end{aligned}$$

  That is, if on input x, auxiliary input y, and coin tosses r, P^* makes V accept with probability greater than $s(n)$, then K will output a witness with probability at least $1-\varepsilon(n)$, given oracle access to the next-message function of P^* (with x, y, and r hardwired in). K is called a *strong knowledge extractor*.

We call $c(\cdot)$ the *completeness error*, $s(\cdot)$ the *soundness error*, and $\varepsilon(\cdot)$ the *extraction error*. We say that (P, V) has *negligible error* if c, s, and ε are negligible. We say that it has *perfect completeness* if $c = 0$.

P is an *efficient prover* if $P(w)$ is computable by a probabilistic polynomial-time algorithm when $w \in R_x$.

If (P, V) is a system of strong proofs of knowledge for relation R according to Definition 5.1, then it is also an interactive proof system for L_R (with the same completeness and soundness errors). This is because if P^* convinces V with probability greater than s, then K must obtain a valid witness with non-zero probability. However, when $x \notin L$, there are no valid witnesses that can be obtained. Therefore, in such a case, V can be convinced with probability at most s.

We are primarily interested in strong proofs of knowledge with *negligible error*. Such proofs guarantee that if P^* convinces with non-negligible probability, then K extracts with overwhelming probability. This is the key property of strong proofs of knowledge which make them both interesting and useful, but is also the source of the lower bound on round complexity that we will prove below.

The definition above is slightly more general than the one in [Gol2], which requires perfect completeness and negligible soundness and extraction errors. For **NP**-relations R, a strong proof of knowledge with even $1 - 1/\text{poly}(n)$ extraction error is also a strong proof of knowledge with negligible extraction error (by running the extractor $\text{poly}(n)$ times, and outputting the first valid witness it outputs).

Feasibility of (1-round) strong proofs of knowledge with no secrecy. We note that if there is no secrecy requirement (like, for example, zero knowledge), then constant-round strong proofs of knowledge for **NP** can be easily constructed by having the prover just send an NP-witness w to the verifier V. The verifier V then outputs 1 if and only if $(x, w) \in R$. The reason that this

constitutes a strong proof of knowledge follows from the fact that the protocol is deterministic and non-interactive. Thus, a prover $P^*(x, y; r)$ either always convinces V or never convinces V. Therefore, if $p(x, y, r) > s(|x|)$, this means that $P^*(x, y; r)$ always sends a valid witness and the extractor K can just output this witness.

Feasibility of nonconstant-round zero-knowledge strong proofs of knowledge. If one-way functions exist, then there exist $f(n)$-round zero-knowledge strong proofs of knowledge with negligible error for any growing function $f(n) = \omega(1)$. This is achieved by a slight modification of the protocol in [Gol2], specifically taking $\Omega(f(n))$ sequential repetitions of a $\log n$-fold parallel repetition of Blum's basic HAMILTONICITY protocol [Blu2]. In this section, we will give evidence that this is optimal.

Variants of Definition 5.1. Definition 5.1 is (a slight generalization of) the definition as presented in [Gol2]. However, we will also consider a number of variants. Specifically, we will consider the case that K may run in *expected*, rather than strict, polynomial time. We will also consider *arguments*, where the strong validity requirement will only need to hold for polynomial-time provers P^*. In such a case, the extractor is not limited to black-box access to P^*, but rather can depend on the specific cheating prover. This is a natural relaxation of the notion of proofs of knowledge. (In fact, the only reason to consider black-box access in the first place, seems to be so that efficient extraction from inefficient provers can also be considered.) Finally, we will consider a strengthening of strong proofs of knowledge, which relates to how the "success probability" of P^* is computed. Specifically, Definition 5.1 requires extraction when the *deterministic* prover $P^*(x, y; r)$ successfully convinces V with probability greater than $s(|x|)$. Another possibility, however, is to require extraction when the *probabilistic* prover $P^*(x, y)$ successfully convinces V with probability greater than $s(|x|)$. It turns out that this makes a very big difference.

Definition 5.2 (variants of strong proofs of knowledge).

1. *Expected instead of strict polynomial time:* Rather than restricting K to strict polynomial time, it is allowed to run in **expected** polynomial time (we stress that this running time is still independent of $p(x, y, r)$).

2. *Arguments instead of proofs and non-black-box instead of black-box extraction:* We modify the strong validity/soundness requirement as follows. For every PPT P^* and every polynomial $q(n)$, there exists a PPT K such that for all inputs $x \in \{0,1\}^*$, all auxiliary inputs $y \in \{0,1\}^{q(n)}$, and all random tapes $r \in \{0,1\}^{q(n)}$, we have

$$\begin{aligned} p(x, y, r) &\stackrel{def}{=} \Pr\left[V \text{ accepts in } (P^*(y; r), V)(x)\right] > s(|x|) \\ \Rightarrow \quad &\Pr\left[K(x, y) = w \text{ s.t. } (x, w) \in R\right] \geq 1 - \varepsilon(|x|) \end{aligned}$$

   If K fulfills this modified condition, then we call it a *strong* **non-black-box** *knowledge extractor*, and we call the protocol a *system of strong* **arguments** *of knowledge*.[9]

3. *Fixed versus variable randomness for P^*:* Denote by $p(x, y)$ the probability that V accepts on input x, when interacting with the prover specified by $P^*(x, y)$. We stress that this probability

[9] It is possible to also consider arguments with black-box extractors and proofs with non-black-box extractors. However, this combination seems to be the most reasonable.

is also over the coins r that are used by P^*. Then, require that if $p(x,y) > s(|x|)$, machine K must output a witness w such that $(x,w) \in R$ with probability at least $1-\varepsilon(|x|)$. That is,

$$\begin{aligned} p(x,y) &\stackrel{def}{=} \Pr\left[V \text{ accepts in } (P^*(y),V)(x)\right] > s(|x|) \\ &\Rightarrow \quad \Pr\left[K^{P^*(x,y,\cdot;\cdot)}(x) = w \text{ s.t. } (x,w) \in R\right] \geq 1-\varepsilon(|x|) \end{aligned}$$

We stress that in this case, K is allowed to set the random coins for $P^*(x,y)$ to be any value it wishes (even if it has only black-box access). Thus, K asks queries of the form $(\alpha_1, \alpha_2, \ldots; r)$, which are answered according to $P^*(x, y, \alpha_1, \alpha_2, \ldots; r)$.

If K fulfills this modified strong validity condition, then we call it an **extra-strong** *knowledge extractor*. Furthermore, we call (P,V) a *system of* **extra-strong** *proofs of knowledge*.

Analogous to auxiliary-input zero knowledge, for strong arguments of knowledge there is a universal prover $P^*_{\text{uni}}(x,z)$ that interprets its auxiliary input z as a circuit C_z and uses C_z as its next message function. If the knowledge extraction condition holds for P^*_{uni}, then it holds for all PPT prover strategies P^*. Note, however, that the extractor K may depend on the polynomial bound $q(n)$ on the size of the circuit given as auxiliary input to P^*_{uni}, so this does give a "universal" extractor in the usual sense; this makes our lower bound stronger.

Strict versus expected polynomial time. We show that, unlike for zero-knowledge simulators and standard proof of knowledge extractors, strict polynomial-time strong and extra-strong extractors can be constructed from expected polynomial-time strong and extra-strong extractors. That is, allowing the more liberal notion of expected polynomial time does not add any power. This means that in our lower bounds, it suffices to rule out the existence of strong and extra-strong proofs and arguments of knowledge with *strict* polynomial-time extractors, respectively.

Proposition 5.3. *Let (P,V) be a system of strong proofs of knowledge for a relation R with an* expected *polynomial-time strong knowledge extractor having extraction error ε. Then, (P,V) is a system of strong proofs of knowledge for R with a* strict *polynomial-time strong knowledge extractor having extraction error $\varepsilon' = O(\varepsilon)$. In particular if ε is negligible, so is ε'. The same holds for extra-strong proofs of knowledge, and for strong and extra-strong arguments of knowledge.*

Proof. We prove the proposition for strong proofs of knowledge; the other cases are proved in exactly the same way. Let (P,V) be a system of strong proofs of knowledge, and let K be a strong knowledge extractor for (P,V) that runs in expected-time $q(n)$ and has soundness error $s(\cdot)$ and extraction error $\varepsilon(\cdot)$.

We begin by constructing a strict polynomial-time machine K' who runs in time $2q(n)$ and, when $p(x,y,r) > s(|x|)$, extracts with probability at most $1/2 - \varepsilon(|x|)$. The machine K' simply invokes K and if K exceeds $2q(n)$ steps, then it outputs time-out. By Markov's inequality, K outputs time-out with probability at most $1/2$. Furthermore, conditioned on this not happening, the probability that K does not obtain a valid witness at most doubles. Thus, by the strong validity property, when $p(x,y,r) > s(|x|)$, machine K' obtains a valid witness with probability at least $1 - 2\varepsilon(|x|)$. In summary, K' runs in time that is (strictly) bound by $2q(|x|)$, and outputs time-out with probability at most $1/2$. Furthermore, conditioned on K' not outputting time-out, it outputs a valid witness with probability $1 - 2\varepsilon(|x|)$.

Given such a K', a strong knowledge extractor K'' is obtained by invoking K' independently n times. The first time that K' outputs a string w and does not output time-out, the extractor K''

outputs w and halts. If K' outputs time-out in all iterations, then K'' halts and outputs failure. Now, the probability that K'' outputs failure is at most 2^{-n}. Furthermore, the probability that K'' outputs a valid witness, conditioned on it *not* outputting failure is at least $1-2\varepsilon(|x|)$. Putting this together, we have that K'' outputs a valid witness with probability at least $(1-2^{-n})(1-2\varepsilon(n)) = 1-\varepsilon'(n)$ for $\varepsilon' = O(\varepsilon)$. K'' thus constitutes a strict polynomial-time strong knowledge extractor for (P,V) with extraction error $O(\varepsilon)$, as required. □

Remark. This method of converting an expected polynomial-time extractor to a strict polynomial-time extractor works for strong proofs of knowledge, but does *not* work for extractors for standard (i.e., not strong) proofs of knowledge. The reason for this difference is that the running time and success probability of strong knowledge extractors are both independent of the probability that P^* convinces V. This is not the case for standard proofs of knowledge. Indeed, an actual separation between expected and strict polynomial-time *black-box* extraction for standard proofs of knowledge has been shown in [BL].

5.2 Triviality

It is well-known that every language in **BPP** has a trivial zero-knowledge interactive proof system, where the prover sends nothing to the verifier and the verifier simply decides membership on its own. For proofs of knowledge, the corresponding notion of triviality does not refer to the complexity of deciding membership in the language, but rather the complexity of *finding witnesses for the given relation.* For example, if f is a one-way permutation, then the relation $R = \{(x,w) : f(w) = x\}$ defines a trivial language $L_R = \{0,1\}^*$, but giving a zero-knowledge proof of knowledge for R is quite nontrivial. We note that zero-knowledge proofs of knowledge for relations of this type are also often used in cryptographic construction. Thus we are led to define the following notion:

Definition 5.4. A (poly-balanced) relation R is *easy to search* if there exists a PPT T and a function $\varepsilon(n) \leq 1 - 1/\text{poly}(n)$ such that

$$\forall\ x \in L_R\ \ \Pr[T(x) \in R_x] \geq 1 - \varepsilon(|x|).$$

We call ε the *error probability* of the search algorithm A. We say that R is easy to search *deterministically* if T is deterministic poly-time (and hence $\varepsilon = 0$).

Notice that according to our definition, a relation is easy to search if witnesses can be found with any noticeable (i.e., inverse polynomial) probability. Below, we will show that if a relation is easy to search in this weak sense, then witnesses can actually be found with all but negligible probability. It therefore suffices to consider this weaker notion.

Before proceeding, we note that for **NP**-relations and **MA**-relations, this is a strengthening of saying that deciding membership is easy. Thus, lower bounds for relations are stronger than lower bounds for languages.

Lemma 5.5. *If R is an* **NP***-relation and R is easy to search, then L_R is in* **RP***. If R is an* **MA***-relation and R is easy to search, then L_R is in* **BPP***.*

Proof. By error reduction on the **MA** proof system for R, there is a PPT algorithm A such that if $(x,w) \in R$ then $A(x,w)$ accepts with probability at least $1-2^{-n}$ and if $x \notin L_R$, then $A(x,w)$ accepts with probability at most 2^{-n}. (If R is an **NP** relation, then R is deterministic.) Now we can use A together with the search algorithm T to decide L_R: given x, run $A(x,T(x))$ many

(specifically, $O(1/(1-\varepsilon(|x|)))$ times independently and accept if at least one of these executions accepts. If $x \in L_R$, then with high probability, at least one execution of $T(x)$ will produce w such that $(x,w) \in R$, and hence A will accept with high probability. If $x \notin L_R$, then A will reject on all executions with high probability (regardless of what T outputs). (If R is an **NP** relation, then A will reject $x \notin L_R$ with probability one, hence we get an **RP** algorithm.) □

Now we show that easy-to-search **MA**-relations have trivial proofs of knowledge. Thus, a lower bound proving that strong proofs of knowledge exist only for easy-to-search **MA**-relations (as we indeed prove below) is tight.

Proposition 5.6. *If R is an* **MA***-relation that is easy to search with error probability $\varepsilon(\cdot)$, then R has a zero-round (perfect, auxiliary-input) zero-knowledge extra-strong proof of knowledge with extraction error $\varepsilon(\cdot)$ and negligible completeness and soundness errors.*

Proof. The verifier decides whether to accept or reject input x using the **BPP** algorithm given by Lemma 5.5 (after reducing its error probability to negligible). The prover does nothing (which implies the zero knowledge property). The knowledge extractor ignores the prover algorithm and simply runs the search algorithm for R. □

Since the error probability of the search algorithm translates into the extraction error of the trivial proof of knowledge, it is desirable to be able to reduce the search error to negligible. This is easy for **NP**-relations: run the search algorithm T polynomially many times and output the first valid witness it finds (if any). For **MA**-relations, it is a bit more subtle. For the sake of the intuition, consider a **MA**-relation for which $(x,w) \in R$ *if and only if* $\Pr[A(x,w)=1] \geq 2/3$ (the definition of **MA**-relations only requires that if $(x,w) \in R$ then $A(x,w)=1$ with probability greater than $2/3$). The problem is that when $x \in L_R$, the search algorithm T may sometimes output a value w with, say, $\Pr[A(x,w)=1] = 2/3 - 2^{-|x|}$ (so $w \notin R_x$). However, it will be infeasible to distinguish such w's from w's in the relation, and thus it seems we cannot reduce the search error probability for R.

Due to the above difficulty, we define extensions of **MA** relations. Specifically, let R be an **MA**-relation and let A be the PPT algorithm that checks witnesses (with error probability $1/3$). Consider the relation $\hat{R} = \{(x,w) : \Pr[A(x,w) = \texttt{accept}] \geq 1/2\}$. Then $R \subseteq \hat{R}$, $\hat{R}$ is an **MA**-relation, and $L_{\hat{R}} = L_R$. (The constant $1/2$ is arbitrary, and anything strictly between $1/3$ and $2/3$ would do.) We call a relation $\hat{R}$ obtained in this way an *extension* of R. Notice that if an extension $\hat{R}$ is easy to search, then this means that it is easy to find a value w such that $\Pr[A(x,w)=1] \geq 1/2$. Such a value may *not* be a witness in the classic sense because (x,w) is not necessarily in R; however, it *is* a proof that $x \in L_R$ (because for $x \notin L_R$, $\Pr[A(x,w)=1] < 1/3$ for every w). Also notice that if R is an **NP**-relation, then R is an extension of itself. We have the following.

Proposition 5.7. *If an* **NP***-relation R is easy to search, then R is easy to search with negligible error probability. If an* **MA***-relation R is easy to search, then every extension $\hat{R}$ of R is easy to search with negligible error probability.*

Proof. We only deal with the **MA** case. Let A be a witness-checking algorithm for R and $\hat{R}$ the corresponding extension. Consider an algorithm $A'(x,w)$ that runs $A(x,w)$ for $O(|x|)$ executions and accepts if at least a .6 fraction of the executions accept. Then if $(x,w) \in R$, $A'(x,w)$ accepts with all but exponentially small probability. If $(x,w) \notin \hat{R}$, then $A'(x,w)$ rejects with all but exponentially small probability. Now let T be the search algorithm for R with error probability $\varepsilon(n)$. We use T and A' to construct a search algorithm T' for $\hat{R}$ as follows. On input x of length

n, T' runs $T(x)$ many times to get witnesses $w_1, \ldots, w_k \leftarrow_{\mathrm{R}} T(x)$, where $k = O(n/(1-\varepsilon(n)))$, and outputs the first w_i such that $A'(x, w_i) = \texttt{accept}$. If there is no such w_i, then T' outputs $\texttt{fail}$.

If $x \in L_R$, then with probability at least $1 - (1-\varepsilon(n))^k \geq 1 - 2^{-n}$, at least one w_i will be in R_x and thus will be accepted by A' with probability at least $1 - 2^{-n}$. In addition, A' will reject any w_i not in $\hat{R}_x$ with all but exponentially small probability. Thus, when $x \in L_R$, the output of $T'(x)$ will be in $\hat{R}_x$ with all but exponentially small probability. □

Proposition 5.7 shows that in order to give a triviality result for zero-knowledge proofs of knowledge for **NP**-relations (or **MA**-relations), it suffices to come up with a probabilistic polynomial-time algorithm that finds witnesses with probability at least $1/\mathrm{poly}(n)$.

Summary of triviality. In the setting of proofs of knowledge, triviality results on languages do not necessarily capture what is required (because, as we have shown, there exist "hard" relations that have trivial languages). We will therefore prove the results of this section for relations, and specifically for **MA**-relations (which of course, includes all **NP**-relations).

In order to further clarify what our results mean, we explain the ramifications of the statement that if a type of interactive proof exists for an **MA**-relation R, then *every extension $\hat{R}$ of R is easy to search* (our triviality results are based on such statements):

1. First, since every **MA**-relation is also an **NP**-relation and every **NP**-relation is an extension of itself, we have that the **NP**-relation is easy to search. Furthermore, by Proposition 5.7, it follows that the **NP**-relation is easy to search with negligible error probability (in other words, witnesses can almost always be found).

2. Next, if R is an **MA**-relation then so is its extension $\hat{R}$, and furthermore $L_{\hat{R}} = L_R$. Thus, by Lemma 5.5, it follows that $L_R \in \mathbf{BPP}$. Therefore, a triviality result on the extension $\hat{R}$ of an **MA**-relation R implies a triviality result (in the usual "membership sense") on the original language L_R as well.

The above justifies proving triviality with respect to extensions of **MA**-relations because this only strengthens our results.

5.3 Tools: Strong Pseudorandomness

In order to prove some of our lower bounds, we need to use a very strong pseudorandom generator, which can be constructed using the following exponential hardness assumption.

Assumption 5.8. There exists a constant δ and a length preserving polynomial-time computable[10] regular[11] one-way function $f : \{0,1\}^* \to \{0,1\}^*$ that is hard to invert for $2^{\delta n}$-sized circuits. That is, for every circuit family $A = \{A_n\}_{n \in \mathbb{N}}$ such that each A_n is of size at most $2^{\delta n}$, and for all sufficiently large n's,

$$\Pr[A_n(f(U_n)) \in f^{-1}(f(U_n))] \leq 2^{-\delta n}$$

As we will show in Section 8, strong assumptions of hardness are necessary for proving lower bounds for strong proofs of knowledge; indeed, under assumptions of sub-exponential "easiness", the lower bounds do not hold. The following theorem is used for proving our non-black-box lower bounds. Loosely speaking, it states that for every polynomial-time distinguisher, there exists a

[10] Actually, for our results it will suffice that f will be computed in $2^{o(n)}$-time.

[11] A function $f : \{0,1\}^n \to \{0,1\}^m$ is *regular* if $|f^{-1}(z)|$ has the same size for every z in the image of f.

pseudorandom generator that receives a very short $O(\log n)$-length seed and outputs a string of length n that cannot be distinguished from random by the distinguisher.

Theorem 5.9 (strong pseudorandomness [HILL, Lub]). *Assuming Assumption* 5.8*, there is a polynomial $t_G(n)$ such that for every polynomial $q(n)$, there exists a constant d and a generator $G : \{0,1\}^{d \log n} \to \{0,1\}^n$ such that G can be computed in time $t_G(n)$, and for every nonuniform probabilistic algorithm D running in time $q(n)$ and all sufficiently large n's,*

$$|\Pr[D(G(U_{d \log n})) = 1] - \Pr[D(U_n) = 1]| < \frac{1}{q(n)}$$

The dependence of G on $q(n)$ is only with respect to the constant d in the seed length. Furthermore, the time it takes to compute G is independent of d and thus of $q(n)$. That is, there exists a single machine M_G running in time $t_G(n)$, such that for all for $s \in \{0,1\}^{d \log n}$, upon input $(1^n, d, s)$ machine M_G computes $G(s)$ that "fools" all distinguishers running in time $q(n)$.

We note that the pseudorandom generators above imply $\mathbf{BPP} = \mathbf{P}$, and even $\mathbf{MA} = \mathbf{NP}$. In fact, they also imply that if R is an **NP**-relation that is easy to search, then R is easy to search deterministically, and if R is an **MA**-relation that is easy to search, then every extension $\hat{R}$ of R is easy to search deterministically.

5.4 Extra-Strong Proofs and Arguments of Knowledge

We begin by showing that extra-strong proofs and arguments of knowledge cannot be obtained for nontrivial languages, even when no secrecy requirements (such as zero knowledge or witness indistinguishability) are imposed. This result holds for proofs and arguments with any (polynomial) number of rounds. Note that by our definition, the impossibility result for proofs is black-box, whereas the impossibility result for arguments holds even for non-black-box extractors. However, any proof system is also an argument. Therefore, the non-black-box result holds also for proof systems.

Theorem 5.10.

1. *If a relation R has an extra-strong proof of knowledge with negligible soundness error, then R is easy to search. In particular, if R is an* **NP***-relation (resp.,* **MA***-relation), then $L_R \in \mathbf{RP}$ (resp., $L_R \in \mathbf{BPP}$).*

2. *Assuming Assumption 5.8, if an* **MA***-relation R has an extra-strong* argument *of knowledge with negligible soundness error and an efficient prover, then every extension $\hat{R}$ of R is easy to search (deterministically). In particular, $L_R \in \mathbf{P}$.*

Proof. Intuitively, extra-strong proofs of knowledge cannot be obtained because a prover may simply refuse to cooperate with the extractor. That is, assume that the extractor's running time is $q(n)$. Then, consider a "cheating" prover P^* who with probability $1 - 1/2q(n)$ does nothing, and with probability $1/2q(n)$ plays the honest prover strategy. Then, on the one hand, the prover convinces V with probability $1/2q(n) > s(n)$. On the other hand, the probability that the extractor will obtain a response from the prover in time $q(n)$ is at most $1/2$. Since it must extract with probability $1 - \varepsilon(|x|)$, it must obtain the witness with no help from the prover in almost half the cases. Thus, an efficient decision procedure can be built from this extractor. The above intuition suffices for obtaining a black-box lower bound. However, if the extractor is given non-black-box

access to the above prover P^*, the following problem arises. In order to construct a P^* who convinces V with probability $1/2q(n)$, we need to give it an explicit witness for the statement being proved. Otherwise, it is not clear how it can convince V in the case that it is supposed to run the honest prover strategy. However, if P^* holds a witness w, then the extractor who is given P^*'s code and auxiliary input can immediately obtain this witness. Thus, we must somehow "hide" the witness from the extractor. This is achieved by having P^* hold an *encryption* of the witness, where the "encryption scheme" has the following two properties. First, the extractor, who runs in time $q(n)$, will only be able to decrypt with probability that is noticeably less than 1. Second, it can easily be decrypted with probability $1/\text{poly}(n)$, using a random "key". Therefore, on the one hand, P^*'s random tape can be used to obtain the witness and convince V with high enough probability (i.e., with probability greater than $s(n)$). On the other hand, the extractor will *not* be able to obtain the witness with high enough probability. We now proceed to the formal proof.

We first prove the theorem for the case of extra-strong *proofs* of knowledge, and thus where the extractor K is given only black-box access to P^* (as in Definition 5.1).

Triviality of extra-strong proofs. Assume that (P, V) is an extra-strong proof of knowledge, with extra-strong knowledge extractor K, negligible soundness error $s(n)$, and completeness and extraction errors $c(n), \varepsilon(n) \leq 1 - 1/\text{poly}(n)$. Let $t(n)$ be the polynomial bounding K's running time, and let $q(n) = 2t(n)/(1 - \varepsilon(n))$.

We define a prover algorithm P^*, from which K will have to successfully extract witnesses. On input $x \in L_R$ of length n, we will give P^* an auxiliary input of the form $y = (w, s)$, where w is a witness valid witness for x and s is a (randomly chosen) string of length $\log q(n)$. Now $P^*(x, (w, s); r)$ works as follows. It examines the first $\log q(|x|)$ bits of its random tape r. If they do not equal s, then P^* aborts without sending any message. Otherwise, P^* correctly proves the proof of knowledge, following P's strategy (using the remainder of its random tape for P's random bits). Now, the probability $p(x, (w, s))$ that $P^*(x, (w, s))$ convinces V equals $(1/q(n)) \cdot (1 - c(n)) = 1/\text{poly}(n)$. Thus, $p(x, w) > s(n)$, Therefore, by the strong validity/soundness property, K must output a valid witness with probability at least $1 - \varepsilon(n) \geq 1/\text{poly}(n)$, while utilizing oracle access to $P^*(x, (w, s), \cdot; \cdot)$.

Given such a K, we construct a search algorithm T for R. Upon input x of length n, machine T chooses $s \leftarrow_{\text{R}} \{0,1\}^{\log q(n)}$. Machine T then invokes $K(x)$, and plays the role of $P^*(x, (w, s))$, answering all of K's oracle queries. Specifically, when K asks a query of the form $(\alpha_1, \alpha_2, \ldots; r)$, machine T first checks if the first $\log q(n)$ bits of r equal s. If yes, then T outputs `fail`. (In this case, T cannot continue emulating $P^*(x, (w, s))$ because it does not know a witness for x.) Otherwise, T emulates $P^*(x, (w, s))$ and hence emulates P^* sending abort. T continues this emulation until K concludes. If K outputs a value w, then T outputs w.

It remains to prove that T constitutes a valid search algorithm for R. Suppose $x \in L_R$. We need to show that T outputs a valid witness with probability at least $1/\text{poly}(n)$. The probability that this occurs is at least the probability that K fails to output a valid witness (when interacting with $P^*(x, (w, s))$ for random s) minus the probability that T outputs `fail`. The probability that K outputs a valid witness is at least $\varepsilon(n)$. Now, we analyze the probability that T outputs `fail`. Since K runs in time that is bound by $q(n)$, the maximum number of oracle queries that it makes is $t(n)$. For each of these oracle calls, the probability that the first $\log q(n)$ bits of r equal s is at most $1/q(n)$. This is due to the fact that K' has only *black-box* access to P^* and so knows nothing about s which is uniformly distributed in $\{0,1\}^{\log q(n)}$. By the union bound over all $t(n)$ oracle calls, we have that the probability of there being some oracle call for which the first $\log q(n)$ bits match is at most $t(n)/q(n) \leq (1 - \varepsilon(n))/2$. This means that T outputs `fail` with

probability at most $(1-\varepsilon(n))/2$. Thus, the probability that $T(x)$ outputs a valid witness is at least $1-\varepsilon(n)-(1-\varepsilon(n))/2 = (1-\varepsilon(n))/2 \geq 1/\text{poly}(n)$, as desired.

Triviality of extra-strong arguments. The fact that the existence of extra-strong arguments of knowledge for a relation R implies that R is easy to search is proven in a similar way to the analogous claim for extra-strong proofs of knowledge. However, some essential changes must be made. Specifically, the extractor K now has *non-black-box* access to the prover. Therefore, if we were to use the same prover strategy $P^*(x,(w,s))$, K would be able to see s and thus set r so that its prefix equals s. Furthermore, K would also see the witness w that $P^*(x,(w,s))$ uses to prove when it does not abort. We must therefore construct a prover that works in a similar way to $P^*(x,(w,s))$, but also "hides" s and w from the extractor K. We do this by using the strong pseudorandom generator of Theorem 5.9. Specifically, the generator is applied to s and the result is used to "encrypt" w.

Formally, let (P,V) be a system of extra-strong arguments of knowledge. Then, construct a cheating prover P^* as follows: First, recall that by Theorem 5.9 (which follows from Assumption 5.8), for every polynomial $q'(n)$ there exists a pseudorandom generator $G : \{0,1\}^{d\log n} \to \{0,1\}^n$, where d is a constant, such that G fools any distinguisher running in time $q'(n)$. Furthermore, there exists a single machine M_G running in time $t_G(n)$, such that M_G receives for input $(1^n,d,s)$ where $s \in \{0,1\}^{d\log n}$ and computes the above mentioned generator G on seed s. Now, let $x \in L_R$ be the statement being proven and let w be a witness such that $(x,w) \in R$. Without loss of generality, assume that $|w| = |x| = n$. Then, the auxiliary input of P^* is a pair $y = (d,u)$, where $u = G(s) \oplus w$ and $s \in \{0,1\}^{d\log n}$ (an appropriate encoding is used to differentiate between d and u). Intuitively, P^* will agree to prove the statement if and only if the first $d\log n$ bits of its random tape correctly "decrypt" y, revealing the witness w. Formally, when $P^*(x,(d,u))$ is invoked, it reads the value of d from its auxiliary input, takes the first $d\log n$ bits of its random tape, denoted s', and computes $w' = G(s') \oplus u$ (using the machine M_G). Then P^* uses the witness w' and proves the proof, playing the honest prover strategy (using all but the first $d\log n$ bits of its random tape for P's random bits). Before proceeding, notice that $P^*(x,(d,u))$ runs in time $t_G(n) + t_P(n)$, where $t_P(n)$ is the polynomial bounding the running time of the honest prover P upon inputs of length n. Now, by the assumption that (P,V) is a system of extra-strong arguments, there exists an extractor K running in some polynomial time $t(n)$ for this prover P^*. Let $q(n) = \max\{t(n) + t_R(n), 2/(1-\varepsilon(n))\}$, where $t_R(n)$ is the running time of the witness-checking algorithm T that accepts $(x,w) \in R$ with probability at least $1-2^{-n}$ and accepts $(x,w) \notin \hat{R}$ with probability at most 2^{-n}.

Consider now the case that $P^*(x,(d,u))$ has auxiliary input $y = (d,u)$, where d defines a generator that fools all distinguishers running in time $q(n)$ (this choice of d will become apparent below) and $u = G(s) \oplus w$ for some $s \in \{0,1\}^{d\log n}$. The probability that $P^*(x,(d,u))$ convinces V to accept is at least the probability that $s' = s$ times the probability that $P(x,w)$ convinces V to accept. By the uniformity of s' (coming from P^*'s random tape) and the completeness/nontriviality property, this equals $(1/n^d)\cdot(1-c(n)) \geq 1/\text{poly}(n)$. (The probability that P^* convinces V may actually be greater if there are numerous witnesses because a number of different strings s' may "decrypt" u to valid witnesses.) This is greater than all negligible functions, and in particular greater than soundness error $s(n)$. Therefore, by the soundness/strong validity property, for $u = w \oplus G(s)$, $K(x,(d,u))$ succeeds in obtaining a valid witness $w' \in R_x$ with probability at least $1-\varepsilon(|x|)$. Intuitively, we will obtain a search algorithm for R because when s is random, the extractor K, who runs in time $q(n)$, cannot distinguish $G(s)$ from U_n, and thus we can run $K(x,(d,U_n))$ and still obtain a witness.

Formally, we claim that when $x \in L_R$, $K(x,(d,U_n))$ obtains a witness $w \in \hat{R}_x$ with probability at least $(1-2^{-n})\cdot(1-\varepsilon(n))/2 \geq 1/\mathrm{poly}(n)$. This holds due to the pseudorandomness of G. Specifically, let D be a distinguisher, given auxiliary input $(x,w) \in R$, who receives $z \in \{0,1\}^n$ and must distinguish the case that $z \leftarrow U_n$ from the case that $z \leftarrow G(U_{d\log n})$. Distinguisher D simply runs $K(x,(d,z \oplus w))$ to obtain a string w', and then runs the witness-checking algorithm $T(x,w')$ that distinguishes the case $(x,w') \in R$ from $(x,w') \notin \hat{R}$. D has running time $t(n)+t'(n) \leq q(n)$. We have already argued that for every $z \leftarrow G(U_{d\log n})$, $K(x,(d,z \oplus w))$ obtains a witness $w' \in R_x$ and thus D accepts with probability at least $1-\varepsilon(n)$. By the pseudorandomness of G against algorithms running in time $q(n)$, $D(U_n)$ must accept with probability at least $1-\varepsilon(n)-q(n) \geq (1-\varepsilon(n))/2 \geq 1/\mathrm{poly}(n)$. That is, $K(x,(d,U_n))$ outputs a $w' \in \hat{R}_x$ with probability at least $1/\mathrm{poly}(n)$, as desired. □

5.5 Constant-Round ZK Strong Proofs and Arguments of Knowledge

In this section, we present our main result regarding strong proofs of knowledge. Specifically, we show that under Assumption 5.8, there do not exist constant-round *zero-knowledge* strong proofs and arguments of knowledge. In fact, the lower bound holds even for *honest-verifier* zero knowledge. We note that the difference between the first and second items in the theorem with respect to $L_R \in \mathbf{BPP}$ or $L_R \in \mathbf{P}$ is due simply to the fact that under Assumption 5.8, $\mathbf{BPP} = \mathbf{P}$.

Theorem 5.11. *Let R be an* **MA***-relation.*

1. *If R has a constant-round* honest-verifier zero-knowledge *strong proof of knowledge with negligible completeness and soundness errors, then R is easy to search. In particular, $L_R \in \mathbf{BPP}$.*
2. *Assuming Assumption 5.8, if R has a constant-round* honest-verifier zero-knowledge *strong argument of knowledge with negligible completeness and soundness errors and an efficient prover, then every extension $\hat{R}$ of R is easy to search (deterministically). In particular, $L_R \in \mathbf{P}$.*

Proof. The idea behind the lower bound is as follows. Assume that there exists a constant-round zero-knowledge strong proof of knowledge (P,V). Then, there must be a round in which V sends a message with entropy $\omega(\log n)$. Otherwise, all of V's messages can be guessed with nonnegligible probability, and V is "close" to being deterministic. This is a contradiction because zero-knowledge verifiers must be probabilistic [GO]. Now, this message of V that has "high" entropy can be used in a similar way to the string s that is used in the proof of Theorem 5.11. That is, it can be used to reduce the probability that P^* convinces V, so that K does not have enough time to obtain any information from P^*. The rest of the proof then continues in the same way. (There are some additional technical details that arise from the fact that this high-entropy message sent by V is not actually uniformly distributed. However, this is solved by hashing the message to extract its randomness.)

As in the proof of Theorem 5.10, we first prove the case of strong *proofs* of knowledge, and thus where K is a black-box extractor.

Triviality of strong proofs. Let (P,V) be a $2k$-round honest-verifier zero-knowledge strong proof of knowledge. We will denote verifier messages by α and prover messages by β. For each $x \in L$ of length n, fix a witness $w_x \in R_x$. Call a sequence r of the prover's coin tosses *accepting* for x if the probability that $V(x)$ accepts when interacting with $P(x,w_x;r)$ is at least $1-4c(n)$, where c is the completeness error. By Markov's inequality, all but $1/4$ fraction of the r's are accepting.

Let $p_1(n), \ldots, p_k(n)$ be polynomials to be specified later, and let $p(n) = \prod_i p_i(n)$. For a setting r of the prover's coin tosses, we call a transcript $h = (\alpha_1, \beta_1, \ldots, \alpha_k, \beta_k)$ *heavy* with respect to r and x if

$$\Pr\left[(P(w_x; r), V)(x) \text{ exchange precisely the messages in } h\right] \geq 1/p(n).$$

Let L' be the set of $x \in L$ such that for at least $1/2$ of the prover's coin tosses r, there exists a heavy transcript with respect to x and r.

We will deal separately with inputs $x \in L'$ and $x \in L \setminus L'$. For instances $x \in L'$, we will show how to use the zero-knowledge simulator to construct a successful prover strategy and thereby extract a witness. For instances $x \notin L'$, we will show that there is a high-entropy verifier message and follow the intuitive argument outlined above.

Claim 5.12. *There is a PPT T_1 such that for every $x \in L'$, $T_1(x)$ outputs an element of R_x with probability at least $1/\mathrm{poly}(|x|)$.*

Proof of Claim. Let $x \in L'$. Then at least a $1/2 - 1/4 = 1/4$ fraction of the r's have a heavy transcript h *and* are accepting. First, we show how given a heavy transcript h for any accepting r, we can construct a successful prover strategy. Consider $P^*(x, h)$ that on input x and auxiliary input h: P^* simply sends the verifier the messages in h, and aborts if the verifier ever sends a message inconsistent with h. We now claim that if h is heavy for an accepting r, then $P^*(x, h)$ convinces V to accept with probability at least $1/p(n) - 4c(n)$. The reason is that P^* behaves exactly the same as $P(x, w_x; r)$ unless V sends something inconsistent with h and V remains consistent with probability at least $1/p(n)$.

Now, if we choose r uniformly at random and generate h according to $\langle P(w; r), V\rangle(x)$, then r will be accepting and h will be heavy wrt r with probability at least $(1/4) \cdot (1/p(n))$. Thus, for $h \leftarrow_{\mathrm{R}} \langle P(w), V\rangle(x)$, we see that $P^*(x, h)$ convinces $V(x)$ to accept with probability at least $(1/4) \cdot (1/p(n)) \cdot (1/p(n) - 4c(n)) \geq 1/\mathrm{poly}(n)$. By honest-verifier zero knowledge, the same must hold (up to a negligible difference in probability) when we generate h according to the honest-verifier simulator $S(x)$. (Formally, consider the distinguisher $D_x(h)$ that simulates the interaction between $P^*(x, h)$ and $V(x)$ and outputs 1 iff V accepts.)

Therefore, the prover strategy $P^{**}(x) = P^*(x, S(x))$ convinces $V(x)$ to accept with probability at least $1/\mathrm{poly}(n) > s(n)$ when $x \in L'$. By the strong validity/soundness property, the strong knowledge extractor $K(x)$ for P^{**} outputs a witness $w \in R_x$ with probability at least $1 - \varepsilon(n) \geq 1/\mathrm{poly}(n)$. Setting $T_1 = K$ completes the proof of the claim. □

Claim 5.13. *There is a PPT T_2 such that for every $x \in L \setminus L'$, $T_2(x)$ outputs an element of $\hat{R}_x$ with probability at least $1/\mathrm{poly}(|x|)$.*

Proof of Claim. Let $x \in L'$. Then greater than a $1/2 - 1/4 = 1/4$ fraction of the prover's coin tosses r have *no* heavy transcript and are accepting. For each such r, we claim that there exists a *partial* transcript $h = (\alpha_1, \beta_1, \ldots, \alpha_i, \beta_i)$ such that in the interaction between $P(x, w; r)$ and $V(x)$,

- h occurs with probability at least $\overline{p}_i(n) \stackrel{def}{=} \prod_{j=1}^{i} (1/p_j(n))$. (The probability here is over V's coin tosses.)
- For every α_{i+1}, the probability that α_{i+1} will be V's next message given that h has occurred is less than $1/p_{i+1}(n)$. That is, conditioned on h, the $(i+1)$'st message of V has min-entropy greater than $\log p_{i+1}(n)$

We call such an h a *good history* for r.

To prove the existence of good histories, consider a r for which the above does not hold. Since the claim doesn't hold, there exists a first verifier message α_1 that occurs with probability least $1/p_1(n)$. Set β_1 to be $P(x, w, \alpha_1; r)$. Since β_1 is deterministically computed, the partial transcript (α_1, β_1) still has probability at least $1/p_1(n)$. Thus, if the above does not hold for (α_1, β_1), it must be due to the second condition. This means that there exists a second verifier message α_2 (in response to (α_1, β_1)) whose conditional probability is at least $1/p_2(n)$. Set β_2 to be $P(x, w; r)$'s response. We obtain that the prefix $(\alpha_1, \beta_1, \alpha_2, \beta_2)$ appears with probability at least $(1/p_1(n)) \cdot (1/p_2(n))$, and so we can continue. Ultimately, we end up with a complete transcript of probability mass at least $\prod_{i=1}^{k}(1/p_i(n)) = p(n)$, contradicting the assumption that r has no heavy transcript.

So fix an accepting r and a good history $h = (\alpha_1, \beta_1, \ldots, \alpha_i, \beta_i)$ for r. Let V_{i+1} be the distribution of the $(i+1)^{\text{th}}$ verifier message given history h, and let $m(n) = \log p_{i+1}(n)$ be the min-entropy of V_{i+1}. We will convert this high-entropy message into an almost-uniform string by hashing (actually, any "strong randomness extractor" [NZ] would do): By the Leftover Hash Lemma [HILL], if we choose $g : \{0,1\}^{\text{poly}(n)} \to \{0,1\}^{m(n)/4}$ from a pairwise independent family, then with high probability $g(V_{i+1})$ has statistical difference at most $2^{-m(n)/3}$ from uniform. Fix such a g.

Now, let K be the strong knowledge extractor for (P, V) and let $t(n)$ be a bound on the running time of K. Then, similarly to the proof of Theorem 5.10, we construct a cheating prover $P^*(x, y)$ to which we give an auxiliary input $y = (w_x, r, h, g, s)$, where w_x is a witness, r are an accepting sequence of coin tosses for P, h a good transcript for r, g the hash function as above, and s a string of length $m(n)/4 = (\log p_{i+1}(n))/4$. For the first i messages, $P^*(x, (w_x, r, h, g, s))$ just sends the prover messages in the history h. If V replies with a message that is not consistent with h, then P^* aborts. Otherwise, if the transcript so far *is* consistent with h, then P^* obtains the $(i+1)^{\text{th}}$ verifier message α_{i+1} and checks if $g(\alpha_{i+1}) = s$. If yes, then P^* completes the proof, following the instructions of the honest prover $P(x, w_x; r)$. Otherwise, P^* halts and outputs nothing.

We argue that for every (r, h, g) as above and every s, the probability that P^* convinces an honest verifier V is at least $1/\text{poly}(n)$. First, recall that h is chosen in such a way that V gives consistent responses with probability at least $\overline{p}_i(n)$. Second, recall that $g(V_{i+1})$ has statistical difference at most $2^{-m(n)/3}$ from the uniform distribution on $\{0,1\}^{m(n)/4}$, so it equals s with probability at least

$$2^{-m(n)/4} - 2^{-m(n)/3} = \Omega(2^{-m(n)/4}) = \Omega(1/p_{i+1}(n)^{1/4}).$$

Thus P^* behaves exactly as $P(x, w_x; r)$ with probability at least $1/\text{poly}(n)$. Since $P(x, w_x; r)$ causes $V(x)$ to reject with probability at most $4c(n)$, which is negligible, we conclude that P^* convinces V to accept with probability at least $1/\text{poly}(n) - 4c(n) = 1/\text{poly}(n)$.

Now we construct a search algorithm T_2 for R, similar to the proof of Theorem 5.10. Upon input x of length n, machine T_2 chooses $i \leftarrow_{\text{R}} \{0, \ldots, k-1\}$, generates a *simulated* random history $h = (\alpha_1, \alpha_2, \ldots, \alpha_i, \beta_i) \leftarrow_{\text{R}} S(x)$, chooses $s \leftarrow_{\text{R}} \{0,1\}^{\log(p_{i+1}(n))/4}$, and chooses a random hash function g. Machine T_2 then invokes $K(x)$, and plays the role of $P^*(x, (w_x, r, h, g, s))$, answering all of K's oracle queries. However, when K asks a query containing an $(i+1)^{\text{th}}$ verifier message α_{i+1}, it checks if $g(\alpha_{i+1}) = s$ and if so, outputs `fail`. (In this case, T_2 cannot continue emulating P^* because it does not know the witness w_x for x nor the coin tosses r.) Otherwise, T_2 emulates P^* and hence aborts. T_2 continues this emulation until K concludes. If K outputs a value w, then T_2 outputs w.

It remains to prove that T_2 constitutes a valid search algorithm for R. Suppose $x \in L_R$. We need to show that T_2 outputs a valid witness $w \in \hat{R}_x$ with probability at least $1/\text{poly}(n)$. This is at least the probability that K outputs a valid witness (when interacting with $P^*(x, (w_x, r, i, h, g, s))$)

minus the probability that T_2 outputs `fail`. For each choice of i, the latter probability is at most $t(n)/p_{i+1}^{1/4}(n)$ (analogous to the bound $t(n)/q(n)$ in the proof of Theorem 5.10).

For the probability that K fails to output a valid witness $w \in \hat{R}_x$, we note that it suffices to bound the probability that K fails to output a witness $w \in R_x$ (in the *original* relation) under the assumption that h is generated according to the real interaction $\langle P(w_x), V\rangle(x)$ rather than by the simulator $S(x)$. This is because pairs $(x,w) \in R$ can be distinguished from pairs $(x,w) \notin \hat{R}_x$ in probabilistic polynomial time. Recall that at least $1/4$ of all r's are accepting and have no heavy transcript. For each such r, there is an i and a good history h for r of length $2i$. This implies that there is a fixed value of i such that at least $1/4k$ fraction of all r's are accepting and have a good history h of length $2i$. For each such r and h, the probability that the real interaction $\langle P(w_x;r), V\rangle(x)$ yields h is at least $\overline{p}_i(n)$. Thus, if we generate a length $2i$ partial transcript $h \leftarrow_{\mathrm{R}} \langle P(w_x), V\rangle(x)$, then h is good history for some accepting r with probability at least $(1/4k)\cdot\overline{p}_i(n) = \Omega(\overline{p}_i(n))$. In such a case, we have shown that $P^*(x,(w_x,r,i,h,g,s))$ convinces $V(x)$ to accept with probability at least $1/\mathrm{poly}(n) > s(n)$, and hence $K(x)$ must output a valid witness $w \in R_x$ with probability at least $1-\varepsilon(n)$. Thus, there is a setting of i such that $K(x)$ outputs a witness (when interacting with P^*) with probability at least $\Omega(1/\overline{p}_i(n))\cdot(1-\varepsilon(n))$.

Putting the above together, there is a setting of i such that $T_2(x)$ will output a witness $w \in \hat{R}_x$ with probability at least $\Omega(1/\overline{p}_i(n))\cdot(1-\varepsilon(n)) - t(n)/p_{i+1}^{1/4}(n)$. Thus, recursively defining $p_{i+1}(n) = \left(n\cdot\overline{p}_i(n)\cdot t(n)/(1-\varepsilon(n))\right)^4$ ensures that T_2 will output a witness with probability at least $1/\mathrm{poly}(n)$. □

To complete the proof, we obtain a single search algorithm by running one of the two search algorithms T_1, T_2 at random.

Triviality for strong arguments. As in the proof of Theorem 5.10, the only difference here is that the witness used by P^* must be "hidden". This is done in the same way. That is, given a parameter d and a generator that takes a seed of length $d\log n$, we give $P^*(x,y)$ the auxiliary input $y = (r,h,g,du,e)$, where $u = G(s)\oplus w$ and $s \in \{0,1\}^{d\log n}$. Then, upon receiving the i^{th} verifier message α_{i+1}, P^* computes $g(\alpha_{i+1})$, and completes the proof using the witness $w' = G(g(\alpha_{i+1}))\oplus u$. The rest of the proof is the same. □

Remark 5.14 (witness-indistinguishable strong proofs of knowledge). We note that Theorem 5.11 can also be used to rule out the possibility of obtaining *witness-indistinguishable* strong proofs of knowledge for any **NP**-complete language. This follows from the Feige–Shamir [FS1] construction of zero-knowledge arguments of knowledge. Specifically, [FS1] show that assuming the existence of one-way functions, any witness-indistinguishable proof (or argument) of knowledge for an **NP**-complete language can be used to obtain a system of zero-knowledge arguments of knowledge for **NP**. It can be verified that if the given witness-indistinguishable proof is a *strong* proof of knowledge, then the resulting zero-knowledge proof is also a *strong* proof of knowledge.

Part II
Protocols

In this part, we present several protocols that under some assumptions satisfy various forms of zero-knowledge. We wish to stress that not all of these assumptions are standard or even likely to be true; indeed, some of them are even conjectured to be false. Our use of such assumptions is therefore to demonstrate the kind of assumption one needs to make, and the kind of obstacles one faces, when trying to prove a negative result.

6 Two Round Zero Knowledge

In this section we present several positive results regarding 2-round plain zero-knowledge systems. Two of these results are proven under assumptions that, although used in previous works, are highly non-standard, and one of the results relies on an assumption that is widely believed to be false. Thus, we consider these results not so much as positive results, but rather as "negative results on negative results", showing what must be tackled in order to extend the results of Section 3. These results also give additional motivation to investigating the validity of these assumptions.

In Section 6.1 we present a two-round (private-coin) plain zero-knowledge *proof* system for a promise problem outside of **BPP** under the "Knowledge of Exponent Assumption" suggested by Damgård [Dam]. Thus, in order to extend Theorem 3.7 to hold for all problems outside of **BPP** (and not just for **NP**-complete languages), one must refute the Knowledge of Exponent Assumption.

In Section 6.2 we construct a two-round public-coin plain zero-knowledge proof system for **NP**, under the assumption that one can use nondeterminism to yield a super-polynomial saving in running time.[12] This assumption is a strengthening of the *negation* of Assumption 3.2 and is mostly believed to be *false*. Thus, we do not view this result as a "true" positive result but rather as showing that assumptions similar to Assumptions 3.2 and 3.5 are *necessary* for proving results such as Theorem 3.4 and Theorem 3.7.

In Section 6.3 we present a two-round public-coin plain zero-knowledge *argument* system for **NP** under a variant of the "Noninteractive CS Proofs Conjecture" suggested by Micali [Mic]. Thus, in order to extend Theorem 3.4 (which currently holds only for proof systems) to hold also for *argument systems*, one would have to disprove Micali's conjecture.

6.1 A Non-trivial 2-Round (Private-Coin) ZK Proof?

Theorem 3.7 gives evidence that 2-round zero-knowledge proofs (with perfect completeness) do not exist for **NP**-complete problems, but does not rule out the possibility that such proof systems exist for some nontrivial languages (i.e. ones outside of **BPP**). In this section, we show that the "Knowledge of Exponent Assumption" of Damgård [Dam] implies that indeed there does exist a nontrivial promise problem with a 2-round ZK proof.

Roughly speaking, the Knowledge of Exponent Assumption says that the only way to generate a Diffie-Hellman tuple is to "know" the corresponding exponent. To make this formal, we require a family of computational groups $\mathcal{G} = \bigcup_k \mathcal{G}_k$ of *prime order*. For simplicity, $\mathcal{G}_k$ can be thought of as consisting of all groups of the form QR_{2q+1}, where QR_{2q+1} is the group of quadratic residues in $\mathbb{Z}^*_{2q+1}$ for a k-bit prime q such that $2q+1$ is prime. More generally, we will only require

[12] See Section 6.2 for a fuller description of the assumption.

basic computational properties of the group, e.g. it is possible to do all the following in time poly(k): given security parameter k, generate the description of a random group $G \leftarrow_{\mathrm{R}} \mathcal{G}_k$; given the description of G, compute the order of G; given the description of G, carry out operations in G; etc. A full list of properties which suffice for our purposes can be found in the definition of a "computational group scheme" in [CS].

We consider the problem of distinguishing Diffie-Hellman non-tuples from Diffie-Hellman from tuples.

Definition 6.1 ((Complement of) the DECISIONAL DIFFIE-HELLMAN problem).

$$\begin{aligned}\overline{\mathrm{DDH}}_Y &= \{(G, g_0, g_1, h_0, h_1) : G \in \mathcal{G}, g_0, g_1, h_0, h_1 \in G \setminus \{1\}, \mathrm{DLog}_{g_0}(h_0) \neq \mathrm{DLog}_{g_1}(h_1)\} \\ \overline{\mathrm{DDH}}_N &= \{(G, g_0, g_1, h_0, h_1) : G \in \mathcal{G}, g_0, g_1, h_0, h_1 \in G \setminus \{1\}, \mathrm{DLog}_{g_0}(h_0) = \mathrm{DLog}_{g_1}(h_1)\}\end{aligned}$$

For each k, we let $\overline{\mathrm{DDH}}_{Y,k} = \{(G, g_0, g_1, h_0, h_1) \in \overline{\mathrm{DDH}}_Y : G \in \mathcal{G}_k\}$.

We require two assumptions about this problem. The first is (even weaker than) the standard DDH Assumption.

Assumption 6.2 ((Weak) DDH Assumption). The promise problem $\overline{\mathrm{DDH}}$ is not in **BPP**. Moreover, if A is any PPT that rejects *all* elements of $\overline{\mathrm{DDH}}_N$ with probability at least 2/3 (over A's coin tosses), then A rejects a *random* element of $\overline{\mathrm{DDH}}_{Y,k}$ with probability at least 1/2 for all sufficiently large k.

The second is a nonstandard assumption, which intuitively says that, given random (G, g_0, g_1), the only way to generate (g_1, h_1) so that (G, g_0, g_1, h_0, h_1) is a Diffie-Hellman tuple is by raising g_0 and g_1 to some known exponent. That is, for any efficient A that generates Diffie-Hellman tuples, there is an efficient $\hat{A}$ that generates the corresponding exponent when given the coin tosses of A.

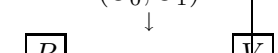
Assumption 6.3 (Knowledge-of-Exponent Assumption). For every PPT A, there exists a PPT $\hat{A}$ such that

$$\Pr_{\substack{G \leftarrow_{\mathrm{R}} \mathcal{G}_k \\ g_0, g_1 \leftarrow_{\mathrm{R}} G \setminus \{1\}, r}} \left[A(G, g_0, g_1; r) = (g_0^x, g_1^x) \text{ for some } x \in \mathbb{Z}^*_{|G|} \text{ and } \hat{A}(G, g_0, g_1; r) \neq x \right] = \mathrm{neg}(k).$$

We do not regard this to be a "reasonable" assumption, but to date it has not been proven false, and our result below shows that proving triviality of 2-round ZK will require such a falsification.[13] Already when introducing this assumption, Damgård [Dam] hinted at the possibility that it can be used to construct 2-round zero-knowledge protocols (but the protocol suggested there is an Identification Scheme rather than a proof of membership, and thus automatically involves a common reference string). More recently, in [HT, BP], this assumption and some variants of it were used to construct 3-round zero-knowledge arguments for **NP**. We use it to construct a 2-round plain zero-knowledge proof system for a nontrivial promise problem.

Theorem 6.4. *If Assumption 6.3 holds, then there is a 2-round plain zero-knowledge proof for a promise problem not in* **BPP**. *Specifically, this promise problem is of the form* $\overline{\mathrm{DDH}}' = (\overline{\mathrm{DDH}}'_Y, \overline{\mathrm{DDH}}'_N)$, *where* $\overline{\mathrm{DDH}}'_Y \subset \overline{\mathrm{DDH}}_Y$ *and* $\overline{\mathrm{DDH}}'_N = \overline{\mathrm{DDH}}_N$.

[13] See [Nao2] for a discussion of why such assumptions are difficult to falsify. Nevertheless, some related assumptions, suggested in [HT], have been falsified in [BP].

Public input: (C_0, C_1), both circuits with n input gates and m output gates	(C_0, C_1) 
Step V1: Choose $b \leftarrow_{\mathrm{R}} \{0,1\}$, $r \leftarrow_{\mathrm{R}} \{0,1\}^n$. Set $\alpha = C_b(r)$	$\xleftarrow{\alpha}$
Step P1: If $\alpha \in \mathrm{Image}(C_0)$, set $c = 0$, else set $c = 1$.	$\xrightarrow{c}$
V accepts if $c = b$ and rejects otherwise.	

Protocol 6.6. Interactive Proof for Disjoint Images

We actually abstract away the number theory by reducing it to the following more general problem:

Definition 6.5 (Disjoint Images). Disjoint Images is the promise problem $\mathrm{DI} = (\mathrm{DI}_Y, \mathrm{DI}_N)$ whose instances consist of pairs of circuits $C_0, C_1 : \{0,1\}^k \to \{0,1\}^\ell$ and

$$\begin{aligned} \mathrm{DI}_Y &= \{(C_0, C_1) : \mathrm{Image}(C_0) \cap \mathrm{Image}(C_1) = \emptyset\} \\ \mathrm{DI}_N &= \{(C_0, C_1) : C_0(U_k) \text{ and } C_1(U_k) \text{ have the same distribution}\} \end{aligned}$$

Disjoint Images is a restriction of Statistical Difference, the complete problem for statistical zero knowledge [SV], and is a generalization of many well-known computational problems such as Graph Nonisomorphism and Quadratic Nonresiduosity. Note that $\overline{\mathrm{DDH}}$ reduces to Disjoint Images under the reduction $\psi(G, g_0, g_1, h_0, h_1) = (C_0, C_1)$ where the circuits $C_0, C_1 : \mathbb{Z}^*_{|G|} \to G^2$ are defined by $C_b(x) = (g_b^x, h_b^x)$. Notice that if $(G, g_0, g_1, h_0, h_1) \in \overline{\mathrm{DDH}}_N$ then $C_0(x)$ and $C_1(x)$ have the same distribution. (Specifically, for some r, we have $C_0(x) = (g_0^x, h_0^x) = (g_0^x, g_0^{rx})$ and $C_1(x) = (g_1^x, h_1^x) = (g_1^x, g_1^{rx})$. Thus, for a random x, the distributions are identical.) In contrast, if $(G, g_0, g_1, h_0, h_1) \in \overline{\mathrm{DDH}}_Y$, then the discrete logs of h_0 and h_1 are different and so the distributions are disjoint. Note also that by encoding the input instance into the description of the circuits, we can ensure that ψ is injective and can be inverted in polynomial time (i.e. is an invertible reduction).

Protocol 6.6 is the interactive proof system for Disjoint Images from [SV]. This proof system is easily seen to be *honest-verifier* perfect zero knowledge, but is unlikely to be zero knowledge for cheating verifiers. Intuitively, the verifier can learn something by sending the prover an α for which it does not know if $\alpha \in \mathrm{Image}(C_0)$ (e.g. when the verifier can determine whether $C_0 : \{0,1\}^k \to \{0,1\}$ is satisfiable by sending $\alpha = 1$ to the prover). However, there might be YES instances of Disjoint Images for which it is infeasible to generate $\alpha \in \mathrm{Image}(C_0)$ without "knowing" that this is the case. The following assumption formalizes this intuition.

Assumption 6.7. There exists a subset $\mathrm{DI}'_Y \subset \mathrm{DI}_Y$ such that

1. The promise problem $\mathrm{DI}' = (\mathrm{DI}'_Y, \mathrm{DI}_N)$ is not in **BPP**.
2. For every PPT A, there exists a PPT $\hat{A}$ such that for every $(C_0, C_1) \in \mathrm{DI}'_Y$

$$\Pr_r[A(C_0, C_1; r) \in \mathrm{Image}(C_0) \iff \hat{A}(C_0, C_1; r) = 0] \geq 1 - \mathrm{neg}(n),$$

where $n = |(C_0, C_1)|$.

Note that, unlike the Knowledge-of-Exponent Assumption, here we require the "extractor" $\hat{A}$ to work with high probability for *every* instance $(C_0, C_1) \in \mathrm{DI}'_Y$, rather than just a random instance. Nevertheless, we can still prove that this assumption follows from the Knowledge-of-Exponent Assumption:

Lemma 6.8. *Assumptions 6.2 and 6.3 imply Assumption 6.7.*

Proof. We use the probabilistic method to choose random subset $\overline{\mathrm{DDH}}'_Y$ of $\overline{\mathrm{DDH}}_Y$, and apply the reduction ψ to Disjoint Images to obtain $\mathrm{DI}'_Y = \psi(\overline{\mathrm{DDH}}'_Y)$ which will almost surely satisfy the requirements of Assumption 6.7. Specifically, for each k, we choose $x_1, \ldots, x_k \leftarrow_{\mathrm{R}} \overline{\mathrm{DDH}}_{Y,k}$ and set $\overline{\mathrm{DDH}}'_{Y,k} = \{x_1, \ldots, x_k\}$, and $\overline{\mathrm{DDH}}'_Y = \bigcup_k \overline{\mathrm{DDH}}'_{Y,k}$. The intuition is that the DDH Assumption ensures that the random instances x_i are "hard" (so even the restricted problem $\overline{\mathrm{DDH}}'$ will not be in **BPP** w.h.p.), and that the DDH Assumption ensures that we can do "extraction" on the random instances x_i with probability $1 - \mathrm{neg}(k)$ (so we will be able to do extraction on *all* the x_i's by a union bound).

To show that $(\mathrm{DI}'_Y, \mathrm{DI}_N)$ is not in **BPP**, it suffices to show that $(\overline{\mathrm{DDH}}'_Y, \overline{\mathrm{DDH}}_N)$ is not in **BPP**. Fix any probabilistic polynomial-time Turing machine A which rejects all elements of $\overline{\mathrm{DDH}}_N$. By the DDH Assumption, A rejects with probability at least $1/2$ on a random instance of $\overline{\mathrm{DDH}}_{Y,k}$ for all sufficiently large k. Thus, for at least .1 fraction of the $\overline{\mathrm{DDH}}_{Y,k}$ instances, A rejects with probability at least .4 over its coin tosses. Thus, when we choose k random instances, with probability at least $1 - \exp(-\Omega(k))$, A will reject at least one element of $\overline{\mathrm{DDH}}'_{Y,k}$ with probability at least .4. Taking $k \to \infty$, we see that A fails to be a **BPP** algorithm for $(\overline{\mathrm{DDH}}'_Y, \overline{\mathrm{DDH}}_N)$ with probability 1. Since there are only countably many PPT algorithms A, we conclude that $(\mathrm{DI}'_Y, \mathrm{DI}_N)$ is not in **BPP** with probability 1.

We now show that the extraction condition (Item 2) in Assumption 6.7 holds with probability 1, using the Knowledge-of-Exponent Assumption. Given any PPT A, we wish to show that A satisfies the second condition (i.e. the "extraction condition") of Assumption 6.7 with high probability. We will do this by construct a related PPT A' to which we will apply the Knowledge-of-Exponent assumption. Intuitively, since the extraction condition Knowledge-of-Exponent holds on random instances with high probability, we will be able to argue that we can extract from A with high probability on all k instances of DI'_Y.

Specifically, we construct A' as follows: On input (G, g_0, h_0) and coin tosses (r, g_1, h_1) (where $g_1, h_1 \in G \setminus \{1\}$), A' computes $(C_0, C_1) = \psi(G, g_0, g_1, h_0, h_1)$, and outputs $A(C_0, C_1; r)$. By the definition of ψ, A' outputs a pair of the form (g_0^x, h_0^x) iff A outputs an element of $\mathrm{Image}(C_0)$.

Using the Knowledge-of-Exponent Assumption, we obtain a PPT $\hat{A}'(G, g_0, h_0; r, g_1, h_1)$ such that

$$\Pr_{\substack{G \leftarrow_{\mathrm{R}} \mathcal{G}_k \\ g_0, h_0, g_1, h_1 \leftarrow_{\mathrm{R}} G \setminus \{1\}, r}} \left[\begin{array}{c} A'(G, g_0, h_0; r, g_1, h_1) = (g_0^x, h_0^x) \text{ for some } x \in \mathbb{Z}^*_{|G|} \\ \text{and } \hat{A}'(G, g_0, h_0; r, g_1, h_1) \neq x \end{array} \right] = \mathrm{neg}(k).$$

Note that, in the above probability, the distribution on (G, g_0, h_0, g_1, h_1) is statistically close to uniform on $\overline{\mathrm{DDH}}_{Y,k}$. (The only difference is that the above probability allows the possibility that $\mathrm{DLog}_{g_0}(h_0) = \mathrm{DLog}_{g_1}(h_1)$, but this only occurs with probability $1/|G| = \mathrm{neg}(k)$.) Thus, with probability at least $1 - 1/k^3$ over $(G, g_0, h_0, g_1, h_1) \leftarrow_{\mathrm{R}} \overline{\mathrm{DDH}}_{Y,k}$, we have

$$\Pr_r \left[\begin{array}{c} A'(G, g_0, h_0; r, g_1, h_1) = (g_0^x, h_0^x) \text{ for some } x \in \mathbb{Z}^*_{|G|} \\ \text{and } \hat{A}'(G, g_0, h_0; r, g_1, h_1) \neq x \end{array} \right] \leq k^3 \cdot \mathrm{neg}(k) = \mathrm{neg}(k).$$

Since we define $\overline{\mathrm{DDH}}'_{Y,k}$ to consist of k random elements of $\overline{\mathrm{DDH}}_{Y,k}$, the following holds with probability at least $1 - 1/k^2$ over the choice of $\overline{\mathrm{DDH}}'_{Y,k}$:

$$\forall (G, g_0, h_0, g_1, h_1) \in \overline{\mathrm{DDH}}'_{Y,k} \qquad \Pr_r \left[\begin{array}{c} A'(G, g_0, h_0; r, g_1, h_1) = (g_0^x, h_0^x) \text{ for some } x \in \mathbb{Z}^*_{|G|} \\ \text{and } \hat{A}'(G, g_0, h_0; r, g_1, h_1) \neq x \end{array} \right] = \mathrm{neg}(k). \tag{3}$$

The probability (over the choice of $\overline{\mathrm{DDH}}'_Y$) that Equation (3) does not hold for some $k \geq k_0$ is at most $\sum_{k=k_0}^{\infty}(1/k^2)$, which tends to 0 as $k_0 \to \infty$. Thus, Equation (3) for all PPT A and all sufficiently large k with probability 1 over the choice of $\overline{\mathrm{DDH}}'_Y$.

Now we translate this into showing that DI' satisfies Assumption 6.7. Recall that $A'(G, g_0, h_0; r, g_1, h_1) = (g_0^x, h_0^x)$ iff $A(C_0, C_1; r) \in \mathrm{Image}(C_0)$, where $(C_0, C_1) = \psi(G, g_0, h_0, g_1, h_1)$. So we need only convert $\hat{A}'$ into an extractor $\hat{A}$ for A. We define $\hat{A}(C_0, C_1; r)$ as follows: compute the instance $(G, g_0, g_1, h_0, h_1) = \psi^{-1}(C_0, C_1)$, run $\hat{A}'(G, g_0, h_0; r, g_1, h_1)$ to obtain $x \in \mathbb{Z}^*_{|G|}$, and output 0 iff $A'(G, g_0, h_0; r, g_1, h_1) = (g_0^x, h_0^x)$.

With this definition, Equation (3) becomes:

$$\forall (C_0, C_1) \in \mathrm{DI}'_Y \qquad \Pr_r[A(C_0, C_1; r) \in \mathrm{Image}(C_0) \iff \hat{A}(C_0, C_1; r) = 0] \geq 1 - \mathrm{neg}(n),$$

for all sufficiently large $n = |(C_0, C_1)|$, as desired. □

To complete the proof, we show that Assumption 6.7 suffices for a nontrivial 2-round zero-knowledge proof.

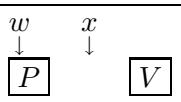
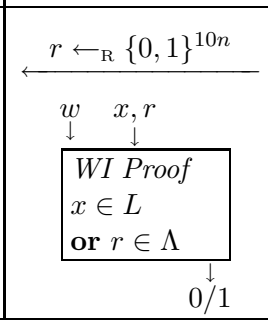
Theorem 6.9. *If Assumption 6.7 holds, then Protocol 6.6 is a 2-round proof system for a nontrivial promise problem (specifically,* $\mathrm{DI}' \notin \mathbf{BPP}$*) that is statistical plain zero knowledge. The protocol has perfect completeness and soundness error* 1/2*. Moreover, the* n*-fold parallel repetition of the proof system is also statistical plain zero knowledge (and has negligible soundness error).*

Proof. We obtain (perfect) completeness and soundness because these properties hold for all instances of Disjoint Images. Thus we need only establish the zero knowledge property on the subproblem DI'. Given a uniform PPT verifier V^*, we apply Item 2 of Assumption 6.7 to $A = V^*$ and obtain PPT $\hat{A}$. Then our simulator S for V^* operates as follows on input (C_0, C_1): Generate random coin tosses r for V^*, let $\alpha = V^*(C_0, C_1; r)$, let $b = \hat{A}(C_0, C_1; r)$, and output the transcript $(\alpha, b; r)$. It follows from the definition of $\hat{A}$ and the prover P that this simulation has negligible statistical difference from $\langle P, V^* \rangle (C_0, C_1)$ whenever $(C_0, C_1) \in \mathrm{DI}'_Y$.

For the parallel repetition of the protocol, we must consider verifiers V^* whose first message consists of an n-tuple $(\alpha_1, \ldots, \alpha_n)$, so we define A to output one of these components at random. That is, we define $A(C_0, C_1; r, i) = V^*(C_0, C_1; r)_i$, and apply Assumption 6.7 to obtain a corresponding extractor $\hat{A}$. Now, the simulator works as follows on input (C_0, C_1): Generate random coin tosses r for V^*, let $(\alpha_1, \ldots, \alpha_n) = V^*(C_0, C_1; r)$, let $b_i = \hat{A}(C_0, C_1; r, i)$ for $i = 1, \ldots, n$, and output the transcript $((\alpha_1, \ldots, \alpha_n), (b_1, \ldots, b_n); r)$. □

6.2 Zero Knowledge from the Power of Nondeterminism

In this section we construct a 2-round public-coin proof system that is plain zero knowledge under the hypothesis that trapdoor permutations exist and (a slight strengthening of) the assumption that $\mathbf{Dtime}(f(n)) \subseteq \mathbf{NP}$ for some super-polynomial function $f(\cdot)$. Note that this means that *every* language that can be recognized in deterministic time $f(n)$ can be solved in nondeterministic polynomial time. We prove the following theorem:

Public input: $x \in \{0,1\}^n$ (statement to be proved is "$x \in L$") **Prover's auxiliary input:** w (a witness that $x \in L$)	
Steps V1 (Send long random string): Verifier sends $r \leftarrow_{\mathrm{R}} \{0,1\}^{10n}$. **Step P2 (WI Proof):** Prover proves to verifier using its input w via a witness-indistinguishable (WI) proof/argument system that either $x \in L$ or $r \in \Lambda$, where $r \in \Lambda$ iff there exists a Turing machine M of description length at most $\frac{\lvert r\rvert}{2}$ such that, on the empty input, M outputs r within $f(n)$ steps. The verifier accepts if the proof is completed successfully.	

The right column contains a schematic description of the protocol as defined in the left column.

Protocol 6.11. Generic bounded auxiliary-input zero-knowledge protocol.

Theorem 6.10. *Suppose that there exist enhanced trapdoor permutations*[14] *and that there is a super-polynomial function $f : \mathbb{N} \to \mathbb{N}$ (i.e., $f(n) = n^{\omega(1)}$) such that* $\mathbf{Dtime}(f(n)) \subseteq \mathbf{NP}$. *Furthermore, suppose that this inclusion is constructive in the following sense: given a Turing machine M and input $x \in \{0,1\}^n$, if $M(x) = 1$ within at most t steps for $t \leq f(n)$, then it is possible to obtain a short (i.e.,* $\mathrm{poly}(n)$*-sized)* **NP** *witness for this fact within* $\mathrm{poly}(t)$ *steps.*
Then, there exists a 2*-round public-coin proof system for L that is* plain *zero knowledge, with perfect completeness, negligible soundness error, and an efficient prover.*[15] *In fact, it is bounded-resettable zero knowledge (in the sense of Theorem 4.4).*

Proof. Let $L \in \mathbf{NP}$. Our 2-round proof system for L will be an implementation of Protocol 6.11, which is suggested in the full version of [Bar].

The results of [Bar] imply that any implementation of Protocol 6.11 (satisfying a certain prover efficiency condition) will be plain zero knowledge (and even zero knowledge to verifiers who have at most n bits of auxiliary input). The prover efficiency condition mentioned above is that given $r \in \Lambda$ and a machine M that outputs r within $t \leq f(n)$ steps, it should be possible to carry out the prover strategy of the WI Proof (Step P2) in time $\mathrm{poly}(t)$. (This is what the simulator does, taking M to be the verifier strategy V^*, with coin tosses reduced to n via a pseudorandom generator.) In [Bar], the WI Proof is implemented by a protocol that has several rounds and is only computationally sound. Thus a *constant-round* zero-knowledge *argument*, rather than a 2*-round* zero-knowledge *proof*, is obtained.

The crucial observation in the proof is that under our assumptions, the language Λ is in fact in **NP**. Therefore, for the second stage we can use a ZAP [DN], which is a 2-round, public-coin witness-indistinguishable (statistically sound) proof system for **NP**. The resulting protocol has 2 rounds because Step V1 of our protocol can be sent along with the verifier message of the ZAP. ZAPs are known to exist if (enhanced) trapdoor permutations exist [DN]. Because a random string r will have Kolmogorov complexity higher than $|r|/2$ with very high probability, and because ZAPS

[14] Informally, an *enhanced trapdoor permutation family* [Gol4] is trapdoor permutation family where it is hard to find a preimage of a point y in the domain even if the adversary is given the coin tosses used to generate y. We assume the existence of such families in order to have the existence of non-interactive zero-knowledge proofs (and hence ZAPs [DN]). See discussion in Goldreich [Gol4, Apdx C.1,C.4.1]

[15] Actually, zero knowledge is obtained even with respect to verifiers with "bounded auxiliary input"; i.e., verifiers who have an a priori polynomial bound on the length of the auxiliary input that they receive.

are proof systems (i.e. are statistically sound), our protocol will be also be statistically sound.

The requirement that the inclusion of $\mathbf{Dtime}(f(n))$ in **NP** be constructive is used to satisfy the prover efficiency condition mentioned above. (The ZAP prover is efficient given an **NP** witness.)

To obtain bounded-resettable zero knowledge, we have the prover use a pseudorandom function for its randomness; the proof that the modified protocol is bounded-resettable zero knowledge is a simpler version of the proof of Theorem 7.1 and hence is omitted here.

□

6.3 A Two-Round Zero-Knowledge Argument for NP?

The technique used above in the proof of Theorem 6.10 can also be used to show a connection between the existence of 2-round plain zero-knowledge arguments and a conjecture of Micali [Mic] regarding the existence of noninteractive (or two-round[16]) CS proofs (or universal arguments [BG]). Specifically, we show that if two-round universal arguments exist (i.e. if Micali's conjecture is true), then **NP** has 2-round plain zero-knowledge argument systems.[17] In other words, in order to extend the impossibility of results of Section 3 to argument systems, one must refute Micali's conjecture.

For our purposes we can use the following somewhat simplified and ad-hoc definition of universal arguments:

Definition 6.12 (Universal Arguments). Let $f : \mathbb{N} \to \mathbb{N}$ be a (polynomial-time computable) super-polynomial function, and let R_f be the following $\mathbf{Ntime}(f(n))$-hard relation: $(M \circ x, w) \in R_f$ (where $x \in \{0,1\}^*$, M is a description of a Turing machine, and $\circ$ denotes concatenation) if $|w| \leq f(|x|)$ and on input x, w, the machine M halts within $|w|$ steps and outputs 1.

A *universal argument* system is a computationally sound argument system for the language $L_f = L_{R_f}$ satisfying the following requirements:

Efficient verification The protocol is polynomially bounded and the verifier runs in probabilistic polynomial time. That is, there is a polynomial $p(\cdot)$ such that the total running time of the verifier (and hence also the total communication complexity of both parties) when interacting in a proof for $M \circ x$ is bounded by $p(|M \circ x|)$.

Efficient prover On input $(M \circ x, w) \in R_f$ the honest prover algorithm P runs in time polynomial in $|M \circ x| + |w|$ (even if this is shorter than $f(|x|)$) and causes the honest verifier to accept with probability 1.

Computational soundness against 2^{n^ϵ}-sized circuits There exists some $\epsilon > 0$ such that if $M \circ x \notin L_f$ then for every 2^{n^ϵ}-sized circuit P^* (where $n = |M \circ x|$), the probability that V outputs 1 after interacting with P^* on input $M \circ x$ is negligible.

Public verifiability The verifier decides whether to accept or reject the proof by applying a public *linear-time* algorithm to the protocol's *transcript* of messages exchanged (which need not include the verifier's coin tosses nor the input statement).[18]

[16] Micali's conjecture refers to CS Proofs that are *noninteractive* in Common Random String Model; this is equivalent to considering *two-round, public-coin* protocols.

[17] Micali himself had observed that his conjecture implies 3-round zero-knowledge arguments for **NP** (personal communication, June 2003).

[18] Note that the algorithm's running time is linear in the length of the transcript and so if the communication complexity of the protocol is shorter than the length of the statement then so will be running time. This property is not important in this section but will be important in Section 7. The requirement of *linear* time is not essential, and any fixed polynomial would do.

A constant-round public-coin protocol satisfying Definition 6.12 (with respect to any function $f(\cdot)$ such that $f(n) < 2^n$) can be constructed under standard assumptions (namely, the existence of hash functions that are collision resistant with respect to subexponential-sized circuits [Kil, Mic, BG]). Micali [Mic] suggested a *two-round public-coin* protocol that may satisfy Definition 6.12 (in this protocol the verifier's first message consists of sending a hash function, which is then used to make the constant-round protocol non-interactive by following the Fiat-Shamir heuristic). We now prove the following theorem:

Theorem 6.13. *Suppose that there exist two-round universal argument systems for* $\mathbf{Ntime}(f(n))$ *for some super-polynomial* $f : \mathbb{N} \to \mathbb{N}$ *and that there exist enhanced trapdoor permutations secure against* 2^{n^ϵ}*-sized circuits for some constant* $\epsilon > 0$*. Then, there exists a 2-round plain zero-knowledge argument system for* **NP***, with perfect completeness, negligible soundness error, and an efficient prover. Furthermore, if the universal argument system is public coin, then so is the obtained zero knowledge system.*

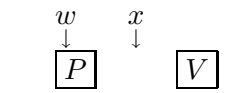
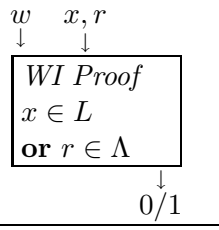
Proof. Under the assumed trapdoor permutation family, we can construct a commitment scheme Com^w (where the superscript w stands for "weak") that can be broken in time $2^{n^{\epsilon/2}}$ where the universal argument system is secure against 2^{n^ϵ}-sized circuits. (When reducing the soundness of our protocol to that of the universal argument, we will need to break the commitment scheme. This "complexity leveraging technique" was first introduced in [CGGM].) We will assume for simplicity that we have a one-round WI system (as the one of [BOV], that requires an additional assumption), even though we can use also the ZAP system of [DN] in its place and hence the result does hold under the stated assumptions.[19] Our two-round zero-knowledge protocol for **NP** is Protocol 6.14. It is a variant of Protocol 6.11 and hence is zero-knowledge for similar reasons.

The completeness property of this protocol is fairly straightforward. To prove computational soundness, we note that if there is a polynomial-time cheating prover P^* that causes the honest verifier to accept $x \notin L$ with non-negligible probability then we can convert it to a $2^{O(n^{\epsilon/2})}$-time cheating prover P^{**} for the universal argument system. Indeed, with extremely high probability the random string r chosen by the verifier is *not* a member of Λ, but by using brute force to extract the message committed to by P^*, the prover P^{**} can obtain a convincing UA proof that $r \in \Lambda$ (note that the commitment must contain such a message since $x \notin L$ and the WI system is statistically (or even perfectly) sound).

The proof of plain zero knowledge is almost identical to the proof in [Bar]. Given a PPT verifier strategy V^*, the simulator uses a pseudorandom generator to reduce the number of coins V^* uses to n. Since V^* is a uniform algorithm, and hence can be assumed to have description length of at most n, we see that this code, along with x and the random coins, can be used as a witness that $r \in \Lambda$ by the simulator. The simulator uses this witness to compute a commitment to the honest prover message of the universal argument system in Step P2a. In Step P2b it then uses the honest prover algorithm of the WI system to prove that either $x \in L$ or z is a commitment to an accepting message. Indistinguishability follows from indistinguishability of the commitment scheme and the WI property of the proof system used in Step P2b.

□

[19] Simply plugging in the ZAP system instead of the one-round system in Step P2b seems to yield a *three-round* protocol, as the ZAP system takes two rounds. However, because ZAPs are public coin, the verifier's first message in the ZAP is independent of the statement proven, and so the verifier can send this message as Step V1c.

Public input: $x \in \{0,1\}^n$ (statement to be proved is "$x \in L$") **Prover's auxiliary input:** w (a witness that $x \in L$)	
Steps V1a (Send long random string): Verifier sends $r \leftarrow_{\mathrm{R}} \{0,1\}^{10n}$.	$r \leftarrow_{\mathrm{R}} \{0,1\}^{10n}$ ⟵
Steps V1b (Send first message of UA): Verifier sends its first message v of a universal argument proof that $r \in \Lambda$. As in Protocol 6.11, $r \in \Lambda$ iff there exists a Turing machine M of description length at most $\frac{\|r\|}{2}$ such that, on the empty input, M outputs r within $f(n)$ steps. Note that Λ is easily reducible to L_f.	⟵ v
Step P2a ("Encrypted" UA): Prover sends $z = \mathsf{Com}^w(0^m)$ where m is the length of the prover's message in a universal argument proof for r. Com^w is a commitment that can be completely broken in time $2^{n^{\epsilon/2}}$.	$z = \mathsf{Com}^w(0^m)$ ⟶
Step P2b (WI Proof): Prover proves to verifier using its input w via a one-message witness-indistinguishable (WI) proof/argument system that either $x \in L$ or z is a commitment to a message p such that (v,p) is an accepting transcript in the universal argument system for the statement $r \in \Lambda$.	

Protocol 6.14. Uniform zero-knowledge using two-round universal arguments.

7 A Constant-Round Public-Coin Bounded-Resettable ZK Argument

In this section, we show that under a strong but standard assumption (namely the existence of strongly collision-resistant hash function and trapdoor permutations) there does exist a constant-round public-coin bounded-resettable zero-knowledge *argument* system for **NP**. Namely, we prove the following theorem:

Theorem 7.1. *Assume that there exist families of hash functions that are collision resistant against 2^{n^ϵ}-sized circuits (for some constant $\epsilon > 0$) and that there exist enhanced trapdoor permutations. Then there exists a constant-round, public-coin bounded-resettable auxiliary-input zero-knowledge argument for* **NP**. *Moreover, the protocol has perfect completeness, negligible soundness error, and an efficient prover.*

This result is interesting for two reasons:

1. This is the first construction of a constant-round argument system that is bounded-resettable zero knowledge in the plain model. This is also the first construction of a bounded-resettable zero-knowledge argument that is public-coin (regardless of the number of rounds).

2. This construction demonstrates that the lower bound of Section 4 for resettable *proofs* cannot be extended to argument systems.

7.1 Proof of Theorem 7.1

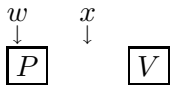
The proof is based on using a ZAP (i.e., a 2-round, public-coin resettable-WI proof system) [DN] as a component in the concurrent zero-knowledge protocol of [Bar]. To prove Theorem 7.1, we construct a constant-round public-coin zero-knowledge argument that remains zero knowledge even under attack by a verifier that is allowed to reset the prover (as in the model of [CGGM]) a fixed polynomial number of times. Our construction is Protocol 7.2, which is a variant of a concurrent zero-knowledge argument of Barak [Bar]. Let c be some constant, the protocol will be zero knowledge against a verifier that can reset the prover at most n^c times, where n is the length of the statement to be proven. We will also ensure that the prover communication complexity of the protocol is at most $O(n)$.

Remarks on implementation of Protocol 7.2.

- We note that under our the assumed hash functions, there exists a constant-round public-coin universal argument system (as per Definition 6.12) for the language Λ [Kil, Mic, Bar, BG].

- We will use the fact that under our assumptions, for every constant $\delta > 0$ we can obtain a universal argument system in which the total communication complexity is of size at most m^δ, where m is the size of the statement proven, and hence the verifier decides its acceptance also by applying a $O(m^\delta)$-time algorithm. Since the communication complexity of the rWI proof is a fixed polynomial in n and the acceptance time of the universal argument, by choosing a small enough $\delta > 0$, we can ensure that the total prover communication of the entire Protocol 7.2 is $O(n)$. (In fact, all messages of the protocol except for the verifier message of Step V3 will be of size $O(n)$.)

<table>
<tr><td>Public input: $x \in \{0,1\}^n$ (statement to be proved is "$x \in L$")

Prover's auxiliary input: w (a witness that $x \in L$)</td><td>  </td></tr>
<tr><td>c is an arbitrary given constant. (The protocol will be zero knowledge against n^c resets where n is the length of the input.) Given c, $\delta > 0$ is set to be a sufficiently small constant so that the prover's communication complexity is at most $O(n)$.)</td><td></td></tr>
<tr><td>Step V1 (Send hash function): Verifier sends h which is a description of random hash function chosen from a collision-resistant hash function collection $\mathcal{H}$. We assume that $h : \{0,1\}^* \to \{0,1\}^{n^\delta}$.

Step P2 (Commitment to "junk"): Prover sends $z = \mathsf{Com}(0^{n^\delta})$.
Step V3 (Send long random string): Verifier sends $r \leftarrow_{\mathrm{R}} \{0,1\}^{n^{2+c}}$.</td><td>$h \leftarrow_{\mathrm{R}} \mathcal{H}$

$z = \mathsf{Com}(0^{n^\delta})$

$r \leftarrow_{\mathrm{R}} \{0,1\}^{n^{2+c}}$</td></tr>
<tr><td>Steps P,V4–7 ("Encrypted" universal argument): The following is repeated twice: the Verifier sends a random string of length n^δ to prover, and prover sends a commitment to a 0^{n^δ}. We denote the transcript of this stage by $\langle \alpha, \beta, \gamma, \delta \rangle$.</td><td>$\alpha \leftarrow_{\mathrm{R}} \{0,1\}^{n^\delta}$
$\beta = \mathsf{Com}(0^{n^\delta})$
$\gamma \leftarrow_{\mathrm{R}} \{0,1\}^{n^\delta}$
$\delta = \mathsf{Com}(0^{n^\delta})$</td></tr>
<tr><td>Step P8 (rWI Proof): Prover proves to verifier using its input w via a resettable-witness-indistinguishable (rWI) proof system that either $x \in L$ or the decommitted messages of the transcript $\langle \alpha, \beta, \gamma, \delta \rangle$ in Steps P,V4–7 form an accepting transcript in a universal argument system for the statement $\langle h, z, r \rangle \in \Lambda$. The language Λ is defined as follows: $\langle h, z, r \rangle \in \Lambda$ iff there exists a Turing machine M and strings s and y such that $|y| \leq \frac{|r|}{2}$, $z = \mathsf{Com}(h(M); s)$ and $M(z, y)$ outputs r within $n^{\log n}$ steps.</td><td>w x, r
$rWI\text{-}proof$
$x \in L$
or
$\langle \alpha, \beta, \gamma, \delta \rangle$
proves that
$\langle z, h, r \rangle \in \Lambda$
0/1</td></tr>
</table>

Protocol 7.2. bounded-rZK public-coin argument

- The commitments used in this protocol are perfectly (or statistically) binding and computationally hiding. For simplicity we assume that these commitments are *noninteractive*; such commitments can be constructed based on one-to-one one-way functions [Blu1] or based on one-way functions and Assumption 3.2 [BOV]. However, our protocol will be secure if the noninteractive commitments are replaced with the two-round scheme of Naor [Nao1], which can be based on any one-way function. Also, for simplicity, we will assume that the WI proof used in Step P8 consists of a single message. Such a system can be constructed based on the existence of trapdoor permutations and Assumption 3.2 [BOV]. However, our construction will remain secure even if we use the two-round WI proof of [DN], which is based only on trapdoor permutations.[20]

- The subexponentially hard hash functions imply subexponentially hard one-way functions, and hence a subexponentially hard commitment scheme via [Nao1]. We set the security parameter of the commitment scheme in such a way that for all the commitments used in the protocol, recovering the plaintext by brute force can be done in time $2^{o(n^\epsilon)}$. (As in the proof of Theorem 6.13, this is used in proving soundness via the "complexity leveraging technique" of [CGGM].)

- To obtain the bounded-resettable zero-knowledge condition, the prover should choose a pseudorandom function, and in each step obtain the random coins needed by applying the pseudorandom function on the message history of the protocol.

7.2 Analyzing Protocol 7.2

In this section we prove that Protocol 7.2 is a bounded-rZK argument. As in [Bar], completeness follows immediately from the definition of the protocol. The soundness condition also follows as in [Bar], with one variant. There, the WI system used in Step P8 is a proof of knowledge, and the knowledge extractor for this proof is used to obtain from a cheating prover an accepting transcript for the universal arguments. Later, a collision for the hash function is obtained from this transcript. Here we use only a proof of membership in Step P8. However, we can still recover an accepting transcript in time $2^{o(n^\epsilon)}$, by using brute force on the commitment scheme. This implies that we can treat the WI system as a proof of knowledge (albeit with a $2^{o(n^\epsilon)}$-time extractor) and hence soundness is obtained by the analysis of [Bar] (since the hash function we use is collision-resistant even against $2^{o(n^\epsilon)}$-time adversaries).

Bounded rZK. We now show why Protocol 7.2 remains zero knowledge against a verifier that is allowed to reset the prover at most n^c times. Let V^* be such a resetting verifier. Our simulator will compute the V^*'s view by executing the verifier, giving it access to a modified version of the honest prover strategy. Specifically, it will deviate from this strategy in the following way:

- In Step P2 it will compute $z = \mathsf{Com}(h(\mathsf{desc}(V^*)))$, where $\mathsf{desc}(V^*)$ denotes the description of the verifier's strategy, with its auxiliary input and coin tosses hardwired in.

- In the *first* time that the simulator receives a string r as verifier message of Step V3 of some session, the simulator will record the history $\gamma_r = (p_1, \ldots, p_k)$ of all messages the verifier

[20] We can replace the noninteractive commitments and WI proofs with the schemes of [Nao1] and [DN] by simply having the receiver/verifier send the first messages of these schemes (which are independent of the input) as the first message of Protocol 7.2.

has received so far across all executions (where $k \leq O(n^c)$, since the protocol has a constant number of rounds). Note that since each prover message in the protocol is of size at most $O(n)$, $|\gamma_r| = O(n^{c+1}) = o(n^{2+c})$ (and hence we can assume $|\gamma_r| \leq \frac{n^{2+c}}{2}$). (Note that even if the simulator receives r again as a verifier message of a later session, it still keeps the original recorded sequence γ_r.)

- When computing the messages of Steps P,V4–7, the simulator will run the honest prover of the universal argument scheme to prove the (true) statement that $\langle h, z, r \rangle \in \Lambda$. It uses the recorded sequence of messages γ_r as auxiliary input to the universal-argument prover algorithm.

We note that the simulator will also use a random function for randomness in each step, that is applied to the message history (of the current session) so far. The simulator can use a *truly random* function, by choosing the function incrementally, tossing fresh coins whenever it needs to compute the function on a new input.

Proving that the simulator's output is indistinguishable from the verifier's view. The proof that the simulator's output is indistinguishable from the verifier's view uses the hybrid argument. We consider the following sequence of hybrids, and claim that each one of them is computationally indistinguishable from the preceding one.

Hybrid H0 Hybrid H0 denotes the verifier's view when mounting a bounded reset attack on the honest prover.

Hybrid H1 This hybrid denotes the verifier's view when the honest prover uses a *truly* random function. Indistinguishability from H0 follows from the security of the pseudorandom function ensemble.

Hybrid H2 We modify H1 by having the prover use a commitment to $h(V^*)$ instead of to 0^n in Step P2 of all sessions. Indistinguishability from $H1$ follows from the security of the commitment scheme. Note that a commitment of Step P2 of some session in Hybrid H1 is either *identical* to a commitment of some previous session (if the history up to that point is identical in both session) or uses completely independent coins from the ones used in all previous sessions. For commitments that use completely independent coins, security follows in a straightforward way from the hiding property of the commitment scheme. However, we need to show that if a commitment in Step P2 in some session i is identical to a commitment in Step P2 for some previous session j in Hybrid H1, then these commitments would remain identical also in Hybrid H2 (where we commit to $h(V^*)$ instead of 0^{n^δ}). However, this will indeed be the case since the history includes the hash function h that is used in the commitment (and the verifier's code V^* is identical across all sessions).

Hybrid H3 We modify H2 by having the prover use commitments to messages computed by the honest universal argument prover in Steps P,V4–7. Indistinguishability from H2 follows from the security of the commitment scheme. Once again we need to verify that identical commitments in Hybrid H2 remain identical in Hybrid H3. However, commitments in Hybrid H2 can be identical only if the history of messages up to the point where the commitment was made was identical. However, in this case the simulator will use exactly the same prover

strategy to compute its messages in the two sessions[21] and hence the commitments to the prover messages will remain identical.

Hybrid H4 We modify H3 by having the prover use the second case in the WI proof of Step P8. Indistinguishability follows from the WI property. Note that in this case, because we assume the rWI proof consists only of a single message (as in the system of [BOV]), which is the last message of the session, we can assume that every two different sessions differ in the history up to this point (as otherwise they are completely identical and hence redundant). We note that H4 is identical to the simulator's output.[22]

8 Zero Knowledge from a Circuit SAT Algorithm

In this section we construct a constant-round public-coin zero-knowledge strong proof of knowledge for **NP**, under the assumption that there exist one-way functions[23] and that there is a nontrivial algorithm for the Circuit Satisfiability problem (CSAT). By nontrivial we mean an algorithm for CSAT that on input a circuit $C : \{0,1\}^k \to \{0,1\}$, runs in time $2^{o(k)} \cdot \text{poly}(|C|)$. Again, this assumption is a strengthening of the *negation* of Assumption 5.8 and seems likely to be false. However, we note that it seems compatible with the assumption that factoring is hard for sub-exponential sized circuits, and thus does not directly contradict most known cryptographic construction. Thus, this result shows that assuming some sort of *exponential* (and not just sub-exponential) lower bound is necessary to rule out zero-knowledge strong proofs of knowledge (as we indeed do). Also, the protocol that we construct is in fact the parallel version of Blum's protocol for Hamiltonicity. The question of whether or not this protocol is zero-knowledge or not has been a long-standing open question, and this result shows that to resolve it negatively will require making some strong assumptions. See Section 9 for more discussion on this and related questions.

In the course of the proof, we prove that if such a non-trivial CSAT algorithm exists, then it is possible to generate in polynomial time an (almost) uniformly distributed witness for every **NP** problem that has at least a $\mu(n)$-fraction of witnesses, where $\mu(\cdot)$ is some fixed negligible function (i.e., $\mu(n) = 2^{-\omega(\log n)}$). This result may be of independent interest.

Notations:

Circuit satisfiability problem. We use the standard definitions for boolean circuits. We will identify a boolean circuit C with its representation as a string in $\{0,1\}^*$. We define $|C|$ to be the length of this representation. The language CSAT consists of all the circuits C that have a *satisfying assignment* (i.e., an input x, such that $C(x) = 1$). The trivial exhaustive-search algorithm decides if C is in CSAT in time $2^k \cdot \text{poly}(|C|)$, where k is the input length of C. We say that CSAT has a *nontrivial algorithm* if there's a (possibly probabilistic) algorithm that decides CSAT in time $T(k) \cdot \text{poly}(|C|)$, where $T(k) = 2^{o(k)}$ and $T(k)$ is computable in time $2^{o(k)}$. By the usual reduction from search to decision, this implies that it is possible to *find* a satisfying assignment to C in time $T(k) \cdot \text{poly}(|C|)$. We note that the existence of such

[21] This is the reason we insisted that the simulator uses the first sequence $\gamma_r = (p_1, \ldots, p_k)$ of prover messages that yields r, so that this sequence, which is used as auxiliary input to the prover strategy, will be identical in the two sessions.

[22] The same reasoning works also if the rWI system consists of two messages, as in the case of the ZAP system of [DN].

[23] Actually, we will need to use a "nice" one-way function; i.e., a one-way function that cannot be inverted in time $h(\cdot)$, where $h(n)$ is a super-polynomial function that is computable in $\text{poly}(n)$ time (e.g., $h(n) = n^{\log \log n}$).

an algorithm is inconsistent with Assumption 5.8 (used in our lower bound for strong proofs of knowledge): for any poly(n)-time computable function $f : \{0,1\}^n \to \{0,1\}^*$ and string y, we can find a preimage of y under f in time $2^{o(n)}$ by finding a satisfying assignment of the circuit C_y defined by $C_y(x) = 1$ iff $f(x) = y$.

An equivalent formulation to the hypothesis that CSAT has a nontrivial algorithm is that there is a PPT A such that $A(1^n, C)$ outputs a satisfying assignment for C (in time poly$(n, |C|)$) provided that C has at most $k(n)$ variables. Another formulation is that it is possible to simulate Turing machines with $k(n)$-*bounded nondeterminism* (i.e., polynomial-time Turing machines that can have access to at most $k(n)$ nondeterministic bits) in probabilistic polynomial time.

Nice one-way functions. A function $f : \{0,1\}^* \to \{0,1\}^*$ is *one-way* if for nonuniform polynomial-time algorithm C (i.e. family of polynomial-sized circuits), there is a negligible function μ such that

$$\Pr_{y \leftarrow_R f(U_n)}[C(y) \in f^{-1}(y)] = \mu(n)$$

where $\mu : \mathbb{N} \to [0,1]$ is some negligible function. An equivalent condition (cf., [Bel]) is that there exists some function $h : \mathbb{N} \to \mathbb{N}$ such that $h(\cdot)$ is super-polynomial (i.e., $h(n) = n^{\omega(1)}$) and such that for every circuit C of size at most $h(n)$, it holds that

$$\Pr_{y \leftarrow_R f(U_n)}[C(f(y)) \in f^{-1}(y)] < \tfrac{1}{h(n)}$$

We say that $f(\cdot)$ is a *nice one-way function* if this condition holds for some super-polynomial function $h(\cdot)$ such that $h(n)$ is computable in poly(n) time (e.g., $h(n) = n^{\log \log n}$).

The main theorem of this section is Theorem 8.1, whose proof is given in the next subsection (Section 8.1).

Theorem 8.1. *Suppose that* Circuit Satisfiability *has a nontrivial algorithm and that there exists a nice one-to-one one-way function Then, for every* **NP** *relation R, there exists a zero-knowledge proof for L_R with the following properties:*

1. *It has 3 rounds.*
2. *It is a (statistically sound) proof with negligible soundness error.*
3. *It is a public-coin protocol.*
4. *It is a strong proof of knowledge for R (with a non-black-box knowledge extractor).*
5. *It is auxiliary-input zero knowledge (with a non-black-box simulator).*

Before describing the protocol, we first state the main implications of our assumptions that will be used the proof.

Lemma 8.2 (noninteractive commitments [Blu1]). *If there exists a nice one-to-one one-way function, then there exists a nice noninteractive commitment scheme* Com*. That is, there is a* poly(n)*-time computable function $h(n) = n^{\omega(1)}$ such that for every n, circuit C of size at most $h(n)$, and every two messages $m_0, m_1 \in \{0,1\}^n$,*[24]

$$|\Pr[C(\mathsf{Com}(m_0)) = 1] - \Pr[C(\mathsf{Com}(m_1)) = 1]| \leq \frac{1}{h(n)}.$$

We call $h(n)$ the security *of the commitment scheme.*

[24] To commit to shorter messages, simply pad them to length n.

The "one-to-one" constraint on the one-way function is only needed to ensure that the commitment scheme is noninteractive, which is needed for our final zero-knowledge proof system to have 3 rounds. We can also obtain a 3-round proof system using the noninteractive commitment scheme of Barak et al. [BOV], which is based on incomparable assumptions. Assuming just the existence of a (nice) one-way function, it is possible to obtain a 4-round proof system with the same properties, using the commitment scheme of Naor [Nao1].

The other implication of our assumptions is given the following lemma, which may be of independent interest. Loosely speaking, this lemma says that if we can solve CSAT for k-variable circuits, then we can sample a random satisfying assignment for any circuit whose fraction of satisfying assignments is larger than 2^{-k}.

Lemma 8.3 (witness generation from a CSAT algorithm). *Suppose there exists a nontrivial algorithm for* CSAT. *Then there exists a* $\mathrm{poly}(n)$*-time computable function* $k(n) = \omega(\log n)$ *and a PPT algorithm* A *such that the following holds. For every* n *and every circuit* C *that accepts more than a* $2^{-k(n)}$ *fraction of inputs, the random variable* $A(1^n, C)$ *has statistical difference at most* $2^{-\Omega(n)}$ *from the uniform distribution on* C*'s satisfying assignments.*

Lemma 8.3 is proven in Section 8.1.1.

8.1 The protocol and its analysis

The idea behind the proof is to use the k-times parallel composition of Blum's zero-knowledge protocol for proving HAMILTONICITY [Blu2], for $k = \omega(\log n)$. The resulting protocol is well known to be a 3-round public-coin proof system for HAMILTONICITY, with negligible soundness error. However, this protocol is *not* known to be zero knowledge, or to be a strong proof of knowledge. Nonetheless, we show that under the assumptions of Theorem 8.1, it is in fact a zero-knowledge strong proof of knowledge.

We prove Theorem 8.1 by fully describing the zero-knowledge protocol that we use and analyzing its properties. First, we note that the assumed existence of a nice one-way function

Given input length n, we choose $k = k(n)$ such that:

- It is possible to find satisfying assignments to circuits C that have at most $2k$ inputs in time $\mathrm{poly}(n, |C|)$.
- It is possible to almost-uniformly generate satisfying assignments for circuits C that accept at least a 2^{-2k} fraction of inputs in time $\mathrm{poly}(n, |C|)$.
- There exists a nice noninteractive commitment scheme (for messages of length up to n) with security 2^{2k}.

By our assumptions and Lemmas 8.2 and 8.3, it is possible to choose such a $k(n)$ so that $k(n)$ can be computed in $\mathrm{poly}(n)$-time and $k(n) = \omega(\log n)$.

Blum's HAMILTONICITY protocol. We let HAM denote the **NP**-complete language of all Hamiltonian graphs (i.e., n vertex graphs that contain the n-cycle as a subgraph). Blum's basic protocol for proving membership in HAM is Protocol 8.4. It is a 3-round public-coin proof for HAM with soundness error equal to $\frac{1}{2}$.

Public input: $x = (x_{i,j})$, the adjacency matrix of an undirected graph on n vertices. **Prover's auxiliary input:** w, a hamiltonian cycle in the graph x	$\underset{\downarrow}{w}\quad\underset{\downarrow}{x}$ $\boxed{P}\qquad\boxed{V}$
Step P1 (Commitment to permuted graph): Prover selects a random permutation φ on the vertices, and computes a commitment to the permuted graph. That is, prover sends the matrix of commitments $C = (C_{i,j})$ where $C_{i,j} = \mathsf{Com}(x_{\varphi(i),\varphi(j)})$. **Step V2 (Send random query):** The verifier selects a random bit $b \leftarrow_{\mathrm{R}} \{0,1\}$ and sends it. **Step P3 (Open commitments):** If $b = 0$ then the prover sends decommitments for all the commitments sent in Step P1 and in addition sends the permutation φ chosen in that step. Otherwise (if $b = 1$) the prover sends decommitments only for the commitments that correspond to the edges of $\varphi(w)$ (i.e., the edges of the permuted Hamiltonian cycle).	$C = \left(\mathsf{Com}(x_{\varphi(i),\varphi(j)})\right)_{i,j\in[n]} \longrightarrow$ $b \leftarrow_{\mathrm{R}} \{0,1\} \longleftarrow$ $\textit{decommitments} \longrightarrow$
If $b = 0$ then the verifier accepts iff all decommitments are valid and form a graph x' such that $x' = \varphi(x)$ (i.e., $x'_{i,j} = x_{\varphi(i),\varphi(j)}$ for all $i, j \in [n]$). If $b = 1$ then the verifier accepts iff the decommitments sent are for a Hamiltonian cycle and all the opened commitments are to the value 1.	

Protocol 8.4. Blum's basic protocol [Blu2]

Our protocol. Our protocol is the $k = k(n)$-times parallel composition of Protocol 8.4. That is, to prove that $x \in \text{HAM}$, the verifier and prover run k independent copies of Protocol 8.4 in parallel, where in each copy x is the common input. The verifier accepts iff all the copies are accepting. The combined protocol is clearly a 3-round public-coin proof system for HAM with soundness error $2^{-k(n)}$ (which is negligible since $k = \omega(\log n)$). Thus, all that remains is to prove that it is zero knowledge and a strong proof of knowledge. This is proven in Claims 8.5 and 8.8.

Claim 8.5. *Under the assumptions of Theorem 8.1, the $k(n)$-times parallel composition of Protocol 8.4 is zero knowledge.*

Proof. We prove the claim in two stages. First, we show a "slow" simulator (i.e., a simulator than runs in roughly 2^k steps) for this protocol whose output is computationally indistinguishable from the verifier's view; this simulator will use the fact that the commitment scheme is secure against 2^{5k}-sized circuits. We will then show that using the almost-uniform generation (via Lemma 8.3), it is possible to "speed up" this simulator and obtain a new simulator that runs in polynomial time, and whose output is statistically indistinguishable from the output of the original simulator.

The "slow" simulator. Our 2^k-time simulator is the natural extension of the simulator of the basic protocol (Protocol 8.4) to the parallel composed protocol. Loosely speaking, the simulator works by choosing $b^1, \ldots, b^k \leftarrow_{\text{R}} \{0,1\}$, and simulating the protocol under the assumption that the verifier's query in the l^{th} copy will be b^l. If this guess was correct, the simulator manages to simulate the entire transcript. Otherwise, the simulator fails (or tries again). We now give a more precise description of the simulator's operation:

Algorithm 8.6 (A "slow" simulator).

Input:

- x : a graph over n vertices.
- Description V^* of verifier's strategy, with its auxiliary input and coin tosses hardwired in. (Actually, this "slow" simulator only requires black-box access to V^*).

1. Choose $b^1, \ldots, b^k \leftarrow_{\text{R}} \{0,1\}$
2. For every $1 \leq l \leq k$ do the following:
   - If $b^l = 0$ then choose φ^l to be a random permutation over $[n]$ and let $C^l = \mathsf{Com}(\varphi^l(x))$ (i.e., C^l is a matrix of commitments such that $C^l_{i,j} = \mathsf{Com}(x_{\varphi(i),\varphi(j)})$).
   - If $b^l = 1$ then choose c^l to be a random cycle on n vertices and compute C^l to be a commitment to the cycle c^l (i.e., $C^l_{i,j} = \mathsf{Com}(1)$ iff (i,j) is an edge in c^l).
3. Feed $C^1, \ldots, C^k$ to the verifier V^* and obtain its response $\beta^1, \ldots, \beta^k$.
4. If $\beta^l \neq b^l$ for some $1 \leq l \leq k$, halt and output `fail`. (In this case, we say that the simulator has *failed*, and otherwise the simulator has *succeeded*.)
5. For every $1 \leq l \leq k$ compute D^l as follows:
   - If $b^l = \beta^l = 0$ then D^l is a string that contains φ^l and all the decommitments of the matrix C^l.

- If $b^l = \beta^l = 1$ then let D^l be a string that contain only the decommitments of C^l that correspond to edges in the cycle c^l. (That is, decommit to all the 1's in C^l.)

6. The output of the simulator is the view $(\langle C^1, \ldots, C^k \rangle, \langle D^1, \ldots, D^k \rangle)$.

The properties of this simulator are given in the following claim:

Claim 8.6.1. *For every polynomial-sized verifier V^* and $x \in$ HAM, on input (x, V^*) Algorithm 8.6 succeeds with probability at least 2^{-2k}. Furthermore, conditioned on success, the output of Algorithm 8.6 is computationally indistinguishable from the view of V^* in an interaction with the honest prover on input x.*

Thus, we can obtain a simulator that succeeds with high probability by running Algorithm 8.6 up to $n \cdot 2^{2k}$ and taking its first successful output. But, since $k = \omega(\log n)$, this takes superpolynomial time. (This is why we call it a "slow" simulator.) However, we will now show how we can use it to construct a *polynomial-time* simulator (and give the proof of Claim 8.6.1 afterwards.) Our polynomial-time simulator is the following algorithm:

Algorithm 8.7 (A "fast" simulator).

Input:

- x : a graph over n vertices.
- Description V^* of verifier's strategy.

1. Using the procedure of Lemma 8.3, generate an almost-uniformly distributed random tape r such that Algorithm 8.6 on input (x, V^*) and random tape r is successful.
2. Output the output of Algorithm 8.6 on input (x, V^*) and random tape r.

By Claim 8.6.1, a fraction of at least 2^{-2k} of the random tapes for Algorithm 8.6 result in a successful iteration. Thus the procedure of Lemma 8.3 will indeed work, and the output of Algorithm 8.7 will be statistically indistinguishable from the output of Algorithm 8.6. By Claim 8.6.1, this means that this output is computationally indistinguishable from the verifier's view.

We see that all that is left to do is to prove Claim 8.6.1

Proof of Claim 8.6.1. Let $x \in$ HAM and let w be a hamiltonian cycle in the graph x. We let S_{hyb} denote the following "hybrid simulator": S_{hyb} gets as auxiliary input x, V^* and the witness w, and follows the strategy of Algorithm 8.6 on input V^*, x with one modification: In Step 2, it computes C^l for each $1 \leq l \leq k$, to be a commitment of a random permutation of the input graph x, regardless of the value of b^l. However, if $b^l = 1$ then let c^l denote the image of the cycle w under the permutation. Note that w is a random n-cycle.

We now make several claims involving this hybrid simulator.

- The output of S_{hyb}, conditioned on success, is *identical* to the view of the verifier in a real execution. This can be verified by inspection.
- S_{hyb} succeeds with probability exactly 2^{-k}. This is because the hybrid simulator generates the message $\langle C^1, \ldots, C^k \rangle$ independently of the choice of $b^1, \ldots, b^k$, and so this message contains no information on this choice.

- The output of S_{hyb} cannot be distinguished (by a poly-sized distinguisher) from the output of S with advantage greater than $\text{poly}(n) \cdot 2^{-2k}$. (Here we do not condition on success.) This is because the commitment scheme has security 2^{2k}. Specifically, note that the only difference between the output of the "hybrid simulator" and the output of the real simulator is that in the real simulator's output, some of the C^l's contain commitments to a random n-cycle (instead of to a random permutation of the input x). The commitments which differ (namely those for edges outside of $\varphi^l(w)$) are never opened. Thus any distinguisher between the output of these simulators can be converted into a distinguisher for the commitment scheme.

- The original simulator S has a success probability of $2^{-k} \pm 2^{-2k}$. This follows from the previous two items.

- The output of S_{hyb} conditioned on success, is computationally indistinguishable from the output of S conditioned on success. This follows from the previous three items. When we condition on success, the advantage of any distinguisher can increase by a factor of at most $1/(2^{-k} - 2^{-2k})$. Therefore, any poly-sized distinguisher has advantage at most $\text{poly}(n) \cdot 2^{-2k}/(2^{-k} - 2^{-2k}) < 2 \cdot \text{poly}(n) \cdot 2{-}k$, which is negligible.

Combining the first item and the last item, we conclude that the output of S conditioned on success is computationally indistinguishable from the V^*'s view of the interaction, completing the proof of the claim. □

□

This concludes the proof of zero-knowledge; we now proceed to show that Protocol 8.4 is also a strong argument of knowledge.

Claim 8.8. *Under the assumptions of Theorem 8.1, the $k(n)$-times parallel composition of Protocol 8.4 is a strong argument of knowledge, with soundness error $s(n) = 2^{-k(n)}$.*

Proof. Let P^* be any polynomial-time prover strategy, with its coin tosses and auxiliary input hardwired in. Suppose that for some x, P^* manages to cause the honest verifier in all k iterations of Protocol 8.4 to accept with probability larger than 2^{-k}. Because the prover P^* is deterministic, so is its first message $\langle C^1, \ldots, C^k \rangle$. This means that there exist at least two different choices of verifier messages $\beta \neq \beta' \in \{0,1\}^k$, such that if P^* is given either β and β', then its response is valid. Finding β and β' can be formulated as a CSAT problem for a $\text{poly}(n)$-sized circuit with $2k$ variables, and thus can be solved under our assumptions in polynomial time. Once we find such β and β', we let l be a coordinate in which they differ. From the prover's response in the l^{th} instance of the protocol we can obtain a Hamiltonian cycle in the graph x (because we obtain both the permutation and the permuted cycle). □

8.1.1 Proof of Lemma 8.3

Our starting point is the fact that it is possible to almost-uniformly generate satisfying assignments to a circuit in probabilistic polynomial time with an **NP** oracle [JVV, BGP]. The key observation is that if the circuit accepts at least a 2^{-k} fraction of inputs, then all the oracle queries can be made instances of CSAT on $O(k)$ variables.

Specifically, we start with the algorithm by Bellare, Goldreich and Petrank [BGP] (see also [Gol3, Lec. 6]), which actually achieves perfect uniform generation. We modify the algorithm and

implement some steps in a different way, in order to ensure that all the queries are on circuits of a small number of variables.

Given a CSAT algorithm $A(1^n, C)$ that finds satisfying assignments when C has at most $k = k(n)$ variables, we will construct a sampler $B(1^n, C)$ that generates almost-uniform satisfying assignments when C has at most $k/2$ variables.

Let n and C be given, and let m be the number of variables in C. By adding dummy variables, we may assume that $m \geq \max\{k, n\}$. We denote by S_C the set $C^{-1}(1)$ (i.e., the set of satisfying assignments for C). We assume that $\Pr[U_m \in S_C] \geq 2^{-k/2}$, or in other words that $|S_C| \geq 2^{m-k/2}$.

t-wise independent hash functions. Recall that a t-wise independent hash function collection is a collection $\mathcal{H}$ of functions from $\{0,1\}^m$ to $\{0,1\}^i$ such that for every t distinct values $x_1, \ldots, x_t \in \{0,1\}^m$, the random variables $h(x_1), \ldots, h(x_t)$ (for a random choice of $h \in \mathcal{H}$) are independently and uniformly distributed over $\{0,1\}^i$. The standard construction of such a family $\mathcal{H}$ consists of all polynomials of degree at most $t-1$ over the field $\mathrm{GF}(2^m)$, truncated to the first i bits. We will use not only the fact that the functions in this family can be evaluated efficiently (as in [BGP]), but also that it can be *inverted* efficiently. Specifically, given a polynomial p of degree at most $t-1$ over $\mathrm{GF}(2^m)$ and a point $\gamma \in \mathrm{GF}(2^m)$, all the elements of $p^{-1}(\beta)$ can be found in time $\mathrm{poly}(m,t)$ [Ber].

We now give an outline of the [BGP] algorithm:

Algorithm 8.9 (Uniform generation with an NP oracle [BGP]).

1. Test if $|S_C| \leq 10m^5$. If so, uniformly generate an element out of S_C.
2. Find $i \in \{0, \ldots, m\}$ such that $|S_C| \in (2^i m^5, 2^{i+2} m^5)$.
3. Choose $h : \{0,1\}^m \to \{0,1\}^i$ at random from an m-wise independent hash function collection.
4. If there exists $\alpha \in \{0,1\}^i$ such that $|h^{-1}(\alpha) \cap S_C| > 6m^5$ then abort.
5. Choose $\alpha \leftarrow_{\mathrm{R}} \{0,1\}^i$.
6. Find all the (at most $6n^5$) elements of $h^{-1}(\alpha) \cap S_C$. Output each such element with probability $\frac{1}{6m^5}$. Otherwise abort (i.e.,abort with probability $1 - \frac{|h^{-1}(\alpha) \cap S_C|}{6m^5}$).

Bellare *et al.* [BGP] show how to implement Algorithm 8.9 using an **NP**-oracle and show that this algorithm does not abort with high probability. Furthermore, they show that conditioned on the algorithm not aborting, the algorithm's output is exactly the uniform distribution over S_C. Note that this algorithm works for every circuit C (and not just circuits with $|S_C| \geq 2^{m-k/2}$). We will show that a variant of this algorithm can be implemented for circuits C such that $|S_C| \geq 2^{m-k/2}$ using only an oracle for CSAT for k-variable $\mathrm{poly}(m)$-sized circuits. We now show how we implement (and sometimes modify) each step of Algorithm 8.9:

Step 1 Note that we assume that $k \leq m$. Thus $|S_C| \geq 2^{m/2}$ and we can skip this step.

Step 2 We will try to run the algorithm with all possible choices for i. We start with $i = m - k/2 - 6\log m$ (as $|S_C| \geq 2^{m-k/2} > 2^{m-k/2-6\log n} \cdot n^5$) and try to run the algorithm described below (i.e., in Steps 3 to 6). We will see below (in Step 6), that if the chosen value i fails some check then we let $i \leftarrow i+1$ and try again. Note that since $k = \omega(\log m)$ we always have that $i \geq m - 2k/3$.

Step 3 This step can be done without using any oracle. We will choose h to be a random polynomial of degree $m-1$ over the field $\mathbb{F}_{2^m}$, where we truncate the output of this polynomial to its first i bits.

Step 4 We skip this step and do not check h for this condition.

Step 5 This step can be done without using any oracle.

Step 6 We first show that we can find a single element of the set $S \stackrel{def}{=} h^{-1}(\alpha) \cap S_C$ using an oracle for CSAT for k-variable poly(n)-sized circuits. We then show how to find all of S's elements.

Finding a single element of S Recall that we used for the hash function h the i-bit truncation of a degree $n-1$ polynomial. Let p denote the untruncated polynomial. Then, each element x of $h^{-1}(\alpha)$ satisfies that $p(x) = \alpha\beta$ for some $\beta \in \{0,1\}^{m-i}$. As mentioned earlier, given $\alpha\beta$, it is possible to find all the (at most $m-1$) elements in $p^{-1}(\alpha\beta)$. Given h and α, we construct a circuit C' that on input β computes all elements of $p^{-1}(\alpha\beta)$ and outputs 1 if one of these elements is a satisfying assignment for C. Note that since $i \geq m - 2k/3$, the number of inputs of C' is at most $2k/3 < k$. Thus, we can find a satisfying assignment β for C' using our presumed CSAT algorithm. Given such β we can check all elements of $p^{-1}(\alpha\beta)$ and at least one of these will be a element of $h^{-1}(\alpha) \cap S_C$.

Finding all the elements of S Once we found one element x of S we can apply the same process to the set $h^{-1}(\alpha) \cap (S_C \setminus \{x\})$ and find another element x'. Note that the set $S_C \setminus \{x\}$ can also be decided by a polynomial-sized circuit. We continue this process for at most $6m^5$ steps. If after $6m^5$ steps we are still not finished then we abort.

Additional check We perform the above procedure for the set $S_C \cap h^{-1}(0^i)$ and verify that it is of size at most $4m^5$. Otherwise, we let $i \leftarrow i+1$ and go back to Step 3.

Analysis. The analysis of [BGP] shows that if $|S_C| > 2^i m^5$ then with $1 - 2^{-\Omega(m)}$ probability it will hold that $|h^{-1}(\alpha) \cap S_C| \in (1 \pm \frac{1}{5})\frac{|S_C|}{2^i}$ for *all* $\alpha \in \{0,1\}^i$. Note that as i gets larger, the quantity $\frac{|S_C|}{2^i}$ gets smaller, and for the first value of i we try in the algorithm (i.e., $i = m - k/2 - 6\log m$), this quantity is at least m^6 (since $|S_C| \geq 2^{m-k/2}$). From this it can be shown that very high probability, the first i for which we pass the check $|h^{-1}(0^i) \cap S_C| \leq 4m^5$ will also satisfy $m^5 \leq |h^{-1}(\alpha) \cap S_C| \leq 6m^5$ for all α. [25] Thus, we will assume that this event always happens (this assumption can cause a statistical difference of at most $2^{-\Omega(m)}$ in the algorithm's output). This means that our algorithm outputs an element with probability at least $\frac{1}{6}$ (since it aborts with probability $1 - \frac{|h^{-1}(\alpha) \cap S_C|}{6m^5}$). If an element is output then it is distributed exactly according to the uniform distribution on S_C, since each $x \in S_C$ has the same probability (i.e. $\frac{1}{6m^5}$) of being output. Using repetition, we can ensure that we output an element with $1 - 2^{\Omega(m)}$ probability. Thus, we see that the output of the algorithm is of statistical distance at most $2^{-\Omega(m)}$ from the uniform distribution over S_C.

□

[25] Consider the first i for which we pass the check. Then we have $4m^5 \geq |h^{-1}(0^i) \cap S_C| \geq (4/5)|S_C|/2^i$, so for all α, $|h^{-1}(\alpha) \cap S_C| \leq (6/5)|S_C|/2^i \leq 6m^5$. Also, since we didn't pass the check at $i-1$, we have $4m^5 < |h^{-1}(0^{i-1}) \cap S_C| \leq (6/5)|S_C|/2^{i-1}$, which implies that at stage i we have $|h^{-1}(\alpha) \cap S_C| \geq (4/5)|S_C|/2^i > (20/12)m^5 > m^5$.

Part III
Conclusions and Open Problems

9 Constant-Round Public-Coin Zero-Knowledge Proofs

A long-standing open problem is whether there exist constant-round public-coin zero-knowledge proof systems. (Recall that [GK1] has constructed constant-round *private-coin* zero-knowledge proof systems, and [Bar] has constructed constant-round public-coin zero-knowledge *argument* systems.) It is not even known whether the 3-round proof systems obtained by parallel repetition of classic protocols, such as the ones for Quadratic Residuosity [GMR], Graph Isomorphism [GMW], Three-Coloring [GMW], and Hamiltonicity [Blu2], are zero knowledge.

Dwork, Naor, Reingold, and Stockmeyer [DNRS] have shown that the above open problem is closely related to the soundness of the Fiat-Shamir heuristic [FS2]. The latter is a popular heuristic for eliminating interaction in (public-coin) protocols, whereby the verifier's messages are replaced by a cryptographic hash of the conversation history. In particular, it is used to convert identification schemes (typically based on zero-knowledge proofs) into digital signature schemes, and many practical digital signature schemes have been obtained in this way (e.g. [Sch, GQ, Oka]). This technique can be proven sound in the random oracle model [PS], but it has no known proof of security in the standard model, where the hash function is implemented by an efficiently computable (and publicly known) algorithm.

The work of Dwork et al. [DNRS] shows that proving that a protocol is not zero-knowledge (at least in some weak sense) is essentially equivalent to exhibiting a family of efficiently computable hash functions ("magic functions") with which the Fiat-Shamir heuristic is sound when applied to that protocol. Thus by constructing constant-round public-coin zero knowledge argument, Barak [Bar] gave an example of a protocol under which the Fiat-Shamir heuristic is completely unsound (for any family of efficiently computable hash functions). Goldwasser and Taumann [GT] subsequently gave a 3-round computationally-sound identification scheme for which the Fiat-Shamir heuristic is completely unsound. These results still leave open the possibility that the Fiat-Shamir heuristic is sound for any *proof* system (for an appropriate family of "magic functions"). Not only would this be significant for the Fiat-Shamir heuristic itself (which is typically applied to proof systems), it would imply that no constant-round public-coin proof system (e.g. the parallel versions of [GMR, GMW, Blu2]) is zero knowledge.

In this section, we observe that there is a very clean and plausible property of a family of hash functions which implies the soundness of the Fiat-Shamir heuristic when applied to any *proof* system. We do not, however, know how to construct such hash functions based on more standard complexity assumptions, and leave this as an intriguing open problem.

We begin by formalizing the syntactic properties of a family of hash functions.

Definition 9.1. For functions $k(n), m(n) \leq \text{poly}(n)$, a *family of hash functions* with *input length* $m(n)$ and *output length* $k(n)$ is a collection $\mathcal{H} = \bigcup_n \{h_i : \{0,1\}^{m(n)} \to \{0,1\}^{k(n)}\}_{i \in \mathcal{I}_n}$, such that

- There is a probabilistic polynomial-time algorithm that given 1^n, outputs $i \leftarrow_{\text{R}} \mathcal{I}_n$. (This distribution need not be uniform.) We require $|i| \geq n$.
- There is a deterministic polynomial-time algorithm that given $i \in \mathcal{I}_n$ and $x \in \{0,1\}^{m(n)}$, outputs $h_i(x)$.

The security property we will formulate is based on the notion of conditional entropy. Recall that the *entropy* of a random variable X is defined as $\mathrm{H}(X) = \mathbb{E}_{x \leftarrow_{\mathrm{R}} X}[\log(1/\Pr[X = x])]$. For jointly distributed random variables (X, Y), the *conditional entropy* of Y given X is defined to be $\mathbb{E}_{y \leftarrow_{\mathrm{R}} Y}[\mathrm{H}(X|_{Y=y}]$, where $X|_{Y=y}$ denotes the conditional distribution of X given that $Y = y$.

Definition 9.2. Let $\mathcal{E}$ be a collection of functions $e : \mathbb{N} \to \mathbb{N}$. We say that a family of hash function $\mathcal{H} = \bigcup_n \{h_i : \{0,1\}^{m(n)} \to \{0,1\}^{k(n)}\}_{i \in \mathcal{I}_n}$ *ensures conditional entropy* (greater than) $\mathcal{E}$ if for every nonuniform probabilistic polynomial-time algorithm A, there exists an $e \in \mathcal{E}$ such that

$$\mathrm{H}(h_I(A(I))|A(I)) > e(n),$$

where the probabilities are taken over $I \leftarrow_{\mathrm{R}} \mathcal{I}_n$ and the coin tosses of A, and $A(I)$ denotes the first $m(n)$ bits of the output of A on input I.

Examples of entropy bounds we will consider below are $\mathcal{E} = \{0\}$, $\mathcal{E} = \{1/n^c : c \in \mathbb{N}\} \stackrel{def}{=} 1/\mathrm{poly}(n)$, and $\mathcal{E} = \{k(n) - c\log n : c \in \mathbb{N}\} \stackrel{def}{=} k(n) - O(\log n)$. (The reason for working with families instead of a single function $e(n)$ is to allow the constant c in the latter two cases to depend on the choice of the algorithm A.)

Why is it plausible that such function families exist? Note that the length of the function-index i can be a larger polynomial than $m(n)$. Thus, even conditioned on $A(I)$, I still has a lot of entropy, and hence we can hope that $h_I(A(I))$ contains some of this entropy. This would be trivial if $k(n)$ were larger than the length of the function-index, because we could just define $h_i(x) = i$ for all i. However, we are typically interested in the case that $k(n)$ is much smaller than $m(n)$, which must be smaller than the length of the function-index. (Note that if $m(n)$ is larger than the length of the function-index, then $A(i) = i$ makes the conditional entropy $H(h_I(A(I))|A(I))$ zero.) We observe that the largest conditional entropy we can hope for is $k(n) - O(\log n)$: if all the h_i's are regular functions (i.e. $h_i(U_{m(n)}) = U_{k(n)}$), then by evaluating $h_i(x)$ on $\mathrm{poly}(n)$ random inputs, a PPT A can with high probability find an input x such that the first $c\log n$ bits of $h_i(x)$ are zero, making $\mathrm{H}(h_I(A(I))|A(I))) \leq \mathrm{H}(h_I(A(I))) \leq n - c\log n + O(1)$, where c is any constant. Our (strongest) conjecture in this section is that there are hash functions for which this attack is essentially the best possible.

Conjecture 9.3. *For every two functions $m(n), k(n) \leq \mathrm{poly}(n)$, there exists a family of hash functions with input length $m(n)$ and output length $k(n)$ that ensures conditional entropy $k(n) - O(\log n)$.*

Before seeing the implications of this conjecture, we establish the following useful lemma.

Lemma 9.4. *If there exists a family of hash functions $\mathcal{H} = \bigcup_n \{h_i : \{0,1\}^{m(n)} \to \{0,1\}^{k(n)}\}_{i \in \mathcal{I}_n}$ that ensures conditional entropy $\mathcal{E}$, then there exists such a family where for every $\alpha \in \{0,1\}^{m(n)}$, $h_I(\alpha)$ is distributed uniformly in $\{0,1\}^{k(n)}$ when $I \leftarrow_R \mathcal{I}_n$.*

Proof. Define $h'_{(i,z)}(x) = h_i(x) \oplus z$. Then, since α is fixed before I is chosen, we immediately obtain that $h'_{(i,z)}(\alpha)$ is uniformly distributed. It remains to show that h' still ensures conditional entropy $\mathcal{E}$. Now, for every nonuniform PPT $A'(i,z)$, we can define a collection of nonuniform PPT

$A_z(i) = A(i,z)$ so that when I and Z are uniform, we have

$$\begin{aligned}
\mathrm{H}(h'_{I,Z}(A'(I,Z))|A'(I,Z)) &\geq \mathrm{H}(h'_{I,Z}(A'(I,Z))|A'(I,Z),Z) \\
&= \mathbb{E}_{z\leftarrow_{\mathrm{R}}Z}[\mathrm{H}(h'_{I,z}(A'(I,z))|A'(I,z))] \\
&= \mathbb{E}_{z\leftarrow_{\mathrm{R}}Z}[\mathrm{H}(h_I(A_z(I)) \oplus z|A_z(I))] \\
&= \mathbb{E}_{z\leftarrow_{\mathrm{R}}Z}[\mathrm{H}(h_I(A_z(I))|A_z(I))] \\
&\geq e(n)
\end{aligned}$$

□

Now we show that the above conjecture implies that there do not exist constant-round public-coin zero-knowledge proof systems.

Theorem 9.5. *Assuming Conjecture 9.3, if a language L has a constant-round public-coin auxiliary-input zero-knowledge proof system, then $L \in \mathbf{BPP}$.*

More generally, assuming Conjecture 9.3 only for input length $m(n)$ and output length $k(n)$, if L has a constant-round public-coin auxiliary-input zero-knowledge proof system in which the total communication is at most $m(n)$ bits and the verifier's messages are of length at most $k(n)$ on inputs of length n, then $L \in \mathbf{BPP}$.

Proof. We begin with the special case of 3-message protocols. Let (P,V) be a 3-round public-coin auxiliary-input zero-knowledge proof for a language L. We denote the three messages by α, β, γ, and consider a family of hash functions $\mathcal{H}$ which has input length $m = m(n) \geq |\alpha|$ and output length $k = k(n) \geq |\beta|$, and ensures entropy $k(n) - O(\log n)$. By Lemma 9.4, we may assume that $h_I(\alpha)$ is uniformly distributed for every α. Intuitively, if the verifier chooses its message β by applying h_I to the prover message α then we are guaranteed two properties: (1) the verifier's message is uniformly distributed and so is a valid message (recall that this is a public-coin proof system); (2) even given I, whatever message $\alpha = A(I)$ a cheating prover (or simulator) generates, the entropy of the verifier's reply $h_I(\alpha)$ is still high. In such a case, it is "hard" to simulate. We now formally prove the theorem.

Consider a cheating verifier $V^*(x,i,\alpha)$ that on input x, auxiliary input i, and prover message α, sends message $\beta = h_i(\alpha)$. By auxiliary-input zero knowledge, there is a PPT simulator S such that for every x and i, $S(x,i)$ is computationally indistinguishable from V^*'s view. We consider the output distribution of $S(x,I)$ where I and the coin tosses of S are chosen uniformly.

When $x \in L$, we claim that $S(x,I)$ outputs an accepting transcript (i.e. (α,β,γ) such that $V(x,\alpha,\beta,\gamma) = \texttt{accept}$) with probability at least $1 - \mathrm{neg}(n)$. The reason is that when I is chosen uniformly, then, for every α, $h_I(\alpha)$ is uniformly distributed, so $\langle P, V^*(I)\rangle(x)$ is distributed identically to $\langle P, V\rangle(x)$, which is accepting with probability $1 - \mathrm{neg}(n)$ by completeness.

When $x \notin L$, we claim that $S(x,I)$ outputs an accepting transcript with probability at most $1/2$. Consider a nonuniform algorithm $A_x(i)$ that runs $S(x,i)$ and outputs the first message α. Then, by the magic function property, $\mathrm{H}(h_I(A_x(I))|A_x(I)) \geq k - c\log n$ for some constant c. In other words, when $(A,B,C) \leftarrow_{\mathrm{R}} S(x,I)$, we have $\mathrm{H}(B|A) \geq k - c\log n$. This implies that with probability at least $3/4$ over $\alpha \leftarrow_{\mathrm{R}} A$, $\mathrm{H}(B|_{A=\alpha}) \geq k - 4c\log n$. For each α, let Acc_α be the set of $\beta \in \{0,1\}^k$ for which there exists γ such that (α,β,γ) is an accepting transcript. By soundness, for every $\alpha \in \{0,1\}^m$, $|\mathrm{Acc}_\alpha| \leq s(n) \cdot 2^k$, where $s(n)$ is the (negligible) soundness error, k is the length of the verifier message β. Thus, if $\mathrm{H}(B|_{A=\alpha})$ has entropy at least $k - O(\log n)$, then $\Pr[B|_{A=\alpha} \in \mathrm{Acc}_\alpha] \leq 1/4$.

(Otherwise $B|_{A=\alpha}$ has entropy at most $(1/4) \cdot \log_2 |\mathrm{Acc}_\alpha| + (3/4) \cdot k + 1 = k - \omega(\log n)$.)[26] To conclude, we can upper bound the probability that $S(x, I)$ outputs an accepting transcript by

$$\begin{aligned}
&\Pr[B \in \mathrm{Acc}_A] \\
&\quad \leq \Pr[B \in \mathrm{Acc}_\alpha | \mathrm{H}(B|_{A=\alpha}) \geq k - 4c \log n] + \Pr[\mathrm{H}(B|_{A=\alpha}) < k - 4c \log n] \\
&\quad \leq 1/4 + 1/4 = 1/2.
\end{aligned}$$

Thus, we can decide L in **BPP** by choosing I uniformly, running $S(x, I)$, and deciding according to whether S outputs an accepting transcript.

For general constant-round proof systems, we use a different hash function for each verifier message. Replace Acc_α with sets Acc_t for each partial transcript t ending in a prover message. If t ends before the i'th verifier message, define Acc_t to be the set of verifier messages for which the maximum conditional acceptance probability is at least $s^{1/2^i}$, where $s = s(n)$ is the soundness error. Refining the above analysis, it can be argued that if the conditional acceptance probability given t is smaller than $s^{1/2^{i-1}}$, then with probability $1 - \epsilon$ over the history generated by the simulator, the next verifier message (computed by the i'th magic function) lands in Acc_t with probability at most ϵ, where ϵ is an arbitrarily small constant. Choosing ϵ to be smaller than $1/(4 \cdot \#\text{rounds})$, we obtain an accepting transcript with probability at most $1/2$. □

Following [DNRS], the above proof actually gives a positive result for the Fiat-Shamir heuristic under Conjecture 9.3. Specifically, it shows that for any polynomial-time P^* and any $x \notin L$, the probability that $P^*(x, i)$ produces an accepting transcript $(\alpha, h_i(\alpha), \gamma)$ is at most $1/2$ when i is chosen randomly: simply replace the simulator S with P^* in the above proof. This success probability (of $1/2$) can probably be made negligible by requiring the prover generate accepting transcripts for polynomially many hash functions chosen randomly from the family (rather that just a single hash function). Also, in typical applications of the Fiat–Shamir heuristic, the prover is typically allowed to choose the instance x; in such a case one should include x in the input to the hash function as well.

The above theorem rules out all constant-round public-coin zero-knowledge proofs (under the conjecture), but many protocols of interest (from the perspective of both zero knowledge itself and the Fiat-Shamir heuristic) possess additional properties that may make ruling out zero knowledge easier (e.g. we may use a weaker conjecture). For example, many popular protocols have 3 rounds, perfect completeness, and soundness that is optimal in the following sense.[27]

Definition 9.6. A 3-round public-coin proof system (P, V) for a language L has *optimal soundness* if for every $x \notin L$ and every first prover message α, there is at most one verifier message β such that $V(x, \alpha, \beta, \gamma) = \texttt{accept}$ for some second prover message γ.

Examples of protocols with optimal soundness are the classic proof systems for Quadratic Residuosity [GMR], Graph Isomorphism [GMW], and Hamiltonicity [Blu2] and the parallel versions of these protocols. (The Three-Coloring protocol of [GMW] does not have optimal soundness.)

For protocols with this property, a weaker version of the conjecture suffices.

[26] Let $C = B|_{A=\alpha}$ and let I be the indicator for the event $C \in \mathrm{Acc}_\alpha$. Then $\mathrm{H}(C) \leq \mathrm{H}(I) + \mathrm{H}(C|I) \leq 1 + \mathrm{H}(C|I)$, where $\mathrm{H}(\cdot|\cdot)$ denotes conditional entropy. By definition, $\mathrm{H}(C|I) = \Pr[I = 1] \cdot \mathrm{H}(C|I = 1) + \Pr[I = 0] \cdot \mathrm{H}(C|I = 0) \leq \Pr[I = 1] \cdot (\log_2 |\mathrm{Acc}_\alpha|) + (1 - \Pr[I = 1]) \cdot k$. The above calculation then follows by noting that this last expression is decreasing in $\Pr[I = 1]$.

[27] Our definition of optimal soundness is weaker than the notion of *special soundness*, introduced in [CSD], which requires that one can actually recover the witness from accepting responses to any two different verifier messages.

Conjecture 9.7. *For every two functions $m(n), k(n) \leq \mathrm{poly}(n)$ such that $k(n) = \omega(n)$, there exists a family of hash functions with input length $m(n)$ and output length $k(n)$ that ensures conditional entropy $1/\mathrm{poly}(n)$.*

Theorem 9.8. *Assuming Conjecture 9.7, if a language L has a 3-round public-coin auxiliary-input zero-knowledge proof system with optimal soundness, then $L \in \mathbf{BPP}$.*

More generally, assuming Conjecture 9.7 only for input length $m(n)$ and output length $k(n)$, if L has a 3-round public-coin auxiliary-input zero-knowledge proof system with optimal soundness where the prover's first message is length at most $m(n)$ and the verifier's message is of length at most $k(n)$, then $L \in \mathbf{BPP}$.

Proof. The proof proceeds as above, with the only modifications being in the case $x \notin L$. Here, we observe that optimal soundness means that for every α, $|\mathrm{Acc}_\alpha| = 1$. We denote the unique element of Acc_α by $f(\alpha)$. We will use this and the magic function property to argue that S outputs a rejecting transcript with probability at least $1/\mathrm{poly}(n)$. Using the same notation as above, we have $\mathrm{H}(B|A) \geq 1/p(n)$ for some polynomial p. This implies that with probability at least $1/(2k \cdot p(n))$ over $\alpha \leftarrow_{\mathrm{R}} A$, $\mathrm{H}(B|_{A=\alpha}) \geq 1/2p(n)$. If this occurs, we have $\Pr[B|_{A=\alpha} \neq f(\alpha)] = \Omega(1/[p(n) \cdot (k + \log n)])$. Thus, the probability that $S(x, I)$ outputs an rejecting transcript is at least

$$\Pr[B \neq f(A)] \geq (1/2p(n)) \cdot \Omega(1/[p(n) \cdot (k + \log n)]) \geq 1/\mathrm{poly}(n),$$

as desired. □

The "ultimate" weakening of the conjecture is to require that the hash function only ensure conditional entropy greater than zero. It turns out that this has the following equivalent formulation:

Conjecture 9.9. *For every two functions $m(n), k(n) \leq \mathrm{poly}(n)$ such that $k(n) = \omega(n)$, there exists a polynomial p such that the following holds. For every nonuniform* deterministic *polynomial-time algorithm A and all sufficiently large n, there exist circuits $C_1, C_2 : \{0,1\}^{m(n)} \to \{0,1\}^{k(n)}$ of size at most $p(n)$ such that $A(C_1) = A(C_2)$ but $C_1(A(C_1)) \neq C_2(A(C_2))$.*

To see how this conjecture follows from the existence of hash functions with input length $m(n)$ and output length $k(n)$ that ensure conditional entropy greater than 0, note that the condition $\mathrm{H}(h_I(A(I))|A(I)) > 0$ is equivalent to the existence of i, j such that $A(i) = A(j)$ but $h_i(A(i)) \neq h_j(A(j))$, and that giving A the index i is equivalent to giving A the circuit that computes h_i. For the converse, consider $\mathcal{H}_n$ which is the uniform distribution on all circuits of size at most $p(n)$ mapping $\{0,1\}^{m(n)}$ to $\{0,1\}^{k(n)}$.

We find Conjecture 9.9 to be very plausible: When $p(n) \gg m(n)$, many different circuits C must map to the same input $z = A(C)$. When the range $\{0,1\}^{k(n)}$ is of superpolynomial size, it is difficult to imagine how a polynomial-time algorithm A can ensure that all such circuits have the same value on z. Moreover, we note that this conjecture only refers to worst-case complexity, and thus has hope of being related to other conjectures in complexity theory. Still, even this weak form of the conjecture implies a negative result for zero-knowledge proofs.

Theorem 9.10. *Assuming Conjecture 9.9, if a language L has a 3-round public-coin auxiliary-input zero-knowledge proof system with optimal soundness and perfect completeness, then the complement of L is in $\mathbf{AM}$.*

More generally, assuming Conjecture 9.9 only for input length $m(n)$ and output length $k(n)$, if L has a 3-round public-coin auxiliary-input zero-knowledge proof system with optimal soundness and perfect completeness in which the prover's first message is length at most $m(n)$ and the verifier's message is of length at most $k(n)$, then the complement of L is in $\mathbf{AM}$.

Public input: x	x $\downarrow$ $\boxed{M}$ $\quad$ $\boxed{A}$
Step A1: Choose $r \leftarrow_{\mathrm{R}} \{0,1\}^s$. **Step M1:** Find C of size at most $p(n)$ such that $S(x, C; r)$ is a rejecting transcript.	$\xleftarrow{r}$ $\xrightarrow{C}$
Accept if C is of size at most $p(n)$ and $S(x, C; r)$ is a rejecting transcript.	

Protocol 9.12. **AM** proof system for $\overline{L}$

Corollary 9.11. *If Conjecture 9.9 holds only for output length $k(n)$ and input length $m(n) = n^c$ for a sufficiently large constant c and* $\mathbf{NP} \not\subseteq \mathbf{coAM}$*, then the $k(n)$-fold parallel repetition of Blum's* HAMILTONICITY *protocol is not auxiliary-input zero knowledge.*

Proof. Let (P, V) be a 3-round public-coin auxiliary-input zero-knowledge proof system with optimal soundness and perfect completeness in which the prover's first message is length at most $m = m(n)$ and the verifier's message is of length at most $k = k(n)$. Let $p(n)$ be the polynomial guaranteed by Conjecture 9.9. Analogously to the above proofs, we consider the cheating verifier $V^*(x, C, \alpha)$ that on input x, auxiliary input C (interpreted as a circuit mapping $\{0,1\}^m \to \{0,1\}^k$), and first prover message α, sends message $\beta = C(\alpha)$. Let $S(x, C)$ be the simulator for V^*. We may assume wlog that $S(x, C)$ always outputs transcripts of the form $(\alpha, C(\alpha), \gamma)$. We call such a transcript *accepting* if it would make the honest verifier V accept.

By perfect completeness, for every $x \in L$ and every circuit C, $S(x, C)$ outputs an accepting transcript with probability $1 - \mathrm{neg}(n)$. By taking polynomially many trials, we can increase this probability to be at least $1 - 2^{-p(n)-n}$. Let $s = s(n)$ be the number of random bits used by the resulting simulator.

Protocol 9.12 contains the **AM** proof system for the complement of L. We now analyze this proof system, beginning with soundness. When $x \in L$, then for every fixed C, the simulator $S(x, C; r)$ outputs a rejecting transcript with probability at most $2^{-p(n)-n}$ when $r \leftarrow_{\mathrm{R}} \{0,1\}^s$. Thus, by a union bound, the probability over r that there exists a C of size at most $p(n)$ such that $S(x, C; r)$ is rejecting is at most $2^{p(n)} \cdot 2^{-p(n)-n} = 2^{-n}$.

For completeness, let $x \in L$, fix any $r \in \{0,1\}^s$, and consider the (deterministic) polynomial-sized circuit $A(\cdot)$ which outputs the first message in $S(x, \cdot; r)$. By Conjecture 9.9, there exist C_1, C_2 of size at most $p(n)$ such that $A(C_1) = A(C_2) = \alpha$ but $C_1(\alpha) \neq C_2(\alpha)$. By optimal soundness, there exists at most one verifier message $\beta \in \{0,1\}^k$ such that (α, β) can be completed to an accepting transcript. At least one of $C_1(\alpha)$ or $C_2(\alpha)$ is not equal to β, and thus the prover M can send a circuit C such that $S(x, C; r)$ is rejecting. □

10 Conclusions

New results on zero knowledge put in question the usefulness of black-box lower bounds for understanding the power of general (non-black-box) zero knowledge. Thus, non-black-box lower bounds and upper bounds are required to close the gaps in our understanding of this field. In this work, we

have shown several such lower bounds. Beyond the lower bounds themselves, our work implies new differences between the power of zero-knowledge *proofs* versus *arguments*. It also implies that in some cases complexity assumptions can and should be used to obtain zero-knowledge lower bounds. Sometimes we will need to use assumptions, such as Assumption 3.2, that are not commonly used in cryptography.

In our opinion, one of the most important open questions is whether there exists a constant-round public-coin zero-knowledge *proof* system for **NP** and the special case of 3-round proof systems obtained by parallel repetition of [GMW, Blu2]. In Section 9, we provided several plausible conjectures that would imply that the answer is no. It is an open problem to establish one of these conjectures based on more standard complexity-theoretic assumptions.

Acknowledgements

We thank Oded Goldreich, Silvio Micali and Luca Trevisan for helpful discussions, and the anonymous FOCS referees for useful comments.

References

[AK] V. Arvind and J. Köbler. On pseudorandomness and resource-bounded measure. *Theoret. Comput. Sci.*, 255(1-2):205–221, 2001.

[BM] L. Babai and S. Moran. Arthur-Merlin Games: A Randomized Proof System and a Hierarchy of Complexity Classes. *J. Comput. Syst. Sci.*, 36:254–276, 1988.

[Bar] B. Barak. How to go beyond the black-box simulation barrier. In *Proc. 42st FOCS*, pages 106–115. IEEE, 2001. Preliminary full version available on `http://www.wisdom.weizmann.ac.il/~boaz`.

[BG] B. Barak and O. Goldreich. Universal Arguments and their Applications. Cryptology ePrint Archive, Report 2001/105, 2001. Preliminary version appeared in CCC' 2002.

[BGGL] B. Barak, O. Goldreich, S. Goldwasser, and Y. Lindell. Resettably-Sound Zero-Knowledge and its Applications. Record 2001/063, Cryptology ePrint Archive, Aug. 2001. Appeared in 42nd FOCS, 2001.

[BL] B. Barak and Y. Lindell. Strict Polynomial-time in Simulation and Extraction. Cryptology ePrint Archive, Report 2002/043, 2002. `http://eprint.iacr.org/`. Extended appeared in STOC' 02.

[BLV] B. Barak, Y. Lindell, and S. Vadhan. Lower bounds for non-black-box zero knowledge. In *Proceedings of the 44th Annual Symposium on Foundations of Computer Science (FOCS '03)*, pages 384–393, Cambridge, MA, 11–14 Oct. 2003. IEEE.

[BOV] B. Barak, S. J. Ong, and S. Vadhan. Derandomization in Cryptography. In D. Boneh, editor, *Advances in Cryptology—CRYPTO '03*, volume 2729 of *Lecture Notes in Computer Science*. Springer-Verlag, 17–21 August 2003.

[Bel] M. Bellare. A note on negligible functions. *Journal of Cryptology*, 15(4):271–284, 2002.

[BGP] M. Bellare, O. Goldreich, and E. Petrank. Uniform Generation of NP-Witnesses Using an NP-Oracle. *Information and Computation*, 163, 2000.

[BP] M. Bellare and A. Palacio. The Knowledge-of-Exponent Assumptions and 3-Round Zero-Knowledge Protocols. Cryptology ePrint Archive, Report 2004/008, 2004. `http://eprint.iacr.org/`.

[Ber] E. R. Berlekamp. Factoring polynomials over large finite fields. *Math. Comput.*, 24, 1970.

[Blu1] M. Blum. Coin Flipping by Phone. In *Proc. 24th IEEE Computer Conference (CompCon)*, pages 133–137, 1982. See also *SIGACT News*, Vol. 15, No. 1, 1983.

[Blu2] M. Blum. How to prove a theorem so no one else can claim it. In *Proceedings of the International Congress of Mathematicians*, pages 1444–1451, 1987.

[BCY] G. Brassard, C. Crépeau, and M. Yung. Everything in NP Can Be Argued in *Perfect* Zero-Knowledge in a *Bounded* Number of Rounds. In *Eurocrypt '89*, pages 192–195, 1989. LNCS No. 434.

[Can] R. Canetti. Universally composable security: a new paradigm for cryptographic protocols (extended abstract). In *42nd IEEE Symposium on Foundations of Computer Science (Las Vegas, NV, 2001)*, pages 136–145. IEEE Computer Soc., Los Alamitos, CA, 2001.

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

[CKPR] R. Canetti, J. Kilian, E. Petrank, and A. Rosen. Black-Box Concurrent Zero-Knowledge Requires $\tilde{\Omega}(\log n)$ Rounds. Record 2001/051, Cryptology ePrint Archive, June 2001. Extended abstract appeared in STOC' 01.

[CKL] R. Canetti, E. Kushilevitz, and Y. Lindell. On the Limitations of Universally Composable Two-Party Computation without Set-up Assumptions. In *EUROCRYPT '03*, pages 68–86, 2003.

[CSD] R. Cramer, B. Schoenmakers, and I. Damgård. Proofs of partial knowledge and simplified design of witness hiding protocols. *CWI Quarterly*, 8(2):111–127, 1995.

[CS] R. Cramer and V. Shoup. Design and Analysis of Practical Public-Key Encryption Schemes Secure against Adaptive Chosen Ciphertext Attack. Cryptology ePrint Archive, Report 2001/108, 2001. `http://eprint.iacr.org/`.

[Dam] I. Damgård. Towards Practical Public-Key Cryptosystems Provably-Secure against Chosen-Ciphertext Attack. In J. Feigenbaum, editor, *Advances in Cryptology—CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 445–456. Springer-Verlag, 1991, 11–15 Aug. 1991.

[DN] C. Dwork and M. Naor. Zaps and Their Applications. In *Proc. 41st FOCS*, pages 283–293. IEEE, 2000.

[DNRS] C. Dwork, M. Naor, O. Reingold, and L. Stockmeyer. Magic functions. In *Proc. 40th FOCS*, pages 523–534. IEEE, 1999.

[DNS] C. Dwork, M. Naor, and A. Sahai. Concurrent Zero Knowledge. In *Proc.* 30*th STOC*, pages 409–418. ACM, 1998.

[FS1] U. Feige and A. Shamir. Zero Knowledge Proofs of Knowledge in Two Rounds. In *Crypto '89*, pages 526–545, 1989. LNCS No. 435.

[FS2] A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In *Crypto '86*, pages 186–194, 1986. LNCS No. 263.

[Gol1] O. Goldreich. A uniform-complexity treatment of encryption and zero-knowledge. *Journal of Cryptology*, 6(1):21–53, 1993.

[Gol2] O. Goldreich. *Foundations of Cryptography: Basic Tools*. Cambridge University Press, 2001.

[Gol3] O. Goldreich. Lecture notes for the course "Randomized Methods in Computation", 2001. Available from `http://www.wisdom.weizmann.ac.il/~oded/rnd.html`.

[Gol4] O. Goldreich. *Foundations of Cryptography: Basic Applications*. Cambridge University Press, 2004.

[GK1] O. Goldreich and A. Kahan. How to Construct Constant-Round Zero-Knowledge Proof Systems for NP. *J. Cryptology*, 9(3):167–189, Summer 1996.

[GK2] O. Goldreich and H. Krawczyk. On the Composition of Zero-Knowledge Proof Systems. *SIAM J. Comput.*, 25(1):169–192, Feb. 1996. Preliminary version appeared in ICALP' 90.

[GMW] O. Goldreich, S. Micali, and A. Wigderson. Proofs that Yield Nothing But Their Validity or All Languages in NP Have Zero-Knowledge Proof Systems. *J. ACM*, 38(3):691–729, July 1991.

[GO] O. Goldreich and Y. Oren. Definitions and Properties of Zero-Knowledge Proof Systems. *J. Cryptology*, 7(1):1–32, Winter 1994. Preliminary version in FOCS' 87.

[GVW] O. Goldreich, S. Vadhan, and A. Wigderson. On Interactive Proofs with a Laconic Prover. *Computational Complexity*, 11:1–53, 2002. Extended abstract in *ICALP '01*.

[GMR] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989. Preliminary version in STOC' 85.

[GT] S. Goldwasser and Y. Tauman. On the (In)security of the Fiat-Shamir Paradigm. Cryptology ePrint Archive, Report 2003/034, 2003. Extended abstract appears in these proceedings.

[GQ] L. C. Guillou and J.-J. Quisquater. A "paradoxical" identity-based signature scheme resulting from zero-knowledge. In *Advances in cryptology—CRYPTO '88 (Santa Barbara, CA, 1988)*, volume 403 of *Lecture Notes in Comput. Sci.*, pages 216–231. Springer, Berlin, 1990.

[HT] S. Hada and T. Tanaka. On the existence of 3-round zero-knowledge protocols. In *Advances in cryptology—CRYPTO '98 (Santa Barbara, CA, 1998)*, volume 1462 of *Lecture Notes in Comput. Sci.*, pages 408–423. Springer, Berlin, 1998.

[HILL] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[IW] R. Impagliazzo and A. Wigderson. **P** = **BPP** if **E** Requires Exponential Circuits: Derandomizing the XOR Lemma. In *Proc. 29th STOC*, pages 220–229. ACM, 1997.

[JVV] M. R. Jerrum, L. G. Valiant, and V. V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical Comput. Sci.*, 43(2-3):169–188, 1986.

[Kil] J. Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *Proc. 24th STOC*, pages 723–732. ACM, 1992.

[KP] J. Kilian and E. Petrank. Concurrent Zero-Knowledge in Poly-logarithmic Rounds. Cryptology ePrint Archive, Report 2000/013, 2000. Preliminary version appeared in STOC' 01.

[KPR] J. Kilian, E. Petrank, and C. Rackoff. Lower bounds for zero knowledge on the Internet. In *Proc. 39th FOCS*, pages 484–492. IEEE, 1998.

[KvM] A. R. Klivans and D. van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM J. Comput.*, 31(5):1501–1526, 2002.

[Lev] L. A. Levin. Universal search problems. *Problemy Peredači Informacii*, 9(3):115–116, 1973.

[Lub] M. G. Luby. *Pseudorandomness and cryptographic applications*. Princeton computer science notes. Princeton University Press, 1996.

[Mic] S. Micali. CS proofs. In *Proc. 35th FOCS*, pages 436–453. IEEE, 1994.

[MV] P. B. Miltersen and N. V. Vinodchandran. Derandomizing Arthur-Merlin games using hitting sets. In *Proc. 40th FOCS*, pages 71–80. IEEE, 1999.

[Nao1] M. Naor. Bit Commitment Using Pseudorandomness. *J. Cryptology*, 4(2):151–158, 1991. Preliminary version in CRYPTO' 89.

[Nao2] M. Naor. On Cryptographic Assumptions and Challenges. In D. Boneh, editor, *Advances in Cryptology—CRYPTO '03*, volume 2729 of *Lecture Notes in Computer Science*. Springer-Verlag, 17–21 August 2003.

[NZ] N. Nisan and D. Zuckerman. Randomness is Linear in Space. *J. Comput. Syst. Sci.*, 52(1):43–52, Feb. 1996.

[Oka] T. Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In *Advances in cryptology—CRYPTO '92 (Santa Barbara, CA, 1992)*, volume 740 of *Lecture Notes in Comput. Sci.*, pages 31–53. Springer, Berlin, 1993.

[PS] D. Pointcheval and J. Stern. Security proofs for signature schemes. In *Advances in cryptology—EUROCRYPT '96 (Saragossa, 1996)*, volume 1070 of *Lecture Notes in Comput. Sci.*, pages 387–398. Springer, Berlin, 1996.

[PRS] M. Prabhakaran, A. Rosen, and A. Sahai. Concurrent Zero Knowledge with Logarithmic Round-Complexity. In *Proc.* 33*rd FOCS*. IEEE, 1992.

[Rey] L. Reyzin. *Zero-Knowledge with Public Keys*. PhD thesis, MIT, 2001.

[RK] R. Richardson and J. Kilian. On the Concurrent Composition of Zero-Knowledge Proofs. In *Eurocrypt '99*, 1999. LNCS No. 1592.

[Ros] A. Rosen. A Note on the Round-Complexity of Concurrent Zero-Knowledge. In *Crypto '00*, 2000. LNCS No. 1880.

[SV] A. Sahai and S. Vadhan. A complete problem for statistical zero knowledge. *Journal of the ACM*, 50(2):196–249, March 2003. Extended abstract in *FOCS '97*.

[Sch] C. P. Schnorr. Efficient Signature Generation by Smart Cards. *Journal of Cryptology*, 4(3):161–174, 1991.

[SU] R. Shaltiel and C. Umans. Simple extractors for all min-entropies and a new pseudo-random generator. In *Proc.* 42*st FOCS*, pages 648–657. IEEE, 2001.

[Sip] M. Sipser. A Complexity Theoretic Approach to Randomness. In *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing*, pages 330–335, Boston, Massachusetts, 25–27 Apr. 1983.

[Sto] L. Stockmeyer. On Approximation Algorithms for #P. *SIAM J. Comput.*, 14(4):849–861, Nov. 1985.

Part IV
Appendices

A Uniform Zero Knowledge

The standard definition of zero knowledge, as presented in Section 2, refers to computational indistinguishability with respect to *nonuniform* distinguishers. Goldreich [Gol1] has given a treatment of zero knowledge that makes sense for uniform distinguishers. In this section, we give analogues of some of our results in this uniform setting.

A.1 Definition

Following [Gol1], for uniform indistinguishability, we incorporate a distribution on the input x. We call an ensemble $\{X_n\}_{n\in\mathbb{N}}$ of probability distributions *polynomial-time samplable* if there is a PPT algorithm M such that $M(1^n)$ is distributed according to X_n for all n.

Definition A.1 (uniform computational indistinguishability). For an index set $I \subseteq \{0,1\}^*$, two ensembles of random variables $\{A_x\}_{x\in I}$ and $\{B_x\}_{x\in I}$ are *uniformly indistinguishable* if for every (uniform) PPT algorithm D and every polynomial-time samplable $\{Z_n = (X_n, Y_n)\}$, there exists a negligible function μ such that for all n,

$$|\Pr[D(X_n, Y_n, A_{X_n}) = 1 \ \&\ X_n \in I \cap \{0,1\}^n] - \Pr[D(X_n, Y_n, B_{X_n}) = 1 \ \&\ X_n \in I \cap \{0,1\}^n]| \leq \mu(n)$$

Intuitively, this definition says that it is infeasible to generate $x \in I$ such that A_x and B_x are distinguishable (by a uniform algorithm D, even it is given additional information y that is correlated with x).

The definition (implicit) in [Gol1] restricts to distributions (X_n, Y_n) such that the support of X_n is contained I (and thus omits the $X_n \in I$ condition in the probabilities). The definition above is stronger, but we believe it to be more natural.

For reasons that will become clear shortly, in this section, we will work with relations that are *strongly poly-balanced* in the sense that there is a polynomial p with nonnegative integer coefficients such that $(x, w) \in R \Rightarrow |w| = p(|x|)$ (instead of just $|w| \leq p(|x|)$). The point is that the length of x can be determined from the length of the pair (x, w).

Definition A.2 (uniform plain zero knowledge). An interactive proof system (P, V) with respect to a strongly poly-balanced relation R is *uniform plain zero knowledge* if for every PPT V^*, there exists a probabilistic polynomial-time *simulator* S such that the ensembles $\{\langle P(w), V^*\rangle(x)\}_{(x,w)\in R}$ and $\{S(x)\}_{(x,w)\in R}$ are uniformly indistinguishable.

The above definition says that for every PPT V^*, there exists a PPT simulator S such that for all PPT D and all samplable distributions $(X'_n, Y'_n) = ((X_n, W_n), Y_n)$, we have

$$\begin{aligned}|&\Pr[D(X_n, W_n, Y_n, \langle P(W_n), V^*\rangle(X_n)) = 1 \ \&\ (X_n, W_n) \in R \cap (\{0,1\}^n \times \{0,1\}^*)] \\ -\ &\Pr[D(X_n, W_n, Y_n, S(X_n)) = 1 \ \&\ (X_n, W_n) \in R \cap (\{0,1\}^n \times \{0,1\}^*)]| \leq \mathrm{neg}(n).\end{aligned}$$

Note that a direct application of the definition of uniform indistinguishability would give the condition $X'_n = (X_n, W_n) \in R \cap \{0,1\}^n$ rather than $(X_n, W_n) \in R \cap (\{0,1\}^n \times \{0,1\}^*)$. The two definitions are equivalent due to the fact that R is strongly poly-balanced.

Observe that the distinguisher is given not only the input X_n, but also the witness W_n and some additional auxiliary information Y_n that may be correlated with X_n and W_n, but the verifier and the simulator are only given the input X_n. Intuitively, this models a situation where verifier starts with no a priori information other than the input X_n, but we want to ensure that receiving the zero-knowledge proof would not help the verifier attack a system whose behavior depends not just on X_n but also on the witness W_n and possibly some other correlated information Y_n.

We note that the relation R has greater significance for uniform zero knowledge than the other forms, because uniform zero knowledge only requires the simulation to be indistinguishable from the verifier's view on inputs x that can be efficiently sampled *together with a witness* w. In case R is the trivial relation for some language L (i.e. $R = L \times \{\lambda\}$), then uniform zero knowledge simply says that the simulator's output is indistinguishable from the verifier's view for all samplable distributions on the input x.

Definition A.3 (uniform auxiliary-input zero knowledge). An interactive proof system (P, V) with respect to relation R is *uniform auxiliary-input zero knowledge* if for every PPT V^* and polynomial p with nonnegative integer coefficients, there exists a PPT S such that the ensembles

$$\{\langle P(w), V^*(z)\rangle(x)\}_{(x,w)\in R, z\in\{0,1\}^{p(|x|)}} \qquad \text{and} \qquad \{S(x,z)\}_{(x,w)\in R, z\in\{0,1\}^{p(|x|)}}$$

are uniformly indistinguishable.

It is known that uniform auxiliary-input zero knowledge with an efficient prover is closed under sequential composition [Gol1].

As with the nonuniform version, here the indistinguishability is required even for distinguishers that have additional a priori information beyond the auxiliary input of the verifier. In this case, it is because we consider efficiently samplable distributions (X_n, W_n, Z_n, Y_n), where X_n is the input, W_n the witness, Z_n the verifier's auxiliary input, and Y_n the distinguisher's additional auxiliary input, and the entire 4-tuple (X_n, W_n, Z_n, Y_n) is provided to the distinguisher.

Also as in the nonuniform case, there is a universal verifier $V^*_{\text{uni}}(x, z)$ that interprets its auxiliary input z as a Boolean circuit C_z and uses C_z as its strategy (i.e. next-message function); if the zero-knowledge condition holds for V^*_{uni}, then it holds for all PPT verifier strategies V^*.

A.2 Triviality

It is well-known that every language in **BPP** have trivial zero-knowledge proofs (where the verifier decides the language on its own, and the prover sends nothing). For uniform zero knowledge, this triviality can be extended to a larger class of languages.

Definition A.4. For a language L and a poly-balanced relation R such that $L = L_R$, we say that L is *uniformly trivial with respect to* R if there is a PPT algorithm A such that the following two conditions hold:

- For every samplable distribution $\{D_n = (X_n, W_n)\}_{n\in\mathbb{N}}$, there exists a negligible function $\mu(n)$ such that $\Pr[A(X_n) \neq \texttt{accept} \ \&\ (X_n, W_n) \in R \cap (\{0,1\}^n \times \{0,1\}^*)] \leq \text{neg}(n)$.
- For all $x \notin L$, $\Pr[A(x) \neq \texttt{reject}] \leq \text{neg}(n)$.

In other words, the algorithm A is correct with high probability on all NO instances, and it is hard to generate YES instances (together with witnesses) on which A errs with nonnegligible probability. An equivalent formulation is to require that for all $x \notin L$, $A(x)$ rejects with probability

at least 2/3, and with probability $1 - \text{neg}(n)$ over $(x, w) \leftarrow_{\text{R}} D_n$, $A(x)$ accepts with probability at least 2/3 or $(x, w) \notin R \cap (\{0,1\}^n \times \{0,1\}^*)$. (Equivalence is seen by amplifying via independent repetitions and majority vote.)

Proposition A.5. *If a language L is uniformly trivial wrt R and $L \in$ **IP**, then L has a uniform auxiliary-input zero-knowledge proof wrt R. Moreover, if R is an **NP**-relation (or even an **MA** relation), then the prover is efficient.*

Proof. Uniform triviality guarantees that the verifier can decide L on its own on "most" instances (just by running the algorithm A in the definition). However, the completeness and soundness conditions of zero-knowledge proofs are universally quantified over *all* instances x. Thus, we have the prover check whether x is an instance where the algorithm A would fail to decide correctly, and in such a case, provide V with a (non-zero-knowledge) proof. These rare instances won't hurt the uniform zero knowledge property because, by definition of uniformly trivial, they are hard to generate.

Let A be the PPT algorithm given by the definition of uniformly trivial. Consider the following interactive proof:

$(P(w), V)(x)$**:**

1. P runs $A(x)$ for n executions, where $n = |x|$. If less than 3/4 of these executions are accepting, then P initiates the (non-ZK) interactive proof for L on input x. (If R is an **NP** or **MA** relation, the non-ZK proof simply amounts to sending w.)

2. V runs $A(x)$ for n executions. If $A(x)$ accepts in a majority of these executions, then V halts and accepts. Otherwise, V accepts according the (non-ZK) interactive proof for L (rejecting if P does not initiate it).

Negligible soundness follows from the definition of trivial and the soundness of the non-ZK proof. For completeness, let $B = \{x \in L : \Pr[A(x) = \texttt{accept}] \leq 2/3\}$. If $x \in B$, then with probability $1 - 2^{-\Omega(n)}$, P will initiate the non-ZK proof for L, and thus A will accept by completeness of the non-ZK proof. If $x \notin B$, then with probability $1 - 2^{-\Omega(n)}$, the majority of V's executions of A will be accepting, and thus V will accept.

For uniform auxiliary-input zero knowledge, let $V^*(x, z)$ be any verifier and let $S(x, z)$ be the simulator that simply simulates V^*'s view as if the prover sends nothing. We argue that the simulation produced by S is good on any samplable distribution (X_n, W_n, Z_n, Y_n). Let $B' = \{x : \Pr[A(x) = \texttt{accept}] \leq 7/8\}$. From the definition of uniformly trivial, it follows that

$$\Pr\left[X_n \in B' \;\&\; (X_n, W_n) \in R \cap (\{0,1\}^n \times \{0,1\}^*)\right] \leq \text{neg}(n).$$

Now consider any fixed (x, w, z, y) such that $x \notin B'$. With probability $1 - 2^{-\Omega(n)}$, at least 3/4 of P's executions of $A(x)$ will accept, and P sends nothing to V. □

It turns out that one-way functions imply non-trivial languages in **NP**.

Proposition A.6. *If one-way functions exist, then there exists an **NP**-relation R such that L_R is not uniformly trivial wrt R.*

Proof. Assume one-way functions exist. By [HILL], there exists an pseudorandom generator [HILL] $G : \{0,1\}^n \to \{0,1\}^{2n}$ (ie the analogue of a pseudorandom generator where the indistinguishability

condition only holds for infinitely many n's). Consider the **NP**-relation $R = \{(x, w) : G(w) = x\}$, so $L_R = \text{Image}(G)$. Suppose L_R were uniformly trivial wrt R. Let W_{2n} be a random variable distributed uniformly over $\{0,1\}^n$ and let $X_{2n} = G(W_{2n})$. Since $\Pr\left[(X_{2n}, W_{2n}) \in R \cap (\{0,1\}^{2n} \times \{0,1\}^*)\right] = 1$, we have

$$\Pr\left[A(X_{2n}) = \texttt{accept}\right] \geq 1 - \text{neg}(n).$$

On the other hand, since L_R contains at most a 2^{-n} fraction of $\{0,1\}^n$, we have

$$\Pr\left[A(U_{2n}) = \texttt{accept}\right] \leq 2^{-n} + \text{neg}(n) = \text{neg}(n).$$

This contradicts the fact that G is an pseudorandom generator. □

In fact, the above proposition only needs the existence of one-way functions which are hard to invert for infinitely many input lengths, referred to as *io-OWF*. In fact, if we modify the definition of trivial so that the PPT A may depend on the distribution (X_n, W_n), then the existence of nontrivial **NP**-relations becomes *equivalent* to the existence of io-OWF. (However, with this modification, then the proof of Proposition A.5 fails.)

A.3 Two-round Zero Knowledge

We begin by presenting an analogue of the Goldreich–Oren result (Theorem 3.1).

Theorem A.7. *If a language L has a 2-round proof or argument system wrt relation R that is uniform* auxiliary-input *zero knowledge, then L is trivial wrt R. In particular, if one-way functions exist, then for every* **NP***-complete relation R, L_R does not have a 2-round uniform auxiliary-input zero-knowledge argument system wrt R.*

Proof. Suppose (P, V) is a 2-round zero-knowledge argument for $L = L_R$ wrt relation R. Consider the verifier $V^*(x, z)$ that sends its auxiliary input z as its first message, and let S be the simulator for V^*. Define a PPT algorithm A as follows.

$A(x)$**, for** $|x| = n$**:**

1. Choose $y \leftarrow_{\text{R}} \{0,1\}^m$, where $m = m(n)$ is the number of coin tosses used by V.
2. Let $z = V(x; y)$.
3. Run $S(x, z)$ and let m be the prover message in the simulated transcript.
4. Accept if $V(x; y)$ would accept prover message m.

We now prove that A satisfies the conditions of uniform triviality. When $x \notin L$, then we can view S defining a PPT prover strategy, and thus $A(x)$ accepts with probability at most 1/3 by the soundness of (P, V). Now, let (X_n, W_n) be any samplable distribution, and consider the PPT distinguisher D defined by $D(x, y, t) = 1$ iff $t = (x, z, m)$ such that $V(x; y)$ would accept prover message m. Let Y_n be distributed uniformly in $\{0,1\}^m$ and $Z_n = V(X_n; Y_n)$. Then,

$$\begin{aligned}
&\Pr\left[A(X_n) \neq \texttt{accept} \;\&\; (X_n, W_n) \in R\right] \\
&= \Pr\left[D(X_n, Y_n, S(X_n, Z_n)) = 0 \;\&\; (X_n, W_n) \in R\right] \\
&\leq \Pr\left[D(X_n, Y_n, \langle P, V^*(Z_n)\rangle(X_n)) = 0 \;\&\; (X_n, W_n) \in R\right] + \text{neg}(n) \\
&\leq \text{neg}(n),
\end{aligned}$$

where the second-to-last inequality follows from uniform auxiliary-input zero knowledge, and the last inequality from completeness of (P, V). □

Now we present uniform analogues of some of our lower bounds.

Theorem A.8. *Under Assumption 3.2, if a language L has a 2-round public-coin proof system that is* uniform plain *zero knowledge wrt a relation R and has an efficient prover, then L is uniformly trivial wrt to R.*

Proof. Most of the proof is the same as that of Theorem A.8. Given the 2-round proof (P, V), we use a pseudorandom generator G against nondeterministic circuits to obtain a derandomized verifier V^*, we let S be its simulator, and construct an algorithm M based on S and V^* in the same way:

$M(x)$**:** Run $S(x)$ many times to obtain transcripts $(G(s_1), \beta_1; s_1), (G(s_2), \beta_2; s_2), \ldots, (G(s_q), \beta_q; s_q)$, where $q = n \cdot 2^\ell$. Accept if $\{s_1, \ldots, s_q\} = \{0,1\}^\ell$ and, for the majority of $s \in \{0,1\}^\ell$, there exists an i such that $s_i = s$ and $V^*(x, G(s_i), \beta_i; s_i)$ accepts.

The proof that M rejects all $x \notin L$ with probability at least $2/3$ is the same as in the nonuniform case. The only difference in the proof is in analyzing M's behavior on $x \in L$. Specifically, we want to show that for any samplable distribution (X_n, W_n), with probability $1 - \mathrm{neg}(n)$ over $(x, w) \leftarrow_{\mathrm{R}} (X_n, W_n)$, $M(x)$ accepts with probability at least $2/3$ or $(x, w) \notin R$.

First, we argue that it suffices to prove this if we replace the runs of $S(x)$ in the definition of M with samples of $\langle P(w), V^* \rangle(x)$. If M only used one sample of $S(x)$, this would follow from the uniform indistinguishability of the ensembles $\{\langle P(w), V^* \rangle(x)\}_{(x,w) \in R}$ and $\{S(x)\}_{(x,w) \in R}$. However, since M uses many samples of $S(x)$, it is also important that both ensembles are polynomial-time samplable given (x, w) (for the single-sample indistinguishability to imply multiple-sample indistinguishability, cf. [Gol2, Thm 3.2.6]). This holds because P is an efficient prover.

Now, once we replace the runs of $S(x)$ with samples of $\langle P(w), V^* \rangle(x)$, it follows that $M(x)$ accepts with probability $1 - 2^{-\Omega(n)}$, just as in the proof of Theorem 3.4. □

Theorem A.9.

1. *If a language L has a constant-round public-coin proof system wrt relation R that is uniform auxiliary-input resettable zero knowledge, then L is uniformly trivial wrt R.*

2. *Under Assumption 3.2, if a language L has a constant-round public-coin proof system wrt relation R that is uniform plain resettable zero knowledge, then L is uniformly trivial wrt R.*

Proof sketch. The proof is identical to that of Theorem A.9, noting that the reduction from a constant-round public-coin proof system to a 2-round public-coin proof system preserves *uniform* resettable zero knowledge, and appealing to Theorems A.7 and A.8 at the end. We can drop the efficient-prover condition because the multiple-sample indistinguishability of $\langle P(w), V^* \rangle(x)$ and $S(x)$ holds by virtue of the resettable zero knowledge property. □