Zero Knowledge with Efficient Provers

Minh-Huyen Nguyen[*]
Harvard University
Division of Engineering and Applied Sciences
Cambridge, MA 02138, USA
mnguyen@eecs.harvard.edu

Salil Vadhan[†]
Harvard University
Division of Engineering and Applied Sciences
Cambridge, MA 02138, USA
salil@eecs.harvard.edu

ABSTRACT

We prove that every problem in **NP** that has a zero-knowledge proof also has a zero-knowledge proof where the prover can be implemented in probabilistic polynomial time given an **NP** witness. Moreover, if the original proof system is statistical zero knowledge, so is the resulting efficient-prover proof system. An equivalence of zero knowledge and efficient-prover zero knowledge was previously known only under the assumption that one-way functions exist (whereas our result is unconditional), and no such equivalence was known for statistical zero knowledge. Our results allow us to translate the many general results and characterizations known for zero knowledge with inefficient provers to zero knowledge with efficient provers.

Categories and Subject Descriptors

F.1.2 [**Modes of Computation**]: Interactive and reactive computation

General Terms

Theory

Keywords

cryptography, computational complexity, language-dependent commitment schemes, zero-knowledge

1. INTRODUCTION

Zero-knowledge proofs [18] have been one of the most fertile sources of interaction between cryptography and complexity theory. From the perspective of cryptography, zero-knowledge proofs provide a powerful building block for secure protocols and serve as a good testbed for understanding new security concerns such as concurrency and composability. From a complexity point of view, zero knowledge enriches the classical study of **NP** proofs with randomness, interaction, and secrecy, and provides an interesting classification of computational problems.

In the past decade, this interaction has yielded a number of very general results about zero-knowledge proofs. These include natural complete problems (or similar characterizations), closure properties, equivalence of private coins and public coins, equivalence of honest-verifier and malicious-verifier zero knowledge, and more. Results of this form were first obtained for the class **SZK** of problems having "statistical" zero-knowledge proofs [28, 31, 15, 17, 32], and were recently extended to the class **ZK** of problems having general, "computational" zero-knowledge proofs [34].[1]

However, there has been a significant gap between this complexity-theoretic study of zero knowledge and the cryptographic applications of zero knowledge. Specifically, the works mentioned above ([28, 31, 15, 17, 32, 34]) allow for a *computationally unbounded* prover, following the standard definition of interactive proofs [18, 2]). Cryptographic applications of zero knowledge, on the other hand, require that the prover strategy be implementable in polynomial time, so that the resulting cryptographic protocol is feasible to execute. We can hope for a polynomial-time prover in such applications because the statements being proven are typically **NP** statements (e.g. that a given ciphertext decrypts to a particular message) and the prover typically has an **NP** witness (e.g. the decryption key) to the statement. Thus we want zero-knowledge proofs for languages in **NP** where the prover can be implemented in polynomial time given an **NP** witness. We refer to a proof system with this property as having an *efficient prover*.

Many existing zero-knowledge proofs have efficient provers, including perfect (or statistical) zero-knowledge proofs for specific problems such as Quadratic Residuosity [18] and Graph Isomorphism [14], as well as the (computational)

[*] Supported by NSF grants CNS-0430336, CCR-0205423, and ONR grant N00014-04-1-0478.

[†] Supported by NSF grants CNS-0430336, CCR-0205423, and CCR-0133096, ONR Grant N00014-04-1-0478.

STOC'06, May21–23, 2006, Seattle, Washington, USA.
Copyright 2006 ACM 1-59593-134-1/06/0005 ...$5.00.

[1] All of these results were known about **ZK** under the assumption that one-way functions exist [14, 21, 7, 19, 26, 20]. The point of [34] is to establish the same results unconditionally.

zero-knowledge proofs for all of **NP** based on one-way functions [14]. However, the general, unconditional results mentioned earlier do not provide efficient provers. For example, the results of [28, 34] provide a generic way to transform any zero-knowledge proof into one that uses public coins. But, even if the original prover is efficient, the prover in the resulting public-coin proof system will run in superpolynomial time (indeed, the best upper bound seems to be probabilistic polynomial time with an **NP** oracle). Thus, even though some of these general transformations have cryptographic significance (e.g. from honest-verifier zero knowledge to malicious-verifier zero knowledge), they cannot actually be used in cryptographic protocols.[2]

In [33], it was shown that this blow-up in prover complexity is inherent in the approach being used. Specifically, if one-way functions exist, then there is no "black-box transformation" from private-coin honest-verifier statistical zero-knowledge proofs to public-coin proofs that preserves prover efficiency, where "black-box transformation" essentially says that the prover and verifier strategies in the new proof system are constructed using the original prover and verifier strategies (as well as the simulator) as black boxes.[3] Indeed, all of the general transformations for **SZK** at the time [28, 31, 15, 17] were black box in this sense.

Recently, Micciancio and Vadhan [24] suggested a "non-black-box" approach with the potential to overcome the above bottleneck, and reduced the question to constructing a certain kind of "instance-dependent" commitment scheme [5, 22] for any **SZK**-complete problem. (This is described in more detail below in Section 1.) However, they only managed to construct such a commitment scheme for a restricted form of an **SZK**-complete problem. In this work, we carry out a variant of their approach (introducing a new type of commitment scheme), and thereby resolve the question of prover efficiency for statistical zero knowledge:

Theorem 1.1. *Every problem in* $\mathbf{SZK} \cap \mathbf{NP}$ *has a statistical zero-knowledge proof with an efficient prover. Moreover, the proof system is public coin and has perfect completeness.*

We note that $\mathbf{SZK} \subseteq \mathbf{AM} \cap \textbf{co-AM}$ [12, 1] and it has been shown that $\mathbf{AM} = \mathbf{NP}$ under plausible complexity assumptions [23, 25], so we conclude that *all* of **SZK** has efficient provers under plausible complexity assumptions.

Combining our techniques with the results of [34], we obtain the analogous result for computational zero knowledge:

Theorem 1.2. *Every problem in* $\mathbf{ZK} \cap \mathbf{NP}$ *has a (computational) zero-knowledge proof with an efficient prover. Moreover, the proof system is public coin and has perfect completeness.*

We recall that the classic Three-Coloring protocol of Goldreich, Micali, and Wigderson [14] shows that if one-way functions exist, all of **NP** (not just $\mathbf{ZK} \cap \mathbf{NP}$) has zero-knowledge proofs with an efficient prover. Thus, the significance of Theorem 1.2 is that it is unconditional.

We can use these theorems to automatically translate general results known about **SZK** and **ZK** to the classes $\mathbf{SZK}_{\mathbf{eff}}$ and $\mathbf{ZK}_{\mathbf{eff}}$ of problems having zero-knowledge proofs with an efficient prover.

Related Work. Bellare and Petrank [6] showed that every problem in **SZK** has a statistical zero knowledge proof where the prover can be implemented in probabilistic polynomial time with an **NP** *oracle*. Bellare and Goldwasser [4] studied the power of the prover in (non-zero-knowledge) interactive proofs. They showed that, under plausible complexity assumptions, there exist languages $L \in \mathbf{NP}$ for which proving membership is harder than decision, in that L does not have an interactive proof where the prover can be implemented in polynomial time given an oracle for L (but no auxiliary-input/**NP** witness). The notions of prover efficiency considered in both of these works [6, 4] are mainly interesting from a complexity-theoretic perspective, and do not match the requirements of cryptographic applications.

Ostrovsky and Wigderson [29, 30] showed that if a hard-on-average problem has a zero-knowledge proof, then one-way functions exist. Thus, for hard-on-average problems, one can use the equivalence between zero knowledge and efficient-prover zero knowledge based on one-way functions. However, this approach does not seem to suffice for proving results about all of **ZK** or **SZK**, because these classes may be nontrivial (not equal to **BPP**), yet not have a hard-on-average problem. Nevertheless, the work of [29, 30] was an inspiration for the work of [34] on computational zero knowledge, which we use in our proof of Theorem 1.2.

Techniques. We begin by recalling the notion of a *commitment scheme*, which is a basic building block for many zero-knowledge proofs. A commitment scheme is a two-stage protocol between a sender and a receiver. In the first stage, the sender 'commits' to a value v, and in the second, the sender 'reveals' this value to the receiver. We want two security properties from a commitment scheme. The *hiding* property says that the receiver does not learn anything about the value v during the commit stage. And the *binding* property says that after the commit stage, there is at most one value that the sender can successfully open (without the receiver rejecting). It is known that (computationally hiding) commitment schemes exist if and only if one-way functions exist [26, 20]. The only way that the existence of one-way functions is used in constructing zero-knowledge proofs for **NP** [14] is to obtain a commitment scheme.

In an *instance-dependent* commitment scheme [5, 22],[4] the sender and receiver strategies also depend on an instance x of some decision problem Π, and we relax the security properties so that we only require hiding if x is a yes instance of Π and only require binding if x is a no instance of Π. Since we do not require the hiding and binding properties to hold simultaneously, instance-dependent commitment schemes do not imply the existence of one-way functions. In particular, they can be constructed unconditionally for cer-

[2] We note that for honest-verifier vs. malicious-verifier zero knowledge, the real bottleneck is the transformation from private coins to public coins; given a public-coin honest-verifier zero-knowledge proof, the transformation to tolerate cheating verifiers [15] preserves prover efficiency.

[3] This should not be confused with notion of (non-)black-box *simulation*, as in the work of Barak [3], which is concerned with how we establish the zero-knowledge property, not how we construct the proof system itself.

[4] Previous works [24, 34] have referred to this as a 'problem-dependent' commitment scheme, but the new terminology 'instance-dependent' seems more accurate.

tain problems. (For example, if Π is Graph Isomorphism, then on instance $x = (G_0, G_1)$, we can take a commitment of $b \in \{0, 1\}$ to be a random isomorphic copy of G_b. This is perfectly hiding when $G_0 \cong G_1$ and perfectly binding when $G_0 \not\cong G_1$. [5])

Itoh, Ohta, and Shizuya [22] showed that this relaxation of commitment schemes is still useful for constructing zero-knowledge proofs. The reason is that many constructions of zero-knowledge proofs (e.g. [14]) from commitment schemes only use the hiding property to prove zero knowledge, which is only required on YES instances, and only use the binding property to prove soundness, which is only required on NO instances. Thus, to construct a zero-knowledge proof for a problem $\Pi \in \mathbf{NP}$, it suffices to have an instance-dependent commitment scheme for Π. Then we can reduce Π to an **NP**-complete problem, such as Three-Coloring, and use the Π-dependent commitment scheme to implement the zero-knowledge proof for Three-Coloring [14].

Micciancio and Vadhan [24] suggested that this approach could be useful for obtaining efficient provers for all of **SZK**. Specifically, if we could show that every problem in **SZK** has a (statistically hiding and statistically binding) instance-dependent commitment scheme, then it would follow that every problem $\mathbf{SZK} \cap \mathbf{NP}$ has a statistical zero-knowledge proof with an efficient prover. The plausibility of this approach was demonstrated in [24] by exhibiting an instance-dependent commitment scheme for a restricted version of the **SZK**-complete problem Statistical Difference [31]. Moreover, in [34], it was shown that all of **SZK** has an instance-dependent commitment scheme, albeit with an inefficient sender, which renders it useless for the question of efficient provers.

In this work, we do *not* show how to construct an instance-dependent commitment scheme for all of **SZK**. Instead, we introduce a new relaxation of commitment schemes, which we call $\binom{2}{1}$*-binding commitment schemes*. These are commitment schemes with two phases, each consisting of a commit stage and a reveal stage. In the first phase, the sender commits to and reveals one value v_1, and subsequently, in the second phase, the sender commits to and reveals a second value v_2. We require that both phases are hiding, but only that one of them is binding. That is, the binding property only requires that with high probability, the sender will be forced to reveal the correct committed value in at least one of the phases (but which of the two phases can be determined dynamically by the malicious sender).

First, we demonstrate that these $\binom{2}{1}$-binding commitment schemes can be used to construct zero-knowledge proofs for **NP**. Roughly speaking, we run *two* executions of, say, the Three-Coloring zero-knowledge proof of [14], and use the first phase of the commitment scheme for the first execution and the second phase for the second execution. Intuitively, the zero knowledge property will hold because both phases are hiding, and soundness because at least one phase is binding. Actually, the protocol and the proof of soundness are complicated by the fact that in the first execution, only a subset of the first-phase commitments are opened (so we do not have as many second-phase commitments as first-phase ones), and that the prover can decide whether to break the binding property in the first phase or second phase differently for each of the commitments. Nevertheless, we can manage these difficulties.

Now, our goal of obtaining efficient provers for **SZK** is reduced to constructing an instance-dependent $\binom{2}{1}$-binding commitment scheme for an **SZK**-complete problem. Unfortunately, we do not know how to do this, either. Instead, for each instance x of our problem Π, we construct polynomially many two-phase commitment schemes, with the guarantee that if x is a YES instance, *at least one* of the schemes is hiding (in both phases), and if x is a NO instance, then all of the schemes are binding. However, it turns out that, with some more work, our zero-knowledge proof construction can be extended to work with such a collection of commitment schemes. And thus we obtain Theorem 1.1. The results of [34] allow us to immediately convert our collection of $\binom{2}{1}$-binding commitment schemes for **SZK** into a collection of $\binom{2}{1}$-binding commitment schemes for **ZK** (now with computational hiding), and thereby obtain Theorem 1.2.

Notation. If X is a random variable taking values in a finite set $\mathcal{U}$, then we write $x \overset{\text{R}}{\leftarrow} X$ to indicate that x is selected according to X. If S is a subset of $\mathcal{U}$, then $x \overset{\text{R}}{\leftarrow} S$ means that x is chosen uniformly from S. We denote by U_n the random variable distributed uniformly over $\{0, 1\}^n$, by $\text{neg}(n)$ an arbitrary negligible function and by $\text{poly}(n)$ 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. We follow the standard definition of zero-knowledge proof as in [13].

2. 1-OUT-OF-2 BINDING COMMITMENTS

We now introduce the notion of $\binom{2}{1}$-binding commitments that will play a central role in establishing our results. These are commitment schemes with two *sequential and related* phases such that in each phase, the sender commits to and reveals a value.

Definition 2.1. *A* 2-phase commitment scheme (S, R) *consists of four interactive protocols:*

- (S_c^1, R_c^1) *the first commitment phase*
- (S_r^1, R_r^1) *the first reveal phase*
- (S_c^2, R_c^2) *the second commitment phase*
- (S_r^2, R_r^2) *the second reveal phase*

1. *In the first commitment phase, S_c^1 receives a private input $\sigma^1 \in \{0, 1\}$ and a sequence of coin tosses r_S* [5]*. S_c^1 and R_c^1 receive as common output a commitment z^1 (without loss of generality, we can assume that z^1 is the transcript of the first commitment phase).*

2. *In the first reveal phase, S_r^1 and R_r^1 receive as common input the commitment z^1 and a bit σ^1. S_r^1 receives as private input r_S. S_r^1 and R_r^1 receive a common output τ. (Without loss of generality, we can assume that τ is the transcript of the first commitment phase and the first reveal phase and includes R_r^1's decision to accept or reject).*

[5] The receiver will also be probabilistic but our protocols will not require that the receiver remembers its coin tosses from previous phases. Indeed, our protocols will be public coin for the receiver.

3. *In the second commitment phase, S_c^2 and R_c^2 receive the common input $\tau \in \{0,1\}^\star$ (where τ denotes the common output of the first reveal phase). S_c^2 receives a private input $\sigma^2 \in \{0,1\}$ and the coin tosses r_S. S_c^2 and R_c^2 receive as common output a commitment z^2 (without loss of generality, we can assume that z^2 is the concatenation of τ and the transcript of the second commitment phase).*

4. *In the second reveal phase, S_r^2 and R_r^2 receive as common input the commitment z^2 and a bit σ^2. S_r^2 receives as private input r_S. At the end of the protocol, R_r^2 accepts or rejects.*

5. $S = (S^1, S^2) = ((S_c^1, S_r^1), (S_c^2, S_r^2))$ *and* $R = (R^1, R^2) = ((R_c^1, R_r^1), (R_c^2, R_r^2))$ *are computable in probabilistic polynomial time* $\mathrm{poly}(n)$ *(where 1^n is the security parameter).*

When defining the security of such a commitment scheme with respect to the receiver, we will restrict our focus on *honest receivers* for simplicity because we will use the compiler of [15] to convert our public-coin honest-verifier zero-knowledge proof into a public-coin zero-knowledge proof that withstands malicious verifiers. Loosely speaking, the hiding property for a 2-phase commitment scheme says that *each* commitment phase is hiding. Note that since the phases are run *sequentially*, the hiding property for the second commitment phase is formalized by including the receiver's view of the first phase.

Definition 2.2 (Hiding property). *We say that the 2-phase commitment scheme (S, R) is* statistically (resp. computationally) hiding for honest receiver *if:*

- *the views of R_c^1 in $(S_c^1(0), R_c^1)$ and $(S_c^1(1), R_c^1)$ are statistically (resp. computationally) indistinguishable.*
- *for every $\sigma^1 \in \{0,1\}$, the distributions*

  $\Big(\mathrm{View}_{R^1}(S^1(\sigma^1), R^1), \mathrm{View}_{R_c^2}(S_c^2(0), R_c^2)(\tau)\Big)$ *and*

  $\Big(\mathrm{View}_{R^1}(S^1(\sigma^1), R^1), \mathrm{View}_{R_c^2}(S_c^2(1), R_c^2)(\tau)\Big)$

  where $\tau = \mathrm{output}(S_r^1, R_r^1)(z^1, \sigma^1)$ are statistically (resp. computationally) indistinguishable.

Loosely speaking, the binding property says that *at least* one of the two phases is binding. In other words, for every sender $S^\star$, there is at most one "bad" phase $j \in \{1, 2\}$ such that given a commitment z^j, $S^\star$ can open z^j successfully both as 0 and 1 with nonnegligible probability (this bad phase is determined dynamically by the cheating sender $S^\star$). In formalizing this, we use the simplifying assumption that both reveal phases are non-interactive and deterministic. In the first reveal phase, the sender just sends a decommitment value to the receiver. In the second reveal phase, without loss of generality the sender sends its coin tosses r_S. Indeed the 2-phase commitments we construct in Section 4 will be of that form.

Definition 2.3 (1-out-of-2 binding property). *We say that the 2-phase commitment scheme (S, R) is* 1-out-of-2 statistically binding *or $\binom{2}{1}$ statistically binding if $\forall S^\star$, with probability at least $1 - \mathrm{neg}(n)$ over $z^1 \leftarrow \mathrm{output}(S^\star, R_c^1)$, there is at least one "improper" value $\sigma^1 \in \{0,1\}$ such that for any $\tau = (S^\star, R_r^1)(z^1, \sigma^1)$ where R_r^1 accepts, we have that with probability at least $1 - \mathrm{neg}(n)$ over $z^2 = (S^\star, R_c^2)(\tau)$, there is at least one value $\sigma^2 \in \{0,1\}$ such that $(S^\star, R_r^2)(z^2, \sigma^2) =$ `reject`.*

We say that the first phase commitment z^1 is broken *if $S^\star$ opens the commitment z^1 to the "improper" value σ^1 specified in the above definition (and the second phase will be binding).*

Note that if $S^\star$ opens the commitment z^1 to the "proper" value $\overline{\sigma^1}$, there is no guarantee that the second phase will be binding.

In Section 4, we will describe how to construct a polynomial collection of 2-phase commitment schemes for a promise problem [6] in **SZK** (resp. **ZK**):

Theorem 2.4. *For every promise problem $\Pi = (\Pi_Y, \Pi_N) \in$ **SZK** (resp. $\Pi \in$ **ZK**), there exists a polynomial-time computable function that maps an instance x of length n to $t = \mathrm{poly}(n)$ 2-phase commitment schemes $\mathrm{Com}_1, \cdots, \mathrm{Com}_t$ such that*

$$\begin{aligned} x \in \Pi_Y \quad &\Rightarrow \quad \exists i \in [t] \ \ \textit{such that } \mathrm{Com}_i \textit{ is statistically} \\ &\qquad\quad \textit{(resp. computationally) hiding} \\ x \in \Pi_N \quad &\Rightarrow \quad \forall i \in [t]\ \mathrm{Com}_i \textit{ is } \tbinom{2}{1} \textit{ statistically binding} \end{aligned}$$

3. ZERO-KNOWLEDGE PROOFS FROM 1-OUT-OF-2 BINDING COMMITMENTS

3.1 Generic Zero-knowledge Proof

It will be convenient to present our protocols based on an abstraction of standard zero-knowledge proofs for **NP**-complete problems [14, 8]. Using the standard zero-knowledge proof for Hamiltonicity [8], we may assume that every problem $\Pi \in$ **NP** has a public-coin (honest verifier) zero-knowledge proof $(P, V)(x)$ of the form:

1. P commits to ℓ bits $(b_1, b_2, \cdots, b_\ell)$, and sends the commitments to V. (In Hamiltonicity, this is a commitment to the adjacency matrix of a permuted graph)

2. V sends a challenge $c \xleftarrow{\mathrm{R}} \{0,1\}^q$. (This tells the prover whether to reveal the permutation or a cycle in the permuted graph)

3. P sends a subset of indices $U \subseteq [\ell]$ of size u, where U is determined by the challenge and the prover's coin tosses. P opens the commitments to b_i for $i \in U$. (U is either the entire graph or a cycle. By using appropriate "dummy" commitments, we can ensure that the subsets of indices are of fixed size u.)

4. V checks that "U is valid with respect to the challenge c" and that the opened commitments are valid. (The verifier will check that either these values correspond to the adjacency matrix of a permuted graph or that they correspond to a Hamiltonian cycle.)

[6] A *promise problem* [11] is specified by two disjoint sets of strings $\Pi = (\Pi_Y, \Pi_N)$, where we call Π_Y the set of yes *instances* and Π_N the set of no *instances*. Such a promise problem is associated with the following computational problem: given an input that is "promised" to lie in $\Pi_Y \cup \Pi_N$, decide whether it is in Π_Y or in Π_N

This proof system has perfect completeness. By repeating Blum's HAMILTONICITY protocol [8] in parallel, we get negligible soundness error when the commitment scheme used is perfectly binding. More generally, we can say that if $x \in \Pi_N$, then with probability $1 - \text{neg}(n)$, either the verifier rejects or the prover breaks one of the commitments. When the commitment scheme used is statistically hiding, the protocol is statistical *honest-verifier* zero-knowledge.

3.2 Zero-knowledge Proof from a Single 1-out-of-2 Binding Commitment

As a warm-up to the construction of zero-knowledge proofs based on the collection of commitments given by Theorem 2.4, we will sketch the construction based on a single $\binom{2}{1}$-binding commitment scheme.

THEOREM 3.1. *Let* $\Pi \in \mathbf{NP}$ *and* (S, R) *be a 2-phase commitment scheme. There exists an interactive protocol* (P', V') *such that:*

- *if* $x \in \Pi_Y$ *and* (S, R) *is statistically (resp. computationally) hiding, then* (P', V') *is honest-verifier statistical (resp. computational) zero-knowledge*
- *if* $x \in \Pi_N$ *and* (S, R) *is 1-out-of-2 statistically binding, then* (P', V') *is statistically sound with negligible soundness error.*

The new protocol (P', V') will consist of two sequential executions of the generic protocol (P, V). The prover will use the first phase of (S, R) in the first execution and the second phase of (S, R) in the second execution. The soundness property will rely on the fact that for each commitment, at least one phase is binding (though it might be a different phase for each commitment). Intuitively, this $\binom{2}{1}$-binding property should ensure that the prover cannot cheat in both executions.

However two difficulties arise at this point. First, the prover only opens u first phase commitments in the first execution, whereas we need ℓ second phase commitments in the second execution. Secondly, the prover only needs to break one first phase commitment to ruin the soundness property in the first execution and we cannot guarantee that the corresponding second phase commitment (known to be binding) will be opened by the prover in the second execution.

In order to manage these difficulties, in the first execution, we make the prover commit to each bit b_i a total of ℓ^2 times using the first phase of (S, R). We denote these first phase commitments $z_{i,j}$ for $i \in [\ell], j \in [\ell^2]$. Hence the prover opens more than enough first phase commitments in the first execution so that the corresponding second phase commitments can be used in the second execution.

The protocol (P', V') is honest-verifier zero-knowledge when $x \in \Pi_Y$ because both phases of the commitment scheme (S, R) are hiding and the generic protocol is honest-verifier zero-knowledge when the commitment scheme used is hiding.

Let $x \in \Pi_N$. Let us consider the soundness property for the first execution of (P, V). By the binding property of $\binom{2}{1}$-binding commitments, with $1 - \text{neg}(n)$ probability, each $z_{i,j}$ has at most one "proper" decommitment value $b^\star_{i,j}$ (i.e. other than the bit σ^1 in Definition 2.3). Consider the sequence $(b^\star_1, \cdots, b^\star_\ell)$ where $b^\star_i$ is the majority of $b^\star_{i,j}$ (over $j \in [\ell^2]$). By soundness of the generic protocol, the verifier would reject with probability $1 - \text{neg}(n)$ if the prover opens consistently with $(b^\star_1, \cdots, b^\star_\ell)$. Thus there must be an index $i^\star$ such that the prover opens inconsistently with $b_{i^\star}$, i.e. at least half of the commitments $\{z_{i^\star,1}, \cdots, z_{i^\star,\ell^2}\}$ are broken and the second phases of these commitments will be statistically binding.

Before the second execution starts, the verifier chooses a random correspondence between the first phase commitments opened in the first execution and the second phase commitments to be used in the second execution. This random "shuffling" guarantees that if the prover cheats in the first execution by breaking at least half of the commitments $\{z_{i^\star,1}, \cdots, z_{i^\star,\ell^2}\}$, then with high probability *every* bit b'_i committed to in the second execution of (P, V) will have at least one binding second phase commitment. Then, by soundness of the generic protocol, the verifier rejects in the second execution with probability $1 - \text{neg}(n)$.

3.3 Zero-knowledge Proof from a Collection of 1-out-of-2 Binding Commitments

We will now sketch how to construct a zero-knowledge proof based on the collection of commitments given by Theorem 2.4.

THEOREM 3.2. *Let* $\Pi \in \mathbf{NP}$ *and* $\text{Com}_1, \cdots, \text{Com}_t$ *be 2-phase commitment schemes (where* $\text{Com}_j = (S_j, R_j)$*). There exists an interactive protocol* (P', V') *such that:*

- *if* $x \in \Pi_Y$ *and one of the commitments* $\text{Com}_1, \cdots, \text{Com}_t$ *is statistically (resp. computationally) hiding, then* (P', V') *is honest-verifier statistical (resp. computational) zero-knowledge*
- *if* $x \in \Pi_N$ *and all commitments* $\text{Com}_1, \cdots, \text{Com}_t$ *are 1-out-of-2 statistically binding, then* (P', V') *is sound with negligible soundness error.*

The new protocol (P', V') will consist of $(t + 1)$ sequential executions of the generic protocol (P, V). In order to preserve the zero-knowledge property of the generic protocol, we need the prover's commitments in each execution to be statistically hiding. Since we are only guaranteed to have *at least one* statistically hiding commitment among $\text{Com}_1, \cdots, \text{Com}_t$ (when $x \in \Pi_Y$), we will use a secret sharing scheme for each bit that the prover must commit to in the generic protocol. Each bit b_i will be shared using t random values and the prover will commit to the jth share of b_i using Com_j. This will ensure that each unopened bit b_i is hidden from the verifier and thus that the protocol is honest-verifier zero-knowledge.

The soundness property will be proven by showing that the prover's commitments are binding in *at least one* of the executions. Similarly to the warm-up case, in each execution of (P, V), for every $j \in [t]$, the prover commits to the jth share *multiple* times using both the first and the second phases of Com_j. For every $j \in [t]$, the verifier chooses a random correspondence between the first phase commitments using Com_j opened in the rth execution ($r \in \{1, \cdots, t+1\}$) and the second phase commitments using Com_j in the remaining $(t - r + 1)$ executions. This random "shuffling" of the commitments using Com_j guarantees that if in the rth execution, the prover cheats by opening inconsistently and breaking some first phase commitments using Com_j, then with high probability, for each of the remaining $(t-r+1)$ executions, for every $i \in [\ell]$, the jth share of b_i will have at least one binding second-phase commitment. Hence *every bit b_i* committed to in the $(t+1)$st execution will be binding and by soundness of the generic protocol the verifier rejects in the $(t+1)$st execution with probability $1-\mathrm{neg}(n)$ (if it hasn't rejected in an earlier execution).

4. CONSTRUCTING 1-OUT-OF-2 BINDING COMMITMENTS

We will need the following basic notation. The *support* of a random variable X is $\mathrm{Supp}(X) = \{x : \Pr[X = x] > 0\}$. A random variable X is *flat* if it is uniform over its support. A circuit $X : \{0,1\}^m \to \{0,1\}^n$ defines a probability distribution on $\{0,1\}^n$ by evaluating X on a uniformly chosen input in $\{0,1\}^m$.

Promise problems and instance-dependent commitment schemes

Recall that a promise problem is specified by two disjoint sets of strings $\Pi = (\Pi_Y, \Pi_N)$, where we call Π_Y the set of YES *instances* and Π_N the set of NO *instances*. The *complement* of a promise problem $\Pi = (\Pi_Y, \Pi_N)$ is the promise problem $\overline{\Pi} = (\Pi_N, \Pi_Y)$.

In a (standard) *instance-dependent* commitment scheme [5, 22], the sender and receiver strategies also depend on an instance x of some problem Π. We require that the scheme is hiding if x is a YES instance of Π and binding if x is a NO instance of Π. Similarly, an *instance-dependent 2-phase* commitment scheme is a 2-phase commitment scheme where both the sender and receiver are given an auxiliary input x that is viewed as an instance of Π. It is required that the scheme is hiding when $x \in \Pi_Y$ and is $\binom{2}{1}$-binding when $x \in \Pi_N$.

To construct the polynomial collection of commitments of Theorem 2.4, we will consider the promise problems Entropy Difference (known to be **SZK**-complete [17]) and Entropy Approximation [16]. The promise problem Entropy Difference $\mathrm{ED} = (\mathrm{ED}_Y, \mathrm{ED}_N)$ is defined by

$$\begin{aligned}\mathrm{ED}_Y &= \{(X,Y) : \mathrm{H}(X) \geq \mathrm{H}(Y)+1\}\\ \mathrm{ED}_N &= \{(X,Y) : \mathrm{H}(X) \leq \mathrm{H}(Y)-1\},\end{aligned}$$

where $\mathrm{H}(\cdot)$ denotes the entropy function defined by $\mathrm{H}(X) = \mathrm{E}_{x \xleftarrow{\mathrm{R}} X}[\log(1/\Pr[X = x])]$.

The promise problem Entropy Approximation $\mathrm{EA} = (\mathrm{EA}_Y, \mathrm{EA}_N)$ is defined by

$$\begin{aligned}\mathrm{EA}_Y &= \{(X,t) : \mathrm{H}(X) \geq t+1\}\\ \mathrm{EA}_N &= \{(X,t) : \mathrm{H}(X) \leq t-1\},\end{aligned}$$

Our goal is to construct some form of an instance-dependent commitment scheme for all of **SZK**. To do so, it suffices to construct one for the **SZK**-complete problem ED. However, rather than work with ED directly, we will reduce it to simpler problems. The starting point is a Cook reduction from ED to EA given by [16]. This reduces the task to constructing instance-dependent commitment schemes for EA and its complement. Indeed, we describe an instance-dependent commitment scheme for EA in Section 4.4; in fact this is a standard 1-phase commitment scheme. However we do not know how to give an instance-dependent 2-phase commitment scheme for the complement of EA but rather for the complement of a restriction of EA, denoted by EA$'$ that we will describe later. Hence we will modify the reduction of [16] to obtain a Cook reduction from ED to EA$'$.

Interactive hashing

A key tool in our constructions of $\binom{2}{1}$-binding commitments is interactive hashing, which was introduced by Naor, Ostrovsky, Venkatesan and Yung [27] for the purposes of constructing perfectly binding commitments from one-way permutations and have been used to construct commitments in other settings as well. We only need an information-theoretic version, which we take from Ding, Harnik, Rosen and Shaltiel [10].

Roughly speaking, in such a protocol A holds an input $W \in \{0,1\}^\ell$ and at the end of protocol A and B agree on a pair (W_0, W_1) (in lexicographic order) such that $W_d = W$ for some $d \in \{0,1\}$. A secure interactive hashing protocol guarantees that B does not learn anything about the value d if A's input W is uniform, and that A cannot control both values W_0 and W_1, i.e. A cannot force both values to lie in a "small" subset $S \subseteq \{0,1\}^\ell$.

Definition 4.1. *A protocol (A,B) is called an η-*uniform interactive hashing protocol *if it is an* efficient *two-party protocol with the following properties:*

Inputs *A has an input string $W \in \{0,1\}^\ell$ and B has no input*

Outputs *A and B output a 2-to-1 hash function $h : \{0,1\}^\ell \to \{0,1\}^{\ell-1}$ and two values $W_0, W_1 \in \{0,1\}^\ell$ (in lexicographic order) such that $h(W_0) = h(W_1) = h(W)$.*

Uniformity *Let $d \in \{0,1\}$ such that $W_d = W$. Conditioned on B's view, the distribution of W_{1-d} over the parties' coin tosses is η-close to the uniform distribution over $\{0,1\}^\ell$.*

Definition 4.2. *Let D denote the distribution of the index $d \in \{0,1\}$ such that the string W_d corresponds to the input of A in the interactive hashing protocol. An interactive hashing protocol is* secure for A *if for every unbounded deterministic $B^\star$ the distributions $\{VIEW_{B^\star}(A(W), B^\star), D\}$ and $\{VIEW_{B^\star}(A(W), B^\star), U_1\}$ are identical when $W \equiv U_\ell$.*

An interactive hashing protocol is (μ, ρ)-secure for B *if for every $S \subseteq \{0,1\}^\ell$ of density at most μ and every unbounded strategy $A^\star$,* $\Pr[W_0, W_1 \in S] < \rho$

Improving on the previous protocol of [27], [10] constructed a constant-round interactive hashing protocol where the messages sent from B are the output of its coin-tosses.

Theorem 4.3. *[10] For every $0 < \mu < 1$, there exists a constant-round η-uniform interactive hashing protocol (A,B) that is secure for A and $(\mu, \mathrm{poly}(\ell)\cdot\mu)$-secure for B, where $\eta < 2^{-\ell}$.*

4.1 Instance-dependent Commitment Scheme for Entropy Approximation

Theorem 4.4. EA *has an instance-dependent commitment scheme with a constant number of rounds.*

We will describe informally an instance-dependent (standard) commitment scheme (S, R) for EA.

On input (X, t), the sender and receiver first apply the lemma given in [16] to transform X into a new distribution X' on $\{0,1\}^{n'}$ such that

$$(X,t) \in \mathrm{EA}_Y \quad \Rightarrow \quad \Delta(X', U_{n'}) \le 2^{-n}$$
$$(X,t) \in \mathrm{EA}_N \quad \Rightarrow \quad |\mathrm{Supp}(X')| \le 2^{-n} 2^{n'}$$

where $\Delta(X, Y)$ denotes the statistical difference between random variables X and Y [7].

Pre-processing: On input (X, t), the sender and receiver apply the above transformation to obtain the circuit encoding the distribution X'.

Commit phase

1. On input X', the sender S generates a random sample $x \leftarrow X'$.
2. (S, R) run the constant-round interactive hashing protocol (A, B) of with S playing $A(x)$ and R playing B. Their common output is a pair (x_0, x_1).
3. S lets $d \in \{0,1\}$ such that $x_d = x$ and sends $c = d \oplus b$.
4. The commitment z is defined as the triple (x_0, x_1, c).

Reveal phase S reveals b and the coin tosses r used to generate the sample x. R checks that $X'(r) = x_{c \oplus b}$.

Intuitively, the hiding property holds if $(X, t) \in \mathrm{EA}_Y$ because $x \overset{\mathrm{R}}{\leftarrow} X'$ is close to uniform and the interactive hashing protocol ensures that B does not learn anything about d when A's input is uniform. If $(X, t) \in \mathrm{EA}_N$, then by security of the interactive-hashing protocol, S cannot force both x_0 and x_1 to be in $\mathrm{Supp}(X')$ since $\mathrm{Supp}(X')$ is an exponentially small fraction of $\{0,1\}^{n'}$.

4.2 1-out-of-2 Binding Commitment for $\overline{\mathrm{EA'}}$

For intuition, we will first consider the construction of an instance-dependent commitment for $\overline{\mathrm{EA}}$ when X is a *flat* distribution. We denote by $\Omega_X(x)$ the set of coin tosses r that generate x.

- If $X \in \overline{\mathrm{EA}}_Y$, then $\forall x \in \mathrm{Supp}(X)$, $\Omega_X(x)$ is of size $2^{m-H(X)} \ge 2^{m-t+1}$.
- If $X \in \overline{\mathrm{EA}}_N$, then $\forall x \in \mathrm{Supp}(X)$, $\Omega_X(x)$ is of size $2^{m-H(X)} \le 2^{m-t-1}$.

This factor 4 gap (which can be amplified) between YES and NO instances of $\overline{\mathrm{EA}}$ is similar to the gap between YES and NO instances of EA we exploited in the previous section. We will therefore make the sender choose a sample $x \leftarrow X$ and apply interactive hashing to the *coin tosses* used to generate x.

Commit phase

1. S generates a sample $x \leftarrow X$ using coin tosses r and sends x. S chooses a random hash function $h : \{0,1\}^m \to \{0,1\}^{m-t}$.
2. (S, R) then run the interactive hashing protocol (A, B) with S playing $A(h, h(r))$ and R playing B. Their common output is a pair (w_0, w_1).
3. S lets $d \in \{0,1\}$ such that $w_d = w$ and sends $c = d \oplus b$.
4. The commitment z is defined as (w_0, w_1, c).

Reveal phase S reveals b and the coin tosses r used to generate the sample x. R checks that $X(r) = x$ and w_d is of the form $(h, h(r))$ and rejects if not.

Intuitively, the hiding property holds if $(X, t) \in \overline{\mathrm{EA}}_Y$ because $(h, h(r))$ is close to uniform given x (after amplifying the gap so that $|\Omega_X(x)| \ge 2^{m-t+n}$). If $(X, t) \in \overline{\mathrm{EA}}_N$, then for every x, $\Omega_X(x)$ is of small size and by security of the interactive hashing protocol S cannot force both w_0 and w_1 to be of the form $(h, h(r))$ for $r \in \Omega_X(x)$.

Let us turn to the case where the distribution X *is not* flat. Note that the binding property is violated in the above commitment scheme for $\overline{\mathrm{EA}}$ if the sample x sent by S is "too heavy", i.e. $\Omega_X(x)$ is of large size. It is known that by taking k independent copies of X, we can flatten the distribution X such that *most* strings in $\mathrm{Supp}(X)$ will not be too heavy. However flattening on its own doesn't solve the problem since there still exist some heavy strings under X and the sender can violate the binding property by choosing such a string x. Nevertheless, flattening the distribution X guarantees that there are *few* heavy strings. Hence we use another execution of interactive hashing a first time to constrain the sender's choice of x, thereby giving us the first commitment phase of our two-phase commitment scheme. We can make this idea work for the following restriction of EA.

The promise problem $\mathrm{EA'} = (\mathrm{EA'}_Y, \mathrm{EA'}_N)$ is defined by:

$$\mathrm{EA'}_Y \;=\; \{(X, t, 1^k) : H(X) \ge t + 1\}$$
$$\mathrm{EA'}_N \;=\; \{(X, t, 1^k) : t - 1/k \le H(X) \le t \;\wedge\; k \ge p(m, n)\},$$

where X is a samplable distribution specified by a circuit mapping $\{0,1\}^m \to \{0,1\}^n$, t a rational parameter, 1^k a security parameter, p is a fixed polynomial to be determined (p will emerge from our construction of a 2-phase commitment scheme for $\overline{\mathrm{EA'}}$).

Theorem 4.5. $\overline{\mathrm{EA'}}$ *has an instance-dependent 2-phase commitment scheme that is statistically hiding on YES instances and* $\binom{2}{1}$*-statistically binding on NO instances .*

Direct product of random variables.

In our construction of an instance-dependent 2-phase commitment scheme for $\overline{\mathrm{EA'}}$, we will take independent copies of a random variable hence we need to consider the behavior of entropy under direct products. If X and Y are random variables, then $X \otimes Y$ denotes the random variable obtained by taking independent random samples $x \overset{\mathrm{R}}{\leftarrow} X$ and $y \overset{\mathrm{R}}{\leftarrow} Y$ and outputting (x, y). We write $\otimes^k X$ to denote the random variable consisting of k independent copies of X.

[7] If X and Y are random variables taking values in $\mathcal{U}$, $\Delta(X, Y) \overset{\mathrm{def}}{=} \max_{S \subset \mathcal{U}} |\Pr[X \in S] - \Pr[Y \in S]| = \frac{1}{2} \sum_{x \in \mathcal{U}} |\Pr[X = x] - \Pr[Y = x]|$

For every X, Y, $\mathrm{H}(X \otimes Y) = \mathrm{H}(X) + \mathrm{H}(Y)$ and thus $\mathrm{H}(\otimes^k X) = k \cdot \mathrm{H}(X)$. Another well-known and useful feature of taking direct products is that it makes "flattens" random variables, so that probability masses become concentrated around $2^{-\mathrm{H}(X)}$. (This is known as the Asymptotic Equipartition Property in information theory; see [9].) Our formalization of it follows [17].

Definition 4.6 (heavy, light and typical elements). *Let X be a random variable taking values in a universe $\mathcal{U}$, x an element of $\mathcal{U}$, and Δ a positive real number. We say that x is* Δ-heavy *(resp.,* Δ-light*) if* $\Pr[X = x] \geq 2^{\Delta} \cdot 2^{-\mathrm{H}(X)}$ (resp., $\Pr[X = x] \leq 2^{-\Delta} \cdot 2^{-\mathrm{H}(X)}$). *Otherwise, we say that x is* Δ-typical.

A natural relaxed definition of flatness follows. The definition links the amount of slackness allowed in "typical" elements with the probability mass assigned to non-typical elements.

Definition 4.7 (nearly flat distributions). *A distribution X is called* Δ-flat *if for every $t \geq 1$ the probability that an element chosen from X is $t \cdot \Delta$-typical is at least $1 - 2^{-t^2}$.*

Lemma 4.8 (Flattening Lemma). *Let X be a distribution, k a positive integer, and $\otimes^k X$ denote the distribution composed of k independent copies of X. Suppose that for all x in the support of X it holds that $\Pr[X = x] \geq 2^{-m}$. Then $\otimes^k X$ is $\sqrt{k} \cdot m$-flat.*

Let $(X_0, t_0, 1^k)$ be an instance of $\overline{\mathrm{EA}'}$. Let $X = \otimes^k X_0$, $m = m_0 \cdot k$, $n = n_0 \cdot k$ and $t = t_0 \cdot k$. We have

$$\begin{aligned} (X_0, t_0) \in \overline{\mathrm{EA}'}_Y &\quad \Rightarrow t - 1 \leq H(X) \leq t \\ (X_0, t_0) \in \overline{\mathrm{EA}'}_N &\quad \Rightarrow H(X) \geq t + k \end{aligned}$$

By taking k independent copies of the original circuit X_0, Lemma 4.8 ensures that $X = \otimes^k X_0$ is Δ-flat for $\Delta = k^{7/12}$ (by setting $k \geq (m_0)^{12}$). The key point is that the entropy gap between the YES and NO instances is linear in k, but the deviation Δ from flatness is sublinear. Specifically, we will use the fact that $k \gg s\Delta$ for $s = k^{1/12}$.

We now describe the 2-phase commitment scheme (S, R) for $\overline{\mathrm{EA}'}$:

Pre-processing: Both the sender and verifier apply the flattening lemma to X_0 to obtain the circuit encoding a distribution X as above.

First Commit phase

1. Let r_S be the sender's coin tosses. $S_c^1(r_S)$ chooses a random hash function $h_1 : \{0,1\}^n \rightarrow \{0,1\}^{t-2s\Delta}$ and a sample x from the distribution X.
2. $(S_c^1(\sigma^1), R_c^1)$ run the constant-round interactive hashing protocol (A, B) for the setting $\mu = 2^{-k}$ with the sender playing A on input $W = (h_1, h_1(x))$ and the receiver playing B. The output of the interactive hashing protocol is a pair (W_0, W_1).
3. S_c^1 lets $d^1 \in \{0,1\}$ such that $W_{d^1} = W$ and sends $c^1 = d^1 \oplus \sigma^1$.
4. The first commitment z^1 is defined as concatenation of the transcript of the interactive hashing protocol and the triple (W_0, W_1, c^1).

First Reveal phase $S_r^1(\sigma^1, r_S, z^1)$ reveals σ^1 and the sample x used. R_r^1 checks that $W_{c^1 \oplus \sigma^1}$ is of the form $(h_1, h_1(x))$ and rejects if not. Their common output, denoted by τ, is the concatenation of the transcripts of the first commitment phase and the first reveal phase.

Second Commit phase

1. $S_c^2(\tau, r_S)$ chooses a random hash function $h_2 : \{0,1\}^m \rightarrow \{0,1\}^{m-t-2s\Delta}$. Let r be the sequence of coin tosses used by S_c^1 to generate x in the first phase.
2. $(S_c^2(\sigma^2, r_S), R_c^2)(\tau)$ run the constant-round interactive hashing protocol (A, B) for the setting $\mu = 2^{-k^{2/3}}$ with the sender playing A on input $W' = (h_2, h_2(r))$ and the receiver playing B. The output of the interactive hashing protocol is a pair (W'_0, W'_1).
3. S_c^2 lets $d^2 \in \{0,1\}$ such that $W'_{d^2} = W'$ and sends $c^2 = d^2 \oplus \sigma^2$.
4. The second commitment z^2 is defined as the concatenation of τ, the transcript of the interactive hashing protocol and the triple (W'_0, W'_1, c^2).

Second Reveal phase $S_r^2(\sigma^2, r_S, z^2)$ reveals σ^2 and the coin tosses r used to generate x. R_r^2 checks that $X(r) = x$ and that $W'_{c^2 \oplus \sigma^2}$ is of the form $(h_2, h_2(r))$ and rejects if not.

4.3 Collection of Commitment Schemes for SZK

Once we have built 2-phase commitment schemes for both EA' and $\overline{\mathrm{EA}'}$, we use the Cook reduction given below to construct a collection of 2-phase commitments as in Theorem 2.4.

Lemma 4.9. *Let (X, Y) be an instance of* ED *where the circuits have input length m and output length n. Then for every $k \geq p(m, n)$,*

$$\begin{aligned} &(X, Y) \in \mathrm{ED}_Y \\ &\quad \Rightarrow \bigvee_{i=0}^{n \cdot k} \left((Y, i/k, 1^k) \in \overline{\mathrm{EA}'}_Y \wedge \bigwedge_{i'=0}^{i} (X, i'/k, 1^k) \in \mathrm{EA}'_Y \right) \\ &(X, Y) \in \mathrm{ED}_N \\ &\quad \Rightarrow \bigwedge_{i=0}^{n \cdot k} \left((Y, i/k, 1^k) \in \overline{\mathrm{EA}'}_N \vee \bigvee_{i'=0}^{i} (X, i'/k, 1^k) \in \mathrm{EA}'_N \right) \end{aligned}$$

For each value of $i \in \{0, \cdots, nk\}$, we build a 2-phase commitment scheme Com_i by combining the commitment scheme for EA' and the 2-phase commitment scheme for $\overline{\mathrm{EA}'}$. More precisely, in each phase of Com_i, we commit to a value a total of $(i + 2)$ times by running in parallel the commitment scheme given by Theorem 4.4 for each instance of EA' and the 2-phase commitment scheme given by Theorem 4.5 for the instance of $\overline{\mathrm{EA}'}$. Therefore, we obtain a collection of $t = (nk + 1)$ 2-phase commitment schemes $\mathrm{Com}_1, \cdots, \mathrm{Com}_t$ as in Theorem 2.4.

5. REFERENCES

[1] W. Aiello and J. Håstad. Statistical zero-knowledge languages can be recognized in two rounds. *J. Comput. Syst. Sci.*, 42(3):327–345, June 1991.

[2] 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.

[3] B. Barak. How to go beyond the black-box simulation barrier. *Proc. of the 42nd FOCS, 2001*, 2001.

[4] M. Bellare and S. Goldwasser. The complexity of decision versus search. *SIAM Journal on Computing*, 23(1):97–119, Feb. 1994.

[5] M. Bellare, S. Micali, and R. Ostrovsky. Perfect zero-knowledge in constant rounds. In *Proceedings of the Twenty Second Annual ACM Symposium on Theory of Computing*, pages 482–493, 1990.

[6] M. Bellare and E. Petrank. Making zero-knowledge provers efficient. In *Proceedings of the Twenty-Fourth Annual ACM Symposium on the Theory of Computing*, pages 711–722, 1992.

[7] M. Ben-Or, O. Goldreich, S. Goldwasser, J. Håstad, J. Kilian, S. Micali, and P. Rogaway. Everything provable is provable in zero-knowledge. In *Advances in Cryptology—CRYPTO ’88*.

[8] M. Blum. How to prove a theorem so no one else can claim it. In *Proceedings of the International Congress of Mathematicians, Vol. 1, 2 (Berkeley, Calif., 1986)*, pages 1444–1451, Providence, RI, 1987.

[9] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, Inc., 2nd edition, 1991.

[10] Y. Ding, D. Harnik, A. Rosen, and R. Shaltiel. Constant-round oblivious transfer in the bounded storage model. In *Proceedings of the First Theory of Cryptography Conference*, pages 446–472, 2004.

[11] S. Even, A. L. Selman, and Y. Yacobi. The complexity of promise problems with applications to public-key cryptography. *Information and Control*, 61(2):159–173, May 1984.

[12] L. Fortnow. The complexity of perfect zero-knowledge. In S. Micali, editor, *Advances in Computing Research*, volume 5, pages 327–343. JAC Press, Inc., 1989.

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

[14] 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(1):691–729, 1991.

[15] O. Goldreich, A. Sahai, and S. Vadhan. Honest verifier statistical zero-knowledge equals general statistical zero-knowledge. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 399–408, Dallas, 23–26 May 1998.

[16] O. Goldreich, A. Sahai, and S. Vadhan. Can statistical zero-knowledge be made non-interactive?, or On the relationship of SZK and NISZK. In *Advances in Cryptology—CRYPTO ’99*.

[17] O. Goldreich and S. Vadhan. Comparing entropies in statistical zero-knowledge with applications to the structure of SZK. In *Proceedings of the Fourteenth Annual IEEE Conference on Computational Complexity*, pages 54–73, 1999.

[18] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, February 1989.

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

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

[21] R. Impagliazzo and M. Yung. Direct minimum-knowledge computations. In *Advances in Cryptology—CRYPTO ’87*, pages 40–51.

[22] T. Itoh, Y. Ohta, and H. Shizuya. A language-dependent cryptographic primitive. *Journal of Cryptology*, 10(1):37–49, 1997.

[23] A. R. Klivans and D. van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. In *Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing*, pages 659–667, 1999.

[24] D. Micciancio and S. Vadhan. Statistical zero-knowledge proofs with efficient provers: lattice problems and more. In D. Boneh, editor, *Advances in Cryptology—CRYPTO ‘03*, pages 282–298.

[25] P. B. Miltersen and N. V. Vinodchandran. Derandomizing arthur-merlin games using hitting sets. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science (FOCS ‘99)*, pages 71–80, 1999.

[26] M. Naor. Bit commitment using pseudorandomness. *Journal of Cryptology*, 4(2):151–158, 1991.

[27] M. Naor, R. Ostrovsky, R. Venkatesan, and M. Yung. Perfect zero-knowledge arguments for np using any one-way permutation. *Journal of Cryptology*, 11(2):87–108, 1998.

[28] T. Okamoto. On relationships between statistical zero-knowledge proofs. *Journal of Computer and System Sciences*, 60(1):47–108, February 2000.

[29] R. Ostrovsky. One-way functions, hard on average problems, and statistical zero-knowledge proofs. In *Proceedings of the Sixth Annual Structure in Complexity Theory Conference*, pages 133–138, 1991.

[30] R. Ostrovsky and A. Wigderson. One-way functions are essential for non-trivial zero-knowledge. In *Proceedings of the Second Israel Symposium on Theory of Computing and Systems*, 1993.

[31] A. Sahai and S. Vadhan. A complete problem for statistical zero knowledge. *Journal of the ACM*, 50(2):196–249, March 2003.

[32] S. P. Vadhan. *A Study of Statistical Zero-Knowledge Proofs*. PhD thesis, Massachusetts Institute of Technology, August 1999.

[33] S. P. Vadhan. On transformations of interactive proofs that preserve the prover’s complexity. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing*, pages 200–207, Portland, OR, May 2000.

[34] S. P. Vadhan. An unconditional study of computational zero knowledge. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS ‘04)*, pages 176–185.