A Complete Promise Problem for Statistical Zero-Knowledge*

Amit Sahai[†]
Department of Computer Science
Massachusetts Institute of Technology
Cambridge, MA 02139
amits@theory.lcs.mit.edu

Salil P. Vadhan[†]
Department of Mathematics
Massachusetts Institute of Technology
Cambridge, MA 02139
salil@math.mit.edu

Abstract

We present a complete promise problem for SZK, *the class of languages possessing statistical zero-knowledge proofs (against an honest verifier). The problem is to decide whether two efficiently samplable distributions are either statistically close or far apart. This characterizes* SZK *with* no reference to interaction or zero-knowledge. *From this theorem and its proof, we are able to establish several other results about* SZK, *knowledge complexity, and efficiently samplable distributions.*

1 Introduction

A revolution in theoretical computer science occurred when it was discovered that NP has complete problems [11, 24, 23]. Most often, this theorem and other completeness results are viewed as negative statements, as they provide evidence of a problem's intractability. These same results, viewed as positive statements, enable one to study an entire class of problems by focusing on a single problem. For example, all languages in NP were shown to have computational zero-knowledge proofs when such a proof was exhibited for GRAPH 3-COLORABILITY [19]. Similarly, the result that IP = PSPACE was shown by giving an interactive proof for QUANTIFIED BOOLEAN FORMULA, which is complete for PSPACE [25, 30]. More recently, the celebrated PCP theorem characterizing NP was proven by designing efficient probabilistically checkable proofs for a specific NP-complete language [3, 4].

In this paper, we present a complete promise problem[1] for SZK, the class of languages possessing statistical zero-knowledge proofs against an honest verifier. For traditional complexity classes, such as NP and PSPACE, the construction of natural complete problems has become a routine task. However, it may come as a surprise that SZK, which is defined in terms of two interacting machines and a simulator, has a complete problem which makes *no reference to interaction or zero-knowledge*. We use this complete problem not as a negative tool, but as a positive tool to derive general results about the entire class. Our hope is that this new characterization will also aid and simplify further research in this area.

Statistical zero-knowledge. Informally, an interactive proof is a protocol in which a computationally unbounded prover P attempts to convince a probabilistic polynomial-time verifier V of an assertion, *i.e.* that a string x is in a language L. Following the framework laid out by Goldwasser, Micali, and Rackoff [22], we consider two probability distributions in defining zero-knowledge:

1. The interaction of P and V from V's point of view.
2. The output of a probabilistic polynomial-time machine not interacting with anyone, called the *simulator*, on input x.

We say an interactive proof system (P, V) for L is *zero-knowledge* against V (the *honest verifier*), if for every input x in L, the two distributions above are "alike." Intuitively, the verifier gains no knowledge by interacting with the prover except that $x \in L$, since it could have run the simulator instead. The specific variants of zero-knowledge differ by the interpretation given to "alike." The most strict interpretation, leading to *perfect zero-knowledge*, requires that the distributions be identical. A slightly relaxed interpretation, leading to *statistical zero-knowledge*, requires that the distributions have negligible statistical deviation from one another. The most liberal interpretation, leading to *computational zero-knowledge*, requires that samples from the two distributions be indistinguishable by any polynomial-time machine. This variant is not pursued in this paper. We focus on the class of languages possessing *statistical* zero-knowledge proof systems against an honest verifier, which we denote SZK. Usually one wants the zero-knowledge condition to hold for all (even dishonest) polynomial-time verifiers. Our results translate to this more

*A more detailed version of this paper will be available from `http://www-math.mit.edu/~salil`.

[†] Supported by DOD/NDSEG Graduate Fellowships.

[1] See Section 2 for the definition of a promise problem.

0272-5428/97 $10.00 © 1997 IEEE

general setting under cryptographic assumptions such as the existence of one-way functions [8, 27, 12, 14, 26].

SZK contains a number of important problems, including GRAPH NONISOMORPHISM [19], a problem which is not known to be in NP. It also contains problems with cryptographic application and significance that are believed to be hard on average [22, 18]. At the same time, the statistical zero-knowledge property has several strong consequences. Unlike a computational zero-knowledge protocol, a statistical zero-knowledge protocol remains zero-knowledge even against a computationally unbounded verifier. In addition, a language which has a statistical zero-knowledge proof must lie low in the polynomial-time hierarchy. In fact, such a language cannot be NP-complete unless the polynomial-time hierarchy collapses [16, 2, 10]. Because SZK contains problems believed to be hard yet cannot contain NP-complete problems, it holds an intriguing position in complexity theory.

The complete problem. The promise problem we show to be complete for SZK is STATISTICAL DIFFERENCE. An instance of STATISTICAL DIFFERENCE consists of a pair of circuits, each of which defines a probability distribution on strings by selecting the input uniformly at random and taking the output. The problem is to decide whether the distributions defined by the two circuits are statistically close or far apart. The gap between 'close' and 'far apart' is what makes it a promise problem and not just a language.

A key ingredient in our proof that all languages in SZK reduce to STATISTICAL DIFFERENCE is a powerful theorem of Okamoto [26], which states that all languages in SZK have public coin, also known as Arthur-Merlin [5], statistical zero-knowledge proofs. In the same paper, Okamoto proves a second theorem, showing that SZK is closed under complementation. Our techniques provide a simpler proof of this second theorem.

From the formal description of our complete problem, it will follow immediately that SZK is a natural generalization of BPP. In the definition of STATISTICAL DIFFERENCE, the circuits can output strings of any length. If we restrict the circuits to have output of logarithmic length, the resulting problem is easily shown to be complete for BPP.

In order to show that STATISTICAL DIFFERENCE is in SZK, we give a simple two-round statistical zero-knowledge proof system for it, implying that every problem in SZK has such a proof system. In our protocol, the verifier flips a coin to select one of the two distributions, and sends the prover a sample from the chosen distribution. The prover then attempts to guess from which distribution the sample came, and the verifier accepts if the prover guesses correctly. Thus, in the entire interaction, the prover sends only *a single bit* to the verifier. We will show that, when the two distributions are statistically far apart, this protocol can be simulated by a polynomial-time simulator with *exponentially small* statistical deviation. Thus, every problem in SZK has a protocol which can be simulated with exponentially small deviation. Moreover, we will show that the simulator deviation can be made exponentially small in a security parameter which can be much larger than the length of the assertion being proved. This is in contrast to the definition of SZK, which only requires that the verifier be able simulate the interaction with statistical deviation that is a *negligible*[2] function of the *assertion length.*

Additional consequences. One interesting result that comes out of our argument is that for every pair of efficiently samplable distributions, we can construct another pair of efficiently samplable distributions such that when the former are statistically close, the latter are statistically far apart, and when the former are far apart, the latter are close.

Our work also has some consequences for knowledge complexity [22, 21]. Knowledge complexity seeks to measure how much knowledge a polynomial-time verifier gains from an interactive proof. Loosely speaking, the definitions of knowledge complexity measure the "amount of help" a verifier needs to generate a distribution that is statistically close to its real interaction with the prover. There are several ways of formalizing the "amount of help" the verifier needs and each leads to a different notion of knowledge complexity. We show that for the weakest of these variants, knowledge complexity collapses by logarithmic additive factors at all levels, and in particular, knowledge complexity $\log n$ equals statistical zero-knowledge. No collapse was previously known for any of the variants of knowledge complexity suggested in [21].

As with zero-knowledge, *perfect* knowledge complexity can also be defined. In this model, we measure the number of bits of help the verifier needs to simulate the interaction *exactly*, rather than statistically closely. Using our complete problem for SZK, we are able to give tighter bounds on the perfect knowledge complexity of statistical zero-knowledge, as studied previously in [1].

To summarize informally, our paper

- Introduces a simple complete promise problem for statistical zero-knowledge,
- Exhibits a simple 2-round statistical zero-knowledge protocol for that problem and thus for all of SZK,
- Gives a simpler proof for Okamoto's second theorem in [26], showing that SZK is closed under complementation,

[2] Recall that a function $f(n)$ is negligible if for any polynomial $p(n)$, $f(n) < 1/p(n)$ for sufficiently large n.

- Shows that SZK naturally generalizes BPP,
- Demonstrates that one can efficiently transform a pair of samplable distributions to invert their statistical relationship,
- Proves that there is a partial collapse in one of the knowledge complexity hierarchies, and
- Establishes tighter bounds on the perfect knowledge complexity of SZK.

Related work. The complexity of statistical zero-knowledge was first considered by Fortnow [16], who showed that SZK $\subset$ co-AM. Aiello and Hastad [2] continued Fortnow's work, showing that SZK $\subset$ AM. De Santis et. al [15] and Damgard and Cramer [13] studied monotone boolean closure properties of SZK. Okamoto [26] first proved closure under complementation and showed that every language in SZK has a public-coin proof system.

Relationships between the various types of knowledge complexity were first explored by Goldreich and Petrank [21]. The computational complexity of languages with low knowledge complexity was studied in [9, 20, 29]. The relationship between perfect and statistical knowledge complexity was explored in [20, 1].

2 Notation and definitions

The problem we prove to be complete for SZK is not a language, but is a *promise problem.* Formally, a promise problem Π consists of two disjoint sets of strings Π_Y and Π_N, where Π_Y corresponds to "yes" instances and Π_N corresponds to "no" instances. Thus it is "promised" that only inputs from $\Pi_Y \cup \Pi_N$ will appear. The *complement* of Π is the promise problem $\overline{\Pi}$, where $\overline{\Pi}_Y = \Pi_N$ and $\overline{\Pi}_N = \Pi_Y$. Note that languages are special cases of promise problems.

Let X and Y be discrete random variables on probability spaces Ω and Γ, respectively. We denote the *statistical difference* between X and Y by $||X - Y||$. Several basic facts about this metric are stated in Appendix A. We write $X \otimes Y$ for the random variable on $\Omega \times \Gamma$ that takes value $(X(\omega), Y(\gamma))$ on sample point (ω, γ). *i.e.* $X \otimes Y$ is a sample of X followed by an independent sample of Y. For any positive integer k, $\otimes^k X$ is the random variable on Ω^k that takes value $(X(\omega_1), X(\omega_2), \ldots, X(\omega_k))$ on sample point $(\omega_1, \ldots, \omega_k)$. If $\Omega = \Gamma$, we denote by (X, Y) the random variable on Ω that takes value $(X(\omega), Y(\omega))$ on sample point ω in Ω.

In this paper, we will consider probability distributions defined both by circuits and probabilistic Turing machines. If A is a probabilistic Turing machine, we use $A(x)$ to denote the output distribution of A on input x. If C is a circuit mapping m-bit strings to n-bit strings, then choosing the input to C uniformly at random from $\{0,1\}^m$ defines a probability distribution on n-bit strings. By abuse of notation, we also denote this probability distribution by C.

Before defining zero-knowledge, we need to introduce some more terminology. A *PPT* algorithm is a probabilistic algorithm which runs in *strict* polynomial time. A function $f(n)$ is *negligible* if for all polynomials $p(n)$, $f(n) \leq \frac{1}{p(n)}$ for all sufficiently large n.

We follow [22] and [17] in defining zero-knowledge. For an interactive protocol (P, V), we let $\text{View}_{P,V}$ be a random variable describing the random coins of V and the messages exchanged between P and V during their interaction on input x. A language L is said to have a *statistical zero-knowledge* proof system (for the honest verifier) with *completeness error* $c(n)$ and *soundness error* $s(n)$ if there exists a PPT verifier V, a PPT simulator S, a prover P, and a negligible function α such that

1. If $x \in L$, then
   $\Pr[(P, V)(x) = \texttt{accept}] \geq 1 - c(|x|)$.
2. If $x \notin L$, then for all P^*,
   $\Pr[(P^*, V)(x) = \texttt{accept}] \leq s(|x|)$.
3. If $x \in L$, then $||S(x) - \text{View}_{P,V}(x)|| \leq \alpha(|x|)$.

A *perfect zero-knowledge proof system* is defined in the same way, except that the third condition is replaced by $||S(x) - \text{View}_{P,V}(x)|| = 0$, where S is allowed to output '$\texttt{fail}$' with probability at most $1/2$ and $S(x)$ denotes the conditional distribution of S given that $S(x) \neq \texttt{fail}$. We let SZK (resp. PZK) denote the class of languages with statistical (resp. perfect) zero-knowledge proof systems with $c(n) = s(n) = 1/3$. As long as $1 - c(n)$ is larger than $s(n)$ by an inverse polynomial, the class of languages having statistical or perfect zero-knowledge proofs does not change [17]. The negligible function α is termed the *simulator deviation.*

An interactive proof system is said to be *public coin* if on every input, the verifier's random coins r can be written as a concatenation of strings $r_1 r_2 \cdots r_l$ such that the i'th message sent from the verifier to the prover is simply r_i.

Observe that like [17], we work with the variant of zero-knowledge in which the simulator is required to run in strict polynomial time, with some probability of failure in the perfect case. The original definition in [22] allows the simulator to run in expected polynomial time, but with zero probability of failure. Our choice is not very restrictive, because we are only discussing honest-verifier statistical zero-knowledge and we know of no particular language which requires an expected polynomial time simulator for the honest verifier. In addition, our techniques can be used to prove that expected polynomial time simulators and strict polynomial time simulators are actually *equivalent* for public-coin statistical zero-knowledge proofs against an honest verifier. The details of this transformation will be discussed in the full version of the paper.

3 Results and discussion

3.1 The complete problem

The main aim of this paper is to demonstrate that SZK consists exactly of the problems that involve deciding whether two efficiently samplable distributions are either far apart or close together. This can be formally described as the following promise problem STATISTICAL DIFFERENCE (abbreviated SD):

$$\begin{aligned} \mathrm{SD}_Y &= \left\{ (C_0, C_1) : \|C_0 - C_1\| > \frac{2}{3} \right\} \\ \mathrm{SD}_N &= \left\{ (C_0, C_1) : \|C_0 - C_1\| < \frac{1}{3} \right\} \end{aligned}$$

In the above definition, C_0 and C_1 are circuits; these define probability distributions as discussed in Section 2. The thresholds of $1/3$ and $2/3$ in this definition are not arbitrary; it is important for the Amplification Lemma of Section 3.2 that $(2/3)^2 > 1/3$.

We wish to prove that SD is "complete" for SZK. In order for this to make sense, we extend SZK to promise problems in the natural way, as previously done by Goldreich and Kushilevitz [18]. That is, we require the completeness and zero-knowledge conditions to hold for inputs in the Y set, and we require the soundness condition to hold for inputs in the N set. For the sake of elegance, we also call this extension SZK. This extension and an alternative notion of completeness is discussed in more detail in the full version of the paper.

We can now state the main theorem of the paper.

Theorem 1 SD *is complete for* SZK.

The most striking thing about Theorem 1 is that it characterizes statistical zero-knowledge *with no reference to interaction.* Future investigation of the properties of SZK as a class can focus on the single problem SD, instead of dealing with complicated protocols and arbitrary languages.

We emphasize that the importance of this result lies in the specific complete problem we present and not simply the *existence* of a complete promise problem. It is fairly straightforward to construct a complete promise problem for PZK involving descriptions of Turing machines for the verifier and simulator. This construction has been extended to SZK by [7], using a result of Bellare [6] on negligible functions. However, in contrast to SD, a complete problem constructed in this manner is essentially restatement of the definition of the class and therefore does not simplify the study of the class at all.

The proof of Theorem 1 will come in Sections 3.3 and 3.4 via two lemmas and a theorem of Okamoto [26]. But first, we observe that a statement analogous to Theorem 1 can be made for BPP, if we generalize BPP to promise problems in the obvious way. The proof is omitted in this abstract.

Proposition 2 *If* SD$'$ *is the promise problem obtained by modifying the definition of* SD *so that* C_0 *and* C_1 *only have 1 bit of output, then* SD$'$ *is complete for* BPP.

Proposition 2 remains true even if we allow C_0 and C_1 to output strings of logarithmic length. Other classes such as P and RP can be obtained by modifying the definition of SD in a similar fashion. This demonstrates that SZK is a natural generalization of these well-known classes.

3.2 An amplification lemma

Fact A.4 gives an efficient technique for increasing the statistical difference between two distributions. The following lemma provides a complementary technique which decreases the statistical difference. Combining these two techniques enables us to amplify the thresholds of $1/3$ and $2/3$ in the definition of SD to δ and $1-\delta$ in time polynomial in $\log \delta^{-1}$.

Lemma 3 *There is a polynomial-time computable function that maps a triple* $(C_0, C_1, 1^k)$*, where* C_0 *and* C_1 *are circuits, to a pair of circuits* (D_0, D_1) *such that* $\|D_0 - D_1\| = \|C_0 - C_1\|^k$.

Proof: For each $b \in \{0,1\}$, let D_b do the following: Randomly choose $(b_1, \ldots, b_k) \in \{0,1\}^k$ such that $b_1 \oplus \cdots \oplus b_k = b$. Output a sample of $C_{b_1} \otimes \cdots \otimes C_{b_k}$. It is easily verified that $\|D_0 - D_1\| = \|C_0 - C_1\|^k$. This construction is a generalization of the technique used in [15] to represent the logical AND of statements about GRAPH NONISOMORPHISM. ■

Combining the constructions of Lemma 3 and Facts A.4 and A.3, we obtain the following amplification lemma.[3] The details are omitted in this abstract.

Lemma 4 (Amplification Lemma) *There is a polynomial-time computable function* **Amplify** *that takes a triple* $(C_0, C_1, 1^k)$*, where* C_0 *and* C_1 *are circuits, and outputs a pair of circuits* (D_0, D_1) *such that*

$$\begin{aligned} \|C_0 - C_1\| < 1/3 &\Rightarrow \|D_0 - D_1\| < 2^{-k} \\ \|C_0 - C_1\| > 2/3 &\Rightarrow \|D_0 - D_1\| > 1 - 2^{-k} \end{aligned}$$

[3] The use of the Amplification Lemma in the rest of this paper could be avoided by modifying the definition of SD_N to have a threshold of $1/|(C_0, C_1)|$. For details, see the preliminary version of this paper, available from http://www-math.mit.edu/~salil.

3.3 A protocol for Statistical Difference

In this section, we show that SD has a simple two-round statistical zero-knowledge protocol, similar to the standard ones for Quadratic Nonresiduosity [22] and Graph Nonisomorphism [19]. Intuitively, if two distributions are statistically far apart, then, when given a random sample from one of the distributions, the prover should have a good chance of guessing which distribution it came from. However, if the two distributions are statistically very close, a prover should not have much better than a 50% chance of guessing correctly. More formally, we have the following 2-round private-coin protocol for SD:

Protocol π

1. V, P: Compute $(D_0, D_1) = \mathbf{Amplify}(C_0, C_1, 1^{n+1})$, where $n = |(C_0, C_1)|$.
2. V: Flip one random coin $r \in \{0, 1\}$. Let z be a sample of D_r. Send z to P.
3. P: If $\Pr[D_0 = z] > \Pr[D_1 = z]$, answer 0, otherwise answer 1.
4. V: Accept if P's answer equals r, reject otherwise.

We establish the following lemma, whose proof is omitted in this abstract.

Lemma 5 *The above is a statistical zero-knowledge protocol for* SD, *with soundness error* $\frac{1}{2} + 2^{-n}$, *and completeness error and simulator deviation both* 2^{-n}. *Thus* SD $\in$ SZK.

Observe that by using a security parameter k rather than n in the call to **Amplify**, both the completeness error and simulator deviation can be reduced to 2^{-k}. Hence all languages in SZK have protocols that can be security-parametrized in this manner. Contrast this with the original definition of SZK [22], which only requires that the simulator deviation vanish as an *negligible* function of the *input length.*

This also demonstrates that SZK is closed under polynomial-time many-one reductions, *i.e.* if A reduces to B and $B \in$ SZK, then $A \in$ SZK: To prove that $x \in A$, apply the reduction to x to obtain y and execute the B-protocol with security parameter $|x|$. The security parameter is essential because an arbitrary reduction could potentially shrink string lengths dramatically, but we want the simulator deviation to be negligible as a function of $|x|$, not $|y|$.

Since the above protocol is nearly identical to the one for Graph Nonisomorphism, ideas useful for that protocol can be applied to SD and thereby all of SZK. For example, [15] gives SZK proofs for all monotone boolean formulae whose atoms are statements about membership in Graph Nonisomorphism. Their techniques generalize readily to SD. Using this in conjunction with Theorem 1 and Corollary 9, we can obtain SZK proofs for all (not necessarily monotone) boolean formulae whose atoms are statements about membership in any SZK language.

3.4 SZK-hardness of SD

The other major lemma we prove to show that SD is complete for SZK follows:

Lemma 6 *Suppose promise problem* Π *has a public coin statistical zero-knowledge proof. Then there exist PPT's* A *and* B *and a negligible function* α *such that*

$$\begin{aligned} x \in \Pi_Y \quad &\Rightarrow \quad \|A(x) - B(x)\| \leq \alpha(|x|), \textit{ and} \\ x \in \Pi_N \quad &\Rightarrow \quad \|A(x) - B(x)\| \geq 1 - 2^{-\Omega(|x|)}. \end{aligned}$$

We defer the proof of this Lemma to Section 4. We first observe how this lemma gives a reduction to SD for problems with public coin statistical zero-knowledge proofs.

Corollary 7 *Suppose promise problem* Π *has a public coin statistical zero-knowledge protocol. Then* Π *is polynomial-time reducible to* $\overline{\text{SD}}$. *(Equivalently,* $\overline{\Pi}$ *is polynomial-time reducible to* SD.*)*

Proof: First apply Lemma 6 to produce A and B, with $p(|x|)$ being a polynomial bound on the running times of $A(x)$ and $B(x)$. Given a string x, we can, by standard techniques,[4] produce in polynomial time circuits C_0 and C_1 describing the computation of A and B, respectively, on x for $p(|x|)$ steps. The inputs to C_0 and C_1 are the first $p(|x|)$ bits on the random tapes of A and B and the outputs are the first $p(|x|)$ positions on the output tapes. Then $\|C_0 - C_1\| = \|A(x) - B(x)\|$, which is at most $\alpha(|x|) < 1/3$ if $x \in \Pi$ and at least $1 - 2^{-|x|} > 2/3$ if $x \notin \Pi$ (for all sufficiently long x). So $x \mapsto (C_0, C_1)$ is a reduction from Π to $\overline{\text{SD}}$ which works for all but finitely many x. ■

The final ingredient in the proof of Theorem 1 is a theorem of Okamoto [26], which we state in terms of promise problems.[5]

Theorem 8 ([26, Thm. 1]) *If a promise problem* Π *has a statistical zero-knowledge proof, then* Π *has a public coin statistical zero-knowledge proof.*

We now show how to use Corollary 7 and Theorem 8 to obtain a simpler proof of Okamoto's Theorem 2 in [26].

[4] See, for example, [28, Thms. 8.1 and 8.2].

[5] Okamoto stated his result in terms of languages. Extending it to promise problems will be discussed in the full version of this paper.

Corollary 9 ([26, Thm. 2]) SZK *is closed under complement, even for promise problems.*

Proof: Suppose $\Pi \in$ SZK. Then by Theorem 8, Π has a public-coin statistical zero-knowledge proof system. Corollary 7 then tells us that $\overline{\Pi}$ reduces to SD, and hence $\overline{\Pi} \in$ SZK. ■

Now it will be easy to show that SD is complete for SZK.

Proof (of Theorem 1): Lemma 5 tells us that SD $\in$ SZK, so we only need to show that every problem in SZK reduces to SD. Suppose $\Pi \in$ SZK. By Corollary 9, $\overline{\Pi} \in$ SZK. Theorem 8 tells us that $\overline{\Pi}$ has a public coin statistical zero-knowledge proof system. We conclude that Π reduces to SD, by Corollary 7. ■

By Theorem 1 and Corollary 9, we see that $\overline{\text{SD}}$ reduces to SD. This is equivalent to the following surprising result:

Proposition 10 *There is a polynomial-time computable function that maps pairs of circuits* (C_0, C_1) *to pairs of circuits* (D_0, D_1) *such that*

$$\|C_0 - C_1\| < 1/3 \quad \Rightarrow \quad \|D_0 - D_1\| > 2/3$$
$$\|C_0 - C_1\| > 2/3 \quad \Rightarrow \quad \|D_0 - D_1\| < 1/3$$

It would be interesting to describe such a transformation explicitly.

3.5 Consequences for knowledge complexity

Due to space constraints, we only give terse definitions of the variants of knowledge complexity we consider. All of the following definitions of knowledge complexity for interactive proof systems come from [21], except for the last which comes from [1].

- **Hint sense:** We say that (P, V) has perfect (resp., statistical) knowledge complexity $k(n)$ in the *hint* sense if there exists a PPT simulator S and a hint function $h: L \rightarrow \{0,1\}^*$ such that $|h(x)| = k(|x|)$ and $\|S(x, h(x)) - \text{View}_{P,V}(x)\|$ is 0 (resp., is bounded by a negligible function of $|x|$.)

- **Strict oracle sense:** (P, V) is said to have perfect (resp., statistical) knowledge complexity $k(n)$ in the *strict oracle* sense if there exists a PPT oracle-machine S and an oracle $\mathcal{O}$ such that on every input x, S queries $\mathcal{O}$ at most $k(|x|)$ times and $\|S^{\mathcal{O}}(x) - \text{View}_{P,V}(x)\|$ is 0 (resp., is bounded by a negligible function of $|x|$.)

- **Average oracle sense:** (P, V) has perfect (resp., statistical) knowledge complexity $k(n)$ in the *average oracle* sense if there exists a PPT oracle-machine S and an oracle $\mathcal{O}$ such that for every input x, the average number of queries S makes to $\mathcal{O}$ is at most $k(|x|)$ and $\|S^{\mathcal{O}}(x) - \text{View}_{P,V}(x)\|$ is 0 (resp., is bounded by a negligible function of $|x|$.)

- **Entropy sense:** (P, V) has perfect (resp., statistical) knowledge complexity $k(n)$ in the *entropy* sense if there exists a PPT oracle-machine S, an oracle $\mathcal{O}$, and a PPT oracle-simulator A such that for all x, $\text{E}_R[\log P_x(R)^{-1}] \leq k(|x|)$, where $P_x(R) = \Pr_\rho[A(x, R; \rho) = S^{\mathcal{O}}(x; R)]$ and $\|S^{\mathcal{O}}(x) - \text{View}_{P,V}(x)\|$ is 0 (resp., is bounded by a negligible function of $|x|$). Here, the notation $M(y; r)$ denotes the output of PPT M on input y and random coins r,

We say that the *knowledge complexity* (in some specified sense) of a language L is $k(n)$ if there exists an interactive proof system (P, V) for L achieving negligible error probablity in both the completeness and soundness conditions such that the knowledge complexity of (P, V) is $k(n)$. The class of languages possessing perfect knowledge complexity $k(n)$ in the hint, strict oracle, average oracle, and entropy senses are denoted by PKC_{hint}, $\text{PKC}_{\text{strict}}$, PKC_{avg}, and PKC_{ent}, respectively. Statistical knowledge complexity is denoted by SKC with the appropriate subscript.

Our first result about knowledge complexity is that the SKC_{hint} hierarchy collapses by logarithmic additive factors. Previously, Goldreich and Petrank [21] have shown that $\text{SKC}_{\text{hint}}(\text{poly}(n)) \subset$ AM and $\text{SKC}_{\text{hint}}(O(\log(n))) \subset$ co-AM; the second of these results can be derived immediately from our result and Fortnow's theorem [16] that SZK $\subset$ co-AM.

Theorem 11 *For any polynomially bounded function* $k(n)$,

$$\text{SKC}_{\text{hint}}(k(n) + \log n) = \text{SKC}_{\text{hint}}(k(n)).$$

For intuition, consider the case that $k(n) = 0$. Loosely speaking, if the verifier is given the hint along with the input, then the original language becomes a statistical zero-knowledge promise problem, so we can apply the results of the previous section. Since statistical zero-knowledge is easily seen to be closed under (polynomially bounded) intersection, closure under union follows from the closure under complementation given by Corollary 9. Thus, if we take the "union over all hints," we obtain a statistical zero-knowledge problem, which is easily seen to be the original language. The formal proof is omitted in this abstract.

The next theorem establishes tighter bounds on the perfect knowledge complexity of SZK. Aiello, Bellare, and Venkatesan [1] have previously demonstrated that every language in SZK has perfect knowledge complexity $n^{-\omega(1)}$ (resp., $1 + n^{-\omega(1)}$) in the entropy (resp. average oracle) sense. Our results improve on these bounds, although the results of [1] also apply to cheating-verifier classes and ours do not. Goldreich, Ostrovsky, and Petrank [20] show that SZK has logarithmic perfect knowledge complexity in the *oracle sense*, so our results are incomparable to theirs. Our result for the strict oracle sense is the first that we know of.

Theorem 12 [6]

1. *For every polynomial-time computable* $f(n) = \omega(\log n)$, $\mathsf{SZK} \subset \mathsf{PKC_{strict}}(f(n))$.
2. $\mathsf{SZK} \subset \mathsf{PKC_{avg}}(1 + 2^{-\Omega(n)})$.
3. $\mathsf{SZK} = \mathsf{PKC_{ent}}(2^{-\Omega(n)})$.

Theorem 1 tells us that every language in SZK has a simple two-round proof system like Protocol π of Section 3.3. Thus, in order to measure the perfect knowledge complexity of SZK and prove Theorem 12, it suffices to analyze this protocol. Intuitively, since the prover is only sending the verifier one bit and this bit is almost always a value the verifier knows, the knowledge complexity of this protocol should be extremely small. However, this argument does not suffice, because the knowledge complexity of a language L is determined only by proof systems for L which achieve *negligible* error probability in both the completeness and soundness conditions. We can overcome this difficulty by performing $\omega(\log n)$ parallel repetitions. The formal proof is omitted.

4 Proof of Lemma 6

For the sake of clarity, we give the proof for languages. The proof for promise problems is nearly identical. The constructions in this lemma and protocol π are carried out for the specific example of Graph Isomorphism in the full version of the paper.

Intuition. Recall that we wish to construct a pair of probabilistic polynomial-time machines A and B such that if $x \in L$, the distributions $A(x)$ and $B(x)$ are statistically very close, but when $x \notin L$, $A(x)$ and $B(x)$ are far apart. We are given that L has a *public-coin* statistical zero-knowledge proof system. A natural place to search for such distributions is in the output of the simulator for this proof system. We think of the simulator as describing the moves of a *virtual prover* and a *virtual verifier*.[7]. We wish to find properties of the simulator's output that (1) distinguish the case $x \in L$ from $x \notin L$, and (2) are captured by the statistical difference of samplable distributions. In the case that $x \in L$, we have strong guarantees on the simulator's output. Namely, it outputs accepting conversations with high probability and its output distribution is statistically very close to the real interaction. When $x \notin L$, there are two cases. If the simulator outputs accepting conversations with low probability, this easily distinguishes it from the simulator output when $x \in L$. However, it is possible that the simulator will output accepting conversations with high probability even when $x \notin L$. This means that the virtual prover is doing quite well in fooling the virtual verifier. This naturally suggest a strategy for a real prover — imitate the virtual prover's behavior. Such a prover, called a *simulation-based prover*, was introduced by Fortnow [16] and is a crucial construct in our proof. The soundness of the proof system tells us that the simulation-based prover cannot hope to convince the real verifier with high probability. There must be a reason for this discrepancy between the success rates of the virtual prover and the simulation-based prover. One possibility is that the virtual verifier's coins in the simulator's output are *far from uniform*, so that the simulation only captures a small fraction of possible verifier states. However, this is not the only difficulty. The responses of the virtual prover may depend on future coins of the virtual verifier, which is impossible in a real public-coin interaction. Note that this is equivalent to the virtual verifier's coins being *dependent on previous messages* of the virtual prover. We will show that these are the only two obstacles the simulation-based prover faces in trying to fool the verifier, and thus they must be present when $x \notin L$. In the case that $x \in L$, however, these difficulties cannot arise since we are guaranteed that the simulator output distribution is very close to that of the real interation. If we could measure the extent to which these anomalies are present by the statistical difference between samplable distributions, we would achieve our objective. This is precisely what we do.

Notation. Let (P, V) be a public coin interactive proof system for a language L which is statistically zero-knowledge against V and let S be a simulator for this proof system. Without loss of generality, it may be assumed that the interaction of P and V on input x always has $2r(|x|)$ exchanged messages, with V sending the first message and each message consisting of exactly $q(|x|)$ bits, for some polynomials q and r. Moreover, it may be assumed that S's output always consists of $2r(|x|)$ strings of length $q(|x|)$. The output of S and the conversation between P and V on input x will be written in the form $S(x) = (c_1, p_1, \ldots, c_r, p_r)_S$ and $(P,V)(x) = (c_1, p_1, \ldots, c_r, p_r)_{(P,V)}$, respectively, where

[6] The $2^{-\Omega(n)}$ in these results can be improved to $2^{-\Omega(n^k)}$ for any constant k by amplifying with security parameter n^k instead of $n + 1$ in Protocol π of Section 3.3.

[7] This terminology is taken from [2]. The cases we consider are quite similar to those analyzed in [16, 2] Because we focus on public coin proofs, many complications that other researchers faced do not arise. This allows us to make some new observations and reach a novel conclusion.

$c_1, \ldots, c_r$ represent the coins of V, $p_1, \ldots, p_r$ represent the prover messages, and $r = r(|x|)$. (Dependence on x will often be omitted in this manner for notational convenience.) We use notation such as $(c_i)_S$ for the random variable obtained by running S once and taking the c_i-component of its output. More generally, partial conversation transcripts will be written like $(c_1, p_1, c_2, p_2)_S$. We call a conversation transcript $(c_1, p_1, \ldots, c_r, p_r)$ which would make V accept (resp., reject) an *accepting conversation* (resp., *rejecting conversation*). We denote by $U(n)$ the uniform distribution on strings of length n.

The proof. In order to formalize the above intuition, a definition of the simulation-based prover needs to be given. This is the prover P^* that imitates the virtual prover, *i.e.* P^* does the following to compute its next message when the current conversation transcript is $(c_1, p_1, \ldots, c_i)$:

> If $S(x)$ outputs conversations that begin with $(c_1, p_1, \ldots, c_i)$ with probability 0, then output $0^{q(|x|)}$.
>
> Else output $y \in \{0,1\}^{q(|x|)}$ with probability
>
> $$p_y = \Pr[S(x) \text{ begins with } (c_1, p_1, \ldots, c_i, y) | \; S(x) \text{ begins with } (c_1, p_1, \ldots, c_i)].$$

In order to analyze the success probability of P^*, we first compare the output of S to the actual conversations between P^* and V. Let ϵ_i be the statistical difference between $(c_1, p_1 \ldots, c_{i-1}, p_{i-1}, c_i)_S$ and $(c_1, p_1 \ldots, c_{i-1}, p_{i-1})_S \otimes U(q(|x|))$. Thus ϵ_i measures how far from uniform the virtual verifier's i-th set of coins are and how far from independent they are from what comes before. The following claim formalizes our intuition that P^* can do as well as the virtual prover, as long as the virtual verifier's coins are near-uniform and near-independent from what preceeds them.

Claim 13 $||S(x) - (P^*, V)(x)|| \leq \sum_{i=0}^{r} \epsilon_i$.

Proof: Let $C_i^S = (c_1, p_1, \ldots, c_i)_S$ be the random variable of partial simulator transcripts ending with the i-th coins of the virtual verifier. Let $P_i^S = (c_1, p_1, \ldots, c_i, p_i)_S$ be the random variable of partial transcripts ending with the i-th virtual prover response. Similarly define C_i^* and P_i^* as partial conversation transcripts of (P^*, V). The aim is to show that at round k, the statistical error grows by at most ϵ_k. Formally, it will be shown by induction on k that

$$\left\|P_k^S - P_k^*\right\| \leq \sum_{i=0}^{k} \epsilon_i$$

The case $k = 0$ is trivial. For general k, first note that since P^* gives a response chosen according to the same distribution as the virtual prover, adding these responses to the conversations cannot increase the statistical difference. That is,

$$\left\|P_{k+1}^S - P_{k+1}^*\right\| = \left\|C_{k+1}^S - C_{k+1}^*\right\|$$

The idea now is to extract the parts of $||C_{k+1}^S - C_{k+1}^*||$ corresponding to ϵ_{k+1} and observe that what is left is simply the error from the previous round. Note that $C_{k+1}^* = P_k^* \otimes U(q(|x|))$, since the real verifier's coins are always uniform and independent from what came before.

Then, applying Fact A.3 and the Triangle Inequality,

$$\begin{aligned} &||C_{k+1}^S - C_{k+1}^*|| \\ &\leq \; \left\|C_{k+1}^S - P_k^S \otimes U(q(|x|))\right\| \\ &\qquad + \left\|P_k^S \otimes U(q(|x|)) - P_k^* \otimes U(q(|x|))\right\| \\ &\leq \; \epsilon_{k+1} + \left\|P_k^S - P_k^*\right\| + ||U(q(|x|)) - U(q(|x|))|| \\ &\leq \; \epsilon_{k+1} + \sum_{i=0}^{k} \epsilon_i . \end{aligned}$$

This completes the induction. Since $P_r^S = S(x)$ and $P_r^* = (P^*, V)(x)$, the Claim is proved. ■

We are now ready to construct the distributions we seek. Let A be the algorithm whose output on input x is $(A_0(x), A_1(x), \ldots, A_r(x))$, all run independently, and let B be the algorithm whose output is $(B_0(x), B_1(x), \ldots, B_r(x))$, all run independently. The components of A and B are described in Table 1.

Here, A_i is a sampling of a partial conversation transcript from S up to the virtual verifier's i-th set of coins, while B_i is a sampling of a partial conversation transcript from S up to the virtual prover's $(i-1)$-th response followed by $q(|x|)$ independent random bits. So, for $i \geq 1$, the statistical difference between A_i and B_i is ϵ_i.

We will show that the statistical difference between A and B is negligible if $x \in L$ and is noticeable if $x \notin L$. Amplifying this gap by repetition will give us Lemma 6.

Claim 14 *There exists a negligible function α such that if $x \in L$, then $||A(x) - B(x)|| \leq \alpha(|x|)$.*

Proof: By Fact A.3, the statistical difference between $A(x)$ and $B(x)$ is bounded above by the sum of the statistical differences between $A_i(x)$ and $B_i(x)$ over $i = 1, \ldots, r(|x|)$. First, let's examine A_0 and B_0. Since $S(x)$ outputs a conversation which makes V accept with probability at least $2/3 - \text{neg}(|x|)$, the Chernoff bound implies that $\Pr[A_0(x) = 1] = 1 - 2^{-\Omega(|x|)}$, so the statistical difference between A_0 and B_0 is negligible. In the real conversations of P and V, the verifier's coins are truly uniform and independent from prior rounds, so $||A_i(x) - B_i(x)||$ should essentially be bounded by the statistical difference between the simulator's output and

Algorithm A		Algorithm B	
$A_0(x)$	Run $S(x)$ for $\lvert x\rvert$ repetitions. Output '1' if the majority are accepting conversations and '0' otherwise.	$B_0(x)$	Output 1.
$A_i(x)$	Run $S(x)$ to output $(c_1, p_1, \ldots, c_i)_{S(x)}$.	$B_i(x)$	Run $S(x)$ and flip $q(\lvert x\rvert)$ more coins to output $(c_1, p_1, \ldots, c_{i-1}, p_{i-1})_{S(x)} \otimes U(q(\lvert x\rvert))$.

Table 1: The components of A and B

the real interaction. This is in fact true, as for each $i \geq 1$, applications of Facts A.2, A.3, and the Triangle Inequality show that $||A_i(x) - B_i(x)|| \leq 2\,||S(x) - (P, V)(x)||$ Since the statistical difference between S and (P, V) is negligible, $||A(x) - B(x)||$ is bounded by the sum of polynomially many negligible functions and is therefore negligible itself. ■

Claim 15 *If $x \notin L$ then* $||A(x) - B(x)|| \geq 1/12r(|x|)$.

Proof: By Fact A.2, it suffices to show that for some i, $||A_i(x) - B_i(x)|| > 1/12r(|x|)$. We deal with two cases depending on the probability that S outputs an accepting conversation.

Case 1: $\Pr[S(x) \text{ accepts}] \leq 5/12$. Then, by the Chernoff bound, $\Pr[A_0(x) = 1] \leq 2^{-\Omega(|x|)}$, so the statistical difference between $A_0(x)$ and $B_0(x)$ is at least $1 - 2^{-\Omega(|x|)} > 1/12r(|x|)$.

Case 2: $\Pr[S(x) \text{ accepts}] > 5/12$. Then, since $\Pr[(P^*, V)(x) \text{ accepts}]$ is at most $1/3$, we must have

$$\sum_{i=0}^{r} \epsilon_i \geq ||S(x) - (P^*, V)(x)|| > 1/12.$$

Thus, at least one ϵ_i must be greater than $1/12r(|x|)$. ■

Now consider the samplable distributions $\hat{A}(x) = \otimes^{s(|x|)} A(x)$ and $\hat{B}(x) = \otimes^{s(|x|)} B(x)$, where $s(n) = n(12r(n))^2$. If $x \in L$, $\left\|\hat{A}(x) - \hat{B}(x)\right\| \leq s(|x|)\,||A(x) - B(x)||$, which is still negligible. If $x \notin L$, then by Fact A.4, $\left\|\hat{A}(x) - \hat{B}(x)\right\| \geq 1 - 2^{-\Omega(|x|)}$.

This completes the proof of Lemma 6. ■

Acknowledgements

We thank our advisor Shafi Goldwasser for getting us started on the topic of statistical zero-knowledge and providing direction and advice throughout our work. We are deeply indebted to Oded Goldreich for many enlightening conversations on this topic and his extensive comments on this paper. We are grateful for Mihir Bellare's valuable suggestions on our presentation. Our thanks also to Erez Petrank for useful discussions on this topic and bringing [1] to our attention, and to anonymous referees for helpful suggestions.

References

[1] W. Aiello, M. Bellare, and R. Venkatesan. Knowledge on the average—perfect, statistical, and logarithmic. In *Proceedings of the Twenty Seventh Annual ACM Symposium on the Theory of Computing*, 1995.

[2] W. Aiello and J. Hastad. Perfect zero-knowledge languages can be recognized in two rounds. In *Proceedings of the Twenty Eighth Annual Symposium on Foundations of Computer Science*, pages 439–448, 1987.

[3] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and hardness of approximation problems. In *Proceedings of the Thirty Third Annual Symposium on Foundations of Computer Science*, pages 14–23, 1992.

[4] S. Arora and S. Safra. Probabilistic checking of proofs. In *Proceedings of the Thirty Third Annual Symposium on Foundations of Computer Science*, pages 2–13, 1992.

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

[6] M. Bellare. A note on negligible functions. Technical Report CS97-529, Department of Computer Science and Engineering, University of California at San Diego, March 1997. Also available from the Theory of Cryptography Library (`http://theory.lcs.mit.edu/~tcryptol`).

[7] M. Bellare, O. Goldreich, and M. Sudan. Personal communication, June 1997.

[8] M. Bellare, S. Micali, and R. Ostrovsky. The (true) complexity of statistical zero-knowledge. In *Proceedings of the Twenty Second Annual ACM Symposium on Theory of Computing*, pages 494–502, 1990.

[9] M. Bellare and E. Petrank. Making zero-knowledge provers efficient. In *Proceedings of the Twenty Sixth Annual ACM Symposium on the Theory of Computing*, 1994.

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

[11] S. Cook. The complexity of theorem-proving procedures. In *Conference Record of Third Annual ACM Symposium on Theory of Computing*, pages 151–158, 1971.

[12] I. Damgard. Interactive hashing can simplify zero-knowledge protocol design. In *Proceedings of Crypto '95, Lecture Notes in Computer Science*, volume 403, pages 100–109. Springer-Verlag, 1994.

[13] I. Damgard and R. Cramer. On monotone function closure of perfect and statistical zero-knowledge. Theory of Cryptography Library: Record 96-03, 1996. `http://theory.lcs.mit.edu/~tcryptol`.

[14] I. Damgard, O. Goldreich, T. Okamoto, and A. Wigderson. Honest verifier vs. dishonest verifier in public coin zero-knowledge proofs. In *Proceedings of Crypto '95, Lecture Notes in Computer Science*, volume 403. Springer-Verlag, 1995.

[15] A. De Santis, G. Di Crescenzo, G. Persiano, and M. Yung. On monotone formula closure of SZK. In *Proceedings of the Thirty Fifth Annual Symposium on Foundations of Computer Science*, pages 454–465, 1994.

[16] 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.

[17] O. Goldreich. *Foundations of Cryptography (Fragments of a Book).* Weizmann Institute of Science, February 1995.

[18] O. Goldreich and E. Kushilevitz. A perfect zero-knowledge proof system for a problem equivalent to the discrete logarithm. *Journal of Cryptology*, 6:97–116, 1993.

[19] O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *Journal of the Association for Computing Machinery*, 38(1):691–729, 1991.

[20] O. Goldreich, R. Ostrovsky, and E. Petrank. Computational complexity and knowledge complexity. In *Proceedings of the Twenty Sixth Annual ACM Symposium on the Theory of Computing*, pages 534–543, 1994.

[21] O. Goldreich and E. Petrank. Quantifying knowledge complexity. In *Proceedings of the Thirty Second Annual Symposium on Foundations of Computer Science*, pages 59–68, 1991.

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

[23] R. M. Karp. Reducibility among combinatorial problems. In J. W. Thatcher and R. E. Miller, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, Inc., 1972.

[24] L. A. Levin. Universal'nyĭe perebornyĭe zadachi (Universal search problems : in Russian). *Problemy Peredachi Informatsii*, 9(3):265–266, 1973.

[25] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic methods for interactive proofs. In *Proceedings of the Thirty First Annual Symposium on Foundations of Computer Science*, pages 1–10, 1990.

[26] T. Okamoto. On relationships between statistical zero-knowledge proofs. In *Proceedings of the Twenty Eighth Annual ACM Symposium on the Theory of Computing*, 1996.

[27] R. Ostrovsky, R. Venkatesan, and M. Yung. Interactive hashing simplifies zero-knowledge protocol design. In *Proceedings of Eurocrypt '93, Lecture Notes in Computer Science.* Springer-Verlag, 1993.

[28] C. H. Papadimitriou. *Computational Complexity.* Addison-Wesley, 1994.

[29] E. Petrank and G. Tardos. On the knowledge complexity of NP. In *Proceedings of the Thirty Seventh Annual Symposium on Foundations of Computer Science*, pages 494–502, 1996.

[30] A. Shamir. IP=PSPACE. In *Proceedings of the Thirty First Annual Symposium on Foundations of Computer Science*, pages 11–15, 1990.

A The statistical difference metric

Let X and Y be random variables (possibly on different probability spaces) taking values in a discrete space D. We define the *statistical difference* of X and Y, denoted $\|X - Y\|$ to be

$$\begin{aligned} \|X - Y\| &= \frac{1}{2}\sum_{z \in D} |\Pr[X = z] - \Pr[Y = z]| \\ &= \max_{T \subset D} |\Pr[X \in T] - \Pr[Y \in T]| \\ &= \Pr[X \in S] - \Pr[Y \in S], \end{aligned}$$

where $S = \{z \in D : \Pr[X = z] > \Pr[Y = z]\}$. We now state several facts about statistical difference.

Fact A.1 (Triangle Inequality) *If X, Y, and Z are random variables, $\|X - Y\| \le \|X - Z\| + \|Z - Y\|$.*

Fact A.2 *If X and Y are random variables and f is any function, then $\|f(X) - f(Y)\| \le \|X - Y\|$.*

Fact A.3 *Suppose X_1 and X_2 are independent random variables on probability space Ω and Y_1 and Y_2 are independent random variables on probability space Γ. Then,*

$$\|(X_1, X_2) - (Y_1, Y_2)\| \le \|X_1 - Y_1\| + \|X_2 - Y_2\|.$$

Fact A.4 *Suppose X and Y are random variables with $\|X - Y\| = \epsilon$. Then, for all k,*

$$\left\|\otimes^k X - \otimes^k Y\right\| \ge 1 - 2e^{-k\epsilon^2/2}.$$