Zero Knowledge and Circuit Minimization[☆]

Eric Allender[1,*]

Department of Computer Science, Rutgers University, Piscataway, NJ, USA

Bireswar Das[2]

IIT Gandhinagar, India

Abstract

We show that every problem in the complexity class SZK (Statistical Zero Knowledge) is efficiently reducible to the Minimum Circuit Size Problem (MCSP). In particular Graph Isomorphism lies in $\mathsf{RP}^{\mathsf{MCSP}}$.

This is the first theorem relating the computational power of Graph Isomorphism and MCSP, despite the long history these problems share, as candidate NP-intermediate problems.

Keywords: Graph Isomorphism, Minimum Circuit Size Problem, NP-Intermediate Problem, Statistical Zero Knowledge

[☆]A preliminary version of this work appeared in Proc. 39th International Symposium on Mathematical Foundations of Computer Science (MFCS '14), Lecture Notes in Computer Science 8635, pp. 25-32, 2014.

[*]Corresponding author

Email addresses: allender@cs.rutgers.edu (Eric Allender), bireswar@iitgn.ac.in (Bireswar Das)

URL: www.cs.rutgers.edu/~allender (Eric Allender), www.iitgn.ac.in/faculty/comp/bireswar.htm (Bireswar Das)

[1]Supported in part by NSF grants CCF-0832787, CCF-1064785, and CCF-1555409.

[2]Work performed in part while a DIMACS postdoctoral fellow at Rutgers University, under support provided by the Indo-US Science and Technology Forum.

Zero Knowledge and Circuit Minimization[☆]

Eric Allender[1,*]

Department of Computer Science, Rutgers University, Piscataway, NJ, USA

Bireswar Das[2]

IIT Gandhinagar, India

1. Introduction

For as long as there has been a theory of NP-completeness, there have been attempts to understand the computational complexity of the following two problems:

- Graph Isomorphism (GI): Given two graphs G and H, determine if there is permutation τ of the vertices of G such that $\tau(G) = H$.
- The Minimum Circuit Size Problem (MCSP): Given a number i and a Boolean function f on n variables, represented by its truth table of size 2^n, determine if f has a circuit of size i. (There are different versions of this problem depending on precisely what measure of "size" one uses (such as counting the number of gates or the number of wires) and on the types of gates that are allowed, etc. For the purposes of this paper, any reasonable choice can be used.)

Cook [Coo71] explicitly considered the graph isomorphism problem and mentioned that he "had not been able" to show that GI is NP-complete. Similarly, it has been reported that Levin's original motivation in defining and studying NP-completeness [Lev73] was in order to understand the complexity of GI [PS03], and that Levin delayed publishing his work because he had hoped to be able to say something about the complexity of MCSP [Lev03]. (Trakhtenbrot has written an informative account, explaining some of the reasons why MCSP held special interest for the mathematical community in Moscow in the 1970s [Tra84].)

For the succeeding four decades, GI and MCSP have been prominent candidates for so-called "NP-Intermediate" status: neither in P nor NP-complete. No connection between the relative complexity of these two problems has been established. Until now.

It is considered highly unlikely that GI is NP-complete. For instance, if the polynomial hierarchy is infinite, then GI is not NP-complete [BHZ87]. Many would conjecture that GI $\in$ P; Cook mentions this conjecture already in [Coo71]. However this is still very much an open question, and the complexity of GI has been the subject of a great deal of research. We refer the reader to [KST93, AT05] for more details.

In contrast, comparatively little was written about MCSP, until Kabanets and Cai revived interest in the problem [KC00], by highlighting its connection to the so-called Natural Proofs barrier to circuit lower bounds [RR97]. Kabanets and Cai provided evidence that MCSP is not in P (or even in P/poly); it is known

[☆]A preliminary version of this work appeared in Proc. 39th International Symposium on Mathematical Foundations of Computer Science (MFCS '14), Lecture Notes in Computer Science 8635, pp. 25-32, 2014.

[*]Corresponding author

Email addresses: allender@cs.rutgers.edu (Eric Allender), bireswar@iitgn.ac.in (Bireswar Das)

URL: www.cs.rutgers.edu/~allender (Eric Allender), www.iitgn.ac.in/faculty/comp/bireswar.htm (Bireswar Das)

[1]Supported in part by NSF grants CCF-0832787, CCF-1064785, and CCF-1555409.

[2]Work performed in part while a DIMACS postdoctoral fellow at Rutgers University, under support provided by the Indo-US Science and Technology Forum.

that $\mathsf{BPP}^{\mathsf{MCSP}}$ contains several problems that cryptographers frequently assume are intractable, including the discrete logarithm, and several lattice-based problems [KC00, ABK+06]. The integer factorization problem even lies in $\mathsf{ZPP}^{\mathsf{MCSP}}$ [ABK+06]. (More background on complexity classes such as $\mathsf{P/poly}, \mathsf{BPP}$, and ZPP can be found in Section 2.)

Is MCSP complete for NP? Krajíček discusses this possibility [Kra11], although no evidence is presented to suggest that this is a likely hypothesis. Instead, evidence has been presented to suggest that it will be difficult to reduce SAT to MCSP. Kabanets and Cai define a class of "natural" many-one reductions; after observing that most NP-completeness proofs are "natural" in this sense, they show that any "natural" reduction from SAT to MCSP yields a proof that $\mathsf{EXP} \not\subseteq \mathsf{P/poly}$. Interestingly, Vinodchandran studies a problem called SNCMP, which is similar to MCSP, but defined in terms of strong nondeterministic circuits, instead of deterministic circuits [Var05]. (SNCMP stands for Strong Nondeterministic Circuit Minimization Problem. Strong nondeterministic circuits provide a characterization of the complexity class $\mathsf{NP/poly} \cap \mathsf{coNP/poly}$. Formally, a strong nondeterministic circuit for inputs of size n has n input gates, and some number of "auxiliary nondeterministic input gates"; in addition to the output gate, there is a second output gate called the *flag*. For any input x, there should be some setting of the auxiliary nondeterministic input gates that causes the flag bit to be turned on. If the flag bit is turned on, then x is accepted iff the output bit is 1 – which means that any two settings of the auxiliary nondeterministic bits that turn on the flag bit must agree on what the output bit is.) Vinodchandran shows that any "natural" reduction from graph isomorphism to SNCMP yields a nondeterministic algorithm for the complement of GI that runs in subexponential time for infinitely many lengths n.

A problem related to MCSP was considered by Ko [Ko91]; Ko studied the set of strings with low time-bounded Kolmogorov complexity, which he called MINKT. (Ko was using a slightly different notion of time-bounded Kolmogorov complexity than the notion that is discussed in Section 2, which is even more closely related to MCSP.) He presented an oracle relative to which MINKT is neither in P nor is NP-complete, even under polynomial-time Turing reducibility.

We show that $\mathsf{GI} \in \mathsf{RP}^{\mathsf{MCSP}}$; our proof also shows that $\mathsf{GI} \in \mathsf{RP}^{\mathsf{SNCMP}}$. Thus, although it would be a significant breakthrough to give a "natural" reduction from GI to SNCMP (since this would provide a subexponential-time nondeterministic algorithm[3] for the non-isomorphism problem), no such obstacle prevents us from establishing an RP-Turing reduction. Similarly, our proof implies that $\mathsf{GI} \in \mathsf{RP}^{\mathsf{MINKT}}$.

One of the more important results about GI is that GI lies in SZK: the class of problems with statistical zero-knowledge interactive proofs [GMW91]. After giving a direct proof of the inclusion $\mathsf{GI} \in \mathsf{RP}^{\mathsf{MCSP}}$ in Section 3, we give a proof of the inclusion $\mathsf{SZK} \subseteq \mathsf{Promise\text{-}BPP}^{\mathsf{MCSP}}$ in Section 4. (We also provide the necessary background regarding "promise problems" at that point.) We conclude with a discussion of additional directions for research and open questions.

But first, in Section 2, we present the basic connection between MCSP and resource-bounded Kolmogorov complexity, which allows us to use MCSP to invert polynomial-time computable functions.

2. Preliminaries and Technical Lemmas

We assume that the reader is familiar with elementary computational complexity theory, as presented in standard textbooks such as [AB09]. Background on $\mathsf{P}, \mathsf{NP}, \mathsf{EXP}$ and all of the other complexity classes that we discuss can be found there.

Here is a quick review of the probabilistic complexity classes that we will be using. A language is in RP if it is accepted by an NP machine with the additional property that, if there is any accepting path at all on input x, then at least half of the $2^{p(|x|)}$ computation paths of length $p(|x|)$ are accepting. Thus $\mathsf{RP} \subseteq \mathsf{NP}$. A language A is in ZPP if both A and its complement are in RP. A probabilistic machine accepting a language $A \in \mathsf{RP}$ will never have an accepting computation path on any input $x \notin A$, but it might have a small number of rejecting paths on inputs $x \in A$. Thus RP is said to consist of problems having probabilistic algorithms

[3] Very recently, precisely such a "significant breakthrough" has been announced. Babai has presented a *deterministic* quasi-polynomial time algorithm for GI [Bab16]. Thus the results of [Var05] have been superseded, insofar as they apply to GI.

with *one-sided error*. The analogous *two-sided error* class is called BPP: A language A is in BPP if there is a probabilistic polynomial-time Turing machine M with the property that, for all x, with probability at least $\frac{3}{4}$, the output of M on input x is the correct answer to the question "Is x in A?".

Complexity classes that are defined by resource-bounded Turing machines (such as RP, BPP, etc.) lend themselves to the definition of reducibilities. Given any set A, the notation RP^A denotes the class of problems that are RP-Turing reducible to A. That is RP^A denotes the class of problems that can be solved by probabilistic oracle Turing machines running in polynomial time, with one-sided error, using oracle A. BPP^A is defined similarly.

P/poly is the class of languages that can be recognized by families of polynomial size circuits.

Some of our main results concern the class SZK, consisting of those problems that have statistical zero-knowledge interactive proofs. This is a notion that was introduced by Goldwasser, Micali, and Rackoff [GMR89], and has subsequently been the subject of intensive investigation. The only facts about SZK that will be required here are introduced in Section 4.

Now, we briefly describe the connection between MCSP and resource-bounded Kolmogorov complexity.

A small circuit for a Boolean function f on n variables constitutes one form of a short description for the bit string of length 2^n that describes the truth table of f. In fact, as discussed in [ABK+06, Theorem 11], there is a version of time-bounded Kolmogorov complexity (denoted KT) that is roughly equivalent to circuit size. That is, if x is a string of length m representing the truth table of a function f with minimum circuit size s, it holds that

$$\left(\frac{s}{\log m}\right)^{1/4} \leq \mathrm{KT}(x) \leq O(s^2(\log s + \log\log m)).$$

The connection with Kolmogorov complexity is relevant, because of this simple observation: The output of a pseudorandom generator consists of strings with small time-bounded Kolmogorov complexity. Thus, with an oracle for MCSP, one can take as input a string x and accept iff x has no circuits of size, say, $\sqrt{|x|}$, and thereby ensure that one is accepting a very large fraction of all of the strings of length n (since most x encode functions that require large circuits), and yet[4] accept no strings x such that $\mathrm{KT}(x) \leq |x|^{1/9}$. Such a set is an excellent test to distinguish the uniform distribution from the distribution generated by a pseudorandom generator. Using the tight connection between one-way functions and pseudorandom generators [HILL99], one obtains the following result:

Theorem 1. *[ABK+06, Theorem 45] Let L be a language containing at least $2^n/n^k$ strings of each length n, for some k, such that, for some $\epsilon > 0$, for every $x \in L$, $KT(x) \geq |x|^\epsilon$. Let $f(y,x)$ be computable uniformly in time polynomial in $|x|$. There exists a polynomial-time probabilistic oracle Turing machine N and polynomial q such that for any n and y*

$$\Pr_{|x|=n,s}[f(y, N^L(y, f(y,x), s)) = f(y,x)] \geq 1/q(n),$$

where x is chosen uniformly at random and s denotes the internal coin flips of N.

That is, let f_y be a collection of functions indexed by a parameter y, where $f_y(x)$ denotes $f(y,x)$. Then, if one has access to an an oracle L that contains many strings but no strings of small KT-complexity, one can use the probabilistic algorithm N to take as input $f_y(x)$ for a randomly-chosen x, and with non-negligible probability find a $z \in f_y^{-1}(f_y(x))$, that is, a string z such that $f_y(z) = f_y(x)$.

Note that there is such a set L that can be recognized in deterministic polynomial time with an oracle for MCSP, as well as with an oracle for SNCMP. (One could also use an oracle for R_{KT}, the KT-random strings: $R_{\mathrm{KT}} = \{x : \mathrm{KT}(x) \geq |x|\}$.) For instance, the set $L = \{f : f$ is a Boolean function on n variables, and $(f, 2^{n/2}) \notin \mathsf{MCSP}\}$ satisfies the hypothesis of Theorem 1.

[4]Here we are oversimplifying slightly. A more rigorous argument applies the techniques of [RR97], as in [ABK+06, Theorem 45].

3. Graph Isomorphism and Circuit Size

Theorem 2. $\mathsf{GI} \in \mathsf{RP}^{\mathsf{MCSP}}$.

Proof. We are given as input two graphs G and H, and we wish to determine whether there is an isomorphism from G to H.

Consider the polynomial-time computable function $f(G, \tau)$ that takes as input a graph G on n vertices and a permutation $\tau \in S_n$ and outputs $\tau(G)$. We will use the notation $f_G(\tau)$ to denote $f(G, \tau)$. That is, f_G takes a permutation τ as input, and produces as output the adjacency matrix of the graph obtained by permuting G according to τ. Observe that f_G is uniformly computable in time polynomial in the length of τ.

Thus, by Theorem 1, there is a polynomial-time probabilistic oracle Turing machine N and polynomial q such that for any n and G

$$\Pr_{\tau \in S_n, s}[f_G(N^{\mathsf{MCSP}}(G, f_G(\tau), s)) = f_G(\tau)] \geq 1/q(n),$$

where τ is chosen uniformly at random and s denotes the internal coin flips of N.

Now, given input (G, H) to GI, our $\mathsf{RP}^{\mathsf{MCSP}}$ algorithm does the following for $100q(n)$ independent trials:

1. Pick τ and probabilistic sequence s uniformly at random.
2. Compute $\tau(G)$.
3. Run $N^{\mathsf{MCSP}}(H, \tau(G), s)$ and obtain output π.
4. Report "success" if $\pi(H) = \tau(G)$.

The $\mathsf{RP}^{\mathsf{MCSP}}$ algorithm will accept if at least one of the $100q(n)$ independent trials is successful.

Note that if H and G are not isomorphic, then there is no possibility that the algorithm will succeed.

On the other hand, if H and G are isomorphic, then $\tau(G)$ does appear in the image of f_H. In fact, the distributions $\tau(G)$ and $\tau(H)$ are identical over τ picked uniformly at random. Thus, with probability at least $1/q(n)$ (taken over the choices of τ and s), the algorithm will succeed in any given trial. Thus the expected number of trials that will succeed is at least 100, and hence, by the Chernoff bounds, the probability of having at least one success is well over $1/2$. □

Since truth-tables that require large strong nondeterministic circuits also require large deterministic circuits, it is immediate that this reduction can be carried out also with SNCMP.

Corollary 3. $\mathsf{GI} \in \mathsf{RP}^{\mathsf{SNCMP}} \cap \mathsf{RP}^{R_{\mathrm{KT}}}$.

4. Zero Knowledge

In this section, we show that all problems in SZK BPP-reduce to MCSP. We begin by recalling some basic facts about SZK.

When Goldwasser, Micali, and Rackoff introduced the notion of interactive proofs [GMR89], they also began the study of designing protocols that would allow a prover to interact with an untrusted verifier, and convince the verifier that a string is in a language, without revealing any information that will enable the verifier to in turn convince another party about the truth of the assertion; the verifier learns nothing by interacting with the prover, other than the truth of the assertion. More formally, the probability distribution on transcripts of interactions between the prover and the verifier is "indistinguishable" from a probability distribution that the verifier can generate alone.

Different notions of "zero knowledge" proof arise, depending on how the word "indistinguishable" is interpreted. Under plausible cryptographic assumptions, PSPACE is precisely the set of languages for which a "zero knowledge" protocol exists, where the distribution on transcripts is "computationally indistinguishable" from a distribution that the verifier can generate [BOGG$^+$90, Sha92].

The notion that is more relevant to the current investigation is the notion that results when "indistinguishable" is interpreted as being "statistically close" – regardless of whether an algorithm can efficiently distinguish between the two. This gives rise to the class known as SZK, for "Statistical Zero Knowledge". For the purposes of the present investigation, the reader will not need the formal (and somewhat complicated) definition of the class SZK; it will suffice to know that SZK has complete problems, and to know the definition of one of those complete problems. For more background and motivation about SZK, we refer the reader to some of the excellent treatments that are available [Gol02, Vad17]. In particular, these are good sources to find out about some of the important natural problems that are known to lie in SZK.

SZK is best defined not as a class of *languages*, but as a class of "promise problems". A promise problem consists of a pair of disjoint languages (Y, N) where Y consists of "yes-instances" and N consists of "no-instances". Thus a language is the special case of a promise problem where $Y \cup N = \{0,1\}^*$. It follows that our theorem relating the complexity of SZK and MCSP is best stated in terms of "promise BPP". That is, we will show that, for every $(Y, N) \in \mathsf{SZK}$ there is a probabilistic polynomial time oracle Turing machine M with the property that $x \in Y$ implies $M(x)$ accepts with probability at least $2/3$ when given oracle MCSP, and $x \in N$ implies $M(x)$ accepts with probability at most $1/3$ when given oracle MCSP. M may exhibit any behavior on inputs outside of $N \cup Y$. We express this using the notation $\mathsf{SZK} \subseteq \mathsf{Promise\text{-}BPP}^{\mathsf{MCSP}}$.

It was shown by Chailloux et al. [CCKV08] that SZK is equal to a class that Ben-Or and Gutfreund [BOG03] defined and called $\mathsf{NISZK}|_h$. Importantly for us, Ben-Or and Gutfreund showed that a promise problem they called IID (Image Intersection Density) is complete for $\mathsf{NISZK}|_h$ (and thus, by [CCKV08], IID is also complete for SZK). That is, for every promise problem (Y, N) in SZK, there is a polynomial-time computable function f such that $f(Y)$ is a subset of the yes-instances of IID, and $f(N)$ is a subset of the no-instances of IID. The yes-instances of IID consist of pairs of circuits (C_0, C_1), each of size n, taking m-bit inputs, such that the distributions $C_0(x)$ and $C_1(x)$ (where x is chosen uniformly at random) have statistical distance at most $1/n^2$. It is important to note that the circuits C_0 and C_1 have m' output bits, for some $m' > 1$, and thus the distributions $C_0(x)$ and $C_1(x)$ are distributions on $\{0,1\}^{m'}$. The no-instances of IID consist of pairs of circuits (C_0, C_1) with the property that $\Pr_{|x|=m}[\exists y\ C_1(y) = C_0(x)] < 1/n^2$.

We will not work directly with IID, but rather with a related problem that is shown to be complete for $\mathsf{NISZK}|_h$ in [BOG03, Lemma 20], which is just like IID but with different parameters. Let us call this problem PIID for "polarized IID". The yes-instances of PIID consist of triples (n, D_0, D_1), where each D_i is an m-input circuit of size at most n^k (for some fixed k), such that the distributions $D_0(x)$ and $D_1(x)$ (where x is chosen uniformly at random) have statistical distance at most $1/2^n$. The no-instances of PIID consist of triples (n, D_0, D_1) with the property that $\Pr_{|x|=m}[\exists y\ D_1(y) = D_0(x)] < 1/2^n$.

Furthermore, we need to make use of the fact that we can assume that the length m of the inputs to the circuits D_0 and D_1 may be assumed without loss of generality to be at least n^δ for some fixed $\delta > 0$. This can be accomplished by simply adding dummy input variables. It is easy to check that adding dummy variables to both circuits does not change the statistical difference. Similarly, this does not alter the probability that the output produced by a random input to the first circuit is in the support of the second circuit.

Theorem 4. $\mathsf{SZK} \subseteq \mathsf{Promise\text{-}BPP}^{\mathsf{MCSP}}$

Proof. It will suffice to show that $\mathsf{PIID} \in \mathsf{Promise\text{-}BPP}^{\mathsf{MCSP}}$.

Consider the polynomial-time computable function $F(C, x)$ that takes a Boolean circuit C on m-bit inputs, and a string x of length m as input, and outputs $C(x)$. We will use the notation $F_C(x)$ to denote $F(C, x)$. Since the length of x is polynomially-related to the size of C in the instances of PIID that we consider, it follows that F_C is uniformly computable in time polynomial in the length of x.

Thus, by Theorem 1, there is a polynomial-time probabilistic oracle Turing machine N and polynomial q such that for any m and C

$$\Pr_{|x|=m,s}[F_C(N^{\mathsf{MCSP}}(C, F_C(x), s)) = F_C(x)] \geq 1/q(m),$$

where x is chosen uniformly at random and s denotes the internal coin flips of N.

Now, given input (n, D_0, D_1) to PIID, our $\mathsf{Promise\text{-}BPP}^{\mathsf{MCSP}}$ algorithm does the following for n^ℓ independent trials (for an ℓ to be determined later):

1. Pick m-bit input x and probabilistic sequence s uniformly at random.
2. Compute $z = D_0(x)$.
3. Run $N^{\mathsf{MCSP}}(D_1, z, s)$ and obtain output y.
4. Report "success" if $D_1(y) = z$.

The $\mathsf{Promise\text{-}BPP}^{\mathsf{MCSP}}$ algorithm will accept if at least $\log n$ of the n^ℓ independent trials are successful.

If (n, D_0, D_1) is a no-instance of PIID, then the probability that any given trial succeeds is at most $1/2^n$. Thus, for all large n the expected number of the n^ℓ trials that will succeed is at most $n^\ell/2^n < 1$. By the Chernoff bounds, the probability that $\log n$ trials will succeed is less than $1/3$.

If (n, D_0, D_1) is a yes-instance of PIID, then $D_0(x)$ and $D_1(x)$ have statistical distance at most $1/2^n$.

Note that

$$\begin{aligned}
&\Pr_{|x|=m,s}[F_{D_1}(N^{\mathsf{MCSP}}(D_1, F_{D_0}(x), s)) = F_{D_0}(x)] \\
&= \sum_z \Pr_{|x|=m,s}[F_{D_1}(N^{\mathsf{MCSP}}(D_1, z, s)) = z | z = F_{D_0}(x)] \Pr[z = F_{D_0}(x)] \\
&= \sum_z \Pr_{|x|=m,s}[F_{D_1}(N^{\mathsf{MCSP}}(D_1, z, s)) = z | z = F_{D_1}(x)] \Pr[z = F_{D_0}(x)]
\end{aligned}$$

Also,

$$\begin{aligned}
&\Pr_{|x|=m,s}[F_{D_1}(N^{\mathsf{MCSP}}(D_1, F_{D_1}(x), s)) = F_{D_1}(x)] \\
&= \sum_z \Pr_{|x|=m,s}[F_{D_1}(N^{\mathsf{MCSP}}(D_1, z, s)) = z | z = F_{D_1}(x)] \Pr[z = F_{D_1}(x)]
\end{aligned}$$

Thus the difference of these two probabilities is

$$\begin{aligned}
&\sum_z \Pr_{|x|=m,s}[F_{D_1}(N^{\mathsf{MCSP}}(D_1, z, s)) = z | z = F_{D_1}(x)] \times \\
&\qquad\qquad (\Pr[z = F_{D_0}(x)] - \Pr[z = F_{D_1}(x)]) \\
&\leq \sum_z 1 \cdot (\Pr[z = F_{D_0}(x)] - \Pr[z = F_{D_1}(x)]) \\
&\leq 1/2^n
\end{aligned}$$

Since $\Pr_{|x|=m,s}[F_{D_1}(N^{\mathsf{MCSP}}(D_1, F_{D_1}(x), s)) = F_{D_1}(x)] > 1/q(m) > 1/q(n^k)$, it follows that each trial has probability at least $1/q(n^k) - 1/2^n$ of success. Thus, the expected number of the n^ℓ trials that will succeed is at least $n^\ell(1/q(n^k) - 1/2^n)$. Picking ℓ so that n^ℓ is enough greater than $q(n^k)$ guarantees that this expected value is at least n. Thus, by the Chernoff bounds the probability that at least $\log n$ trials succeed is greater than $2/3$. □

In the above proof, notice that we obtain one-sided error on those instances (n, D_0, D_1) of PIID where $\Pr_{|x|=m}[\exists y\ D_1(y) = D_0(x)] = 0$, instead of merely being bounded by $1/2^n$. In particular, the promise problem known as $\overline{\mathsf{SD}^{1,0}}$ (consisting of pairs of circuits (D_0, D_1) where, for the yes-instances, D_0 and D_1 represent identical distributions, and the no-instances have disjoint images) is in $\mathsf{Promise\text{-}RP}^{\mathsf{MCSP}}$. Thus $\mathsf{RP}^{\mathsf{MCSP}}$ contains all of the problems that were shown to be reducible to $\overline{\mathsf{SD}^{1,0}}$ by Arvind and Das [AD08]. Furthermore, $\overline{\mathsf{SD}^{1,0}}$ has been shown to be complete for the class of problems that have "V-bit" perfect zero knowledge protocols [KMS15]. This class contains most of the problems that are known to have perfect zero-knowledge protocols.

It is perhaps worth mentioning that many of the promise problems in SZK are, in fact, *languages* (in the sense that the union of the yes-instances and the no-instances is all of Σ^*). All such languages lie in $\mathsf{BPP}^{\mathsf{MCSP}}$; there is no need to refer to "promise problems" in such cases.

We remark that it could still be possible that all of the promise problems in SZK have a solution in $\mathsf{BPP}^{\mathsf{MCSP}}$ (i.e., via an oracle machine that exhibits BPP-like behavior on all inputs). The following theorem shows that an even stronger conclusion holds in the (unlikely) case that (a) $\mathsf{MCSP} \in \mathsf{P/poly}$, and (b) there is a polynomial-time oracle Turing machine with an oracle for MCSP that, on input 0^n, outputs an element of the set $L = \{f : f \text{ is a Boolean function on } m \text{ variables, and } (f, 2^{m/2}) \notin \mathsf{MCSP}\}$ having length $\geq n$.[5]

Theorem 5. *If* $\mathsf{MCSP} \in \mathsf{P/poly}$ *and there is a polynomial-time oracle Turing machine with an oracle for* MCSP *that, on input* 0^n*, outputs an element of* L *of length* $\geq n$*, then every promise problem in* $\mathsf{Promise\text{-}BPP}^{\mathsf{MCSP}}$ *has a solution in* $\mathsf{P}^{\mathsf{MCSP}}$*.*

Proof. Lemma 35.3 of [ABK+06] shows that $\mathsf{BPP}^{R_{\mathrm{KT}}} = \mathsf{P}^{R_{\mathrm{KT}}}$ if $R_{\mathrm{KT}} \in \mathsf{P/poly}$ and there is a $\mathsf{P}^{R_{\mathrm{KT}}}$ algorithm that produces a long-enough element of R_{KT}, on input 0^n.

The proof of Lemma 35.3 proceeds along the following lines, to simulate a BPP oracle Turing machine M with oracle R_{KT}:

1. On input x of length n, produce an element y of R_{KT} of length n^k (for a suitably-large value of k).
2. The string y is the truth table of a sufficiently hard function, so that one can use the Impagliazzo-Wigderson generator [IW97] to simulate the BPP oracle machine M *deterministically* with oracle R_{KT}.

Clearly, under our assumptions, the same proof carries over using MCSP as an oracle, instead of R_{KT}. Thus $\mathsf{BPP}^{\mathsf{MCSP}} = \mathsf{P}^{\mathsf{MCSP}}$ if $\mathsf{MCSP} \in \mathsf{P/poly}$ and our other assumption holds.

Furthermore, given any promise problem in $\mathsf{Promise\text{-}BPP}^{\mathsf{MCSP}}$, one can follow the same strategy. For inputs that satisfy the promise, the simulation behaves correctly. For inputs that do not satisfy the promise, the deterministic oracle machine gives some (arbitrary) answer. □

Corollary 6. *If* $\mathsf{MCSP} \in \mathsf{P/poly}$*, and there is a polynomial-time oracle Turing machine with an oracle for* MCSP *that, on input* 0^n*, outputs an element of* L *of length* $\geq n$*, then every promise problem in* SZK *has a solution in* $\mathsf{P}^{\mathsf{MCSP}}$*.*

5. Conclusions and Open Problems

We are the first to admit that there appears to be no reason why these results could not have been proved earlier. The techniques involved have been available to researchers for years, and the proofs have much the same flavor as the reductions of factoring, discrete logarithm, and other cryptographic problems to MCSP that were presented in [ABK+06]. Perhaps the only missing ingredient is that the earlier work involved using MCSP (or, equivalently, R_{KT}) to break pseudorandom generators that were constructed from one-way functions that people actually believed *were* cryptographically secure. In contrast, the functions f_G considered here have never seemed like promising candidates to use, in constructing pseudorandom generators.

It is natural to wonder if better reductions are also possible. Is $\mathsf{GI} \in \mathsf{P}^{\mathsf{MCSP}}$? Or in $\mathsf{ZPP}^{\mathsf{MCSP}}$? Does $\mathsf{RP}^{\mathsf{MCSP}}$ contain all problems with perfect zero knowledge protocols?

Equally temptingly, is it possible to build on these ideas to reduce larger classes to MCSP? For instance, is $\mathsf{NP} \cap \mathsf{coNP} \subseteq \mathsf{Promise\text{-}BPP}^{\mathsf{MCSP}}$? Or is $\mathsf{AM} \cap \mathsf{coAM} \subseteq \mathsf{Promise\text{-}BPP}^{\mathsf{MCSP}}$? (Note that $\mathsf{SZK} \subseteq \mathsf{Promise\text{-}AM} \cap \mathsf{Promise\text{-}AM}$ [For89, AH91].) The Wikipedia article on "NP-Intermediate Problems" (from August 30, 2010 to June 11, 2014) said "... MCSP is believed to be NP-complete" [Wik14]. We are unaware of much evidence for this "belief" being very widespread in the complexity theory community, but it is certainly an intriguing possibility.

[5] We view condition (a) as very unlikely, although condition (b) is quite plausible. For our purposes, the language L in condition (b) could just as easily be replaced by $L_c = \{f : f \text{ is a Boolean function on } m \text{ variables, and } (f, 2^{m/c}) \notin \mathsf{MCSP}\}$, for any constant $c > 2$. Such a condition holds if the Impagliazzo-Wigderson derandomization hypothesis [IW97] holds. That is, if there is a language in $\mathsf{DTIME}(2^{O(n)})$ that requires circuits of size $2^{\epsilon n}$, then there is a constant c such that a polynomial-time routine (with no oracle), on input 0^n, outputs an element of L_c having length at least n.

Alternatively, is it possible to tie MCSP more closely to SZK? For instance, what is the complexity of the promise problem whose yes-instances consist of strings with KT-complexity at most $\sqrt{n}$, and whose no-instances consist of strings with KT-complexity $> n/2$?

Acknowledgments

We acknowledge helpful comments from Lance Fortnow, Valentine Kabanets, Rahul Santhanam, Bruce Kapron, Salil Vadhan, Ilya Volkovich, and the anonymous referees.

References

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity: A modern approach*. Cambridge University Press, 2009.

[ABK+06] E. Allender, H. Buhrman, M. Koucký, D. van Melkebeek, and D. Ronneburger. Power from random strings. *SIAM Journal on Computing*, 35:1467–1493, 2006.

[AD08] V. Arvind and Bireswar Das. SZK proofs for black-box group problems. *Theory Comput. Syst.*, 43(2):100–117, 2008.

[AH91] William Aiello and Johan Håstad. Statistical zero-knowledge languages can be recognized in two rounds. *J. Comput. Syst. Sci.*, 42(3):327–345, 1991.

[AT05] V. Arvind and J. Torán. Isomorphism testing: Perspective and open problems. *Bulletin of the EATCS*, 86, 2005.

[Bab16] Laszlo Babai. Graph isomorphism in quasipolynomial time. In *ACM Symposium on Theory of Computing (STOC)*, pages 684–697, 2016.

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

[BOG03] Michael Ben-Or and Danny Gutfreund. Trading help for interaction in statistical zero-knowledge proofs. *J. Cryptology*, 16(2):95–116, 2003.

[BOGG+90] Michael Ben-Or, Oded Goldreich, Shafi Goldwasser, Johan Håstad, Joe Kilian, Silvio Micali, and Phillip Rogaway. Everything provable is provable in zero-knowledge. In *Advances in Cryptology-Crypto'88*, pages 37–56. Springer, 1990.

[CCKV08] André Chailloux, Dragos Florin Ciocan, Iordanis Kerenidis, and Salil P. Vadhan. Interactive and noninteractive zero knowledge are equivalent in the help model. In *Theory of Cryptography, Fifth Theory of Cryptography Conference (TCC)*, volume 4948 of *Lecture Notes in Computer Science*, pages 501–534. Springer, 2008.

[Coo71] S. A. Cook. The complexity of theorem-proving procedures. In *ACM Symposium on Theory of Computing (STOC)*, pages 151–158, 1971.

[For89] Lance Fortnow. The complexity of perfect zero-knowledge. In S. Micali, editor, *Randomness and Computation*, pages 327–343. JAI Press, Greenwich CT, 1989.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems. *Journal of the ACM*, 38(3):691–729, 1991.

[Gol02] Oded Goldreich. Zero-knowledge twenty years after its invention. Technical Report TR02-063, Electronic Colloquium on Computational Complexity (ECCC), 2002.

[HILL99] J. Håstad, R. Impagliazzo, L. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28:1364–1396, 1999.

[IW97] R. Impagliazzo and A. Wigderson. $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In *ACM Symposium on Theory of Computing (STOC)*, pages 220–229, 1997.

[KC00] V. Kabanets and J.-Y. Cai. Circuit minimization problem. In *ACM Symposium on Theory of Computing (STOC)*, pages 73–79, 2000.

[KMS15] Bruce M. Kapron, Lior Malka, and Venkatesh Srinivasan. A framework for non-interactive instance-dependent commitment schemes (NIC). *Theoretical Computer Science*, 593:1–15, 2015.

[Ko91] Ker-I Ko. On the complexity of learning minimum time-bounded Turing machines. *SIAM Journal on Computing*, 20(5):962–986, 1991.

[Kra11] Jan Krajíček. *Forcing with Random Variables and Proof Complexity*. Cambridge University Press, 2011.

[KST93] Johannes Köbler, Uwe Schöning, and Jacobo Torán. *The Graph Isomorphism Problem: Its Structural Complexity*. Birkhauser Verlag, Basel, Switzerland, Switzerland, 1993.

[Lev73] L. A. Levin. Universal sequential search problems. *Problems of Information Transmission*, 9:265–266, 1973.

[Lev03] L. Levin. Personal communication. 2003.

[PS03] Sriram Pemmaraju and Steven Skiena. *Computational Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*. Cambridge University Press, New York, NY, USA, 2003.

[RR97] A. Razborov and S. Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55:24–35, 1997.

[Sha92] Adi Shamir. IP = PSPACE. *J. ACM*, 39(4):869–877, 1992.

[Tra84] B. A. Trakhtenbrot. A survey of Russian approaches to perebor (brute-force searches) algorithms. *IEEE Annals of the History of Computing*, 6(4):384–400, 1984.

[Vad17] Salil Vadhan. *A Study of Statistical Zero-Knowledge Proofs*. Springer, 2017. To appear.

[Var05] Vinodchandran N. Variyam. Nondeterministic circuit minimization problem and derandomizing Arthur-Merlin games. *Int. J. Found. Comput. Sci.*, 16(6):1297–1308, 2005.

[Wik14] Wikipedia. http://en.wikipedia.org/wiki/NP-intermediate, 2014.