Electronic Colloquium on Computational Complexity, Revision 5 of Report No. 23 (2011)

Input-Oblivious Proof Systems and a Uniform Complexity Perspective on P/poly

Oded Goldreich[*] and Or Meir[†]
Department of Computer Science
Weizmann Institute of Science
Rehovot, Israel.

April 13, 2015

Abstract

An input-oblivious proof system is a proof system in which the proof does not depend on the claim being proved. Input-oblivious versions of $\mathcal{NP}$ and $\mathcal{MA}$ were introduced in passing by Fortnow, Santhanam, and Williams (CCC 2009), who also showed that those classes are related to questions on circuit complexity.

In this paper we wish to highlight the notion of input-oblivious proof systems, and initiate a more systematic study of them. We begin by describing in detail the results of Fortnow *et al.*, and discussing their connection to circuit complexity. We then extend the study to input-oblivious versions of $\mathcal{IP}$, $\mathcal{PCP}$, and $\mathcal{ZK}$, and present few preliminary results regarding those versions.

Keywords: NP, IP, PCP, ZK, P/poly, MA, BPP, RP, E, NE, EXP, NEXP.

[*]Partially supported by the Israel Science Foundation (grant No. 1041/08) and the Minerva Foundation with funds from the Federal German Ministry for Education and Research.

[†]Research supported by the Adams Fellowship Program of the Israel Academy of Sciences and Humanities.

ISSN 1433-8092

Contents

1 Introduction

Various types of proof systems play a central role in the theory of computation. In addition to NP-proof systems, which provide the definitional pillar of $\mathcal{NP}$, probabilistic proof systems giving rise to classes such as $\mathcal{IP}$, $\mathcal{ZK}$ and $\mathcal{PCP}$ have also played a major role. (For further background, see [10, Chap. 9].)

In all these cases, the verification procedure is personified by a player, called the verifier, which interacts (implicitly or explicitly) with a more powerful entity, called the prover. The nature of this interaction may vary according to the type of proof system being considered, but in all cases the interaction may depend on a common input, which represents the claim being proved and verified. In particular, in all cases, the actions of the prescribed prover may depend on this common input.

In this paper, following the work of Fortnow, Santhanam, and Williams [8], we ask how is the expressive power of these proof systems effected when the prescribed prover is only given the length of the claim to be proved. We stress that we restrict the power of the prover being referred to in the completeness condition (i.e., the prescribed/honest prover), but maintain the original formulation of the soundness condition (i.e., the cheating prover). That is, we ask what is the expressive power of the input-oblivious versions of each of these proof systems (where input-oblivious restriction applies only to the honest provers not to cheating ones). We start with the case of $\mathcal{NP}$, studied in [8].

1.1 The case of $\mathcal{NP}$

Recall that $S \in \mathcal{NP}$ if there exists a polynomial-time (verification) procedure V and a polynomial p such that

Completeness: For every $x \in S$ there exists $w \in \{0,1\}^{p(|x|)}$ such that $V(x,w) = 1$.

Soundness: For every $x \notin S$ and every w, it holds that $V(x,w) = 0$.

We ask whether (for this procedure V or for an alternative one) it holds that for every $n \in \mathsf{N}$ there exists $w \in \{0,1\}^{p(n)}$ such that for every $x \in S_n \overset{\text{def}}{=} S \cap \{0,1\}^n$ it holds that $V(x,w) = 1$. Such a string w may be considered a universal NP-witness (for all $x \in S_n$), and its existence yields a poly(n)-sized circuit for deciding S_n (i.e., $S \in \mathcal{P}/\text{poly}$). But does every set in $\mathcal{NP} \cap \mathcal{P}/\text{poly}$ have such universal NP-witnesses? Denoting the class of sets having input-oblivious NP-proofs by $\mathcal{ONP}$, Fortnow *et al.* [8] showed that

Theorem 1.1 (on the power of input-oblivious NP-proofs [8]):

1. *$\mathcal{ONP} = \mathcal{NP}$ if and only if $\mathcal{NP} \subset \mathcal{P}/\text{poly}$.*
2. *If $\mathcal{NE} \neq \mathcal{E}$, then $\mathcal{ONP} \neq \mathcal{P}$.*
3. *$\mathcal{ONP} \subseteq \mathcal{NP} \cap \mathcal{P}/\text{poly}$.*

While the proofs of all items of Theorem 1.1 are quite easy, we find the foregoing assertions quite interesting. In particular, we highlight the fact that the first item provides a uniform complexity formulation of the conjecture $\mathcal{NP} \not\subset \mathcal{P}/\text{poly}$. We mention that it is not clear whether or not $\mathcal{ONP} = \mathcal{NP} \cap \mathcal{P}/\text{poly}$; similarly, it is not clear whether or not $\mathcal{BPP} \cap \mathcal{NP} \subseteq \mathcal{ONP}$ (or whether "$\mathcal{BPP} \subseteq \mathcal{NP}$ implies $\mathcal{BPP} \subseteq \mathcal{ONP}$").

1.2 Other input-oblivious proof systems

While the first part of this paper is devoted to surveying the input-oblivious aspect of the work of Fortnow *et al.* [8], in the second part we define and study input-oblivious versions of various forms of probabilistic proof systems. In particular, we consider input-oblivious versions of interactive proof systems (i.e., $\mathcal{IP}$), zero-knowledge proof systems (i.e., $\mathcal{ZK}$), and probabilistically checkable proof systems (i.e., $\mathcal{PCP}$).

We define the input-oblivious version of $\mathcal{IP}$, which we denote $\mathcal{OIP}$, as an interactive proof system in which the actual interaction takes place before the input is revealed to the prover (and the verifier). For this class, we obtain the following sharp characterization.

Theorem 1.2 (on the power of input-oblivious interactive proofs): $\mathcal{OIP} = \mathcal{IP} \cap \mathcal{P}/\text{poly}$.

Considering the zero-knowledge restriction of $\mathcal{OIP}$, we define the class $\mathcal{OZK}$ (i.e., the input-oblivious version of $\mathcal{ZK}$), and show that it is unlikely to contain all of $\mathcal{OIP}$. In particular, $\mathcal{OZK} \setminus \mathcal{BPP}$ may only contain sets for which it is hard to find YES-instances. On the other hand, assuming that $\mathcal{NE} \not\subseteq \mathcal{BPE}$ and that one-way functions exist, the class $\mathcal{OZK}$ extends beyond $\mathcal{BPP}$.

We also consider $\mathcal{OPCP}$, the input-oblivious version of $\mathcal{PCP}[O(\log n), O(1)]$. Finally, we observe that Item 2 of Theorem 1.1 also holds for the class $\mathcal{OPCP}$; in fact, *we show that any sparse NP-set is in* $\mathcal{OPCP}$ (see Theorem 4.3), but it is unclear whether or not $\mathcal{ONP} = \mathcal{OPCP}$ (and even whether $\mathcal{RP} \subseteq \mathcal{OPCP}$).

Related work. As mentioned above, Fortnow, Santhanam, and Williams [8] defined the class $\mathcal{ONP}$, and proved Theorem 1.1. They also defined the class $\mathcal{OMA}$, the input-oblivious version of Merlin-Arthur games ($\mathcal{MA}$), and showed that $\mathcal{ONP} = \mathcal{NP}$ if and only if $\mathcal{NP} \subseteq \mathcal{OMA}$. However, their main interest was actually in the class $\mathcal{ONP}/1$ (cf. [8, Thm. 11]) as well as in other classes of the form $\mathcal{C}/1$ (cf. [8, Thm. 12]). In particular, in [8, Thm. 11] they show that, for every c, it holds that $\mathcal{NP}$ *has no size* n^c *circuits if and only if* $\mathcal{ONP}/1$ *has no size* n^c *circuits.*

Prior to [8], Aaronson [1] defined an input-oblivious version of $\mathcal{NP} \cap \text{co}\mathcal{NP}$, which he denoted $\mathcal{YP}$, and proved an analog of Theorem 1.1 for this class (see [1, Thm 4.4]). He also explicitly discusses the view of input-oblivious proof systems as "untrusted advice", which is emphasized in Section 1.3 below.

Chakaravarthy and Roy [7] considered an input-oblivious version of the symmetric alternation class $\mathcal{S}_2$, and showed that this new class, denoted $\mathcal{O}_2$, contains $\mathcal{BPP}$. They also showed that if $\mathcal{NP} \subset \mathcal{P}/\text{poly}$, then the Polynomial-time Hierarchy collapses to $\mathcal{O}_2$. We note that it is not clear how $\mathcal{O}_2$ relates to $\mathcal{NP}$, but it is syntactically obvious that $\mathcal{O}_2$ contains the class $\mathcal{ONP}$.

Older works of Schöning and Ko on robust oracle machines [21, 14, 22] and of Arvind et. al. on computationally-bounded provers [2] made some of the observations of [8] and of ours using a completely different terminology. We explain their terminology and their relevant results in the appendix.

1.3 Connection to circuit lower bounds

An additional motivation for the study of input-oblivious proof systems comes from their connection to circuit complexity. As we explain below, input-oblivious proof systems may be viewed as a restriction of $\mathcal{P}/\text{poly}$ to advice strings that can be verified. As such, input-oblivious proof systems are strictly weaker than $\mathcal{P}/\text{poly}$ (e.g., $\mathcal{ONP}$ is *strictly* contained in $\mathcal{P}/\text{poly}$). However, it turns out that in some cases, the computational limitations of input-oblivious proof systems imply corresponding limitations on $\mathcal{P}/\text{poly}$ (e.g., $\mathcal{NP} \not\subseteq \mathcal{ONP}$ implies $\mathcal{NP} \not\subseteq \mathcal{P}/\text{poly}$). Thus, proving that certain classes do not have small circuits is equivalent to proving that these classes have no input-oblivious proof systems. Details follow.

Recall that $\mathcal{P}$/poly may be viewed as the class of sets that can be decided by a Turing machine that takes advice. The advice is an arbitrary string of polynomial length, which may depend on the length of the input but not on the input itself. Consider the function $f : \mathsf{N} \to \{0,1\}^*$ that maps each input length to its corresponding advice string. The definition of $\mathcal{P}$/poly places no restrictions on the complexity of computing f, and in particular f is not even required to be computable. This feature of the advice makes $\mathcal{P}$/poly a powerful class, which can even compute functions that are not computable by Turing machines.

It is a natural question to ask what happens when we place computational restrictions on f. The first restriction that may come to mind is to require that $f(n)$ is computable in time poly(n). However, restricting $\mathcal{P}$/poly in this way results in the class $\mathcal{P}$, and is therefore not very interesting.

A second natural restriction is requiring the function f to be verifiable. In other words, we require that although we may not be able to compute the advice efficiently, we can at least verify its correctness. This idea can be realized in few possible ways, and the input-oblivious proof systems defined below can be thought as such realizations

The notions of input-oblivious proof systems (e.g., $\mathcal{ONP}$) may be useful towards studying the circuit complexity of their standard counterparts (resp., $\mathcal{NP}$), because on the one hand these input-oblivious proof systems are strictly weaker than $\mathcal{P}$/poly, and on the other hand they retain much of the power of $\mathcal{P}$/poly. For example, consider the class $\mathcal{ONP}$: on the one hand, $\mathcal{ONP}$ is contained in $\mathcal{NP}$, and is therefore strictly weaker than $\mathcal{P}$/poly (since it can not decide uncomputable functions). On the other hand, Theorem 1.1 shows that *if* $\mathcal{P}$/poly *contains* $\mathcal{NP}$*, then so does* $\mathcal{ONP}$, and in this sense $\mathcal{ONP}$ is quite powerful. A particularly interesting corollary of this theorem is that proving super-polynomial circuit lower bounds for $\mathcal{NP}$ is equivalent to separating $\mathcal{ONP}$ from $\mathcal{NP}$.

The foregoing discussion is not restricted to $\mathcal{ONP}$. In Section 3 we consider the class $\mathcal{OIP}$, which is the input-oblivious version of $\mathcal{IP}$. The class $\mathcal{OIP}$ may also be thought of as the class that results from restricting the advice of $\mathcal{P}$/poly (i.e., the above function f) to be verifiable by an interactive protocol. We show that

$$\mathcal{OIP} = \mathcal{IP} \cap \mathcal{P}/\text{poly}\,, \text{ which equals } \mathcal{PSPACE} \cap \mathcal{P}/\text{poly}.$$

This equality can be interpreted as saying that $\mathcal{OIP}$ is a powerful restriction of $\mathcal{P}$/poly. It also implies that proving super-polynomial circuit lower bounds for $\mathcal{PSPACE}$ is equivalent to separating $\mathcal{OIP}$ from $\mathcal{IP}$.

An additional example is the class $\mathcal{OMA}$, the input-oblivious version of $\mathcal{MA}$ (see Section 3). The class $\mathcal{OMA}$ may also be thought of as the class that results by restricting the advice of $\mathcal{P}$/poly to be verifiable in probabilistic polynomial-time (rather than in deterministic polynomial-time). Babai *et al.* [5] showed that if $\mathcal{EXP} \subseteq \mathcal{P}$/poly then $\mathcal{EXP} = \mathcal{MA}$, and their proof implicitly yields the stronger conclusion $\mathcal{EXP} = \mathcal{OMA}$ (as observed by [8]). The latter result may be viewed as saying that $\mathcal{OMA}$, while being a restriction of $\mathcal{P}$/poly, is still sufficiently powerful to contain $\mathcal{EXP}$ if $\mathcal{P}$/poly contains $\mathcal{EXP}$. This implies that in order to prove super-polynomial circuit lower bounds for $\mathcal{EXP}$, it suffices to separate $\mathcal{EXP}$ from $\mathcal{OMA}$.

Similarly, Impagliazzo *et al.* [12] showed that $\mathcal{NEXP} \subseteq \mathcal{P}$/poly implies $\mathcal{NEXP} = \mathcal{MA}$, and implicitly that $\mathcal{NEXP} \subseteq \mathcal{P}$/poly implies $\mathcal{NEXP} = \mathcal{OMA}$ (as observed by [8]). This result too may be interpreted as saying that in order to prove super-polynomial circuit lower bounds for $\mathcal{NEXP}$, it suffices to separate $\mathcal{NEXP}$ from $\mathcal{OMA}$.

We conclude that input-oblivious proof systems such as $\mathcal{ONP}$, $\mathcal{OMA}$, and $\mathcal{OIP}$ can be viewed as powerful restrictions of $\mathcal{P}$/poly, and therefore may serve as a useful target for research on lower bounds.

1.4 Organization and a piece of notation

In Section 2 we survey the study of input-oblivious NP-proof systems ($\mathcal{ONP}$), which goes back to Fortnow *et al.* [8]. The study of general input-oblivious interactive proof systems (i.e., $\mathcal{OIP}$) and the special case of input-oblivious $\mathcal{MA}$ are presented in Section 3. Other forms of input-oblivious probabilistic proof systems are investigated in Section 4. While Section 2 is mainly a reformulation of material that appeared in [8], Sections 3 and 4 are our main contributions.

Recurring notation. For an arbitrary set $S \subseteq \{0,1\}^*$ and $n \in \mathsf{N}$, we denote by S_n the set $S \cap \{0,1\}^n$.

2 Input-Oblivious NP-Proof Systems ($\mathcal{ONP}$)

The content of this section (except for Claim 2.2) appeared in [8], but our presentation is somewhat different. We also call attention to an open problem at the end of this section. In continuation to the discussion in the introduction, the input-oblivious version of NP-proof systems is defined as follows:

Definition 2.1 (input-oblivious NP-proofs – $\mathcal{ONP}$): *A set S has an* input-oblivious NP-proof system *if there exists a polynomial-time algorithm V and a polynomial p such that the following two conditions hold.*

Completeness: *For every $n \in \mathsf{N}$, there exists $w \in \{0,1\}^{p(n)}$ such that for every $x \in S_n \stackrel{\text{def}}{=} S \cap \{0,1\}^n$ it holds that $V(x,w) = 1$. We call w a* universal witness.

Soundness: *For every $x \notin S$ and every w, it holds that $V(x,w) = 0$.*

The class $\mathcal{ONP}$ consists of all sets having input-oblivious NP-proof systems.

Clearly, $\mathcal{ONP} \subseteq \mathcal{NP} \cap \mathcal{P}/\text{poly}$, since the "universal NP-witnesses" (guaranteed by the completeness condition) can be used as non-uniform advice. We also observe that universal NP-witnesses can be used to derandomize $\mathcal{RP}$.

Claim 2.2 (universal NP-witnesses for sets in $\mathcal{RP}$): *$\mathcal{RP} \subseteq \mathcal{ONP}$.*

Proof: Let $S \in \mathcal{RP}$. Using error reduction, we obtain a polynomial-time algorithm A and a polynomial p such that for every $x \in S$ it holds that $\Pr_{r\in\{0,1\}^{p(|x|)}}[A(x,r)=1] > 1 - 2^{-|x|}$ (whereas $A(x,r) = 0$ for every $x \notin S$ and r). Thus, there exists a string $r \in \{0,1\}^{p(n)}$ such that $A(x,r) = 1$ for every $x \in S_n$, which yields the desired universal NP-witness (w.r.t $V = A$). ■

Proving Theorem 1.1. We next establish the remaining claims of Theorem 1.1 (i.e., Items 1 and 2).

Claim 2.3 [8, Prop. 5]: *$\mathcal{ONP} = \mathcal{NP}$ if and only if $\mathcal{NP} \subset \mathcal{P}/\text{poly}$.*

Proof: Clearly, if $\mathcal{ONP} = \mathcal{NP}$, then $\mathcal{NP} = \mathcal{ONP} \subset \mathcal{P}/\text{poly}$. The proof of the opposite direction uses one main idea of the proof of the Karp–Lipton theorem (i.e., $\mathcal{NP} \subset \mathcal{P}/\text{poly}$ implies that the Polynomial-time Hierarchy collapses to its second level [13]). We follow the presentation in [10, Sec. 3.2.3], where the hypothesis is shown to yield polynomial-size circuits for finding NP-witnesses.

Specifically, consider any NP-complete set S, and recall that searching NP-witnesses for $x \in S$ is reducible to deciding S; that is, for any NP-witness relation R that characterizes S (i.e., $S = \{x : \exists w\ (x,w) \in R\}$), solving the search problem associated with R is reducible to deciding S (cf. [10, Thm. 2.16]). Now, assuming that $\mathcal{NP} \subset \mathcal{P}/\text{poly}$, it follows that this search problem can be solved by polynomial-sized circuits (i.e., by applying the said reduction and using the circuits guaranteed for deciding $S \in \mathcal{NP} \subset \mathcal{P}/\text{poly}$).

The input-oblivious NP-proof system for S will use these (witness finding) circuits as universal witnesses; that is, consider V such that $V(x,w) = 1$ if and only if w is a description of a circuit C_w and $(x, C_w(x)) \in R$. We use w as a universal witness for length n if it describes a poly(n)-size witness-finding circuit for instance length n.

Finally, since S is NP-complete (and $S \in \mathcal{ONP}$), it follows that $\mathcal{NP} = \mathcal{ONP}$. Specifically, we use the fact that if S' is Karp-reducible to a set in $\mathcal{ONP}$, then $S' \in \mathcal{ONP}$. This fact is obvious if the reduction is length-regular (i.e., it maps instances of the same length to instances of the same length). In general, when reducing S' to S, we may use as universal witnesses for S'_n the concatenation of universal witnesses for S_m for $m = 1, ..., \text{poly}(n)$. ■

Remark 2.4 (a crucial point in the proof of Claim 2.3): *The reader may wonder why the strategy used in the proof of Claim 2.3 does not yield $\mathcal{ONP} = \mathcal{NP} \cap \mathcal{P}/\text{poly}$. The point is that the reduction of the search problem of R to S uses the fact that S is NP-complete.*[1] *But $\mathcal{NP} \cap \mathcal{P}/\text{poly}$ is unlikely to contain NP-complete sets, since this would imply $\mathcal{NP} \subset \mathcal{P}/\text{poly}$, thus the hypothesis that $S \in \mathcal{NP} \cap \mathcal{P}/\text{poly}$ does not seem to imply that S is in $\mathcal{ONP}$. However, for any $S \in \mathcal{NP} \cap \mathcal{P}/\text{poly}$ with witness relation R, if the search problem R is reducible to S, then $S \in \mathcal{ONP}$.*

Claim 2.5 (mentioned without proof in [8]): *If $\mathcal{NE} \neq \mathcal{E}$* (resp., $\mathcal{NE} \not\subseteq \mathcal{BPE}$)*, then $\mathcal{ONP} \neq \mathcal{P}$* (resp., $\mathcal{ONP} \not\subseteq \mathcal{BPP}$).

Proof: As is well-known [6], $\mathcal{NE} \neq \mathcal{E}$ (resp., $\mathcal{NE} \not\subseteq \mathcal{BPE}$) if and only if there exists a unary set in $\mathcal{NP} \setminus \mathcal{P}$ (resp., $\mathcal{NP} \setminus \mathcal{BPP}$). Hence, it suffices to show that any unary set $S \in \mathcal{NP}$ is in $\mathcal{ONP}$. But this is obvious, since we may use the NP-witnesses for the unary members of S as universal NP-witnesses; that is, the universal witness for n-bit long inputs equals the NP-witness used for 1^n (and is defined arbitrarily if $1^n \notin S$). ■

On sparse sets. The argument used in the proof of Claim 2.5 can be generalized for demonstrating that every sparse NP-set is in $\mathcal{ONP}$, where a set S is sparse if $|S_n| \leq \text{poly}(n)$. The key idea is that if proving membership of any n-bit long string (in S_n) can be done by using one of poly(n)-many NP-witnesses, then concatenating these witnesses yields a universal NP-witness. The same argument can be applied to show that $\mathcal{NE} = \mathcal{ONE}$, where $\mathcal{ONE}$ is the universal witness analogue of $\mathcal{NE}$. The reason is that for sets in $\mathcal{ONE}$, the the universal witness is allowed to be exponentially long, and there are at most 2^n YES-instances of length n. An equivalent result, using different terminology, was stated in [22, Thm 8] (see the appendix).

Proposition 2.6 *The following holds:*

- *If $S \in \mathcal{NP}$ is sparse, then $S \in \mathcal{ONP}$.*
- *$\mathcal{NE} = \mathcal{ONE}$.*

[1]This reduction is a composition of a reduction of the search problem of R to deciding the auxiliary NP-set $S'_R = \{(x,w') : \exists w''\ (x, w'w'') \in R\}$ and a reduction of S'_R to $S = \{x : \exists w\ (x,w) \in R\}$. The latter reduction exists since S is NP-complete.

Proof: For the first part of the proposition, let $S \in \mathcal{NP}$ be a sparse set. Let p be a polynomial such that $|S_n| \leq p(n)$ for every n, and let V be a polynomial-time algorithm such that $x \in S$ if and only if $V(x,w)=1$ for some $w \in \{0,1\}^{\text{poly}(|x|)}$. Consider a procedure V' such that $V'(x,w')=1$ if and only if w' contains a substring w that is an NP-witness for the membership of x in the set S. Fixing an NP-witness w_x for each $x \in S$, let w'_n be the concatenation of all the w_x's for $x \in S_n$; that is, $w'_n = w_{x_1}, ..., w_{x_t}$, where $t \leq p(n)$ and x_i is the i^{th} string in S_n. It follows that w'_n is a universal witness for length n, and hence $S \in \mathcal{ONP}$.

For the second part of the proposition, we use the same argument. Let $S \in \mathcal{NE}$. As before, we construct a universal witness for S_n by concatenating the witnesses of all the YES-instances of S_n. There are at most 2^n such YES-instances, and the length of each witness is at most $2^{O(n)}$. Hence, the total length of the universal witness is at most $2^{O(n)}$, as required. ■

On co-sparse sets. Note that, while every co-sparse set is in $\mathcal{P}/\text{poly}$, it is unclear whether every co-sparse NP-set is in $\mathcal{ONP}$, where a set S is called co-sparse if $|S_n| \geq 2^n - \text{poly}(n)$. We mention that relative to a random oracle, there exists a co-sparse set in $\mathcal{NP} \setminus \mathcal{ONP}$ (e.g., viewing the random oracle as a sequence of random functions $F_n : [t(n)] \times \{0,1\}^n \to \{0,1\}^n$, consider the set $S_n = \{x : F_n^{-1}(x) \neq \emptyset\}$).[2]

Open Problem 2.7 : *Does $\mathcal{ONP}$ contain all co-sparse NP-sets?*

3 Input-Oblivious Interactive Proof Systems ($\mathcal{OIP}$)

In this section we introduce and study the input-oblivious version of $\mathcal{IP}$. When defining an input-oblivious version of $\mathcal{IP}$, we should make sure that the prover is oblivious to the input throughout the interaction. In particular, we should make sure that the verifier does not communicate the input to the prover.

The simplest way to guarantee this feature is to decouple the interaction into two stages: In the first stage, both parties are only presented with the length of the input, and in the second stage the verifier is given the actual input but is disconnected from the prover. Thus, the verifier is decomposed into two parts, denoted V_1 and V_2, and its decision regarding the input x is written as $V_2(x, (P, V_1)(1^{|x|}))$, where $(P, V_1)(1^n)$ denotes the output of V_1 after interacting with the prover P on common input 1^n. Note that the said output of V_1 may contain its entire view of the interaction with P, and that without loss of generality V_2 may be deterministic (since its coins may be tossed and recorded by V_1).

Definition 3.1 (input-oblivious interactive proofs – $\mathcal{OIP}$): *A set S has an* input-oblivious interactive proof system *if there exists a probabilistic polynomial-time interactive machine V_1 and a polynomial-time algorithm V_2 such that the following two conditions hold.*

[2]Each x is in S_n with probability $1-(1-2^{-n})^{t(n)\cdot 2^n} = 1-e^{-t(n)}$. Hence, for $t(n) = \ln(2^n/n) = \Theta(n)$, with probability $1-\exp(-\Omega(n))$ it holds that $|S_n| = 2^n - \Theta(n)$, since each n-bit string is in $\{0,1\}^n \setminus S_n$ with probability $n \cdot 2^{-n}$ and these events are independent. It follows that S is co-sparse with probability 1. To show that S is not in $\mathcal{ONP}$ relative to the random oracle, observe that accepting a string $x \in \{0,1\}^n$ requires querying the oracle F_n on some element of $F_n^{-1}(x)$. However, with probability $1-\exp(-\Omega(n))$ (over the choice of F_n), it holds that $|F_n^{-1}(x)| \leq n^2$, whereas the elements of $F_n^{-1}(x)$ are uniformly and independently distributed in $[t(n)] \times \{0,1\}^n$. Furthermore, the sets $F_n^{-1}(\cdot)$ corresponding to different n-bit strings are almost $2^{n/3}$-wise independent. Hence, an polynomial-time oracle machine that is given a generic input $x \in \{0,1\}^n$, oracle acess to a random F_n, and a poly(n)-bit long string w_n (which may depend on F_n but not on x) is unlikely to make a query in $F_n^{-1}(x)$.

completeness: *There exists a strategy P such that, for every $x \in S$, it holds that*

$$\Pr[V_2(x,(P,V_1)(1^{|x|}))=1] \geq 2/3.$$

If the latter probability always equals 1, then we say that the system has perfect completeness*.*

soundness: *For every $x \notin S$ and every strategy P, it holds that* $\Pr[V_2(x,(P,V_1)(1^{|x|})=1] \leq 1/3$.

The class $\mathcal{OIP}$ consists of all sets having input-oblivious interactive proof systems.

As in the case of $\mathcal{ONP}$, the soundness condition of $\mathcal{OIP}$ maintains the analogous condition of $\mathcal{IP}$. Moreover, note that when the verifier V_1 is deterministic, the class $\mathcal{OIP}$ collapses to $\mathcal{ONP}$, as expected.

Theorem 3.2 (on the power of input-oblivious interactive proofs): $\mathcal{OIP} = \mathcal{IP} \cap \mathcal{P}/\mathrm{poly}$. *Furthermore, each set in $\mathcal{IP} \cap \mathcal{P}/\mathrm{poly}$ has an input-oblivious interactive proof system* with perfect completeness.

Proof: To see that $\mathcal{OIP}$ is contained in $\mathcal{P}/\mathrm{poly}$, we first apply error reduction to an input-oblivious interactive proof system for any $S \in \mathcal{OIP}$ such that the error probability on instances of length n is smaller than 2^{-n}. Thus, there exists an output of V_1 (after interacting with P on 1^n), denoted y, such that for every $x \in \{0,1\}^n$ it holds that $x \in S$ if and only if $V_2(x,y) = 1$. This output (i.e., y) can be used as non-uniform advice, which implies that $S \in \mathcal{P}/\mathrm{poly}$.

We now assume that $S \in \mathcal{IP} \cap \mathcal{P}/\mathrm{poly}$, and let $\{C_n\}$ be a family of polynomial-size circuits deciding S. On common input 1^n, the (input oblivious) prover sends C_n to the verifier V_1, and proves to it that C_n is correct (i.e., that for every $x \in \{0,1\}^n$ it holds that $C_n(x) = 1$ if and only if $x \in S$). Note that the latter assertion can be verified in polynomial-space, and hence it can be proved by an interactive proof (with perfect completeness) [18, 19]. The output of V_1 equals C_n if V_1 was convinced by the proof, and is the identically zero circuit otherwise. Finally, on input x and y (representing V_1's output), algorithm V_2 outputs $C_y(x)$, where C_y is the circuit represented by the string y. Thus, $S \in \mathcal{OIP}$ (and furthermore S has an input-oblivious interactive proof with perfect completeness). ■

The class $\mathcal{OMA}$. The class $\mathcal{OMA}$, defined by [8], is the input-oblivious version of $\mathcal{MA}$, which in turn is a randomized version of $\mathcal{NP}$ (in which the final verification of witnesses is probabilistic). In terms of input-oblivious interactive proofs (i.e., $\mathcal{OIP}$), the class $\mathcal{OMA}$ contains sets having a uni-directional interactive proof system of perfect completeness (in which, first the prover sends a message, and then the verifier tosses some coins). We observe that Lautemann's argument [17], which has been used to show $\mathcal{BPP} \subseteq \mathcal{MA}$, allows showing that $\mathcal{BPP} \subseteq \mathcal{OMA}$. An equivalent result, using different terminology, was proved in [2, Prop. 4.1] (see the appendix).

Proposition 3.3 : *$\mathcal{BPP} \subseteq \mathcal{OMA}$.*

Proof: Let $S \in \mathcal{BPP}$ and consider an algorithm A such that $\Pr_{r\in\{0,1\}^{p(|x|)}}[A(x,r)=\chi_S(x)] > 1 - 2^{-|x|}$, where $\chi_S(x) = 1$ if $x \in S$ and $\chi_S(x) = 0$ otherwise. Recall that the standard argument asserts that for every $x \in S$ there exists $s_1, ..., s_{p(|x|)} \in \{0,1\}^{p(|x|)}$ such that for every $r \in \{0,1\}^{p(|x|)}$ it holds that $\bigvee_{i\in[p(|x|)]} A(x, r\oplus s_i) = 1$, whereas for any $x \notin S$ and every $s_1, ..., s_{p(|x|)} \in \{0,1\}^{p(|x|)}$ it holds that $\Pr_{r\in\{0,1\}^{p(|x|)}}[\bigvee_{i\in[p(|x|)]} A(x, r\oplus s_i) = 1]$ is smaller than $p(|x|)/2^{|x|}$. We just note that, for every

$n \in \mathsf{N}$, there exists $s_1, ..., s_{p(|x|)+|x|} \in \{0,1\}^{p(|x|)}$ such that for every $x \in S_n$ and every $r \in \{0,1\}^{p(|x|)}$ it holds that $\bigvee_{i \in [p(|x|)+|x|]} A(x, r \oplus s_i) = 1$ (This uses the same standard argument, but takes a union bound over all x's as well as over all r's). We conclude by taking $s_1, ..., s_{p(|x|)+|x|} \in \{0,1\}^{p(|x|)}$ to be the universal witness. ■

4 Input-Oblivious Versions of PCP and ZK

In this section, we consider input-oblivious versions of the classes $\mathcal{PCP}$ (Probabilistically Checkable Proofs) and $\mathcal{ZK}$ (Zero-Knowledge interactive proofs). In both cases, we provide evidence that the said classes extend beyond the obvious (e.g., beyond $\mathcal{P}$), but note that they are unlikely to contain all of $\mathcal{ONP}$.

4.1 Input-Oblivious PCP

For sake of simplicity, we focus on PCP systems of logarithmic[3] randomness complexity and constant query complexity, and identify such systems with the term PCP. Furthermore, we explicitly upper bound the proof length (by a polynomial), since such a bound does not follow automatically in our setting.[4]

Definition 4.1 (input-oblivious probabilistically checkable proofs – $\mathcal{OPCP}$): *A set S has an* input-oblivious PCP system *if there exists a probabilistic polynomial-time oracle machine V of logarithmic randomness complexity and constant query complexity such that the following two conditions hold.*

Completeness: *For every $n \in \mathsf{N}$ there exists a $\text{poly}(n)$-bit long string π_n such that, on input any $x \in S_n$ and access to the oracle π_n, machine V always accepts x.*

Soundness: *For every $x \notin S$ and every string π, on input x and access to oracle π, machine V rejects x with probability at least $\frac{1}{2}$.*

The class $\mathcal{OPCP}$ consists of all sets having input-oblivious PCP systems.

Clearly, $\mathcal{OPCP} \subseteq \mathcal{ONP}$, but the converse may not hold. Still, Claim 2.5 extends to $\mathcal{OPCP}$.

Proposition 4.2 (on the power of input-oblivious PCPs): *If $\mathcal{NE} \neq \mathcal{E}$ (resp., $\mathcal{NE} \not\subseteq \mathcal{BPE}$), then $\mathcal{OPCP} \neq \mathcal{P}$ (resp., $\mathcal{OPCP} \not\subseteq \mathcal{BPP}$).*

Proof: As in the proof of Claim 2.5, it suffices to show that every unary set $S \in \mathcal{NP}$ is in $\mathcal{OPCP}$. Applying the PCP Theorem (cf. [4, 3]) to the inputs in S along with corresponding NP-witnesses, we obtain the desired input-oblivious PCP; that is, the proof oracle for the single n-bit long string that may be in S is obtained by applying the standard transformation to $(1^n, w_n)$, where w_n is the NP-witness for $1^n \in S$. ■

[3]Here, "logarithmic" means $O(\log n)$.

[4]As pointed out by Thilo Mie, failing to upper bound the proof length yields a class that equals $\mathcal{PCP}$, since a universal exponentially long proof can contain the list of proofs for all inputs of length n.

Digest. The proof of Proposition 4.2 does not use the fact that $S \in \mathcal{ONP}$, but rather uses the fact that $S \in \mathcal{NP}$ is a unary set. This seems required since in standard PCP constructions the proof oracle depends on the input (and not only on the corresponding NP-witness). Thus, *it is not clear that a universal NP-witness yields a universal PCP proof oracle.* Note that it is not even clear whether $\mathcal{RP}$ is in $\mathcal{OPCP}$, although $\mathcal{RP} \subseteq \mathcal{ONP}$ (and clearly $\mathcal{P} \subseteq \mathcal{OPCP}$). On the other hand, we show (next) that any sparse NP-set is in $\mathcal{OPCP}$.

Theorem 4.3 (sparse NP-sets are in OPCP): *If $S \in \mathcal{NP}$ is sparse, then $S \in \mathcal{OPCP}$.*

The claim extends to sets that are Karp-reducible to a sparse NP-set.

Proof: By Proposition 2.6, it holds that $S \in \mathcal{ONP}$, but as noted above, the proof uses more than this fact. Instead, as in the proof of Proposition 2.6, we consider a single NP-witness w_x for each $x \in S$, and let π_x denote the corresponding proof oracle obtained by applying the PCP Theorem to the pair (x, w_x). It is tempting to let the proof oracle be the concatenation of the proof oracles obtained for each string in S_n, but if the verifier does not know the index of x in S_n then it does not know which portion of the oracle to access. We solve this problem by using a suitable hashing scheme that maps elements of S_n to indices in $[3p(n)]$, where $|S_n| \leq p(n) = \text{poly}(n)$, such that (w.h.p.) no two (specific) elements are mapped to the same index.

Specifically, we use a poly(n)-size family of efficiently computable hashing functions, H_n, that map $\{0,1\}^n$ to $[3p(n)]$ such that for every two distinct $a, b \in \{0,1\}^n$ it holds that $\Pr_{h \in H_n}[h(a) = h(b)] < 1/2p(n)$; such constructions are presented in [11, 15, 16, 20]. On input $x \in \{0,1\}^n$, the verifier selects uniformly $h \in H_n$, and accesses the portion of the proof oracle that corresponds to the index $(h, h(x)) \in [\text{poly}(n)] \times [3p(n)]$. This portion of the oracle is supposed to contain a proof that there exists $z \in S_n$ such that $h(z) = v$, where $v \leftarrow h(x)$. Note that the latter NP-assertion refers only to h and v (and n), and so we may use any NP-witness for it (and obtain a corresponding PCP oracle proof). We stress that this portion of the proof oracle is not the aforementioned π_z, but it is rather a proof oracle that is derived for the NP-assertion *there exists $z \in S_n$ such that $h(z) = v$*. Note that this assertion corresponds to the pair $(h, v) \in [\text{poly}(n)] \times [3p(n)]$.

Hence, the completeness condition is satisfied by a proof oracle that is a concatenation of proofs for the various possible values of (h, v), whereas on input $x \in S_n$ the verifier always accesses a portion that corresponds to a valid assertion (since it uses $v = h(x)$). The soundness condition holds because any $x \in \{0,1\}^n \setminus S_n$ is mapped with constant probability to an h-image (for a random $h \in H_n$) that has no h-preimage in S_n.

The last statement follows by taking the union bound over S_n. Note that we do not need to take the union bound over $\{0,1\}^n \setminus S_n$, since the hash function h needs only be good (i.e., $h^{-1}(h(x)) \cap S_n = \emptyset$) for the specific input $x \notin S_n$ that is given to the verifier. ■

Let us end this subsection by re-iterating that, while $\mathcal{P} \subseteq \mathcal{OPCP}$ clearly holds, it is unclear whether $\mathcal{RP}$ is in $\mathcal{OPCP}$ (although $\mathcal{RP} \subseteq \mathcal{ONP}$). More generally, we ask:

Open Problem 4.4 : *Does $\mathcal{OPCP} = \mathcal{ONP}$?*

4.2 Input-Oblivious ZK

The class $\mathcal{OZK}$ consists of sets having an input-oblivious interactive proof system in which the prescribed prover is zero-knowledge in the standard (complexity-theoretic) sense.[5] This definition

[5]The standard (complexity theoretic) definition of zero-knowledge requires efficient simulation of the view of the prescribed verifier (of the interaction with the prover); a stronger definition, commonly used in cryptography (cf. [9, Sec. 4.3.1]), requires efficient simulation of the view of arbitrary probabilistic polynomial-time adversaries. We note that our positive results extend also to the general (i.e., adversarial verifier) notion of zero-knowledge.

requires efficient simulation of the (prescribed) verifier's view of the interaction, based solely on the verifier's actual input. Indeed, here we refer to the verifier as the combination of the two stages, denoted V_1 and V_2 in Definition 3.1, and note that this combined verifier gets the actual input (rather than merely its length).[6] Still the view of the verifier is actually the view of its first stage (i.e., V_1), which is its sole interactive part, and in this part the common input is only the length (presented in unary). Denoting the view of V_1 when interacting with P on common input 1^n by $\mathtt{VIEW}_{V_1}^{P}(1^n)$, we obtain the following definition.

Definition 4.5 (input-oblivious zero-knowledge interactive proofs – $\mathcal{OZK}$): *Let S, V_1 and V_2 be as in Definition 3.1; that is, suppose that V_1 and V_2 are the two parts of an input-oblivious interactive proof system for S. We say that this proof system is* zero-knowledge *if for some prover strategy P that satisfies the completeness condition in Definition 3.1 there exists a probabilistic polynomial-time machine M* (called the simulator) *such that the probability ensembles $\{\mathtt{VIEW}_{V_1}^{P}(1^{|x|})\}_{x\in S}$ and $\{M(x)\}_{x\in S}$ are computationally indistinguishable. That is, for every family of polynomial-size circuits $\{C_n\}_{n\in\mathbb{N}}$ and for every positive polynomial p and all sufficiently long $x \in S$, it holds that*

$$\left|\Pr[C_{|x|}(M(x))=1] \;-\; \Pr[C_{|x|}(\mathtt{VIEW}_{V_1}^{P}(1^{|x|}))=1]\right| \;<\; 1/p(|x|).$$

(Note that this definition is equivalent to one in which the circuits obtain also x as input.)

Clearly, $\mathcal{BPP} \subseteq \mathcal{OZK}$ (since any set in $\mathcal{BPP}$ has an input-oblivious interactive proof system in which the prescribed prover does nothing, and hence can easily be simulated). It turns out that $\mathcal{OZK}$ is likely to extend beyond $\mathcal{BPP}$ (see Proposition 4.7), but this occurs only in the case of sets for which it is hard to find YES-instances of any desired length (see Proposition 4.6).

Proposition 4.6 (on the limitation of input-oblivious zero-knowledge proofs): *If $S \in \mathcal{OZK}$ and there exists a probabilistic polynomial-time algorithm A such that* $\Pr[A(1^n) \in S_n] \geq 2/3$ *holds for all sufficiently large n, then $S \in \mathcal{BPP}$.*

Proof: We may weaken the definition of zero-knowledge, and only consider simulating the output of the first stage (rather than the verifier's view of this stage). That is, referring to the notation in Definition 4.5, we consider the requirement that, on input $x \in S$, one can efficiently simulate $(P, V_1)(1^{|x|})$; that is, there exists a probabilistic polynomial-time machine M such that $\{M(x)\}_{x\in S}$ and $\{(P, V_1)(1^{|x|})\}_{x\in S}$ are computationally indistinguishable (by polynomial-size circuits). Let S and A be as in the hypothesis, and let P, V_1, M be as above (and V_2 as in Definition 4.5). By using error-reduction (via sequential repetitions of V_1), we may assume (w.l.o.g.) that the interactive proof system has error probability at most 0.1 (rather than at most 1/3). Using the guarantee regarding M, for all but finitely many $z \in S$ and all $x \in \{0,1\}^{|z|}$, it holds that

$$\left|\Pr[V_2(x, M(z))=1] \;-\; \Pr[V_2(x, (P, V_1)(1^{|z|}))=1]\right| \;<\; 0.01,$$

because otherwise x can be incorporated in a polynomial-size circuit that distinguishes $M(z)$ from $(P, V_1)(1^{|z|})$. Thus, for all but finitely many x, it holds that

$$\left|\Pr[V_2(x, M(A(1^{|x|})))=1] \;-\; \Pr[V_2(x, (P, V_1)(1^{|x|}))=1]\right| \;<\; 0.35,$$

because $\Pr[A(1^{|x|})\in S_{|x|}] \geq 2/3$. This suggests an efficient probability decision procedure for S: *On input x, invoke $V_2(x, M(A(1^{|x|})))$, and rule accordingly.* Observe that this decision procedure has error probability at most $0.1 + 0.35 = 0.45$, because $\Pr[V_2(x, (P, V_1)(1^{|x|})) \neq \chi_S(x)] \leq 0.1$ (where $\chi_S(x) = 1$ if $x \in S$ and $\chi_S(x) = 0$ otherwise). It follows that $S \in \mathcal{BPP}$. ■

[6] A stronger requirement (which mandates simulating the first stage based solely on the length of the actual input) is discussed in Remark 4.9.

Proposition 4.7 (on the power of input-oblivious zero-knowledge proofs): *If $S \in \mathcal{ZK}$ and $|S_n| \leq 1$ for all sufficiently large n, then $S \in \mathcal{OZK}$. Thus, if $\mathcal{NE} \not\subseteq \mathcal{BPE}$ and one-way functions exist, then $\mathcal{OZK} \not\subseteq \mathcal{BPP}$.*

Proof: We first consider an arbitrary set $S \in \mathcal{ZK}$ such that $|S_n| \leq 1$, and show that it is in $\mathcal{OZK}$. On input 1^n, the prover first determines the unique n-bit string in S_n (or halts if no such string exists), sends it to the verifier (i.e., V_1), and then the two parties proceed using the standard zero-knowledge proof (i.e., applying it to the n-bit long string sent by the prover). In the second stage, the verifier (i.e., V_2) checks whether the actual input equals the n-bit long string sent by the prover (at the beginning of the interaction). Specifically, suppose that $(P, V_1)(1^n)$ outputs (z, σ), where z represents the n-bit long string sent by the prover and $\sigma \in \{0,1\}$ is the verifier's decision at the end of the zero-knowledge proof. Then, $V_2(x, (z, \sigma)) \stackrel{\text{def}}{=} \sigma \wedge (x = z)$. This interaction is simulated by using the actual input (i.e., x) for the first message and simulating the zero-knowledge proof using the guaranteed simulator. Thus, $S \in \mathcal{OZK}$.

Now, assuming $\mathcal{NE} \not\subseteq \mathcal{BPE}$ (and the existence of one-way functions), we obtain a set $S \in \mathcal{ZK} \setminus \mathcal{BPP}$ such that $|S_n| \leq 1$ by considering a unary set in $\mathcal{NP} \setminus \mathcal{BPP}$ and employing a standard zero-knowledge proof for sets in $\mathcal{NP}$. Specifically, as in the proof of Claim 2.5, we use the fact that $\mathcal{NE} \not\subseteq \mathcal{BPE}$ implies the existence of a unary set S in $\mathcal{NP} \setminus \mathcal{BPP}$ (cf. [6]). Recall that the standard construction of zero-knowledge proofs for sets in $\mathcal{NP}$ uses any one-way function [9, Sec. 4.4]. Hence, $S \in \mathcal{ZK}$ and $|S_n| \leq 1$ for all n, and $S \in \mathcal{OZK}$ by the foregoing paragraph. ■

Remark 4.8 ($\mathcal{OZK}$ may extend beyond $\mathcal{ONP}$): *While $\mathcal{OZK} \subseteq \mathcal{OIP}$ holds trivially, assuming that $\mathcal{NE} \neq \mathcal{ESPACE}$ yields that $\mathcal{OZK}$ extends beyond $\mathcal{NP}$. Analogously to the proof of Proposition 4.7, the foregoing assumption yields a unary set in $\mathcal{PSPACE} \setminus \mathcal{NP}$, and using zero-knowledge proofs for sets in $\mathcal{IP}$* (cf. [9, Thm. 4.4.12]) *we are done.*

Remark 4.9 (strong zero-knowledge): *We say that an input-oblivious interactive proof system is* strongly zero-knowledge *if one can efficiently simulate the verifier's view of the first stage based solely on $1^{|x|}$* (rather than based on x). *It is easy to see that such proof systems exist only for sets in $\mathcal{BPP}$, even if it is only required to efficiently simulate the verifier's output of the first stage* (i.e., $(P, V_1)(1^{|x|})$) *based on $1^{|x|}$. This holds by following the proof of Proposition 4.6, except that here there is no need to find a string $z \in S_{|x|}$* (because the simulator is supposed to work based solely on $|x|$).

Acknowledgments

We thank Andrew Drucker for calling our attention to [8]. We also thank Scott Aaronson for calling our attention to [1]. We are grateful to Thilo Mie for calling our attention to the fact that the oracle length should be explicitly upper bounded in Definition 4.1. Finally, we thank the anonymous referees for numerous comments which improved the presentation of this work. In particularm we thank one of the referees for bringing the works of [21, 14, 22, 2] to our attention.

References

[1] S. Aaronson. The Learnability of Quantum States. In *Proceedings of the Royal Society* A463(2088), 2007.

[2] V. Arvind, J. Köbler and R. Schuler On Helping and Interactive Proof Systems. *Algorithms and Computation, 5th International Symposium, ISAAC '94*, 137-145, 1994

[3] S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy. Proof Verification and Intractability of Approximation Problems. *JACM*, Vol. 45, pages 501–555, 1998. Preliminary version in *33rd FOCS*, 1992.

[4] S. Arora and S. Safra. Probabilistic Checkable Proofs: A New Characterization of NP. *JACM*, Vol. 45, pages 70–122, 1998. Preliminary version in *33rd FOCS*, 1992.

[5] L. Babai, L. Fortnow and C. Lund. Non-Deterministic Exponential Time has Two-Prover Interactive Protocols. *Computational Complexity*, Vol. 1, pages 3–40, 1991.

[6] R.V. Book. Tally Languages and Complexity Classes. *Information and Control*, Vol. 26 (2), pages 186-193, 1974.

[7] V.T. Chakaravarthy and S. Roy. Oblivious Symmetric Alternation. In *23rd STACS*, Springer LNCS 3884, pages 230–241, 2006.

[8] L. Fortnow, R. Santhanam, and R. Williams. Fixed-Polynomial Size Circuit Bounds. In *24th IEEE Conference on Computational Complexity*, pages 19–26, 2009.

[9] O. Goldreich. *Foundation of Cryptography – Basic Tools.* Cambridge University Press, 2001.

[10] O. Goldreich. *Computational Complexity: A Conceptual Perspective.* Cambridge University Press, 2008.

[11] O. Goldreich and A. Wigderson. Tiny Families of Functions with Random Properties: A Quality–Size Trade–off for Hashing. *Journal of Random structures and Algorithms*, Vol. 11 (4), pages 315–343, 1997. Preliminary version in *26th STOC*, 1994.

[12] R. Impagliazzo, V. Kabanets and A. Wigderson. In Search of an Easy Witness: Exponential Time vs. Probabilistic Polynomial Time. In *16th IEEE Conference on Computational Complexity*, pages 2–12, 2001.

[13] R.M. Karp and R.J. Lipton. Some Connections Between Nonuniform and Uniform Complexity Classes. In *12th STOC*, pages 302-309, 1980.

[14] Ker-I Ko On helping by robust oracle machines. *Theoretical Computer Science*, 52:15-36, 1987.

[15] H. Krawczyk. LFSR-based Hashing and Authentication. In *CRYPTO'94*, LNCS (Vol. 839), Springer, pages 129–139, 1994.

[16] H. Krawczyk. New Hash Functions For Message Authentication. In *EuroCrypt'95*, LNCS (Vol. 921), Springer, pages 301–310, 1995.

[17] C. Lautemann. BPP and the Polynomial Hierarchy. *IPL*, Vol. 17, pages 215–217, 1983.

[18] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic Methods for Interactive Proof Systems. *JACM*, Vol. 39, No. 4, pages 859–868, 1992. Preliminary version in *31st FOCS*, 1990.

[19] A. Shamir. IP = PSPACE. *JACM*, Vol. 39, No. 4, pages 869–877, 1992. Preliminary version in *31st FOCS*, 1990.

[20] A. Srinivasan and D. Zuckerman. Computing with Very Weak Random Sources. *SICOMP*, Vol. 28 (4), pages 1433–1459, 1999. Preliminary version in *35th FOCS*, 1994.

[21] U. Schöning Robust algorithms: a different approach to oracles. *Theoretical Computer Science*, 40:57-66, 1985.

[22] U. Schöning Robust oracle machines. *Mathematical Foundations of Computer Science, MFCS'88*, 93-106, 1988.

A On robust oracle machines and computationally-bounded provers

In this appendix, we compare the results that are discussed in this paper with related works of Schöning and Ko on robust oracle machines [21, 14, 22] and of Arvind et. al. on computationally-bounded provers [2].

A robust oracle machine [21] is an oracle machine that always accepts the same language, regardless of the oracle that it is given. Schöning [21] considered such machines that run in polynomial-time if given access to a "helpful" oracle, but may have a longer running time if given access to an "unhelpful" oracle. This notion bears strong resemblance to proof systems, with the helpful and unhelpful oracles corresponding to honest and dishonest provers respectively. Indeed, the motivation of [21] was to construct an oracle machine that works even if the oracle makes mistakes, and hence can not be trusted.

Ko [14] considered a variant of this notion in which the helpful oracle only needs to help the machine on YES-instances, which is even more similar to the standard definitions of proof systems. He defined the class $\mathcal{P}_{1-help}$, which is the class of robust oracle machines that accept YES-instances in polynomial time if given oracle access to a helpful oracle, but may have a longer running time otherwise. Schöning [22] defined a randomized version of this class, namely $\mathcal{BPP}_{1-help}$. Furthermore, [14] and [22] observed that that $\mathcal{P}_{1-help} = \mathcal{NP}$ and that $\mathcal{BPP}_{1-help} = \mathcal{MIP}$, respectively.

The aforementioned works also studied variants of those classes when the helpful oracles are required to be computationally bounded. In particular:

- Schöning [22] considered the class $\mathcal{P}_{1-help}(\mathcal{SPARSE})$, the variant of $\mathcal{P}_{1-help}$ in which the helpful oracle is a sparse language. It is not hard to see that $\mathcal{P}_{1-help}(\mathcal{SPARSE})$ is equivalent to the class $\mathcal{ONP}$, which was discussed in Section 2. Specifically, he observed that sparse $\mathcal{NP}$-languages belong to $\mathcal{P}_{1-help}(\mathcal{SPARSE})$, which is equivalent to the first item of Proposition 2.6.
- Ko [14] observed that $\mathcal{NP}$ is equivalent to $\mathcal{P}_{1-help}(\mathcal{NP})$, in which the helpful oracle belongs to $\mathcal{NP}$. This observation implies Claim 2.3.

Arvind et. al [2] studied interactive proof systems in which the provers are computationally bounded. In particular, they studied the class $\mathcal{MIP}[\mathcal{P}/\text{poly}]$, which is the class of multi-prover interactive proofs where the provers are polynomial-size circuits. They observed that $\mathcal{MIP}[\mathcal{P}/\text{poly}] = \mathcal{BPP}_{1-help}(\mathcal{P}/\text{poly})$. Using the latter equivalence, it is not hard to see that both classes are equal to the class $\mathcal{OMA}$, which was discussed in Section 3. Moreover, they made the following observations:

- $\mathcal{MIP}[\mathcal{P}/\text{poly}]$ contains $\mathcal{BPP}$, which is the same as our Proposition 3.3.
- $\mathcal{MIP}[\mathcal{P}/\text{poly}]$ is contained in $\mathcal{P}/\text{poly}$, which extends the observation that $\mathcal{ONP} \subseteq \mathcal{P}/\text{poly}$, which is discussed in Section 2.
- $\mathcal{MIP}[\mathcal{P}/\text{poly}]$ is not contained in $\mathcal{BPP}$ unless $\mathcal{NE}$ is contained $\mathcal{BPE}$, which extends Claim 2.5.