Electronic Colloquium on Computational Complexity
ECCC TECHNICAL REPORTS SERIES 1994 REPORT NR: TR94-007

Computational Complexity and Knowledge Complexity[†]

Oded Goldreich[‡] Rafail Ostrovsky[§] Erez Petrank[¶]

Received *November 21, 1994*

Abstract. We study the computational complexity of languages which have interactive proofs of logarithmic knowledge complexity. We show that all such languages can be recognized in $\mathcal{BPP}^{\mathcal{NP}}$. Prior to this work, for languages with greater-than-zero knowledge complexity (and specifically, even for knowledge complexity 1) only trivial computational complexity bounds (i.e., recognizability in $\mathcal{PSPACE} = \mathcal{IP}$) were known. In the course of our proof, we relate statistical knowledge-complexity with perfect knowledge-complexity; specifically, we show that, for the honest verifier, these hierarchies coincide, up to a logarithmic additive term (i.e., $\mathcal{SKC}(k(\cdot)) \subseteq \mathcal{PKC}(k(\cdot) + \log(\cdot))$).

Keywords: Zero-Knowledge, Interactive Proofs, Knowledge Complexity, Randomness, Complexity Classes, Cryptography

[†] An extended abstract of this paper appeared in the *26th ACM Symposium on Theory of Computing* (STOC 94), held in Montreal, Quebec, Canada, May 23-25, 1994.

[‡] Department of Applied Mathematics and Computer Science, Weizmann Institute of Science, Rehovot, Israel. E-mail: oded@wisdom.weizmann.ac.il. Supported by grant no. 92-00226 from the United States — Israel Binational Science Foundation, Jerusalem, Israel.

[§] Computer Science Division, University of California at Berkeley, and International Computer Science Institute, Berkeley, CA 94720. E-mail: rafail@cs.Berkeley.EDU. Supported by an NSF Postdoctoral Fellowship and ICSI.

[¶] Computer Science Department, Technion – Israel Institute of Technology, Haifa 32000, Israel. E-mail: erez@cs.technion.ac.il.

1 Introduction

The notion of knowledge-complexity was introduced in the seminal paper of Goldwasser Micali and Rackoff [GMR-85, GMR-89]. Knowledge-complexity (KC) is intended to measure the *computational advantage* gained by interaction. Satisfactory formulations of knowledge-complexity, for the case that it is not zero, have recently appeared in [GP-91]. A natural suggestion, made by Goldwasser, Micali and Rackoff, is to classify languages according to the knowledge-complexity of their interactive-proofs [GMR-89]. We feel that it is worthwhile to give this suggestion a fair try.

The lowest level of the knowledge-complexity hierarchy is the class of languages having interactive proofs of knowledge-complexity zero, better known as zero-knowledge. Actually, there are three hierarchies extending the three standard definitions of zero-knowledge; namely *perfect*, *statistical* and *computational*. Let us denote the corresponding hierarchies by $\mathcal{PKC}(\cdot)$, $\mathcal{SKC}(\cdot)$, and $\mathcal{CKC}(\cdot)$. Assuming the existence of one-way functions, the third hierarchy collapses, namely $\mathcal{CKC}(0) = \mathcal{IP} = \mathcal{CKC}(\text{poly})$ [GMW-86, IY-87, B+ 88]. Put differently, the zero level of *computational* knowledge-complexity extends to the maximum possible. Anyhow, in the rest of this paper we will be only interested in the other two hierarchies.

Previous works have provided information only concerning the zero level of these hierarchies. Fortnow has pioneered the attempts to investigate the computational complexity of (perfect/statistical) zero-knowledge [F-89], and was followed by Aiello and Hastad [AH-87]. Their results can be summarized by the following theorem that bounds the computational complexity of languages having zero-knowledge proofs.

Theorem [F-89, AH-87]:

$$\mathcal{SKC}(0) \subseteq \mathcal{AM} \bigcap \text{co-}\mathcal{AM}$$

Hence, languages having statistical zero-knowledge must lie in the second level of the polynomial-time hierarchy. Needless to say that $\mathcal{PKC}(k(\cdot)) \subseteq \mathcal{SKC}(k(\cdot))$, for any function k and in particular for $k \equiv 0$.

On the other hand, if we allow polynomial amount of knowledge to be revealed, then every language in $\mathcal{IP}$ can be proven.

Theorem [LFKN-90, Sh-90]:

$$\mathcal{PKC}(\text{poly}(\cdot)) = \mathcal{IP} = \mathcal{PSPACE}$$

As indicated in [GP-91], the first equality is a property of an adequate definition (of knowledge complexity) rather than a result.

In this paper we study the class of languages that have interactive-proofs with logarithmic knowledge-complexity. In particular, we bound the computational complexity of such languages, showing that they can be recognized by probabilistic polynomial-time machines with access to an NP oracle.

Main Theorem:

$$\mathcal{SKC}(O(\log(\cdot))) \subseteq \mathcal{BPP}^{\mathcal{NP}}$$

We recall that $\mathcal{BPP}^{\mathcal{NP}}$ is contained in the third level of the polynomial-time hierarchy ($\mathcal{PH}$). It is believed that $\mathcal{PH}$ is a proper subset of $\mathcal{PSPACE}$. Thus, assuming $\mathcal{PH} \subsetneq \mathcal{PSPACE}$, our result yields the first proof that there exist languages in $\mathcal{PSPACE}$ which cannot be proven by an interactive-proof that yields $O(\log n)$ bits of knowledge. In other words, there exist languages which do have interactive proofs but only interactive proofs with super-logarithmic knowledge-complexity.

Prior to our work, there was no solid indication[1] that would contradict the possibility that all languages in $\mathcal{PSPACE}$ have interactive-proofs which yield only *one bit of knowledge*. The only attempt to bound the computational complexity of languages having interactive proofs of low knowledge-complexity was done by Bellare and Petrank. Yet, their work refers only to languages having interactive proofs that are both of *few rounds* and of *low knowledge complexity* [BP-92]. Specifically, they showed that if a language L has a $r(n)$-round interactive-proof of knowledge-complexity $O(\frac{\log n}{r(n)})$ then the language can be recognized in $\mathcal{BPP}^{\mathcal{NP}}$.

Our proof of the Main Theorem consists of two parts. In the first part, we show that the procedure described by Bellare and Petrank [BP-92] suffices for recognizing languages having interactive proofs of logarithmic *perfect* knowledge complexity. To this end, we use a more careful analysis than the one used in [BP-92]. In the second part of our proof we transform interactive proofs of *statistical* knowledge complexity $k(n)$ into interactive proofs of *perfect* knowledge complexity $k(n)+\log n$. This transformation refers only to knowledge-complexity with respect to the honest verifier, but this suffices since the first part of our proof applies to the knowledge-complexity with respect to the honest verifier. Yet, the transformation is interesting for its own sake, and a few words are in place.

The question of whether statistical zero-knowledge equals perfect zero-knowledge is one of the better known open problems in this area. The question has been open also for the case of zero-knowledge with respect to the honest verifier. We show that for every poly-time computable function $k: \mathsf{N} \mapsto \mathsf{N}$ (and in particular for $k \equiv 0$)

$$\mathcal{SKC}(k(\cdot)) \subseteq \mathcal{PKC}(k(\cdot) + \log(\cdot))$$

This result may be considered an indication that these two hierarchies may collide.

Techniques Used

As stated above, the first part of our proof consists of presenting a more careful analysis of an existing procedure, namely the procedure suggested by Bellare and Petrank in [BP-92]. Their procedure, in turn, is a culmination of two sequences of works discussed bellow.

The first sequence originates in Fortnow's definition of a simulator-based prover [F-89]. Fortnow [F-89], and consequently Aiello and Hastad [AH-87], used the simulator-based prover in order to infer, by way of contradiction, bounds on the sizes of specific sets. A more explicit usage of the simulator-based prover was introduced by Bellare, Micali and Ostrovsky [BMO-90]; specifically, they have suggested to use a PSPACE-implementation of the

[1] Alas, if one had been willing to assume that all languages in $\mathcal{PSPACE}$ have interactive proofs of *logarithmically many rounds*, an assumption that we consider unreasonable, then the result in [BP-92] would have yielded a proof that $\mathcal{PSPACE}$ is not contained in $\mathcal{SKC}(1)$, provided (again) that $\mathcal{PH} \subsetneq \mathcal{PSPACE}$.

simulator-based prover, instead of using the original prover (of unbounded complexity) witnessing the existence of a zero-knowledge interactive proof system. (Thus, they obtained a bound on the complexity of provers required for zero-knowledge proof systems.) Ostrovsky [Ost-91] suggested to use an implementation of the interaction between the verifier and the simulation-based prover as a procedure for deciding the language. Furthermore, assuming that one-way functions do not exist, he used "universal extrapolation" procedures of [ILu-90, ILe-90] to approximate the behavior of the simulator-based prover. (Thus, assuming that one-way function do not exists, he presented an efficient procedure that decides languages in $\mathcal{SKC}(0)$ and inferred that one-way functions are essential to the non-triviality of statistical zero-knowledge). Bellare and Petrank distilled the decision procedure from the context of one-way functions, showing that the simulator-based prover can be implemented using a perfect universal extrapolator (also known as a "uniform generation" procedure) [BP-92]. The error in the implementation is directly related to the deviation of the uniform generation procedure.

The second sequence of works deals with the two related problems of approximating the size of sets and uniformly generating elements in them. These problems were related by Jerrum et. al. [JVV-86]. Procedures for approximating the size of sets were invented by Sipser [Si-83] and Stockmeyer [St-83], and further improved in [GS-89, AH-87], all using the "hashing paradigm". The same hashing technique, is the basis of the "universal extrapolation" procedures of [ILu-90, ILe-90]. However, the output of these procedures deviates from the objective (i.e., uniform distribution on the target set) by a non-negligible amount (i.e., 1/poly(T) when running for time T). On the other hand, Jerrum et. al. have also pointed out that (perfect) uniform generation can be done by a $\mathcal{BPP}^{\Sigma_2^P}$-procedure [JVV-86]. Bellare and Petrank combined the hashing-based approximation methods with the ideas of [JVV-86] to obtain a $\mathcal{BPP}^{\mathcal{NP}}$-procedure for uniform generation with exponentially vanishing error probability [BP-92]. Actually, if the procedure is allowed to halt with no output with constant (or exponentially vanishing) probability, then its output distribution is exactly uniform on the target set.

Motivation for studying KC

In addition to the self-evident fundamental appeal of knowledge complexity, we wish to point out some practical motivation for considering knowledge-complexity greater than zero. In particular, cryptographic protocols that release a small (i.e., logarithmic) amount of knowledge may be of practical value, especially if they are only applied once or if one can obtain sub-additive bounds on the knowledge complexity of their repeated executions. Note that typically, a (single application of a) sub-protocol leaking logarithmically many bits (of knowledge) does not compromise the security of the entire protocol. The reason being that these (logarithmically many) bits can be guessed with non-negligible probability, which in turn means that any attack due to the "leaked bits" can be simulated with non-negligible probability without them.

But why use low knowledge-complexity protocols when one can use zero-knowledge ones (see, [GMW-86, GMW-87])? The reason is that the non-zero-knowledge protocols may be more efficient and/or may require weaker computational assumptions (see, for example, [OVY-91]).

Remarks

A remark *concerning two definitions.* Throughout the paper, $\mathcal{SKC}(k(\cdot))$ and $\mathcal{PKC}(k(\cdot))$ denote the classes of knowledge-complexity *with respect to the honest verifier.* Note that the Main Theorem is only strengthen by this, whereas the transformation (mentioned above) is indeed weaker. Furthermore, by an interactive proof we mean one in which the *error probability is negligible* (i.e., smaller than any polynomial fraction). A few words of justification appear in Section 2.

A remark *concerning Fortnow's paper* [F-89]. In course of this research, we found out that the proof that $\mathcal{SKC}(0) \subseteq$ co-$\mathcal{AM}$ as it appears in [F-89] is not correct. In particular, there is a flaw in the AM-protocol presented in [F-89] for the complement language (see Appendix A). However, the paper of Aiello and Hastad provides all the necessary machinery for proving Fortnow's result as well [AH-87, H-94]. Needless to say that the basic approach presented by Fortnow (i.e., looking at the "simulator-based prover") is valid and has inspired all subsequent works (e.g., [AH-87, BMO-90, Ost-91, BP-92, OW-93]) as well as the current one.

2 Preliminaries

Let us state some of the definitions and conventions we use in the paper. Throughout this paper we use n to denote the length of the input x. A function $f : \mathsf{N} \rightarrow [0,1]$ is called *negligible* if for every polynomial p and all sufficiently large n's $f(n) < \frac{1}{p(n)}$.

2.1 Interactive proofs

Let us recall the concept of interactive proofs, presented by [GMR-89]. For formal definitions and motivating discussions the reader is referred to [GMR-89]. A protocol between a (computationally unbounded) *prover* P and a (probabilistic polynomial-time) *verifier* V constitutes an interactive proof for a language L if there exists a negligible function $\epsilon : \mathsf{N} \rightarrow [0,1]$ such that

1. **Completeness:** If $x \in L$ then
$$\Pr\left[(P,V)(x) \, \text{accepts}\right] \; \geq \; 1 - \epsilon(n)$$

2. **Soundness:** If $x \notin L$ then for any prover P^*
$$\Pr\left[(P^*,V)(x) \, \text{accepts}\right] \; \leq \; \epsilon(n)$$

Remark: Usually, the definition of interactive proofs is robust in the sense that setting the error probability to be bounded away from $\frac{1}{2}$ does not change their expressive power, since the error probability can be reduced by repetitions. However, this standard procedure is not applicable when knowledge-complexity is measured, since (even sequential) repetitions may increase the knowledge-complexity. The question is, thus, what is the *right* definition. The definition used above is quite standard and natural; it is certainly less arbitrary then setting the error to be some favorite constant (e.g., $\frac{1}{3}$) or function (e.g., 2^{-n}). Yet, our techniques yield non-trivial results also in case one defines interactive proofs with non-negligible error probability (e.g., constant error probability). For example, languages having interactive proofs with error probability 1/4 and perfect knowledge complexity 1 are also in $\mathcal{BPP}^{\mathcal{NP}}$. For more details see Appendix B. Also note that we have allowed two-sided error probability; this strengthens our main result but weakens the statistical to perfect transformation[2].

2.2 Knowledge Complexity

Throughout the rest of the paper, we refer to knowledge-complexity *with respect to the honest verifier*; namely, the ability to simulate the (honest) verifier's view of its interaction with the prover. (In the stronger definition, one considers the ability to simulate the point of view of *any efficient verifier* while interacting with the prover.)

We let $(P,V)(x)$ denote the random variable that represents V's view of the interaction with P on common input x. The view contains the verifier's random tape as well as the sequence of messages exchanged between the parties.

We begin by briefly recalling the definitions of perfect and statistical zero-knowledge. A protocol (P,V) is *perfect zero-knowledge* (resp., *statistical zero-knowledge*) if there is a probabilistic polynomial time simulator M such that the random variable $M(x)$ is distributed identically to $(P,V)(x)$ (resp., the statistical difference between $M(x)$ and $(P,V)(x)$ is a negligible function in $|x|$).

Next, we present the definitions of perfect (resp., statistical) knowledge-complexity which we use in the sequel. These definitions extend the definition of perfect (resp., statistical) zero-knowledge, in the sense that knowledge-complexity zero is exactly zero-knowledge. Actually, there are two alternative formulations of knowledge-complexity, called the *oracle version* and the *fraction version*. These formulations coincide at the zero level and differ by at most an additive constant otherwise [GP-91]. For further intuition and motivation see [GP-91]. It will be convenient to use both definitions in this paper[3].

By the *oracle formulation*, the knowledge-complexity of a protocol (P,V) is the number of oracle (bit) queries that are needed to simulate the protocol efficiently.

Definition 2.1 (knowledge complexity — oracle version): *Let k: $\mathsf{N} \to \mathsf{N}$. We say that an interactive proof (P,V) for a language L has* perfect (resp., statistical) knowledge complexity $k(n)$ in the oracle sense *if there exists a probabilistic polynomial time oracle machine M and an oracle A such that:*

1. *On input $x \in L$, machine M queries the oracle A for at most $k(|x|)$ bits.*
2. *For each $x \in L$, machine M^A produces an output with probability at least $\frac{1}{2}$, and given that M^A halts with an output, $M^A(x)$ is* identically distributed (resp., statistically close) *to $(P,V)(x)$.*

[2] Suppose you had a transformation for the one-sided case. Then, given a two-sided interactive proof of some statistical knowledge complexity you could have transformed it to a one-sided error proof of the same knowledge complexity (cf., [GMS-87]). Applying the transformation for the one-sided case would have yielded an even better result.

[3] The analysis of the [BP-92] procedure is easier when using the fraction version, whereas the transformation from statistical to perfect is easier when using the oracle version.

In the *fraction formulation*, the simulator is not given any explicit help. Instead, one measures the density of the largest subspace of simulator's executions (i.e., coins) which is identical (resp., close) to the (P,V) distribution.

Definition 2.2 (knowledge complexity — fraction version): *Let* $\rho\colon\ \mathsf{N} \to (0,1]$. *We say that an interactive proof* (P,V) *for a language* L *has* perfect (resp., statistical) knowledge-complexity $\log_2(1/\rho(n))$ in the fraction sense *if there exists a probabilistic polynomial-time machine* M *with the following "good subspace" property. For any* $x \in L$ *there is a subset of* M*'s possible random tapes* S_x*, such that:*

1. *The set* S_x *contains at least a* $\rho(|x|)$ *fraction of the set of all possible coin tosses of* $M(x)$*.*
2. *Conditioned on the event that* $M(x)$*'s coins fall in* S_x*, the random variable* $M(x)$ *is* identically distributed (resp., statistically close) *to* $(P,V)(x)$*. Namely, for the perfect case this means that for every* $\bar{c}$

$$\mathrm{Prob}(M(x,\omega)=\bar{c}\,|\,\omega\in S_x) = \mathrm{Prob}((P,V)(x)=\bar{c})$$

where $M(x,\omega)$ *denotes the output of the simulator* M *on input* x *and coin tosses sequence* ω*.*

As mentioned above, these two measures are almost equal.

Theorem [GP-91]: The fraction measure and the oracle measure are equal up to an additive constant.

Since none of our results is sensitive to a difference of an additive constant in the measure, we ignore this difference in the subsequent definition as well as in the statement of our results.

Definition 2.3 (knowledge complexity classes):

- $\mathcal{PKC}(k(\cdot))$ = *languages having interactive proofs of perfect knowledge complexity* $k(\cdot)$*.*
- $\mathcal{SKC}(k(\cdot))$ = *languages having interactive proofs of statistical knowledge complexity* $k(\cdot)$*.*

2.3 The simulation based prover

An important ingredient in our proof is the notion of a simulation based prover, introduced by Fortnow [F-89]. Consider a simulator M that outputs conversations of an interaction between a prover P and a verifier V. We define a new prover P^*, called *the simulation based prover*, which selects its messages according to the conditional probabilities induced by the simulation. Namely, on a partial history h of a conversation, P^* outputs a message α with probability

$$\mathrm{Prob}(P^*(h)=\alpha) \stackrel{\mathrm{def}}{=} \mathrm{Prob}(M_{|h|+1}=h\circ\alpha \;\big|\; M_{|h|}=h)$$

where M_t denotes the distribution induced by M on t-long prefixes of conversations. (Here, the length of a prefix means the number of messages in it.)

It is important to note that the behavior of P^* is *not* necessarily close to the behavior of the original prover P. Specifically, if the knowledge complexity is greater than 0 and we consider the simulator guaranteed by the fraction definition, then P^* and P might have quite a different behavior. Our main objective will be to show that even in this case P^* still behaves in a manner from which we can benefit.

3 The Perfect Case

In this section we prove that the Main Theorem holds for the special case of *perfect* knowledge complexity. Combining this result with the transformation (Theorem 2) of the subsequent section, we get the Main Theorem.

Theorem 1 $\mathcal{PKC}(O(\log n)) \subseteq \mathcal{BPP}^{\mathcal{NP}}$

Our proof follows the procedure suggested in [BP-92], which in turn follows the approach of [F-89, BMO-90, Ost-91] while introducing a new uniform generation procedure which builds on ideas of [Si-83, St-83, GS-89, JVV-86] (see introduction).

Suppose that (P, V) is an interactive proof of perfect knowledge complexity $k(\cdot) = O(\log n)$ for the languages L, and let M be the simulator guaranteed by the fraction formulation (i.e., Definition 2.2). We consider the conversations of the original verifier V with the simulation-based-prover P^* (see definition in Section 2.3). We are going to show that the probability that the interaction (P^*, V) is accepting is negligible if $x \notin L$ and greater than a polynomial fraction if $x \in L$. Our proof differs from [BP-92] in the analysis of the case $x \in L$ (and thus we get a stronger result although we use the same procedure). This separation between the cases $x \notin L$ and $x \in L$ can be amplified by sequential repetitions of the protocol (P^*, V). So it remains to observe that we can sample the (P^*, V) interactions in probabilistic polynomial-time having access to an NP oracle. This observation originates from [BP-92] and is justified as follows. Clearly, V's part of the interaction can be produced in polynomial-time. Also, by the uniform generation procedure of [BP-92] we can implement P^* by a probabilistic polynomial time machine that has access to an NP oracle. Actually, the implementation may fail with negligible probability, but this does not matter. Thus, it remains only to prove the following lemma.

Lemma 3.1

1. *If $x \in L$ then the probability that (P^*, V) outputs an accepting conversation is at least $\frac{1}{2} \cdot 2^{-k}$.*
2. *If $x \notin L$ then the probability that (P^*, V) outputs an accepting conversation is negligible.*

Remark: In [BP-92], a weaker lemma is proven. Specifically, they show that the probability that (P^*, V) output an accepting conversation (on $x \in L$) is related to $2^{-k \cdot t}$, where t is the number of rounds in the protocol. Note that in our proof t could be an arbitrary polynomial number of rounds.

proof: The second part of the lemma follows from the soundness property as before. We thus concentrate on the first part. We fix an arbitrary $x \in L$ for the rest of the proof and allow ourselves not to mention it in the sequel discussion and notation. Let $k = k(|x|)$ and q be the number of coin tosses made by M. We denote by $\Omega \overset{\text{def}}{=} \{0,1\}^q$ the set of all possible coin tosses, and by S the "good subspace" of M (i.e., S has density 2^{-k} in Ω and for ω chosen uniformly in S the simulator outputs exactly the distribution of the interaction (P, V)).

Consider the conversations that are output by the simulator on $\omega \in S$. The probability to get such a conversation when the simulator is run on ω uniformly selected in Ω, is at least 2^{-k}. We claim that the probability to get these conversations in the interaction (P^*,V) is also at least 2^{-k}. This is not obvious, since the distribution produced by (P^*,V) may not be identical to the distribution produced by M on a uniformly selected $\omega \in \Omega$. Nor is it necessarily identical to the distribution produced by M on a uniformly selected $\omega \in S$. However, the prover's moves in (P^*,V) are distributed as in the case that the simulator selects ω uniformly in Ω, whereas the verifier's moves (in (P^*,V)) are distributed as in the case that the simulator selects ω uniformly in S. Thus, it should not be too surprising that the above claim can be proven.

However, we need more than the above claim: It is not enough that the (P^*,V) conversations have an origin in S, they must be *accepting* as well. (Note that this is not obvious since M simulates an interactive proof that may have two-sided error.) Again, the density of the accepting conversations in the "good subspace" of M is high (i.e., $\geq 1-\epsilon$), yet we need to show that this is the case also for the (P^*,V) interaction. Actually, we will show that the probability than an (P^*,V) conversation is accepting and "has an origin" in S is at least $\frac{1}{2}\cdot 2^{-k}$.

Let us begin the formal argument with some notations. For each possible history of the interaction, h, we define subsets of the random tapes of the simulator (i.e., subsets of Ω) as follows. Ω_h is the set of $\omega \in \Omega$ which cause the simulator to output a conversation with prefix h. S_h is the subset of ω's in Ω_h which are also in S. A_h is the set of ω's in S_h which are also accepting.

Thus, letting $M_t(\omega)$ denote the t-message long prefix output by the simulator M on coins ω, we get

$$\begin{aligned}
\Omega_h &\stackrel{\text{def}}{=} \{\omega : M_{|h|}(\omega)=h\} \\
S_h &\stackrel{\text{def}}{=} \Omega_h \cap S \\
A_h &\stackrel{\text{def}}{=} \{\omega \in S_h : M(\omega) \text{ is accepting}\}
\end{aligned}$$

Let C be a random variable representing the (P^*,V) interaction, and χ be an indicator so that $\chi(\bar{c})=1$ if the conversation $\bar{c}$ is accepting and $\chi(\bar{c})=0$ otherwise. Our aim is to prove that $\text{Prob}(\chi(C)=1) \geq \frac{1}{2}\cdot 2^{-k}$. Note that

$$\begin{aligned}
\text{Prob}(\chi(C)=1) &= \sum_{\bar{c}} \text{Prob}(C=\bar{c})\cdot\chi(\bar{c}) \\
&\geq \sum_{\bar{c}} \text{Prob}(C=\bar{c})\cdot\frac{|A_{\bar{c}}|}{|\Omega_{\bar{c}}|}
\end{aligned}$$

The above expression is exactly the expectation value of $\frac{|A_c|}{|\Omega_c|}$. Thus, we need to show that:

$$\text{Exp}_{\bar{c}}\left(\frac{|A_{\bar{c}}|}{|\Omega_{\bar{c}}|}\right) > \frac{1}{2}\cdot 2^{-k} \tag{1}$$

where the expectation is over the possible conversations $\bar{c}$ as produced by the interaction (P^*,V). Once Equation (1) is proven, we are done. Denote the empty history by λ. To prove Equation (1) it suffices to prove that

$$\text{Exp}_{\bar{c}}\left(\frac{|A_{\bar{c}}|}{|\Omega_{\bar{c}}|}\cdot\frac{|A_{\bar{c}}|}{|S_{\bar{c}}|}\right) \geq \frac{|A_\lambda|}{|\Omega_\lambda|}\cdot\frac{|A_\lambda|}{|S_\lambda|} \tag{2}$$

since using $\frac{|A_\lambda|}{|S_\lambda|} > \sqrt{\frac{1}{2}}$ and $\frac{|S_\lambda|}{|\Omega_\lambda|} \geq 2^{-k}$, we get

$$\begin{aligned}
\mathrm{Exp}_{\bar{c}}\left(\frac{|A_{\bar{c}}|}{|\Omega_{\bar{c}}|}\right) &\geq \frac{|A_\lambda|}{|\Omega_\lambda|} \cdot \frac{|A_\lambda|}{|S_\lambda|} \\
&= \left(\frac{|A_\lambda|}{|S_\lambda|}\right)^2 \cdot \frac{|S_\lambda|}{|\Omega_\lambda|} \\
&\geq \frac{1}{2} \cdot 2^{-k}
\end{aligned}$$

The proof of Equation (2) is by induction on the number of rounds. Namely, for each round i, we show that the expected value of $\frac{|A_h|}{|\Omega_h|} \cdot \frac{|A_h|}{|S_h|}$ over all possible histories h of i rounds (i.e., length i) is greater or equal to the expected value of this expression over all histories h' of $i-1$ rounds. In order to show the induction step we consider two cases:

1. the current step is by the prover (i.e., P^*); and
2. the current step is by the verifier (i.e., V).

In both cases we show, for any history h,

$$\mathrm{Exp}_m\left(\frac{|A_{hom}|}{|\Omega_{hom}|} \cdot \frac{|A_{hom}|}{|S_{hom}|}\right) \geq \frac{|A_h|}{|\Omega_h|} \cdot \frac{|A_h|}{|S_h|} \tag{3}$$

where the expectation is over the possible current moves m, given history h, as produced by the interaction (P^*, V).

Technical Claim

The following technical claim is used for deriving the inequalities in both cases.

Claim 3.2 *Let x_i, y_i, $1 \leq i \leq n$ be positive reals. Then,*

$$\sum_{i=1}^{n} \frac{{x_i}^2}{y_i} \geq \frac{(\sum_{i=1}^n x_i)^2}{\sum_{i=1}^n y_i}$$

Proof: The Cauchy-Schwartz Inequality asserts:

$$\left(\sum_{i=1}^{n} {a_i}^2\right) \cdot \left(\sum_{i=1}^{n} {b_i}^2\right) \geq \left(\sum_{i=1}^{n} a_i \cdot b_i\right)^2$$

Setting $a_i \stackrel{\text{def}}{=} \sqrt{y_i}$ (we can do this since y_i is positive) and $b_i \stackrel{\text{def}}{=} \frac{x_i}{a_i}$, and rearranging the terms, we get the desired inequality. □

Prover Step – denoted α

Given history h, the prover P^* sends α as its next message with probability $\frac{|\Omega_{ho\alpha}|}{|\Omega_h|}$. Thus,

$$\mathrm{Exp}_\alpha\left(\frac{|A_{ho\alpha}|}{|\Omega_{ho\alpha}|} \cdot \frac{|A_{ho\alpha}|}{|S_{ho\alpha}|}\right) = \sum_\alpha \frac{|\Omega_{ho\alpha}|}{|\Omega_h|} \cdot \frac{|A_{ho\alpha}|}{|\Omega_{ho\alpha}|} \cdot \frac{|A_{ho\alpha}|}{|S_{ho\alpha}|}$$

$$
\begin{aligned}
&= \frac{1}{|\Omega_h|} \cdot \sum_\alpha \frac{|A_{h\circ\alpha}|^2}{|S_{h\circ\alpha}|} \\
&\geq \frac{|A_h|}{|\Omega_h|} \cdot \frac{|A_h|}{|S_h|}
\end{aligned}
$$

The inequality is justified by using the Technical Claim and noting that $\sum_\alpha |A_{h\circ\alpha}| = |A_h|$ and $\sum_\alpha |S_{h\circ\alpha}| = |S_h|$.

Verifier Step – denoted β

By the perfectness of the simulation, when restricted to the good subspace S, we know that given history h, the verifier V sends β as its next message with probability $\frac{|S_{h\circ\beta}|}{|S_h|}$. Thus,

$$
\begin{aligned}
\mathrm{Exp}_\beta \left(\frac{|A_{h\circ\beta}|}{|\Omega_{h\circ\beta}|} \cdot \frac{|A_{h\circ\beta}|}{|S_{h\circ\beta}|} \right) &= \sum_\beta \frac{|S_{h\circ\beta}|}{|S_h|} \cdot \frac{|A_{h\circ\beta}|}{|\Omega_{h\circ\beta}|} \cdot \frac{|A_{h\circ\beta}|}{|S_{h\circ\beta}|} \\
&= \frac{1}{|S_h|} \cdot \sum_\beta \frac{|A_{h\circ\beta}|^2}{|\Omega_{h\circ\beta}|} \\
&\geq \frac{|A_h|}{|\Omega_h|} \cdot \frac{|A_h|}{|S_h|}
\end{aligned}
$$

The inequality is justified by using the Technical Claim and noting that $\sum_\beta |A_{h\circ\beta}| = |A_h|$ and $\sum_\beta |\Omega_{h\circ\beta}| = |\Omega_h|$.

Having proven Equation (3) for both cases, Equation (2) follows and so does the lemma. □

4 The Transformation

In this section we show how to transform *statistical* knowledge complexity into *perfect* knowledge complexity, incurring only a logarithmic additive term. This transformation combined with Theorem 1 yields the Main Theorem.

Theorem 2 *For every (poly-time computable) $k : \mathsf{N} \mapsto \mathsf{N}$,*

$$\mathcal{SKC}\,(k(\cdot)) \subseteq \mathcal{PKC}\,(k(\cdot) + O(\log(\cdot)))$$

We stress again that these knowledge complexity classes refer to the honest verifier and that we don't know whether such a result holds for the analogous knowledge complexity classes referring to arbitrary (poly-time) verifiers.

proof: Here we use the oracle formulation of knowledge complexity (see Definition 2.1). We start with an overview of the proof. Suppose we are given a simulator M which produces output that is *statistically close* to the real prover-verifier interaction. We change both the interactive proof and its simulation so that they produce exactly the same distribution space. We will take advantage of the fact that the prover in the interactive proof and the oracle that "assists" the simulator are both infinitely powerful. Thus, the modification to the prover's program and the augmentation to the oracle need not be efficiently computable. We stress that the modification to the simulator itself will be efficiently computable. Also, we maintain the original verifier (of the interactive proof), and thus the resulting interactive proof is still sound. Furthermore, the resulting interaction will be statistically close to the original one (on any $x \in L$) and therefore the completeness property of the original interactive proof is maintained (although the error probability here may increase by a negligible amount).

Preliminaries

Let $L \in \mathcal{SKC}(k(\cdot))$, and (P, V) be the guaranteed interactive proof. Without loss of generality, we may assume that all messages are of length 1. This message-length convention is merely a matter of encoding.

Recall that Definition 2.1 only guarantees that the simulator produces output with probability $\geq \frac{1}{2}$. Yet, employing Proposition 3.8 in [GP-91], we get that there exists an oracle machine M, that after asking $k(n) + 2\log\log n$ queries, *always* produces an output so that the output is statistically close to the interaction of (P, V). Let A denote the associated oracle, and let $M' \stackrel{\text{def}}{=} M^A$ and P' and V' be the simulation-based prover and verifier[4] induced by M' (i.e., $(P', V') = M'$).

In the rest of the presentation, we fix a generic input $x \in L$ and omit it from the notation.

notations: Let $[A, B]_i$ be a random variable representing the i-message (i-bit) long prefix of the interaction between A and B (the common input x is implicit in the notation). We denote by $A(h)$ the random variable representing the message sent by A after interaction-history h. Thus, if the i^{th} message is sent by A, we can write $[A, B]_{i-1} \circ A([A, B]_{i-1}) = [A, B]_i$. By $X \stackrel{\text{s}}{=} Y$ we denote the fact that the random variables X and Y are statistically close.

Using these notations we may write for every $h \in \{0,1\}^i$ and $\sigma \in \{0,1\}$:

$$\text{Prob}(P'(h) = \sigma) \;=\; \text{Prob}\left([M']_{i+1} = h \circ \sigma | [M']_i = h\right)$$

and similarly,

$$\text{Prob}(V'(h) = \sigma) \;=\; \text{Prob}\left([M']_{i+1} = h \circ \sigma | [M']_i = h\right).$$

Claim 4.1 *The distribution induced by (P', V) is statistically close to the distributions induced by both $M' = (P', V')$ and (P, V).*

proof: By definition, the distributions produced by $M' = (P', V')$ and (P, V) are statistically close. Thus, we have

$$[P, V]_i \stackrel{\text{s}}{=} [P', V']_i, \quad \text{for every } i \tag{4}$$

We prove that $[P', V]$ is statistically close to $[P', V']$ by induction on the length of the interaction. Assuming that $[P', V]_i \stackrel{\text{s}}{=} [P', V']_i$, we wish to prove it for $i + 1$. We distinguish two cases. In case the $i + 1^{\text{st}}$ move is by the prover, we get

$$\begin{aligned} [P', V]_{i+1} &= [P', V]_i \circ P'([P', V]_i) \\ &\stackrel{\text{s}}{=} [P', V']_i \circ P'([P', V']_i) \\ &= [P', V']_{i+1} \end{aligned}$$

[4] A simulator-based verifier is defined analogously to the simulator-based prover. It is a fictitious entity which does not necessarily coincide with V.

(use the induction hypothesis for $\overset{s}{=}$). In case the $i+1^{\text{st}}$ move is by the verifier, we get

$$\begin{aligned}
[P',V]_{i+1} &= [P',V]_i \circ V([P',V]_i) \\
&\overset{s}{=} [P',V']_i \circ V([P',V']_i) \\
&\overset{s}{=} [P,V]_i \circ V([P,V]_i) \\
&= [P,V]_{i+1} \\
&\overset{s}{=} [P',V']_{i+1}
\end{aligned}$$

where the first $\overset{s}{=}$ is justified by the induction hypothesis and the two others by Eq. (4). We stress that since the induction hypothesis is used only once in the induction step, the statistical distance is linear in the number of induction steps (rather than exponential). □

Motivating discussion: Note that the statistical difference between the interaction (P',V) and the simulation $M' = (P',V')$ is due solely to the difference between the proper verifier (i.e., V) and the verifier induced by the simulator (i.e., V'). This difference is due to V' putting too much probability weight on certain moves and thus also too little weight on their sibling messages (recall that a message in the interaction contains one bit). In what follows we deal with two cases.

The first case is when this difference between the behavior of V' (induced by M') and the behavior of the verifier V is "more than tiny". This case receives most of our attention. We are going to use the oracle in order to move weight from a verifier message β that gets too much weight (after a history h) to its sibling message $\beta \oplus 1$ that gets too little weight (after the history h) in the simulation. Specifically, when the new simulator M'' invokes M' and comes up with a conversation that has $h \circ \beta$ as a prefix, the simulator M'' (with the help of the oracle) will output (a different) conversation with the prefix $h \circ (\beta \oplus 1)$ instead of outputting the original conversation. The simulator M'' will do this with probability that exactly compensates for the difference between V' and V. This leaves one problem. How does the new simulator M'' come up with a conversation that has a prefix $h \circ (\beta \oplus 1)$? The cost of letting the oracle supply the rest of the conversation (after the known prefix $h \circ (\beta \oplus 1)$) is too high. We adopt a "brutal" solution in which we truncate all conversations that have $h \circ (\beta \oplus 1)$ as a prefix. The truncation takes place both in the interaction (P'',V), where P'' stops the conversation after $\beta \oplus 1$ (with a special STOP message) and in the simulation where the oracle recognizes cases in which the simulator M'' should output a truncated conversation. These changes make M'' and V behave exactly the same on messages for which the difference between V' and V is more than tiny. Naturally, V immediately rejects when P'' stops the interaction abruptly, so we have to make sure that this change does not foil the ability of P'' to convince V on an input $x \in L$. It turns out that these truncations happen with negligible probability since such truncation is needed only when the difference between V and V' is more than tiny. Thus, P'' convinces V on $x \in L$ almost with the same probability as P' does.

The second possible case is that the difference between the behavior of V and V' is tiny. In this case, looking at a full conversation $\bar{c}$, we get that the tiny differences sum up to a small difference between the probability of $\bar{c}$ in the distributions of M' and in the distribution of (P',V). We correct these differences by lowering the probabilities of all conversations in the new simulator. The probability of each conversation is lowered so that its relative weight (relatively to all other conversations) is equal to its relative weight in the interaction (P'', V). Technically, this is done by M'' not producing an output in certain cases that M' did produce an output.

Technical remark: The oracle can be used to allow the simulator to toss bias coins when the simulator does not "know" the bias. Suppose that the simulator needs to toss a coin so that it comes-up `head` with probability $\frac{N}{2^m}$, where $N < 2^m$ and both N and m are integers. The simulator supplies the oracle with a uniformly chosen $r \in \{0,1\}^m$ and the oracle answers `head` if r is among the first N strings in $\{0,1\}^m$ and `tail` otherwise. A similar procedure is applicable for implementing a lottery with more than two a-priori known values. Using this procedure, we can get extremely good approximations of probability spaces at a cost related to an a-priori known upper bound on the size of the support (i.e., the oracle answer is logarithmic in the size of the support).

Definition: Let $\epsilon \stackrel{\rm def}{=} \frac{1}{O(t)}$ where t is the number of rounds in the interaction (P, V).

- Let h be a partial history of the interaction and β be a possible next move by the verifier. We say that β is *weak* with respect to h if

$$\mathrm{Prob}(V'(h)=\beta) < (1-\epsilon)\cdot \mathrm{Prob}(V(h)=\beta)$$

- A conversation $\bar{c} = (c_1, ..., c_t)$ is i-weak if c_i is weak with respect to $(c_1, ..., c_{i-1})$, otherwise it is *i-good.* (Note that a conversation can be i-weak only if the $i^{\rm th}$ move is a verifier move.)
- A conversation $\bar{c} = (c_1, ..., c_t)$ is *i-critical* if it is i-weak but j-good for every $j < i$. A conversation $\bar{c} = (c_1, ..., c_t)$ is *i-co-critical* if the conversation obtained from $\bar{c}$, by complementing (only) the $i^{\rm th}$ bit, is i-critical. (Note that a conversation can be i-critical only for a single i, yet it may be i-co-critical for many i's.)
- A conversation is *weak* if it is i-weak for some i, otherwise it is *good.*

Claim 4.2 *(P', V) outputs weak conversations with negligible probability.*

proof: Recall that $[P', V] \stackrel{\rm s}{=} [P', V']$ and that the same holds also for prefixes of the conversations. Namely, for any $1 \leq i \leq t$, $[P', V]_i \stackrel{\rm s}{=} [P', V']_i$. Let us define a prefix $h \in \{0,1\}^i$ of a conversation to be *bad* if either

$$\mathrm{Prob}([P', V']_i = h) < \left(1 - \frac{\epsilon}{2}\right) \cdot \mathrm{Prob}([P', V]_i = h)$$

or

$$\mathrm{Prob}([P', V']_i = h) > \left(1 + \frac{\epsilon}{2}\right) \cdot \mathrm{Prob}([P', V]_i = h)$$

The claim follows by combining two facts.

Fact 4.3 *The probability that (P', V) outputs a conversation with a bad prefix is negligible.*

proof: Define B_i to be the set of bad prefixes of length i. By the statistical closeness of $[P',V]_i$ and $[P',V']_i$, we get that

$$\Delta \stackrel{\rm def}{=} \sum_{h\in B_i} |{\rm Prob}([P',V]_i\!=\!h) - {\rm Prob}([P',V']_i\!=\!h)| \le \gamma$$

for some negligible fraction γ. On the other hand, Δ can be bounded from bellow by

$$\sum_{h\in B_i} {\rm Prob}([P',V]_i\!=\!h)\cdot \left|1-\frac{{\rm Prob}([P',V']_i\!=\!h)}{{\rm Prob}([P',V]_i\!=\!h)}\right|$$

which by definition of B_i is at least

$${\rm Prob}([P',V]_i\!\in\! B_i)\cdot\left|\pm\frac{\epsilon}{2}\right|$$

Thus, ${\rm Prob}([P',V]_i\!\in\! B_i) \le \frac{2\gamma}{\epsilon}$ and the fact follows. □

Fact 4.4 *If a conversation $\bar{c}=(c_1,...,c_t)$ is weak then it contains a bad prefix.*

proof: Suppose that $\beta \stackrel{\rm def}{=} c_{i+1}$ is weak with respect $h \stackrel{\rm def}{=} (c_1,...,c_i)$. If h is a bad prefix then we are done. Otherwise it holds that

$${\rm Prob}([P',V']_i\!=\!h) < \left(1+\frac{\epsilon}{2}\right)\cdot{\rm Prob}([P',V]_i\!=\!h)$$

Using the fact that β is weak with respect to h, we get

$$\begin{aligned}{\rm Prob}([P',V']_{i+1}\!=\!h\circ\beta) &< \left(1+\frac{\epsilon}{2}\right)\cdot(1-\epsilon)\cdot{\rm Prob}([P',V]_{i+1}\!=\!h\circ\beta)\\ &< \left(1-\frac{\epsilon}{2}\right)\cdot{\rm Prob}([P',V]_{i+1}\!=\!h\circ\beta)\end{aligned}$$

which implies that $h\circ\beta$ is a bad prefix of $\bar{c}$. □

Combining Facts 4.3 and 4.4, Claim 4.2 follows. □

Claim 4.5 *Suppose that $\bar{c}=(c_1,...,c_t)$ is a good conversation. Then, the probability that $\bar{c}$ is output by M' is at least $(1-\epsilon)^{\lceil t/2\rceil}\cdot{\rm Prob}([P',V]\!=\!\bar{c})$. Furthermore, for $l<k$, if $\bar{c}=(c_1,...,c_t)$ is i-good for every $i\in\{l+1,...,k\}$, then*

$${\rm Prob}\left([M']_k\!=\!\gamma\,|\,[M']_l\!=\!h\right) \ge (1-\epsilon)^{\lceil\frac{k-l}{2}\rceil}\cdot{\rm Prob}\left([P',V]_k\!=\!\gamma\,|\,[P',V]_l\!=\!h\right)$$

where $\gamma\stackrel{\rm def}{=}(c_1,...,c_k)$ and $h\stackrel{\rm def}{=}(c_1,...,c_l)$

proof: To see that this is the case, we write the probabilities step by step conditioned on the history so far. We note that the prover's steps happen with equal probabilities in both sides of the inequality, and therefore can be reduced. Since the relevant verifier's steps are not weak, we get the mentioned inequality. The actual proof proceeds by induction on $k-l$.

Clearly, if $k - l = 0$ the claim holds. We note that if $k - l = 1$ the claim also holds since step k in the conversation is either a prover step or a k-good verifier step.

To show the induction step we use the induction hypothesis for $k - l - 2$. Namely,

$$\begin{aligned} &\text{Prob}\left([M']_{k-2} = (c_1, \ldots, c_{k-2}) \mid [M']_l = (c_1, \ldots, c_l)\right) \\ &\quad \geq (1 - \epsilon)^{\lceil \frac{k-l}{2} \rceil - 1} \cdot \text{Prob}\left([P', V]_{k-2} = (c_1, \ldots, c_{k-2}) \mid [P', V]_l = (c_1, \ldots, c_l)\right) \end{aligned} \tag{5}$$

Steps $k - 1$ and k include one prover message and one verifier message. Assume, without loss of generality, that the prover step is $k - 1$. Since P' is the simulator based prover, we get

$$\begin{aligned} &\text{Prob}\left([M']_{k-1} = (c_1, \ldots, c_{k-1}) \mid [M']_{k-2} = (c_1, \ldots, c_{k-2})\right) \\ &\quad = \text{Prob}\left([P', V]_{k-1} = (c_1, \ldots, c_{k-1}) \mid [P', V]_{k-2} = (c_1, \ldots, c_{k-2})\right) \end{aligned} \tag{6}$$

Since step k of the verifier is good, we also have:

$$\begin{aligned} &\text{Prob}\left([M']_k = (c_1, \ldots, c_k) \mid [M']_{k-1} = (c_1, \ldots, c_{k-1})\right) \\ &\quad \geq (1 - \epsilon) \cdot \text{Prob}\left([P', V]_k = (c_1, \ldots, c_k \mid [P', V]_{k-1} = (c_1, \ldots, c_{k-1})\right) \end{aligned} \tag{7}$$

Combining Equations 5, 6, and 7, the induction step follows and we are done. □

Dealing with weak conversations

We start by modifying the prover P', resulting in a modified prover, denoted P'', that stops once it gets a verifier message which is weak with respect to the current history; otherwise, P'' behaves as P'. Namely,

Definition (modified prover - P''): For any $h \in \{0,1\}^*$ and $\beta \in \{0,1\}$,

$$P''(h \circ \beta) = \begin{cases} STOP & \text{if } \beta \text{ is weak with respect to } h. \\ P'(h \circ \beta) & \text{Otherwise} \end{cases}$$

We assume that the verifier V stops and rejects immediately upon receiving an illegal message from the prover (and in particular upon receiving this STOP message).

Next, we modify the simulator so that it outputs either good conversations or truncated conversations which are originally i-critical. Jumping ahead, we stress that such truncated i-critical conversations will be generated from both i-critical and i-co-critical conversations. The modified simulator, denoted M'', proceeds as follows[5]. First, it invokes M' and obtains a conversation $\bar{c} = (c_1, ..., c_t)$. Next, it queries the augmented oracle on $\bar{c}$. The oracle answers probabilistically and its answers are of the form (i, σ), where $i \in \{1, ..., t\}$ and $\sigma \in \{0, 1\}$. The probability distribution will be specified below, at this point we only wish to remark that the oracle only returns pairs (i, σ) for which one of the following three conditions holds

1. $\bar{c}$ is good, $i = t$ and $\sigma = 0$ (if $\bar{c}$ is good and is not i-co-critical for any i's then the oracle always answers this way);

[5] We stress that P'' is not necessarily the simulator-based prover of M''.

2. $\bar{c}$ is i-critical and $\sigma = 0$;
3. $\bar{c}$ is i-co-critical and $\sigma = 1$.

Finally, the new simulator (M'') halts outputting $(c_1, ..., c_{i-1}, c_i \oplus \sigma)$, which in case $\sigma = 1$ is not a prefix of $\bar{c}$. Note that i may be smaller than t, in which case M'' outputs a truncated conversation which is always i-critical; otherwise, M'' outputs a non-truncated conversation. It remains to specify the oracle's answer distribution.

Let us start by considering two special cases. In the first case, the conversation generated by M' is i-critical, for some i, but is not j-co-critical for any $j < i$. In this case the oracle always answers $(i, 0)$ and consequently the simulator always outputs the i-bit long prefix. However, this prefix is still being output with too low probability. This will be corrected by the second case hereby described. In this ("second") case, the conversation $\bar{c}$ generated by M' is good and i-co-critical for a single i. This means that the i-bit long prefix is given too much probability weight whereas the prefix obtained by complimenting the i^{th} bit gets too little weight. To correct this, the oracle outputs $(i, 1)$ with probability q and $(t, 0)$ otherwise, where q will be specified. What happens is that the M'' will output the "i-complimented prefix" with higher probability than with which it has appeared in M'. The value of q is determined as follows. Denote $p \stackrel{\text{def}}{=} \text{Prob}(V(c_1, ..., c_{i-1}) = c_i \oplus 1)$ and $p' \stackrel{\text{def}}{=} \text{Prob}(V'(c_1, ..., c_{i-1}) = c_i \oplus 1)$. Then, setting q so that $p' + (1 - p') \cdot q = p$ (i.e., $q = \frac{p-p'}{1-p'}$) allows the simulator to output the prefix $(c_1, ..., c_{i-1}, c_i \oplus 1)$ with the right probability.

In the general case, the conversation generated by M' may be i-co-critical for many i's as well as j-critical for some (*single*) j. In case it is j-critical, it can be i-co-critical only for $i < j$. Let us consider the sequence of indices, $(i_1, ..., i_l)$, for which the generated conversation is critical or co-critical (i.e., the conversation is i_k-co-critical for all $k < l$ and is either i_l-critical or i_l-co-critical). We consider two cases. In both cases the q_k's are set as in the above example; namely, $q_k = \frac{p_k - p'_k}{1-p'_k}$, where $p_k \stackrel{\text{def}}{=} \text{Prob}(V(c_1, ..., c_{i_k-1}) = c_{i_k} \oplus 1)$ and $p'_k \stackrel{\text{def}}{=} \text{Prob}(V'(c_1, ..., c_{i_k-1}) = c_{i_k} \oplus 1)$.

1. The generated conversation, $\bar{c} = (c_1, ..., c_t)$, is i_k-co-critical for every $k < l$ and is i_l-critical. In this case, the distribution of the oracle answers is as follows. For every $k < l$, the pair $(i_k, 1)$ is returned with probability $(\prod_{j<k}(1 - q_j)) \cdot q_k$; whereas the pair $(i_l, 0)$ appears with probability $\prod_{j<l}(1 - q_j)$. We stress that no other pair appears in this distribution.[6]

2. The generated conversation, $\bar{c} = (c_1, ..., c_t)$, is i_k-co-critical for every $k \leq l$. In this case, the distribution of the oracle answers is as follows. For every $k \leq l$, the pair $(i_k, 1)$ is returned with probability $(\prod_{j<k}(1 - q_j)) \cdot q_k$; whereas the pair $(t, 0)$ appears with probability $\prod_{j\leq l}(1 - q_j)$. Again, no other pair appears in this distribution.

Claim 4.6

1. $[P'', V] \stackrel{\text{s}}{=} [P', V]$;

2. Each conversation of (P'', V)*, be it a complete* (P', V)*-conversation or a truncated (i.e., critical) one, is output by* M'' *with probability that is at least a* $(1 - \epsilon)^t > \frac{3}{4}$ *fraction of the probability that it appears in* $[P'', V]$.

[6] Indeed the reader can easily verify that these probabilities sum up to 1.

proof: The weak conversations are negligible in the output distribution of (P', V) (see Claim 4.2). The only difference between $[P'', V]$ and $[P', V]$ originates from a different behavior of P'' on weak conversations, specifically P'' truncates them while P' does not. Yet, the distribution on the good conversations remains unchanged. Therefore the distribution of $[P'', V]$ is statistically close to the distribution of $[P', V]$, and we are done with Part (1).

For Part (2) let us start with an intuitive discussion which may help reading through the formal proof that follows. First, we recall that the behavior of the simulation M' in prover steps is identical to the behavior of the interaction (P', V) in prover's steps. This follows simply from the fact that P' is the simulation based prover of M'. We will show that this property still holds for the new interaction (P'', V) and the new simulation M''. We will do this by noting two different cases. In one case, the prover step is conducted by P'' exactly as it is done by P' and then M'' behaves exactly as M'. The second possible case is that the prover step contains the special message STOP. We shall note that this occurs with exactly the same probability in the distribution (P'', V) and in the distribution of M''.

Next, we consider the verifier steps. In the construction of M'' and P'' we considered the behavior of M' and V on verifier steps and made changes when these differences were not "tiny". We called a message β weak with respect to a history h, if the simulator assigns the message β (after outputting h) a probability which is smaller by a factor of more than $(1-\epsilon)$ from the probability that the verifier V outputs the message β on history h. We did not make changes in messages whose difference in weight (between the simulation M' and the interaction (P', V)) were smaller than that. In the proof, we consider two cases. First, the message β is weak with respect to the history h. Clearly, the sibling message $\beta \oplus 1$ is getting too much weight in the simulation M'. So in the definition of M'' we made adjustments to move weight from the prefix $h \circ (\beta \oplus 1)$ to the prefix $h \circ \beta$. We will show that this transfer of weight exactly cancels the difference between the behavior of V and the behavior of M'. Namely, the weak messages (and their siblings) are assigned exactly the same probability both in M'' and by V. Thus, we show that when a weak step is involved, the behavior of (P'', V) and the behavior of M'' are exactly equivalent. It remains to deal with messages for which the difference between the conditional behavior of V and M' is "tiny" and was not considered so far. In this case, M'' behaves like M'. However, since the difference is so tiny, we get that even if we accumulate the differences throughout the conversation, they sum up to at most the multiplicative factor $3/4$ stated in the claim.

Let us begin the formal proof by writing again the probability that (P'', V) outputs $\bar{c}$ as the product of the conditional probabilities of the t steps. Namely,

$$\prod_{i=1}^{t} \text{Prob}\left([P'', V]_{i+1} = h_i \circ c_{i+1} \mid [P'', V]_i = h_i\right)$$

where $h_i \stackrel{\text{def}}{=} (c_1, ..., c_i)$. We do the same for the probability that M'' outputs a conversation $\bar{c}$. We will show by induction that each step of any conversation is produced by M'' with at least $(1-\epsilon)$ times the probability of the same step in the (P'', V)-interaction. Once we have shown this, we are done. Clearly this claim holds for the null prefix. To prove the induction step, we consider the two possibilities for the party making the $i + 1^{\text{st}}$ step.

$i + 1^{\text{st}}$ step is by the prover: Consider the conditional behavior of M'' given the history so far. We will show that this behavior is identical to the behavior of P'' on the same partial history.

A delicate point to note here is that we may talk about the behavior of M'' on a prefix h_i only if this prefix appears with positive probability in the output distribution $[M'']_i$. However, by the induction hypothesis any prefix that is output by $[P'', V]_i$ appears with positive probability in $[M'']_i$.

We partition the analysis into two cases.

1. First, we consider the case in which the last message of the verifier is weak with respect to the history that precedes it. Namely, $h = h' \circ \beta$ and β is weak with respect to h'. In this case, both in the interaction (P'', V) and in the simulation M'', the next message of the prover is set to STOP with probability 1. Namely,

$$\begin{aligned}\text{Prob}\left(M'' = h \circ \text{STOP} \mid [M'']_i = h\right) &= 1 \\ &= \text{Prob}\left(P''(h) = \text{STOP}\right)\end{aligned}$$

2. The other possible case is that the last message of the verifier is not weak with respect to its preceding history. In this case, the simulator M'' behaves like M' and the prover P'' behaves like P'. (Note that the changes in critical and co-critical steps apply only to verifier steps.) Thus,

$$\begin{aligned}\text{Prob}\left([M'']_{i+1} = h \circ \alpha \mid [M'']_i = h\right) &= \text{Prob}\left([M']_{i+1} = h \circ \alpha \mid [M']_i = h\right) \\ &= \text{Prob}\left(P'(h) = \alpha\right) \\ &= \text{Prob}\left(P''(h) = \alpha\right)\end{aligned}$$

To summarize, the conditional behavior of M'' in the prover steps and the conditional behavior of P'' are exactly equal.

$i+1^{st}$ step is by the verifier: Again, we consider the conditional behavior of M'' given the history so far. Let us recall the second modification applied to M' when deriving M''. This modification changes the conditional probability of the verifier steps in the distribution of M' in order to add weight to steps having low probability in the simulation. We note that this modification is made only in critical or co-critical steps of the verifier. Consider a history h_i which might appear in the interaction (P'', V) and a possible response β of V to h_i. Again, by the induction hypothesis, h_i has a positive probability to be output by the simulation M'' and therefore we may consider the conditional behavior of M'' on this history h_i. There are three cases to be considered, corresponding to whether either β or $\beta \oplus 1$ or none is weak with respect to h_i.

We start with the simplest case in which neither β nor $\beta \oplus 1$ is weak (w.r.t. h_i). In this case, the behavior of M'' is identical to the behavior of M' since the oracle never sends the message $(i+1, \sigma)$ in this case. However, by the fact that β is not weak, we get that

$$\begin{aligned}(1-\epsilon) \cdot \text{Prob}(V(h) = \beta) &\leq \text{Prob}\left([M']_{i+1} = h \circ \beta \mid [M']_i = h\right) \\ &= \text{Prob}\left([M'']_{i+1} = h \circ \beta \mid [M'']_i = h\right)\end{aligned}$$

and we are done with this simple case.

We now turn to the case in which β is weak (w.r.t. h_i). In this case, given that M'' has produced the prefix h_i, it produces $h_i \circ \beta$ whenever M' produces the prefix $h_i \circ \beta$. Furthermore, with conditional probability q (as defined above), M'' produces the prefix $h_i \circ \beta$ also in case M' produces the prefix $h_i \circ (\beta \oplus 1)$. As above, we define

$$\begin{aligned} p &\stackrel{\rm def}{=} \text{Prob}\,(V(h_i) = \beta) \\ p' &\stackrel{\rm def}{=} \text{Prob}\,(V'(h_i) = \beta) \end{aligned}$$

Since V' is the simulation (M') based verifier, we may also write

$$p' = \text{Prob}\,([M']_{i+1} = h_i \circ \beta \mid [M']_i = h_i) \tag{8}$$

Also, recall that q was defined as $\frac{p-p'}{1-p'}$. Now, using these notations:

$$\begin{aligned} \text{Prob}\,([M'']_{i+1} = h_i \circ \beta \,|[M'']_i = h_i) &= \text{Prob}\,([M']_{i+1} = h_i \circ \beta \,|[M']_i = h_i) \\ &\quad + \frac{p-p'}{1-p'} \cdot \text{Prob}\,([M']_{i+1} = h_i \circ (\beta \oplus 1) \,|[M']_i = h_i) \end{aligned}$$

Using Equation (8), we get

$$\begin{aligned} &= p' + \frac{p-p'}{1-p'} \cdot (1-p') \\ &= p \\ &= \text{Prob}\,(V(h) = \beta) \end{aligned}$$

Finally, we turn to the case in which $\beta \oplus 1$ is weak (w.r.t. h_i). Again, this means that β is co-critical in $\bar{c}$. Given that M'' has produced the prefix h_i, it produces $h_i \circ \beta$ only when M' produces the prefix $h_i \circ \beta$, and furthermore, M'' does so only with probability $1-q$ (where q is again as defined above). We denote p and p', with respect to the critical message $\beta \oplus 1$. Namely,

$$\begin{aligned} p &\stackrel{\rm def}{=} \text{Prob}\,(V(h_i) = \beta \oplus 1) \\ p' &\stackrel{\rm def}{=} \text{Prob}\,(V'(h_i) = \beta \oplus 1) \\ &= \text{Prob}\,([M']_{i+1} = h_i \circ (\beta \oplus 1) \mid [M']_i = h_i) \end{aligned}$$

Thus, recalling that $q = \frac{p-p'}{1-p'}$, we get

$$\begin{aligned} \text{Prob}\,([M'']_{i+1} = h_i \circ \beta \,|[M'']_i = h_i) &= (1 - \frac{p-p'}{1-p'}) \cdot \text{Prob}\,([M']_{i+1} = h_i \circ \beta \,|[M']_i = h_i) \\ &= \frac{1-p}{1-p'} \cdot (1-p') \\ &= 1-p \\ &= \text{Prob}\,(V(h_i) = \beta) \end{aligned}$$

This completes the proof of Claim 4.6. □

Lowering the probability of some simulator outputs

After handling the differences between M' and (P', V) which are not tiny, we make the last modification, in which we deal with tiny differences. We do that by lowering the probability that the simulator outputs a conversation, in case it outputs this conversation more frequently than it appears in (P'', V). The modified simulator, denoted M''', runs M'' to obtain a conversation $\bar{c}$. (Note that M'' always produces output.) Using the further-augmented oracle, M''' outputs $\bar{c}$ with probability

$$p_{\bar{c}} \stackrel{\text{def}}{=} \frac{3}{4} \cdot \frac{\text{Prob}([P'', V] = \bar{c})}{\text{Prob}([M''] = \bar{c})}$$

Note that $p_{\bar{c}} \leq 1$ holds due to Part 2 of Claim 4.6.

Claim 4.7

1. *M''' produces output with probability $\frac{3}{4}$;*
2. *The output distribution of M''' (i.e., in case it has output) is identical to the distribution $[P'', V]$.*

proof: The probability that M''' produces an output is exactly:

$$\sum_{\bar{c}} \text{Prob}\left([M''] = \bar{c}\right) \cdot p_{\bar{c}} = \frac{3}{4}$$

As for part (2), we note that the probability that a conversation $\bar{c}$ is output by M''' is exactly $\frac{3}{4} \cdot \text{Prob}\left([P'', V] = \bar{c}\right)$. Since the simulator halts with an output with probability exactly $\frac{3}{4}$, we get that given that M''' halts with an output, it outputs $\bar{c}$ with probability exactly $\text{Prob}\left([P'', V] = \bar{c}\right))$ and we are done. □

An important point not explicitly addressed so far is whether all the modifications applied to the simulator preserve its ability to be implemented by a probabilistic polynomial-time with bounded access to an oracle. Clearly, this is the case with respect to M'' (at the expense of additional $1 + \log_2 t = O(\log n)$ oracle queries). Yet, regarding the last modification there is a subtle points which needs to be addressed. Specifically, we need to verify that the definition of M''' is implementable; namely, that M''' can (with help of an augmented oracle) "sieve" conversations with *exactly* the desired probability. Note that the method presented above (in the "technical remark") may yield exponentially small deviation from the desired probability. This will get very close to a perfect simulation, but yet will not achieve it.

To this end, we modify the "sieving process" suggested in the technical remark to deal with the specific case we have here. But first we modify P'' so that it makes its random choices (in case it has any) by flipping a polynomial number of unbiased coins.[7] This rounding does change a bit the behavior of P'', but the deviation can be made so small that the above assertions (specifically Claim 4.6) still hold.

[7] The implementation of P'' was not discussed explicitly. It is possible that P'' uses an infinite number of coin tosses to select its next message (either 0 or 1). However, an infinite number of coin tosses is not really needed since rounding the probabilities so that a polynomial number of coins suffices, causes only exponentially small rounding errors.

Consider the specific sieving probability we need here. Namely: $p_{\bar{c}} = \frac{3}{4} \cdot \frac{a/b}{c/d}$, where $\frac{a}{b} = \text{Prob}([P'', V] = \bar{c})$ and $\frac{c}{d} = \text{Prob}([M''] = \bar{c})$. A key observation is that c is the number of coin tosses which lead M'' to output $\bar{c}$ (i.e., using the notation of the previous section, $c = |\Omega_{\bar{c}}|$). Observing that b is the size of probability space for $[P'', V]$ and using the above modification to P'', we rewrite $p_{\bar{c}}$ as $\frac{3ad}{4b} \cdot \frac{1}{c} = \frac{e}{c2^f}$, where e and $f = \text{poly}(n)$ are some non-negative integers.

We now note, that the oracle can allow the simulator to sieve conversations with probability $\frac{e}{c}$ ($f = 0$), for any $0 \leq e \leq c$ in the following way. M''' sends to the oracle the random tape ω that it has tossed for M'', and the oracle sieves only e out of the possible c random tapes which lead M'' to output $\bar{c}$. The general case of $p_{\bar{c}} = \frac{e}{c2^f}$ is deal by writing $p_{\bar{c}} = \frac{q}{c} + \frac{r}{c2^f}$, where $q = \lfloor e/2^f \rfloor$ and $r = e - q2^f < 2^f$. To implement this sieve, M''' supplies the oracle with a uniformly chosen f-bit long string (in addition to ω). The oracle sieves out q random-tapes (of M'') as before, and uses the extra bits in order to decide on the sieve in case ω equals a specific (different) random-tape.

Combining Claims 4.1, 4.6 (part 1), and 4.7, we conclude that (P'', V) is an interactive proof system of *perfect* knowledge complexity $k(n) + O(\log n)$ for L. This completes the proof of Theorem 2. ■

5 Concluding Remarks

We consider our main result as a very first step towards a classification of languages according to the knowledge complexity of their interactive proof systems. Indeed there is much to be known. Below we first mention two questions which do not seem too ambitious. The first is to try to provide evidence that NP-complete languages cannot be proven within low (say logarithmic or even constant) knowledge complexity. A possible avenue for proving this conjecture is to show that languages having logarithmic knowledge complexity are in co-$\mathcal{AM}$, rather than in $\mathcal{BPP}^{\mathcal{NP}}$ (recall that NP is unlikely to be in co-$\mathcal{AM}$ - see also [BHZ-87]). The second suggestion is to try to provide indications that there are languages in $\mathcal{PSPACE}$ which do not have interactive proofs of linear (rather than logarithmic) knowledge complexity. The reader can easily envision more moderate and more ambitious challenges in this direction.

Another interesting question is whether all levels greater then zero of the knowledge-complexity hierarchy contain strictly more languages than previous levels, or if some partial collapse occurs. For example, it is open whether constant or even logarithmic knowledge complexity classes do not collapse to the zero level.

Regarding our transformation of statistical knowledge complexity into perfect knowledge complexity (i.e., Theorem 2), a few interesting questions arise. Firstly, can the cost of the transformation be reduced to bellow $O(\log n)$ bits of knowledge? A result for the special case of statistical zero-knowledge will be almost as interesting. Secondly, can one present an analogous transformation that preserves *one-sided error probability* of the interactive proof? (Note that our transformation introduces a negligible error probability into the completeness condition.) Finally, can one present an analogous transformation that applies to knowledge complexity *with respect to arbitrary verifiers*? (Our transformation applies only to knowledge complexity *with respect to the honest verifier.*)

6 Acknowledgement

We thank Leonard Shulman for providing us with a simpler proof of Claim 3.2.

References

[AH-87] W. Aiello and J. Håstad. Perfect Zero-Knowledge can be Recognized in Two Rounds. *Proceedings of the 28th Annual IEEE Symposium on the Foundations of Computer Science,* IEEE (1987).

[BMO-90] M. Bellare, S. Micali and R. Ostrovsky. The (True) Complexity of Statistical Zero-Knowledge. *Proceedings of the 22nd Annual ACM Symposium on the Theory of Computing,* ACM (1990).

[BP-92] M. Bellare and E. Petrank. Making Zero-Knowledge Provers Efficient. *Proceedings of the 24rd Annual ACM Symposium on the Theory of Computing,* ACM (1992)

[B+ 88] M. Ben-Or, S. Goldwasser, O. Goldreich, J. Håstad, J. Kilian, S. Micali and P. Rogaway. Everything Provable is Provable in Zero-Knowledge. *Advances in Cryptology — Proceedings of CRYPTO 88,* Lecture Notes in Computer Science 403, Springer-Verlag (1989). S. Goldwasser, ed.

[BHZ-87] R. Boppana, J. Håstad and S. Zachos. Does *co-NP* Have Short Interactive Proofs". *Information Processing Letters*, Vol 25 (1987), No. 2, pp 127–132.

[F-89] L. Fortnow. The Complexity of Perfect Zero-Knowledge. *Advances in Computing Research (ed. S. Micali)* Vol. 18 (1989).

[GMS-87] O. Goldreich, Y. Mansour and M. Sipser. Interactive Proof Systems: Provers that never Fail and Random Selection. *Proceedings of the 28th Annual IEEE Symposium on the Foundations of Computer Science,* IEEE (1987).

[GMW-86] O. Goldreich, S. Micali, and A. Wigderson, "Proofs that Yield Nothing But their Validity and a Methodology of Cryptographic Protocol Design", *Proc. 27th FOCS 86,* See also *Jour. of ACM.* Vol 38, No 1, July 1991, pp. 691–729.

[GMW-87] O. Goldreich, S. Micali, and A. Wigderson, "How to Play any Mental Game or a Completeness Theorems for Protocols of Honest Majority", STOC87.

[GP-91] O. Goldreich and E. Petrank. Quantifying Knowledge Complexity. *Proceedings of the 32nd Annual IEEE Symposium on the Foundations of Computer Science,* IEEE (1991).

[GMR-85] S. Goldwasser, S. Micali, and C. Rackoff. The Knowledge Complexity of Interactive Proofs. *Proceedings of the 17th Annual ACM Symposium on the Theory of Computing,* ACM (1985).

[GMR-89] S. Goldwasser, S. Micali, and C. Rackoff. The Knowledge Complexity of Interactive Proofs. *SIAM J. Comput.* **18** (1), 186-208 (February 1989).

[GS-89] S. Goldwasser, and M. Sipser, Private Coins vs. Public Coins in Interactive Proof Systems, *Advances in Computing Research (ed. S. Micali),* 1989, Vol. 5, pp. 73-90.

[H-94] J. Håstad. Perfect Zero-Knowledge in $\mathcal{AM} \cap$ co-$\mathcal{AM}$. Unpublished 2-page manuscript explaining the underlying ideas behind [AH-87]. 1994.

[ILu-90] R. Impagliazzo and M. Luby, One-Way Functions are Essential for Complexity Based Cryptography, *30th FOCS*, pp. 230–235, 1990.

[ILe-90] R. Impagliazzo and L.A. Levin, No Better Ways to Generate Hard NP Instances than Picking Uniformly at Random, *31st FOCS*, pp. 812-821, 1990.

[IY-87] R. Impagliazzo and M. Yung. Direct Minimum-Knowledge computations. *Advances in Cryptology — Proceedings of CRYPTO 87,* Lecture Notes in Computer Science 293, Springer-Verlag (1987).

[JVV-86] M. Jerrum, L. Valiant and V. Vazirani. Random Generation of Combinatorial Structures from a Uniform Distribution. *Theoretical Computer Science* **43**, 169-188 (1986).

[LFKN-90] C. Lund, L. Fortnow, H. Karloff and N. Nisan. Algebraic Methods for Interactive Proof Systems. *Proceedings of the 31st Annual IEEE Symposium on the Foundations of Computer Science,* IEEE (1990).

[Ost-91] R. Ostrovsky. One-Way Functions, Hard on Average Problems, and Statistical Zero-Knowledge Proofs. *Proceedings of Structures In Complexity Theory 6th Annual Conference* IEEE (1991).

[OW-93] R. Ostrovsky and A. Wigderson. One-Way Functions are Essential For Non-Trivial Zero-Knowledge, *Proc. 2nd Israeli Symp. on Theory of Computing and Systems*, 1993.

[OVY-91] R. Ostrovsky, R. Venkatesan and M. Yung. Fair Games Against an All-Powerful Adversary. *AMS DIMACS Series in Discrete Mathematics and Theoretical Computer Science.* Vol 13. (Jin-Yi Cai ed.) pp. 155-169.

[Sh-90] A. Shamir. IP=PSPACE. *Proc. 22nd ACM Symp. on Theory of Computing,* pages 11–15, 1990.

[Si-83] M. Sipser. A Complexity Theoretic Approach to Randomness. *Proceedings of the 15th Annual ACM Symposium on the Theory of Computing,* ACM (1983).

[St-83] L. Stockmeyer. The Complexity of Approximate Counting. *Proceedings of the 15th Annual ACM Symposium on the Theory of Computing,* ACM (1983).

A APPENDIX: A Flaw in [F-89]

In [F-89], Fortnow presents a constructive method for proving that $\mathcal{SZK} \stackrel{\rm def}{=} \mathcal{SKC}(0)$ is contained in co-$\mathcal{AM}$. Given an interactive proof (P, V) for a languages L and a (statistical) zero-knowledge simulator M (for the honest verifier V), he constructs a two-round protocol (P', V'). This protocol was claimed to constitute an interactive proof system for $\overline{L}$. This claim, as we are going to show, is wrong. Yet, the result $\mathcal{SZK} \subseteq$ co-$\mathcal{AM}$ does hold, since the work of Aiello and Hastad contains the necessary refinements which enable to present a modified AM-protocol for $\overline{L}$ (see [AH-87, H-94]). Furthermore, Fortnow's basic approach is valid, and indeed it was used in subsequent works (e.g., [AH-87, BMO-90, Ost-91, BP-92, OW-93]).

Fortnow's basic approach starts with the observation that the simulator M must behave differently on $x \in L$ and $x \notin L$. Clearly, the difference cannot be recognized in polynomial-time, unless $L \in \mathcal{BPP}$. Yet, stronger recognition devices, such as interactive proofs should be able to tell the difference. Fortnow suggests a characterization of the simulator's behavior on $x \in L$ and uses this characterization in his protocol for $\overline{L}$, yet this characterization is wrong. Aiello and Hastad present a refinement of Fortnow's characterization [AH-87], their characterization is correct and can be used to show that $\mathcal{SZK} \subseteq \mathcal{AM}$ (which is the goal of their paper) as well as $\mathcal{SZK} \subseteq$ co-$\mathcal{AM}$.

Fortnow's characterization

Given an interactive proof (P, V) for L and a simulator M, and fixing a common input $x \in \{0,1\}^*$, the following sets are defined. Let us denote by t the number of random bits that the verifier V uses on input x, and by q the number of random bits used by the simulator M. For every conversation prefix, h, we consider the set of the verifier's coin tosses which are consistent with h (the conversation so far). We denote this set by R_1^h. Namely, for $h = (\alpha_1, \beta_1, ..., \alpha_i, \beta_i)$ (or $h = (\alpha_1, \beta_1, ..., \alpha_i, \beta_i, \alpha_{i+1})$), $r \in R_1^h$ iff $V(x, r, \alpha_1, ..., \alpha_j) = \beta_j$ for every $j \leq i$, where $V(x, r, \bar{\alpha})$ denotes the message sent by V on input x random-tape r and prover message-sequence $\bar{\alpha}$. The set R_1^h depends only on the verifier V. Next, we consider sets R_2^h which are subsets of the corresponding R_1^h's. Specifically, they contain only r's that can appear with h in an accepting conversation output by the simulator M. Namely, $r \in R_2^h$ iff $r \in R_1^h$ and there exists $\omega \in \{0,1\}^q$ so that $M(x, \omega)$ is an accepting conversation with prefix h. (Here $M(x, \omega)$ denotes the conversation output by M on input x and simulator-random-tape ω.)

Motivation: For simplicity, suppose that the simulation is perfect (i.e., M witnesses that (P, V) is *perfect* zero-knowledge) and that (P, V) has one sided error (i.e., "perfect completeness"). Then, for every $x \in L$ and every possible h, we must have $R_2^h = R_1^h$ (otherwise the simulation is not perfect). However, if $x \notin L$ then there must exist h's so that R_2^h is much smaller than R_1^h. Otherwise the simulator-based prover (for M) will always convince V to accept x, thus violating the soundness condition of (P, V). The problem with the above dichotomy is that it is "too existential" and thus it is not clear how to use it. Instead Fortnow claimed a dichotomy which is more quantitative.

A False Characterization: Let $\mathrm{pref}(\bar{c})$ denote the set of all message-subsequences in the conversation $\bar{c}$.

- if $x \in L$ then
$$\mathrm{Prob}_{\omega}(\forall h \in \mathrm{pref}(M(x,\omega))\ \left|R_2^h\right| \approx_1 \left|R_1^h\right|) > \frac{3}{4}$$
- if $x \notin L$ then
$$\mathrm{Prob}_{\omega}(\forall h \in \mathrm{pref}(M(x,\omega))\ \left|R_2^h\right| \approx_2 \left|R_1^h\right|) < \frac{1}{4}$$

where the probability (in both cases) is taken uniformly over $\omega \in \{0,1\}^q$. We did not specify what is meant by $\approx_i$. One may substitute $\alpha \approx_1 \beta$ by $\alpha \geq \frac{1}{2} \cdot \beta$, and $\alpha \approx_2 \beta$ by $\alpha \geq \frac{1}{4} \cdot \beta$. The gap between the two is needed for the approximate lower/upper bound protocols.

A Counterexample

The mistake is in the second item of the characterization. The false argument given in [F-89] confuses between the probability distribution of conversations output by the simulator and the probability distribution of the conversations between a simulator-based prover (denote P^*) and the verifier. These distributions are not necessarily the same (note that we are in case $x \notin L$). Consequently, the probability that "good" conversations (i.e., conversations for which $|R_2| \approx |R_1|$ for all prefixes) occur in the (P^*, V) interaction is not the same as the probability that the simulator outputs "good" conversations. This point is ignored in [F-89] and leads there to the false conclusion that the characterization holds. Bellow, we present an interactive proof (P, V) and a (perfect) zero-knowledge simulator for which the characterization fails.

The interactive proof that we present is for the empty language Φ. This interactive proof is perfect zero knowledge for the trivial reason that the requirement is vacuous. Yet, we present a simulator for this interactive proof which, for every $x \in \{0,1\}^* = \overline{\Phi}$, outputs "good" conversation with probability close to 1. Thus, the characterization fails.

The interactive proof (from the verifier's point of view – input $x \in \{0,1\}^n$):

- The verifier uniformly selects $\alpha \in \{0,1\}^n$ and sends α to the prover.
- The verifier waits for the prover's message $\beta \in \{0,1\}^n$.
- Next, the verifier uniformly selects $\gamma \in \{0,1\}^n$ and sends γ to the prover.
- The verifier accepts iff either $\alpha = 0^n$ or $\beta = \gamma$.

Regardless of the prover's strategy, the verifier accepts each $x \in \{0,1\}^n$ with negligible probability; specifically $2^{-n} + (1-2^{-n}) \cdot 2^{-n}$. Thus, the above protocol indeed constitutes an interactive proof for the empty language Φ.

The simulator operates as follows (on input $x \in \{0,1\}^n$):

- With probability $1-\epsilon$, the simulator M outputs a conversation uniformly distributed in $0^n \times \{0,1\}^{2n}$. (ϵ is negligible, say $\epsilon = 2^{-n}$)
- With probability ϵ, the simulator M outputs a conversation uniformly distributed in $(\{0,1\}^n - 0^n) \times \{0,1\}^{2n}$.

Claim: In contradiction to the characterization, for every $x \in \{0,1\}^* = \overline{\Phi}$,

$$\mathrm{Prob}_\omega(\forall h \in \mathrm{pref}(M(x,\omega))\ \left|R_2^h\right| = \left|R_1^h\right|) \geq 1 - \epsilon$$

Proof: It suffices to show that every conversation of the form $0^n\beta\gamma$ satisfies $R_2 = R_1$ for all its prefixes. First observe that $R_1^\lambda = \{0,1\}^{2n} = R_2^\lambda$, since for every $\alpha\gamma \in \{0,1\}^{2n}$ the simulator outputs the accepting conversation $\alpha\gamma\gamma$ with non-zero probability. Similarly, $R_1^{0^n} = 0^n\{0,1\}^n = R_2^{0^n}$. Next, for every $\beta \in \{0,1\}^n$, we have $R_1^{0^n\beta} = 0^n\{0,1\}^n = R_2^{0^n\beta}$, since for every $\gamma \in \{0,1\}^n$ the simulator outputs the accepting conversation $0^n\beta\gamma$ with non-zero probability. (Here we use the fact that the verifier always accepts when $\alpha = 0^n$.) Similarly, $R_1^{0^n\beta\gamma} = 0^n\gamma = R_2^{0^n\beta\gamma}$. $\Box$

Conclusion

The source of trouble is that the definition of the sets R_2^h's does not take into account the probability weight assigned by the simulator to ω's that witness the assertion "the simulator outputs an accepting conversation that starts with h". Indeed, this is exactly the nature of the refinement suggested by Aiello and Hastad [AH-87].

B APPENDIX: Applying our techniques for non-negligible error probabilities

As explained in the introduction, the notion of an interactive proof with bounded knowledge complexity is not robust under changes in the allowed error probability. Throughout the paper, we use the natural definition of interactive proofs in which the error probability is negligible. However, our techniques yield non-trivial results also in the case one defines interactive proofs with some specific non-negligible error probability. In this appendix we explain how such assertions may be obtained, and state such results for two special cases.

Denote by $\epsilon_c(n)$ (an upper bound on) the probability that the verifier rejects an input x although $x \in L$ and the prover plays honestly. This is the error probability related to the completeness condition. Similarly, denote by $\epsilon_s(n)$ (an upper bound on) the probability that the verifier accepts $x \notin L$ when the prover follows its optimal strategy (not necessarily following the protocol). This is the error probability related to the soundness condition. We say that an interactive proof has error probabilities (ϵ_s, ϵ_c) if its error probability in the soundness condition is bounded by ϵ_s and its error probability in the completeness condition is bounded by ϵ_c.

B.1 The perfect case

In this subsection, we consider the more restricted case of *perfect* knowledge complexity, and derive Theorem 3 which is the analogue of Theorem 1 for the case that the error probabilities are not negligible. Following the definitions in Section 3, we denote the simulation based prover by P^*.

Let us follows the steps of the proof of our main theorem and observe which assertions hold for the case of non-negligible error probability. We begin by observing that the following generalization of Lemma 3.1 holds:

Lemma B.1 *Let (P,V) be an interactive proof for L with error probabilities $(\epsilon_s(n), \epsilon_c(n))$ and with knowledge complexity $k(n)$, then*

1. *If $x \in L$ then the probability that (P^*, V) outputs an accepting conversation is at least $(1-\epsilon_c(n))^2 \cdot 2^{-k(n)}$, where $n = |x|$.*
2. *If $x \notin L$ then the probability that (P^*, V) outputs an accepting conversation is at most $\epsilon_s(n)$, where $n = |x|$.*

The proof of this lemma is identical to the proof of Lemma 3.1, except that here $\frac{|A_\lambda|}{|S_\lambda|} = 1 - \epsilon_c(n)$. As explained in Section 3, an efficient machine with access to an NP oracle can sample conversations in (P^*, V). By Lemma B.1, this would yield an accepting conversation with probability at most $\epsilon_s(n)$ in the case $x \notin L$ and at least $(1-\epsilon_c(n))^2 \cdot 2^{-k(n)}$ when $x \in L$. In case these two probabilities differ sufficiently (i.e., by more then a polynomial fraction), we can use standard amplification techniques to get a probabilistic algorithm that determines whether $x \in L$ with error probability less than $1/3$ (or negligible, or 2^{-n}). To summarize, we get the following theorem for perfect knowledge complexity.

Theorem 3 *If a language L has an interactive proof with perfect knowledge complexity $k(n)$ and error probabilities (ϵ_s, ϵ_c) and if there exists a polynomial $p(n)$ such that*

$$(1-\epsilon_c(n))^2 \cdot 2^{-k(n)} > \epsilon_s(n) + \frac{1}{p(n)}$$

then $L \in \mathcal{BPP}^{\mathcal{NP}}$.

Examples: Theorem 3 implies, for example, that if a language L has an interactive proof of knowledge complexity 1 and error probability 1/4 (both in the soundness condition and in the completeness condition), then L is in $\mathcal{BPP}^{\mathcal{NP}}$. Another interesting example is the case of one-sided error (i.e., $\epsilon_c = 0$). Theorem 3 implies that for any polynomial $p(\cdot)$, if a language L has a one-sided error interactive proof (P, V) of knowledge complexity at most $\log_2\left(\frac{p(\cdot)}{2}\right)$ and error probability $\epsilon_s \leq \frac{1}{p(\cdot)}$, then L is in $\mathcal{BPP}^{\mathcal{NP}}$.

B.2 The general (statistical) case

Unfortunately, the analogue result for statistical knowledge complexity is not as clean, and has various different formulations according to possible properties of the error probabilities. Let us explain how such a result can be obtain, and give a specific example for the special case in which $\epsilon_c = 0$, i.e., the original interaction has one-sided error.

Recall that the proof for the negligible error-probability case uses the transformation from statistical to perfect knowledge complexity and then uses Theorem 1. This transformation increases the knowledge complexity by a logarithmic additive term. In view of Lemma B.1, it is desirable not to increase the knowledge complexity without concurrently decreasing the error probability. Thus, before applying the transformation, we reduce the error probability by iterating the protocol as many times as possible while maintaining logarithmic knowledge complexity.

Specifically, denote the length of the interaction by $l(n)$. Also, fix an input x of length n, and let $l = l(n)$, $k = k(n)$, $\epsilon_s = \epsilon_s(n)$ and $\epsilon_c = \epsilon_c(n)$. The transformation from statistical to perfect knowledge complexity (as described in Section 4) increases the knowledge complexity by $1 + \log_2 l$. We begin by running the original protocol (P, V) sequentially $t \stackrel{\text{def}}{=} \lceil (\log_2 l)/k \rceil$ times. These repetitions yield a new protocol (P', V') whose length is $t \cdot l$, its knowledge complexity is bounded by $t \cdot k < (k-1) + \log_2 l$, and its error probability decreases. To compute the decrease in the error probabilities, we partition the analysis into two cases according to whether the original interaction has one sided error or not.

If the original interaction has one sided error, i.e., the verifier always accepts when $x \in L$, then the new verifier V' accepts only if all repetitions of the original protocols end accepting. The error probabilities in this case decrease from $(\epsilon_s, 0)$ to $(\epsilon_s^t, 0)$. In the case where the original interactive proof was not one sided, the verifier counts the number of original interactions that end with the original verifier accepting. The new verifier accepts if this number is greater than $\frac{\epsilon_s + (1-\epsilon_c)}{2} \cdot t$. In order to compute the new error probabilities we may apply the Chernoff bound and get an upper bound on the new error probabilities which depends on t, on the difference between $1 - \epsilon_c$ and ϵ_s, and of-course on ϵ_s and ϵ_c themselves.

Next, we apply the transformation of Section 4 ("from statistical to perfect knowledge complexity") and get a new interactive proof (P'', V'') for L which has knowledge complexity

$k-1+\log_2 l+1+\lceil\log_2(l\cdot t)\rceil$, where the additional $1+\lceil\log_2(l\cdot t)\rceil$ term comes from the transformation. Finally, if the resulting parameters of (P'',V'') satisfy the conditions stated in Theorem 3, then we get that the language L is in $\mathcal{BPP}^{\mathcal{NP}}$. Let us provide full details for the special (yet important) case of one sided error (i.e., $\epsilon_c=0$).

In the special case of one-sided error, we end up using Theorem 3 for an interactive proof with knowledge complexity $k+\log_2 l+\lceil\log_2(l\cdot t)\rceil$ and (one-sided) error probability ${\epsilon_s}^t$. Thus, we get the following theorem for statistical knowledge complexity:

Theorem 4 *Suppose that a language L has an interactive proof of statistical knowledge complexity $k(n)$, one-sided error probability $\epsilon_s(n)$, and with length $l(n)$ so that there exists a polynomial $p(n)$ for which the following inequality holds*

$$\frac{1}{2\cdot 2^{k(n)}\cdot l(n)^2\cdot\left\lceil\frac{\log_2 l(n)}{k(n)}\right\rceil}\geq\epsilon_s(n)^{\lceil(\log_2 l(n))/k(n)\rceil}+\frac{1}{p(n)}$$

Then $L\in\mathcal{BPP}^{\mathcal{NP}}$.