Hashing Functions can Simplify Zero-Knowledge Protocol Design (too)

Ivan Damgård* Oded Goldreich† Avi Wigderson‡

November 28, 1994

Abstract

In *Crypto93*, Damgård showed that any constant-round protocol in which the verifier sends only independent, random bits and which is zero-knowledge against the *honest* verifier can be transformed into a protocol (for the same problem) that is zero-knowledge *in general*. His transformation was based on the interactive hashing technique of Naor, Ostrovsky, Venkatesan and Yung, and thus the resulting protocol had very large round-complexity.

We adopt Damgård's methods, using ordinary hashing functions, instead of the abovementioned interactive hashing technique. Typically, the protocols we derive have much lower round-complexity than those derived by Damgård's transformation. As in Damgård's transformation, our transformation preserves statistical/perfect zero-knowledge and does not rely on any computational assumptions. However, unlike Damgård's transformation, the new transformation is not applicable to argument systems or to proofs of knowledge.

---

*Dept. of Computer Science, Aarhus Univesity, Denmark and BRICS, Basic Research In Computer Science, Centre of the Danish National Research Foundation.

†Dept. of Applied Math. and Computer Science, Weizmann Institute of Science, Rehovot, Israel. Work done while visiting BRICS, Basic Research In Computer Science, Centre of the Danish National Research Foundation. Supported in part by grant No. 92-00226 from the United States – Israel Binational Reseach Foundation (BSF), Jerusalem, Israel.

‡Institute for Computer Science, Hebrew University, Jerusalem, Israel. Work done while visiting BRICS, Basic Research In Computer Science, Centre of the Danish National Research Foundation. Supported in part by grant No. xx-00yyy from the United States – Israel Binational Reseach Foundation (BSF), Jerusalem, Israel.

1 Introduction

Zero-knowledge proof systems, introduced by Goldwasser, Micali and Rackoff [13], are a key tool in the design of cryptographic protocols. The results of Goldreich, Micali and Wigderson [12] guarantee that such proof systems can be constructed for any NP-statement, provided that one-way functions exist. However, the general construction presented in [12] and subsequent works may yield quite inefficient proof systems for particular applications of interest. Thus, developing methodoligies for the design of zero-knowledge proofs is still of interest.

Designing proof systems which are merely zero-knowledge with respect to the honest verifier (i.e., the verifier specified for the system) is much easier than constructing proof systems which are zero-knowledge in general (i.e., with respect to any efficient strategy of trying to extract knowledge from the specified prover). For example, the simple 1-round interactive proof for Graph Non-Isomorphism [1] is zero-knowledge with respect to the honest verifier. Yet, cheating verifiers may extract knowledge from this system and a non-trivial modification, which utilizes proofs of knowledge and increases the number of rounds, is required to make it zero-knowledge in general. Likewise, assuming the existence of one-way function, there exist constant-round interactive proofs for any NP-language which are zero-knowledge with respect to the honest verifier. Yet, constant-round interactive proofs for NP which are zero-knowledge in general are known only under seemingly stronger assumptions and are also more complex (cf., [9]).

In view of the relative simplicity of designing protocols which are zero-knowledge with respect to the honest verifier, a transformation of such protocols into protocols which are zero-knowledge in general (i.e., wrt any verifier) may be very valuable. Assuming various intractability assumptions, such transformations have been presented by Bellare et. al. [2], and Ostrovsky et. al. [18]. A transformation which does not rely on any intractability assumptions has been presented by Damgård in *Crypto93*. His transformation (of honest-verifier zero-knowledge into general zero-knowledge) has two shortcomings. Firstly, it can be applied only to constant-round protocols of the Arthur-Merlin type (i.e., in which the verifier's messages are uniformly distributed in the set of strings of specified length). Secondly, the transformation produces protocols of very high round complexity; specifically, the round complexity of the resulting protocol is linear in the randomness complexity of the original one.

In this paper, we improve the round complexity of Damgård's transformation, while preserving the class of interactive proos to which it can be applied. Our transformation

[1] To be convinced that G_0 and G_1 are not isomorphic, the verifier randomly selects n random isomorphic copies of each graph, randomly shuffles all these copies together, and asks the prover to specify the origin of each copy.

only increases the number of rounds by a factor of two. However, it also increase the error probability of the proof system by a non-negligible amount which can be made arbitrarily small. This increase is inevitible in view of a result of Goldreich and Krawcyzk [10], see discussion in subsection 4.4. Thus, to get proof systems with negligible error probability, one may repeat the protocols resulting from our transformation a non-constant number of times. Still, the resulting proof systems will have much lower round complexity than those resulting from Damgård's transformation.

We preserve some of the positive properties of Damgård's transformation. In particular, our transformation does not rely on any computational assumptions and preserves perfect and almost-perfect zero-knowledge. However, unlike Damgård's transformation, the new transformation is not applicable to argument systems (i.e., the BCC model [4]) or to proofs of knowledge.

Our transformation builds on Damgård's work [6]. In his transformation, the random messages sent by the verifier (in each round) are replaced by a multi-round interactive hashing protocol, which in turn originates in the work of Naor, Ostrovsky, Venkatesan and Yung [17]. Instead, in our transformation, the random messages sent by the verifier are replaced by a 3/2-round protocol, called Random Selection. The Random Selection protocol uses a family of ordinary hashing functions; specifically, we use a family of t-wise indepedent functions, for some parameter t (which is certainly polynomial in the input).

We believe that the Random Selection protocol may be of independent interest. Thus a few words are in place. The goal of this protocol is to allow two parties to select a "random" n-bit string. There is a parameter ε which governs the quality of this selection and the requirement is asymmetric with respect to the two parties. Firstly, it is required that if the first party follows the protocol then, no matter how its counterpart plays, the output of the protocol will be at most ε away (in norm-1) from uniform. Secondly, it is required that if the second party follows the protocol then, no matter how its counterpart plays, no string will appear as output of the protocol with probability greater than $\text{poly}(n/\varepsilon) \cdot 2^{-n}$. Our Random Selection protocol has the additional property of being simulatable in the sense that, given a possible outcome, it is easy to generate a (random) transcript of the protocol which ends with this outcome.

Other Related Work

The idea of transforming honest verifier zero-knowledge into zero-knowledge in general was first studied by Bellare, Micali and Ostrovsky [2]. Their transformation needed a computational assumption of a specific algebraic type. Since then several constructions have reduced the computational assumptions needed. The latest in this line of work is by Ostrovsky, Venkatesan and Yung [18], who give a transformation which is based on interactive hashing and preserved statistical zero-knowledge. Their transformation relies on existence of a one-way permutation. The transformation works for any protocol, provided that the verifier is probabilistic polynomial-time.

An indirect way of converting protocols which are zero-knowledge with respect to the honest verifier into ones which are zero-knowledge in general, is available through a recent result of Ostrovsky and Wigderson [19]. They have proved that the existence of honest verifier zero-knowledge proof system for a language which is "hard on the average" implies the existence of one-way functions. Combined with the results of [12] and [15, 3], this yields a (computational and general) zero-knowledge proof for the same language. Thus, computational honest-verifier zero-knowledge interactive proofs, for "hard on the average" languages, get transformed into computational zero-knowledge interactive proofs for these languages. However, perfect honest-verifier zero-knowledge proofs (for such languages) do not get transformed into *perfect* zero-knowledge proofs.

A two-party protocol for random selection, with unrelated properties, has been presented in [8]. This protocol guarantees that, as long as one party plays honestly, the outcome of the protocol hits any set $S \subset \{0,1\}^n$ with probability at most $\tilde{O}(\sqrt{|S|/2^n})$, where $\tilde{O}(\varepsilon) \stackrel{\text{def}}{=} \varepsilon \cdot \text{polylog}(1/\varepsilon)$.

Another two-party protocol for random selection, with other unrelated properties, has been presented in [11]. Loosely speaking, this protocol allows a computationally restricted party, interacting with a powerful and yet untrustful party, to uniformly select an element in an easily recognizable set $S \subset \{0,1\}^n$.

2 Some Remarks Concerning Definitions

We assume that the reader is familiar with the various definitions of interactive proofs (i.e., the GMR model). Below we merely point out some less familiar definitions that we are going to use.

Following many works, we denote by $(P,V)(x)$ a random variable representing the transcript of the interaction between prover P and verifier V, on common input x.

In this paper we use a somewhat non-standard definition of zero-knowledge. This definition is very convenient for our purposes. Furthermore, we believe that it is nicer in general. Below, we present only the honest-verifier variant of perfect zero-knowledge. We trust the reader to generate the other variants by himself/herself.

Definition 1 (perfect zero-knowledge wrt honest verifier): *Let (P,V) be an interactive proof for language L. We say that P is* perfect zero-knowledge with respect to the honest verifier *if there exists a probabistic polynomial-time machine M and a positive polynomial $p()$ so that for every $x \in L$*

- *with probability at least* $1/p(|x|)$*, on input* x*, machine* M *halts with output;* (otherwise, it halts with no output).
- *given that on input* x *machine* M *halts with output, the output is distributed identically to* $(P,V)(x)$*.*

In the above definition, we require M to run in strictly polynomail-time (whereas the traditional definition allows it to run in *expected* polynomial-time). However, unlike in the traditional definition, we allow the machine to stop without output. All we require is that with non-negligible probability the machine stops with output. Clearly, the new definition implies the traditional one (since we can repeatedly invoke a strict simulator untill it stops with output). Also, most zero-knowledge proofs can be show zero-knowledge also under the new definition.[2] However, we do not know if the traditional definition implies the new one in general. Actually, we believe that it does not. In case the reader is concerned of this issue, he/she can augment the above definition by allowing the simulator both to run in expected polynomial-time and still have output only with non-negligible probability. This augmented definition is clearly equivalent to the traditional one and yet is somewhat more convenient for our purposes.

For the purpose of a motivating discussion in subsection 4.4, we use the notion of *black-box* zero-knowledge. Loosely speaking, black-box zero-knowledge is a strengthening of the ordinary notion of zero-knowledge. Recall that (ordinary) zero-knowledge means that the interaction of the prover with any efficient verifier can be efficiently simulated. Thus, this definition allows to use a different simulator for each verifier and furthermore make no requirement regarding the relation among the various simulators. For black-box zero-knowledge we require that there exists a universal simulator, which given access to any efficient verifier, can simulate the interaction of the prover with this verifier. For further details – see [10].

3 Random Selection

We consider a randomized two-party protocol for selecting strings. The two parties to the protocol are called the *challenger* and the *responder*. These names are supposed to reflect the asymmetric requirements (presented below) as well as the usage of the protocol in our zero-knowledge transformation. Loosely speaking, we require that

[2]This includes, for example, the perfect zero-knowledge proofs for Graph Isomorphism and the computational zero-knowledge proofs for NP, but not the perfect zero-knowledge proof for Graph Non-Isomorphism [12].

- if the challenger follows the protocol then, no matter which strategy is used by the responder, the output of the protocol is almost uniformly distributed;
- if the responder follows the protocol then, no string may appear with probability much greater than its probability under the uniform distribution. Furthermore, for any string which may appear as output, when an arbitrary challenger strategy is used, one can efficiently generate a random transcript of that protocol ending with this output.

We postpone the formal specification of these properties to the analysis of the protocol presented below. Actually, we present two version of the protocol.

Construction 1 (Random Selection Protocol – two versions): *Let n and $m < n$ be integers*[3]*, and $H_{n,m}$ be a family of functions, each mapping the set of n-bit long strings onto*[4] *the set of m-bit long strings.*

C1: *the challenger uniformly selects $h \in H_{n,m}$ and sends it to the responder;*

R1:
- (version 1): *the responder uniformly selects $x \in \{0,1\}^n$, computes $\alpha = h(x)$ and sends α to the challenger;*
- (version 2): *the responder uniformly selects $\alpha \in \{0,1\}^m$ and sends it to the challenger;*

C2: *the challenger uniformly selects a preimage of α under h and outputs it.*

We remark that if version 1 is used and both parties follow the protocol then the output is uniformly distributed in $\{0,1\}^n$. However, the interesting case is when one of the parties deviates from the protocol. In this case, the protocol can be guaranteed to produce "good" output, provided that "good" families of hash functions are being used as $H_{n,m}$. These functions must have relatively succient representation as well as strong random properties. Furthermore, given a function h, it should be easy to evaluate h on a given image and to generate a random preimage (of a given range element) under h. Using the algorithmic properties of $H_{n,m}$ it follows that the instructions specified in the above protocol can be implemented in probabilistic poly(n/ε)-time, which for $\varepsilon = 1/\text{poly}(n)$ means poly(n)-time.

Construction 2 (Preferred family $H^t_{n,m}$): *Let n, $m < n$ and $t = \text{poly}(n)$ be integers. We associate $\{0,1\}^n$ with the finite field $GF(2^n)$ and consider the set of $(t-1)$-degree*

[3] In particular, we will use $m \stackrel{\text{def}}{=} n - 4\log_2(n/\varepsilon)$, where ε is an error-bound parameter.

[4] We stress that each function in $H_{n,m}$ rages over all $\{0,1\}^m$. Thus, the challenger may always respond in step C2 even if the responder deviates from the protocol or version 2 is used.

polynomials over this field. For each such polynomial f, we consider the function h so that, for every $x \in \{0,1\}^n$, $h(x)$ is the m most significant bits of $f(x)$. The family $H^t_{n,m}$ consists of all such functions h. The canonical description of a function $h \in H^t_{n,m}$ is merely the sequence of t smallest coefficients of the corresponding polynomial. Finaly, we modify the functions in $H^t_{n,m}$ so that for each $h \in H^t_{n,m}$ and every $x' \in \{0,1\}^m$ it holds $h(x'0^{n-m}) \stackrel{\rm def}{=} x'$.

In the sequel, we will use the family $H_{n,m} \stackrel{\rm def}{=} H^n_{n,m}$. We now list the following, easy to verify, properties of the above family.

P1 There is a poly(n)-time algorithm that, on input a function $h \in H^t_{n,m}$ and a string $x \in \{0,1\}^n$, outputs $h(x)$.

P2 The number of preimages of an image y under $h \in H^t_{n,m}$ is bounded above by $2^{n-m} \cdot t$; furthermore, there exists a poly($2^{n-m}t$)-time algorithm that, on input y and h, outputs the set $h^{-1}(y) \stackrel{\rm def}{=} \{x : h(x) = y\}$. (The algorithm works by trying all possible extensions of y to an element of $GF(2^t)$; for each such extension it remains to find the roots of a degree $t-1$ polynomial over the field.)

P3 $H^t_{n,m}$ is a family of almost t-wise independent hashing functions in the following sense: for every t distinct images, $x_1, ..., x_t \in (\{0,1\}^n - \{0,1\}^m 0^{n-m})$, for a uniformly chosen $h \in H^t_{n,m}$, the random variables $h(x_1), ..., h(x_t)$ are indepedently and uniformly distributed in $\{0,1\}^m$.

3.1 The output distribution for honest challeger

We now turn to analyze the output distribution of the above protocol, assuming that the challenger plays according to the protocol. In the analysis we allow the responder to deviate arbitrarily from the protocol and thus as far as this analysis goes the two versions in Construction 1 are equivalent. The analysis is done using the "random" properties of the family $H^t_{n,m}$. Recall that the statistical difference between two random variable X and Y is

$$\frac{1}{2} \sum_{\alpha} |\mathrm{Prob}(X = \alpha) - \mathrm{Prob}(Y = \alpha)|$$

We say that X is ε-*away* from Y if the statistical difference between them is ε.

Proposition 1 *Let n be an integer, $\varepsilon \in [0,1]$ and $m \stackrel{\rm def}{=} n - 4\log_2(n/\varepsilon)$. Suppose that $H_{n,m}$ is a family of almost n-wise independent hashing functions. Then, no matter which strategy is used by the responder, provided that the challenger follows the protocol, the output of the protocol is at most $(2\varepsilon + 2^{-n})$-away from uniform distribution.*

proof: Recall that an equivalent definition of the statistical difference between two random variables, X and Y, is

$$\max_S\{|\mathrm{Prob}(X\in S)-\mathrm{Prob}(Y\in S)|\}$$

In our case, one random variable is the output of the protocol whereas the other is uniformly distributed. Thus, it suffices to upper bound the difference between the probability that the output hits an arbitrary set S and the density of S (in $\{0,1\}^n$). Furthermore, it suffices to consider sets S of density greater/equal to one half (i.e., $|S| \geq \frac{1}{2}\cdot 2^n$). Let us denote by $\alpha^* : H_{n,m} \mapsto \{0,1\}^m$ an arbitrary strategy employed by the responder. Then, under the conditions of the proposition, the output of the protocol uniformly distributed in the random set $h^{-1}(\alpha^*(h))$, where h is uniformly selected in $H_{n,m}$. Thus, for a set S, the probability that the output is in S equals

$$\mathrm{Exp}_{h\in H_{n,m}}\left(\frac{|h^{-1}(\alpha^*(h))\cap S|}{|h^{-1}(\alpha^*(h))|}\right) \tag{1}$$

For an arbitrarily fixed set S, we can bound the expression in Eq. 1 by considering the event in which a uniformly chosen $h \in H_{n,m}$ satisfies

$$\frac{|h^{-1}(\alpha)\cap S|}{|h^{-1}(\alpha)|} \not\in [(1\pm 2\varepsilon)\rho(S)] \quad \text{for all } \alpha\in\{0,1\}^m. \tag{2}$$

where $\rho(S) \stackrel{\text{def}}{=} \frac{|S|}{2^n}$. Whenever this event occurs, Eq. 1 is in the interval $[(1-2\varepsilon)\rho(S),(1+2\varepsilon)\rho]$ and so the statistical difference is at most 2ε. Thus, it remains to upper bound the probability that the above event does not hold. We first note that when estimating the cardinality of the sets $h^{-1}(\alpha)$ and $h^{-1}(\alpha)\cap S$ we may ignore the contribution of the preimages in $\{0,1\}^m 0^{n-m}$, since there is at most one such elements (i.e., $\alpha 0^{n-m}$). Fixing an arbitrary α and using the t-moment method, with $t=n$, we get

$$\begin{aligned}\mathrm{Prob}_{h\in H_{n,m}}\left(|h^{-1}(\alpha)\cap S|\not\in[(1\pm\varepsilon)\rho(S)2^{n-m}]\right) &< \left(\frac{t}{\varepsilon\cdot\rho(S)}\cdot 2^{-(n-m)/2}\right)^n\\ &< \left(\frac{n}{n^2}\right)^n\\ &< 2^{-2n}\end{aligned}$$

Thus, with overwhelmingly high probability, $|h^{-1}(\alpha)\cap S| \in [(1\pm\varepsilon)\rho(S)\cdot 2^{n-m}]$, for all $\alpha\in\{0,1\}^m$. By a similar argument, with overwhelmingly high probability, $|h^{-1}(\alpha)| \in [(1\pm\varepsilon)\cdot 2^{n-m}]$, for all $\alpha\in\{0,1\}^m$. Thus, with overwhelmingly high probability (i.e., at least $1-2^{-n}$), the event in Eq. 2 holds. ■

3.2 The output distribution for honest responder

We now show that no matter what strategy is used by the challenger, if the responder follows the protocol then the set of possible outputs of the protocol must constitute a non-negligible fraction of the set of n-bit long strings. This claim holds for both versions of Construction 1. Furthermore, we show that no single string may appear with probability which is much more than 2^{-n} (i.e., its probability weight under the uniform distribution).

Proposition 2 *Suppose that $H_{n,m} = H^t_{n,m}$ is a family of hashing functions satisfying property (P2), for some $t = \mathrm{poly}(n)$. Let C^* be an arbitrary challenger strategy. Then, for every $x \in \{0,1\}^n$, the probability that an execution of* version 1 *of the protocol with challenger strategy C^* ends with output x is at most $(t \cdot 2^{n-m}) \cdot 2^{-n}$.*

proof: We consider an arbitrary (probabilistic) strategy for the challenger, denoted C^*. Without loss of generality, we may assume that the first message of this strategy is an element of $H_{n,m}$ (messages violating this convention are treated/interpreted as a fixed function $h_0 \in H_{n,m}$). Similarly, we may assume that the second message of the challenger, given partial history (h, α), is an element of $h^{-1}(\alpha)$ (again, messages violating this convention are interpreted as, say, the lexicographically first element of $h^{-1}(\alpha)$). Finally, it suffices to consider deterministic strategies for the challenger; since, given a probabilistic strategy C^*, we can uniformly select a sequence r respresenting the outcome of the coin tosses of C^* and consider the strategy $\mathbf{c}(\cdot) \stackrel{\text{def}}{=} C^*_r(\cdot) \stackrel{\text{def}}{=} C^*(r, \cdot)$.

We now upper bound the probability that an execution of the protocol with challenger strategy $\mathbf{c}$ ends with output x. We denote by $h \stackrel{\text{def}}{=} c(\lambda)$ the first message of strategy $\mathbf{c}$. Now, the protocol may end with output x only if the responder chose the message $\alpha \stackrel{\text{def}}{=} h(x)$. Thus, the probability that the responder choose α is exactly $|\{x' : h(x') = \alpha\}| \cdot 2^{-n}$. By property (P2), for each $h \in H_{n,m}$ and $\alpha \in \{0,1\}^m$, the cardinality of the set $h^{-1}(\alpha)$ is at most $t \cdot 2^{n-m}$. The proposition follows. ■

Proposition 3 *Let C^* be an arbitrary challenger strategy. Then, for every $x \in \{0,1\}^n$, the probability that an execution of* version 2 *of the protocol with challenger strategy C^* ends with output x is at most 2^{-m}. Furthermore, for every deterministic challenger strategy $\mathbf{c}$, exactly 2^m strings may appear as output, each with probability exactly 2^{-m}.*

proof: Fix a deterministic strategy $\mathbf{c}$ and a string $x \in \{0,1\}^n$. As in the previous proof, we may assume that $h \stackrel{\text{def}}{=} \mathbf{c}(\lambda) \in H_{n,m}$ and $\mathbf{c}(\alpha) \in h^{-1}(\alpha)$. Denoting $h \stackrel{\text{def}}{=} \mathbf{c}(\lambda)$, version 2 terminates with output x if and only if the responder chooses the message $\alpha \stackrel{\text{def}}{=} h(x)$ and $x = \mathbf{c}(\alpha)$. Since α is selected uniformly in $\{0,1\}^m$, the proposition follows. ■

3.3 Simultability property of the protocol

We conclude the analysis of the above protocol by showing that, one can efficiently generate random transcripts of the protocol having this output. Throughout this analysis, we assume that the responder follows the instruction specified by the protocol. As in the proof of the last two propositions, it suffices to consider an arbitrary deterministic challenger strategy, denoted $\mathbf{c}$.

Now, suppose that $H_{n,m} = H_{n,m}^t$ is a family of hashing functions satisfying property (P1), for some $t = \text{poly}(n)$. Then, on input x and access to a function $\mathbf{c}: \{0,1\}^* \mapsto \{0,1\}^*$, we can easily test if $\mathbf{c}(h(x)) = x$, where $h \stackrel{\text{def}}{=} \mathbf{c}(\lambda)$. In case the above condition holds, the triple $(h, h(x), x)$ is the only transcript of the execution of the protocol, with challenger strategy $\mathbf{c}$, which ends with output x. Otherwise, there is no execution of the protocol, with challenger strategy $\mathbf{c}$, which ends with output x. Thus,

Proposition 4 *Consider executions of the Random Selection protocol in which the challenger strategy, denoted* $\mathbf{c}$*, is an arbitrary function and the responder plays according to the protocol. There exists a polynomial-time oracle machine that, on input* $x \in \{0,1\}^n$ *and* $h \in H_{n,m}$ *and oracle access to a function* $\mathbf{c}$*, either generates the unique transcript of a* $\mathbf{c}$*-execution which outputs* x *or indicates that no such execution exists.*

3.4 Setting the Parameters

Proposition 1 motivates us to set ε (the parameter governing the approximation of the output in case of honest challenger) as small as possible. On the other hand, Propositions 2 and 3 motivates us to maintain the difference $n - m$ small and in paricular logarithmic (in n). Recalling that $n - m = 4\log_2(n/\varepsilon)$, this suggests setting $\varepsilon = 1/p(n)$ for some fixed positive polynomial p.

4 The Zero-Knowledge Transformation

Our transformation is restricted to interactive proofs in which the verifier sends the outcome of every coin it tosses. Such interactive proofs are called *Arthur-Merlin games* [1] or *public-coins interactive proofs* (cf., [14]). Note that in such interactive proofs the verifier moves, save the last, may consist merely of tossing coins and sending their outcome. (In its last move the verifier decides, based on the entire history of the communication, whether to accept the input or not.) Without loss of generality, we may assume that in every round of such an interactive proof the verifier tosses at least $4\log(|x|/\varepsilon)$ coins, where x is the common input to the interactive proof and ε specifies the desired bound on the statistical distance (between one round in the resulting interactive proof and the original one). Furthermore, assume for sake of simplicity that at each round the verifier tosses the same number of coins, denoted n.

4.1 The Transformation

In the following description, we use the second version of the Random Selection protocol presented in Construction 1. This simplifies the construction of the simulator for the transformed interactive proof. The first version can be used as well, at the expense of some modification in the simulator construction.

The protocol transformation consists of replacing each verifier move (except the last, decision move) by an execution of the Random Selection protocol, in which the verifier plays the role of the challenger and the prover plays the role of the responder. Thus, each round of the original interactive proof, consisting of a random message sent by the verifier followed by a respond of the prover, is relaced by two rounds in which the three first messages are of the Random Selection protocol and the fourth message is the prover respond. Namely,

Construction 3 (transformation of round i in (P, V) interaction): *Let (P, V) be an interactive proof system in which the verifier V only uses public coins, let $\varepsilon(n) = 1/\mathrm{poly}(n)$ be the desired error in the Random Selection protocol, $m \stackrel{\text{def}}{=} m(n) \stackrel{\text{def}}{=} n - 4\log_2(n/\varepsilon(n))$ and $H_{n,m}$ be as specified in Construction 2* (for $t = n$). *The i^{th} round of the (P, V) interaction, on common input x, is replaced by the following two rounds of the resulting interactive proof (P', V'). Let $(h_1, \alpha_1, r_1, \beta_1, ..., h_{i-1}, a_{i-1}, r_{i-1}, \beta_{i-1})$ be the history so far of the interaction between prover P' and verifier V'. Then, the next two rounds consist of an execution of the* (second version of the) *Random Selection protocol follows by P' mimicing the response of P. Namely, in the first round, the verifier V' uniformly selects $h_i \in H_{n,m}$ and sends it to the prover P' who replies with a_i uniformly selected in $\{0, 1\}^m$. In the second round, the verifier V' uniformly selects $r_i \in h_i^{-1}(a_i)$ and sends it to the prover P' who replies with $\beta_i \stackrel{\text{def}}{=} P(x, r_1, ..., r_i)$.*

The final decision of the new verifier V' mimics the one of the original verifier V; namely,

$$V'(h_1, \alpha_1, r_1, \beta_1, ..., h_t, a_t, r_t, \beta_t) = V(r_1, \beta_1, ..., r_t, \beta_t)$$

4.2 Preservation of Completeness and Soundness

In this subsection, we may assume that V' follows the interactive proof. Thus, if for some $x \in \{0, 1\}^*$, prover P always convinces V on common input x then P' always convinces

V' on this common input. We stress that both V' and P' run in polynomial-time when given oracle access to V and P, respecitely. Thus, the new verifier is a legitimate one. Furthermore, if the original prover P, working in polynomial-time with help of a suitable auxiliary input, could convince the original verifier to accept some common input, then the resulting prover P' could do the same (i.e., can convince V' to accept this common input, while working in polynomial-time with help of the same auxiliary input).

We have just seen that the completeness properties of the original interactive proof is preserved, by the transformation, in a strong sense. Soundness properties are preserved as well, but with some slackness which results from the imperfectness of the Random Selection protocol. In particular,

Proposition 5 *Let $\mu : \{0,1\}^* \mapsto [0,1]$ be a function bounding the probability that verifier V accepts inputs when interacting with any* (possibly cheating) *prover. Namely, $\mu(x)$ is a bound on the probability that V accepts x. Suppose that on input x, the interactive proof (P, V) runs for $t(|x|)$ rounds. Then, $\mu'(x) \stackrel{\text{def}}{=} \mu(x) + O(t(|x|) \cdot \varepsilon(|x|))$ is a function bounding the probability that verifier V' accepts inputs when interacting with any* (possibly cheating) *prover.*

proof: Recall that V' plays the role of the challenger in the Random Selection protocol. Thus, the proposition follows quite immediately from Proposition 1. ■

We stress that the above proposition remains valid no matter which of the two version of Random Selection is used. The same holds with respect to the comments regarding completeness (made above).

4.3 Zero-Knowledge

In this subsection, we may assume that P' follows the interactive proof. Assuming that P is zero-knowledge with respect to the verifier V, we prove that P' is zero-knowledge with respect to any probabilistic polynomial-time verifier strategy. Furthermore, this statement holds for the three versions of zero-knowledge; specifically, perfect, almost-perfect (statistical), and computational zero-knowledge.

Proposition 6 *Let (P, V) be a constant-round Arthur-Merlin interactive proof. Suppose that P is perfect* (resp. almost-perfect) [resp. computational] *zero-knowledge* with respect to the honest verifier V *over the set $L \subseteq \{0,1\}^*$. Then P' is perfect* (resp. almost-perfect) [resp. computational] *zero-knowledge* (with respect to any probabilistic polynomail-time verifier) *over the set $L \subseteq \{0,1\}^*$.*

proof: Let M be a simulator witnessing the hypothesis of the proposition. We start by considering the case of perfect zero-knowledge. Then, for every $x \in L$, with non-negligible probability $M(x)$ halts with output, and given that this happens the output is distributed identically to $(P, V)(x)$. For every verifier strategy V^* interacting with P', we construct a simulator M^* as follows. Again, by uniformly selecting and fixing coin tosses for V^*, *we may assume that* V^* *is deterministic.*

The Simulator M^*: On input x, the simulator invokes M and assuming $M(x)$ halts with output, sets $(r_1, \beta_1, ..., r_t, \beta_t) \stackrel{\text{def}}{=} M(x)$; otherwise M^* also halts with no output. The simulator M^* now tries to form transcripts of the Random Selection protocol which end with output r_1, r_2 through r_t, respectively. (Here we use the simulatability of the Random Selection protocol.) A transcript with output r_1 is formed as follows. M^* feeds V^* with input x and obtains h_1, which can be assumed as in Propositions 2 and 3 to be in $H_{n,m}$. Next, M^* computes $a_1 = h_1(r_1)$ and feeds V^* with (x, a_1). If V^* replies with r_1, we've succeeded in forming a transcript for the first invokation of Random Selection and we proceed to the next. Otherwise, M^* halts with no output. We note that for the next invokations of Random Selection, V^* is fed with the entire history so far; for example, to obtain h_2 we feed V^* with (x, a_1, β_1) and next we feed it with (x, a_1, β_1, a_2), where $a_2 = h_2(r_2)$. If all t rounds were completed successfully, M^* halts with output $(h_1, a_1, r_1, \beta_1, ..., h_t, a_t, r_t, \beta_t)$.

The following observation which follows from Proposition 4 simplifies our analysis. Suppose that $(r_1, \beta_1, ..., r_t, \beta_t)$ is a transcript of a (P, V) interaction on common input x. Then, there exists at most one $(P', V^*)(x)$-transcript that matches it. Namely, there is a unique sequence of h_i's and a_i's so that $h_1 = V^*(x)$, $a_1 = h_1(r_1)$, $h_2 = V^*(x, a_1, \beta_1)$, $a_2 = h_2(r_2)$ and so on. It follows that once M has output a transcript the entire operation of M^* is determined. In particular, all invokations of V^* are on inputs which are already determined.

The above construction will be used also in case of almost-perfect and computational zero-knowledge. However, we start by analyzing it in case of perfect zero-knowledge. The next two claims establish that P' is perfect zero-knowledge in this case.

Claim 1: If M perfectly simulates (P, V) then M^* produces output with non-negligible probability (i.e., there exists a positive polynomial p such that, for every x, on input x machine M produces output with probability at least $1/p(|x|)$).

proof: It suffices to bound the fraction of r's which appear as output of the Random Selection protocol when the challenger uses strategy V^* (with adequately added auxiliary inputs, i.e., x for the first invokation, (x, a_1, β_1) for the second, and so on). By Proposition 3 and the setting of the parameters in the construction of (P', V'), it follows that this fraction is bounded by a non-negligible function of $|x|$, denoted $f(|x|)$. Thus, $M^*(x)$ produces an output with probability at least $p(|x|) \cdot f(|x|)^t$, where $p(|x|)$ is a lower bound on the probability that $M(x)$ produces output. The claim follows. $\Box$

Claim 2: If the output distribution $M(x)$ equals the distribution $(P,V)(x)$ then the output distribution $M^*(x)$ equals the distribution $(P',V^*)(x)$.

proof: Consider a generic transcript, $(h_1,a_1,r_1,\beta_1,...,h_t,a_t,r_t,\beta_t)$, in the support of either distributions (i.e., $M^*(x)$ or $(P',V*)(x)$). As always[5], such a transcript is totally determined by the prover messages, namely the subtranscript $(a_1,\beta_1,...,a_t,\beta_t)$. We first prove that the subtranscript $(a_1,...,a_t)$ appears with equal probability in both distributions; specifically, it appears with probability $(2^{-m})^t$ in both.

- For the distribution $(P',V^*)(x)$, this is obvious by the definition of Random Selection (version 2).
- For the distribution $M^*(x)$, we note that the subtranscript $(a_1,...,a_t)$ appears in an output only if r_i's used to produce it (see construction of M^*) meet a condition that can be satisfied by exactly one sequence of r_i's (i.e., if $r_1 = V^*(x,a_1)$, $r_2 = V^*(x,a_1,\beta_1,a_2)$ and so on). By the fact that M perfectly simulates (P,V) it follows that the r_i's are uniformly distributed.

Thus, each $(a_1,...,a_t)$ subtranscript appear with the same probability in both distributions $M^*(x)$ and $(P',V^*)(x)$. Now, using again the fact that M perfectly simulates (P,V), we conclude that each $(a_1,\beta_1,...,a_t,\beta_t)$ subtranscript appear with the same probability in the two distributions. $\Box$

This concludes our treatment of perfect zero-knowledge. In the other cases (i.e., almost-perfect and computational zero-knowledge), we use the same simulator M^* and adapt the analysis as follows. We start with the case of almost-perfect zero-knowledge, where again we use a pair of claims to establish the validity of the simulation.

Claim 3: If the output distribution ensemble $\{M(x)\}_{x\in L}$ is statistically close to the ensemble $\{(P,V)(x)\}_{x\in L}$ then M^* produces output with non-negligible probability.

proof sketch: The proof follows by an adaptation of the proof of Claim 1. The key observation is that a distribution which is statistically close to uniform (here we refer to the sequence of r_i's produced by M) must hit a non-negligible fraction of the sequences (here we refer to the r_i's produced by Random Selection with V^*) with non-negligible probability. $\Box$

Claim 4: Let M be as in Claim 3. Then the output distribution ensemble $\{M^*(x)\}_{x\in L}$ is statistically close to the ensemble $\{(P',V^*)(x)\}_{x\in L}$

[5] Recall that V^* is determinitic.

proof sketch: We repeat the argument of Claim 2 noting that all equality assertions should be replaced by statistical closeness. Specifically, the r_i's can not be guaranteed to be uniformly distributed but rather statistically close to such a distribution. It follows that the distribution of a_i's in the output of the simulation is statistically close to uniform. (Note that the statistical difference between the a_i's and the uniform distribution may be larger by a polynomial factor than the corresponding difference observed on the r_i's, but still this is negligible.) Similarly, we can argue that the augmentation by the β_i's is statistically close (rather than equal) in the two distributions. $\square$

Finally, we deal with the most complicated case, namely the case of computational zero-knowledge. The following pair of claims is a computational analogue of the previous pair.

Claim 5: Suppose that the output distribution ensemble $\{M(x)\}_{x\in L}$ is computationally indistinguishable from the ensemble $\{(P,V)(x)\}_{x\in L}$. Then M^* produces output with non-negligible probability.

proof sketch: The proof follows by an adaptation of the proof of Claim 1. Here we note that a distribution which is computationally indistinguishable from the uniform distribution (i.e., the sequence of r_i's produced by M) must hit a polynomial-time recognizable set (i.e., the set of r_i's produced by Random Selection with V^*) with probability which may differ from the density of the (recognizable) set by at most a negligible amount. Thus, if the recoginzable set has non-negligible density, as is the case here, then it must be hit with non-negligible probability. $\square$

Claim 6: Let M be as in Claim 5. Then the output distribution ensemble $\{M^*(x)\}_{x\in L}$ is computationally indistinguishable from the ensemble $\{(P',V^*)(x)\}_{x\in L}$.

proof sketch: Here making claims regarding the output distribution of M^* requires a "simulation argument". Specifically, assume towards contradiction, that there exists an algorithm, denoted D^*, that can distinguish the output distribution of $\{M^*(x)\}_{x\in L}$ from the distribution ensemble $\{P',V^*)(x)\}_{x\in L}$. Then, we can construct an algorithm D that can distinguish the output distribution of $\{M(x)\}_{x\in L}$ from the distribution ensemble $\{P,V)(x)\}_{x\in L}$, in contradiction to the hypothesis regarding M. Given a (P,V)-transcript (from either distributions), algorithm D produces a (P',V^*)-transcript by employing the same construction as M^*, specifically by tring to simulate the Random Selection protocol. Clearly, for the $(P,V)(x)$ distribution this must succeeds with non-negligible probability. Also, the success probability on the $M(x)$ distribution must be very close, otherwise we immidiately get a distinguisher. Observe that the extended transcripts produced from the the $(P,V)(x)$ distribution are distributed alike $(P',V^*)(x)$; whereas the extended transcripts produced from the the $M(x)$ distribution are distributed alike $M^*(x)$. Thus, invoking D^* on the extended transcript produced as specified allows us to distinguish the two indistinguishable ensembles, $\{M(x)\}_{x \in L}$ and $\{P, V)(x)\}_{x \in L}$, in contradiction to the hypothesis. $\square$

Thus, in all three cases considered, the corresponding zero-knowledge claim holds. $\blacksquare$

We remark that the above proposition remains valid even if one uses the first version of the Random Selection protocol. However, a slightly more complex simulator will have to be used. The reason being that in the first version (of the Random Selection protocol) the a_i's are not selected uniformly but are rather weighted by the number of their preimages under the corresponding h_i's. Thus, r_i's which are mapped to a_i's with small preimage may be less likely in the real interactions. To compensate for this phenomenon, one may modify the simulator so that it skews the probabilities in the same manner. Namely, when producing a transcript with less likely r_i's, the simulator will discard it with some probability. The required probability (with which to discard transcripts) can be easily computed.

4.4 Conclusions

Combining Propositions 5 and 6, we get

Theorem 1 *Let $\mu : \mathsf{N} \mapsto [0,1]$. Suppose L has a constant-round Arthur-Merlin proof system, with error bound μ, which is perfect* (resp. almost-perfect) [resp. computational] *zero-knowledge* with respect to the honest verifier*. Then, for every positive polynomial $p(\cdot)$, L has a constant-round Arthur-Merlin proof system, with error bound $\mu'(n) \stackrel{\text{def}}{=} \mu(n) + \frac{1}{p(n)}$, which is perfect* (resp. almost-perfect) [resp. computational] *zero-knowledge* (with respect to any probabilistic polynomial-time verifier)*. Furthermore, the zero-knowledge property can be demonstrated using a black-box simulation. Also, if the original system had no error on inputs in L then the same holds for the new system.*

Theorem 1 does not preserve the error probability of the original system. This seems inevitible, in light of [10]. Recall that there are languages believed not to be in $\mathcal{BPP}$ which have constant-round Arthur-Merlin proof systems, with exponentially small error probability, which are zero-knowledge with respect to the honest verifier. For example, Graph Isomorphism has such a system (for perfect zero-knowledge), and assuming the existence of one-way functions, every language in $\mathcal{NP}$ has such a system (for computational zero-knowledge) [12]. Now, a stronger version of Theorem 1, say one in which $\mu'(n) - \mu(n)$ is a negligible function of n, would imply that these languages have constant-round Arthur-Merlin (balck-box) zero-knowledge proof systems (with negligible error probability). But, according to [10], languages having constant-round Arthur-Merlin (balck-box) zero-knowledge proof systems lie in $\mathcal{BPP}$. Needless to say that $\mathcal{NP}$ and even Graph Non-Isomorphism are believed not to lie in $\mathcal{BPP}$.

References

[1] L. Babai. *Trading Group Theory for Randomness*, Proc. of 17th STOC, pages 421–420, 1985.

[2] M. Bellare, S. Micali and R. Ostrovsky: *The (true) Complexity of Statistical Zero-Knowledge*, Proc. of STOC 90.

[3] M. Ben-Or, O. Goldreich, S. Goldwasser, J. Håstad, J. Killian, S. Micali and P. Rogaway: *Everything Provable is Provable in Zero-Knowledge*, Proc. of Crypto 88.

[4] G. Brassard, D. Chaum and C. Crépeau: *Minimum Disclosure Proofs of Knowledge*, JCSS.

[5] G. Brassard, C. Crépeau and M. Yung, "Everything in NP can be Argued in Perfect Zero-Knowledge in a Constant Number of Rounds", proceedings of 16th ICALP, pp. 123–136, 1989.

[6] I. Damgård: *Interactive Hashing can Simplify Zero-Knowledge Protocol Design Without Computational Assumptions*, Proc. of Crypto 93.

[7] U. Feige and A. Shamir, "Zero-Knowledge Proofs of Knowledge in Two Rounds", Advances in Cryptology – Crypto89 (proceedings), pp. 526–544, 1990.

[8] O. Goldreich, S. Goldwasser and N. Linial, "Fault-Tolerant Computation without Assumptions: the Two-Party Case", Proc. of 32nd FOCS, pp. 447–457, 1991.

[9] O. Goldreich and A. Kahan, How to Construct Constant-Round Zero-Knowledge Proof Systems for NP, submitted for publication in Journal of Crypology, 1993.

[10] O. Goldreich and H. Krawcyzk, "On the Composition of Zero-Knowledge Proof Systems", proceedings of 17th ICALP, pp. 268–282, 1990.

[11] O. Goldreich, Y. Mansour and M. Sipser: *Proofs that Never Fail and Random Selection*, Proc. of FOCS 87.

[12] O. Goldreich, S. Micali and A. Wigderson: *Proofs that yield Nothing but their Validity and a Methodology of Cryptographic Protocol Design*, Proc. of FOCS 86.

[13] S. Goldwasser, S. Micali and C. Rackoff: *The Knowledge Complexity of Interactive Proof Systems*, SIAM J. Computing, Vol. 18, pp. 186–208, 1989.

[14] S. Goldwasser and M. Sipser. *Private Coins versus Public Coins in Interactive Proof Systems*, Proc. of 18th STOC, pages 59–68, 1986.

[15] R. Impagliazzo and M. Yung, *Direct Minimum-Knowledge Computations*, Advances in Cryptology - Crypto87 (proceedings), 1987, pp. 40–51.

[16] M. Naor: *Bit Commitments from Pseudorandomness*, Proc. of Crypto 89.

[17] M. Naor, R. Ostrovsky, R. Venkatesan and M. Yung: *Zero-Knowledge Arguments for NP can be Based on General Complexity Assumptions*, Proc. of Crypto 92.

[18] R. Ostrovsky, R. Venkatesan and M. Yung: *Interactive Hashing Simplifies Zero-Knowledge Protocol Design*, Proc. of EuroCrypt 93.

[19] R. Ostrovsky and A. Wigderson: *One-Way Functions are Essential for Non-Trivial Zero-Knowledge*, Proc. 2nd Israel Symp. on Theory of Computing and Systems, 1993.