A Uniform-Complexity Treatment of Encryption and Zero-Knowledge*

Oded Goldreich
Department of Computer Science
Weizmann Institute of Science
Rehovot, Israel.
oded@wisdom.weizmann.ac.il

July 1991, revised July 1998

Abstract

We provide a treatment of encryption and zero-knowledge in terms of uniform complexity measures. This treatment is appropriate for cryptographic settings modeled by probabilistic polynomial-time machines. Our uniform treatment allows to construct secure encryption schemes and zero-knowledge proof systems (for all $\mathcal{NP}$) using only uniform complexity assumptions.

We show that uniform variants of the two definitions of security, presented in the pioneering work of Goldwasser and Micali, are in fact equivalent. Such a result was known before only for the non-uniform formalization.

Non-uniformity is implicit in all previous treatments of zero-knowledge in the sense that a zero-knowledge proof is required to "leak no knowledge" on *all* instances. For practical purposes, it suffices to require that it is *infeasible to find* instances on which a zero-knowledge proof "leaks knowledge". We show how to construct such zero-knowledge proof systems for every language in $\mathcal{NP}$, using only a uniform complexity assumption. Properties of uniformly zero-knowledge proofs are investigated and their utility is demonstrated.

Remark: This version contains proofs for two claims made (without proof) in the original version. Specifically, we refer to the proofs of Remarks 10 and 18 given in a new appendix.
(O.G., July 1998)

*Appeared in *Journal of Cryptology*, Vol. 6, No. 1, (1993), pp. 21–53.

Contents

Reader, I here put into thy hands what has been the diversion of some of my idle and heavy hours. If it has the good luck to prove so of any of thine, and thou hast but half so much pleasure in reading as I had in writing it, thou wilt as little think thy money, as I do my pains, ill bestowed.

[J. Locke, 1690].

1 Introduction

In this paper we provide a treatment of the notions of secure encryption and zero-knowledge proofs in terms of uniform complexity. These two notions were treated before mainly in terms of non-uniform complexity.

In the seminal work of Goldwasser and Micali [18], two definitions of security are presented and considered. The formalization is carried out in terms of non-uniform complexity, but this is not essential to the results presented which include a one-directional implication between these definitions. In [23], non-uniform variants of the two definitions are proved equivalent (but the argument uses non-uniformity in an essential way). In this paper, we present several uniform variants of the two definitions and demonstrate that they are equivalent. It follows from [4, 29, 13, 21, 20] (resp. [18, 29]), that uniformly secure private-key (resp. public-key) encryption schemes can be constructed based on the existence of one-way functions (resp. trapdoor permutations).

All previous formalizations of zero-knowledge, including and following the introduction of the concept in the pioneering work of Goldwasser, Micali and Rackoff [19], are non-uniform in the sense that they require that the proof system "leaks no knowledge" on *all* instances. It is thus not surprising that all the constructions of zero-knowledge proof systems for $\mathbf{NP}$-complete languages (e.g. [15]), employ *non-uniform* complexity assumptions. Clearly, for practical purposes it suffices to require that it is *infeasible to find* instances on which a zero-knowledge proof system "leak knowledge". Such (possibly weaker) zero-knowledge proof systems can be employed within cryptographic protocols, and in particular in the automatic generation of cryptographic protocols for any computable game (cf. [30, 16]). We formalize this (possibly weaker) variant of zero-knowledge, and show (following the ideas in [15]) how to construct such zero-knowledge proof systems for every language in $\mathbf{NP}$ using only a uniform complexity assumption (e.g. the existence of one-way functions).

What is gained by a uniform complexity treatment

We first stress that by uniform or non-uniform complexity treatment of cryptographic primitives we merely refer to the modelling of the adversary. The honest (legitimate) parties are always modelled by uniform complexity classes (most commonly probabilistic polynomial-time). The alternative treatments refer only to the way the adversary is modelled: namely, by uniform or non-uniform complexity classes. We remind the reader that natural uniform complexity classes (e.g., $\mathbf{BPP}$) are known to be contained in their non-uniform counterpart (e.g., non-uniform-$\mathbf{P}$ also denoted $\mathbf{P/poly}$), but the converse is false (e.g., non-uniform classes may contain even non-recursive languages). Hence, a non-uniform adversary is never weaker than a uniform one, and furthermore it might be stronger.

While the question of what is the right model of "real world" cryptography (i.e., uniform vs. non-uniform) is somewhat controversial, we believe that it is important to demonstrate that it is possible to carry out most of the cryptographic theory, developed in recent years, also in the model of uniform computations. Furthermore, the uniform treatment has advantages discussed below.

Typically, the theorems in the cryptographic theory are reductions of the security of a cryptographic primitive to an intractability assumption. More specifically, their proof of security show how to transform an adversarial procedure which "breaks" the primitive into an algorithm contradicting the intractability assumption. A uniform transformation implies a non-uniform one, whereas the converse is not always true. Hence, the uniform transformation supplied by the uniform treatment is technically superior to the non-uniform one. In particular, probabilistic polynomial-time reductions imply reduction by non-uniform polynomial-size circuits. In addition, uniform reductions tend to point to the essential ideas more than non-uniform reductions, in which essential objects may be incorporated into the algorithms and become implicit.

Another reason to prefer the uniform treatment is that it is based only on uniform intractability assumptions. These assumptions are seemingly weaker (and never stronger) than their non-uniform counterparts. In particular, the uniformly secure primitives (i.e. encryption and zero-knowledge proofs for all languages in $\mathbf{NP}$) can be constructed using only uniform complexity assumptions. This should be confronted with the (seemingly stronger) non-uniform assumptions used in the construction of the non-uniformly secure primitives (cf. [18, 15]).

However, something is lost when relying on these (seemingly weaker) uniform assumptions. Namely, the security we obtain is only against the (seemingly weaker) uniform adversaries. We believe that this loss in security is immaterial. Our belief is based on the thesis that uniform complexity is the right model of "real world" cryptography. We believe that it is reasonable to consider only objects (i.e., inputs) generated by uniform and efficient procedures and the effect that these objects have on uniformly and efficient observers (i.e., adversaries). In particular, schemes secure against probabilistic polynomial-time adversaries can be used in any setting consisting of probabilistic polynomial-time machines with inputs generated by probabilistic polynomial-time procedures. We believe that the cryptographic setting is such a case.

Organization: In Section 2 we present basic conventions that will be used throughout the paper. In particular, Section 3 which treats secure encryption uses conventions presented in subsection 2.1, whereas Section 4 which treats zero-knowledge uses conventions presented in subsection 2.2. Some concluding remarks appear in Section 5.

2 Conventions

Throughout this paper we mainly consider probability distributions which can be sampled in polynomial-time. Following are our basic conventions, which make some abuse of classical terms.

Definition 1 (random variables): *A* random variable *is a sequence of variables* $\{X_n\}_{n\in\mathbb{N}}$ *defined over a probability space such that there exists a polynomial* $Q(\cdot)$ *so that (for all* n*)* X_n *ranges over the set of strings of length at most* $Q(n)-1$*. We denote by* $\mathrm{Prob}(X_n=\alpha)$ *the probability that* X_n *equals* α*, where the probability is taken over this space.*

For simplicity, we encode the elements of $\cup_{i\le Q(n)}\{0,1\}^i$ by $\{0,1\}^{Q(n)}$, and hence assume that X_n in the above definition ranges over $\{0,1\}^{Q(n)}$.

Definition 2 (polynomial-time random variables): *A random variable* $\{X_n\}_{n\in\mathbb{N}}$ *is called* polynomial-time *if there exists a probabilistic polynomial-time algorithm* A *such that* $\mathrm{Prob}(A(1^n)=\alpha) = \mathrm{Prob}(X_n=\alpha)$*.*

2.1 Encryption Schemes

We now recall the definition of an encryption scheme. This definition says nothing about the security of the scheme (which is the subject of the next section).

Definition 3 (encryption): *An* encryption scheme *is a triple,* (G,E,D)*, of probabilistic polynomial-time algorithms satisfying the following two conditions*

1. *On input* 1^n*, algorithm* G *(called the* key generator*) outputs a pair of* n*-bit long strings. Let* $G_1(1^n)$ *denote the first string in this pair, and* $G_2(1^n)$ *denote the second one.*
2. *For every pair* (e,d) *in the range of* $G(1^n)$*, algorithms* E *(*encryption*) and* D *(*decryption*) satisfy for each* $\alpha\in\{0,1\}^n$*:*
$$\mathrm{Prob}(D(d,E(e,\alpha))=\alpha)=1$$
(e,d) is a pair of "corresponding" encryption/decryption keys. $E(e,\alpha)$ *is the encryption of cleartext* $\alpha\in\{0,1\}^n$ *using the encryption key* e*, whereas* $D(d,\beta)$ *is the decryption of the ciphertext* β *using the decryption key* d*.*

At this stage, encryption using a key of length n is defined only for messages of length n; generalization is postponed to Convention 2. The convention $|e|=|d|=n$ for every (e,d) in the range of $G(1^n)$ can be drastically relaxed. Clearly, the length of e and d must be polynomial in n. One the other hand, since n also serves as the "security parameter" (see definitions in Section 3), n must be polynomial in $|d|$ (in which case "polynomial in n" is equivalent to "polynomial in $|d|$"). Hence, using polynomial padding, we may assume, without loss of generality, that $|e|=|d|=n$. Condition (2) in Definition 3 may be relaxed so that inequality may occur with negligible probability. For simplicity, we chose to adopt here the more conservative requirement.

Definition 3 does not distinguish "private-key" encryption schemes from "public-key" ones. The difference between the two is in the security definitions: in a public-key scheme the "breaking algorithm" gets e (the encryption key) as an additional input (and thus $e\neq d$ follows); while in private-key schemes e is not given to the "breaking algorithm" (and thus one may assume, without loss of generality, that $e=d$).

Convention 1 : In the sequel we write $E_e(\alpha)$ instead of $E(e, \alpha)$ and $D_d(\beta)$ instead of $D(d, \beta)$. Whenever there is little risk of confusion, we drop these subscripts.

Convention 2 : Messages of length not equal to n (the length of the encryption key) are encrypted by breaking them into blocks of length n and possibly padding the last block. We extend the notation so that

$$E_e(\alpha_1 \cdots \alpha_l \alpha_{l+1}) = E_e(\alpha_1) \cdots E_e(\alpha_l) \cdot E_e(\alpha_{l+1} p)$$

where $|\alpha_1| = \cdots = |\alpha_l| = n$, $|\alpha_{l+1}| \leq n$, and p is some standard padding of length $n - |\alpha_{l+1}|$.

Remark 1 : The above convention may be interpreted in two ways. First, it waves the extremely restricting convention by which the encryption scheme can be used only to encrypt messages of length equal to the length of the key. Second, it allows to reduce the security of encrypting many messages using the same key to the security of encrypting a single message. Convention 2 is used in an essential way in the proof of Proposition 2.

The next convention regarding encryption schemes introduces a breach of security: namely, the length of the cleartext is always revealed by encryption schemes which follow this convention. However, as we show in Appendix A, some information about the length of the cleartext must be leaked by any encryption scheme.

Convention 3 : The encryption algorithm maps messages of the same length to cryptograms of the same length. Namely, there exists a polynomial $Q(\cdot)$ such that for all $e, \alpha \in \{0,1\}^n$, the random variable $E_e(\alpha)$ ranges over $\{0,1\}^{Q(n)}$.

Convention 3 is used in an essential way in the proof of Proposition 2.

2.2 Interactive Machines (Protocols)

The notion of an *interactive machine*, suggested by M. Blum, is the key to a formalization of the notion of a protocol (cf. [19, 15]). We assume that the reader is familiar with this notion, and present some conventions regarding interactive machines.

Convention 4 : Let $\Pi = (A, B)$ be a pair of (probabilistic) interactive machines (i.e. a "two-party protocol"). We denote by $B_{A(x,y)}(x, z)$ the output (distribution) of machine B on input x, z when interacting with $A(x, y)$. In this notation, the string x is a common input to both machines, while y is an auxiliary input to machine A, and z is an auxiliary input to machine B. (The coin tosses of both parties are implicit in the notation.)

Remark 2 : In the context of zero-knowledge interactive proof systems for a language $L \in \mathbf{NP}$, the string $x \in L$ is the common input, the string y is an NP-witness (known to the prover) for $x \in L$, and the string z is partial information a-priori known to the verifier.

Convention 5 : The running time of an interactive machine is considered as a function of the length of the common input. Thus, when saying that the interactive machine A is *polynomial-time* we mean that there exists a polynomial $Q(\cdot)$ such that for every interactive machine B^*, and every triple (x, y, z), the number of steps machine A makes on input x, y when interacting with $B^*(x, z)$ is at most $Q(|x|)$. We say that the protocol $\Pi = (A, B)$ is *polynomial-time* if both machines A and B are (probabilistic) polynomial-time. In the sequel, we will consider only probabilistic polynomial-time protocols.

3 Uniform Security of Encryption Schemes

In this section we present several uniform-complexity variants of the two definitions of security introduced by Goldwasser and Micali [18], and prove the equivalence of these variants. The first definition, called *semantic security*, is the most natural one. Semantic security is a computational complexity analogue of Shannon's definition of perfect privacy [27]. Loosely speaking, an encryption scheme is semantically secure if the encryption of a message does not yield any information on the message to an adversary which is computationally restricted (e.g. to polynomial-time). The second definition has a more technical flavour. It interprets security as the unfeasibility of distinguishing between encryptions of a given pair of messages. This definition is technically useful in demonstrating the security of a proposed encryption scheme, and for arguments concerning properties of cryptographic protocols which utilize an encryption scheme.

3.1 Semantic Security

Loosely speaking, semantic security as defined in [18] means that whatever can be efficiently computed from the encryption of the message, can be efficiently computed given only the length of the message. Here we augment this definition by requiring that the above remains valid in presence of auxiliary partial information about the message. Namely, whatever can be efficiently computed from the encryption of the message and additional partial information about the message, can be efficiently computed given only the length of the message and the same partial information. In the non-uniform case the augmented definition collides with the original one, but in the uniform case (considered here) the augmented definition seems stronger.

Definition 4 *: An encryption scheme, (G, E, D), is* semantically secure *if for every probabilistic polynomial-time algorithm A, there exists a probabilistic polynomial-time algorithm A', such that for every polynomial-time random variable $\{X_n\}_{n\in\mathbb{N}}$, every polynomial-time computable function $h : \{0,1\}^* \mapsto \{0,1\}^*$, every function $f : \{0,1\}^* \mapsto \{0,1\}^*$, every constant $c > 0$ and all sufficiently large n*

$$\mathrm{Prob}\left(A(E_{G_1(1^n)}(X_n), h(X_n), 1^n) = f(X_n)\right) < \mathrm{Prob}\left(A'(|X_n|, h(X_n), 1^n) = f(X_n)\right) + \frac{1}{n^c}$$

The probability in the above terms is taken over the probability space underlying X_n and the internal coin tosses of algorithms G, E, A and A'.

Remark 3 : The primary role of the input 1^n is to allow both A and A' to run $\mathrm{poly}(n)$ steps. This is crucial in case of algorithm A' (to guarantee that A' and A have essentially the same running time with respect to X_n, even in case $h(X_n)$ is very short). It is also crucial that the length of X_n (at least up to a polynomial factor) is given as an additional input to A' (otherwise no encryption scheme may be secure - see Appendix A).

Remark 4 : The function h provides both algorithms with partial information on X_n. These algorithms then try to find the value $f(X_n)$. In the definition of semantic security appearing in [18], the function $h : \{0,1\}^* \mapsto \{0,1\}^*$ does not appear (i.e. $h(x) = \lambda$, for all x). In the non-uniform case these formalizations are equivalent[1], but in the uniform case our choice seems stronger. The function h plays an essential role in the proof of Proposition 2.

[1] See footnote in Remark 8.

Remark 5 : As in [18], we do not require that the function $f : \{0,1\}^* \mapsto \{0,1\}^*$ is even computable. This seems strange at first glance, but as we shall see in the sequel the meaning of semantic security is essentially that the distributions $A(E(X_n), h(X_n), 1^n)$ and $A'(|X_n|, h(X_n), 1^n)$ are statistically close. Note that the last statement does not refer to the function f. On the other hand, in view of our results (see subsection 3.3), restricting f to be polynomial-time computable yields an equivalent definition (both in the uniform and non-uniform case).

Remark 6 : The order of quantifiers in Definition 4 (i.e., $\forall A \exists A' \forall \{X_n\}_{n\in\mathbb{N}} \forall h \forall f$) is the "strongest" one possible. Yet, in view of our results (combined with Remark 9 below), this formalization is equivalent to the formalization in which the quantifiers are in the "weakest" possible order (i.e., $\forall A \forall \{X_n\}_{n\in\mathbb{N}} \forall h \forall f \exists A'$).

Remark 7 : The definition presented above corresponds to "private-key" encryption schemes. To derive a definition of security for "public-key" encryption schemes the string $G_1(1^n)$ (i.e. the public-key) should be given to algorithm A as an additional input [2]. A seemingly stronger definition of semantic security allows $\{X_n\}_{n\in\mathbb{N}}$ to be computed in probabilistic polynomial-time from the public-key (and not only from 1^n). However, the known constructions of public-key encryption schemes (secure in the sense of Definition 4) are also secure in this stronger sense.

Remark 8 : To derive a non-uniform version of Definition 4, we replace everywhere the term "probabilistic polynomial-time algorithm" by the term "family of polynomial-size circuits". However, in this case the definition is also equivalent to one in which $\{X_n\}_{n\in\mathbb{N}}$ may be an arbitrary random variables and $h : \{0,1\}^* \mapsto \{0,1\}^*$ may be an arbitrary function[3].

Remark 9 : In the definition of semantic security appearing in [18], the term $\max_{u,v}\{\mathrm{Prob}(f(X_n)=v|h(X_n)=u)\}$ appears instead of the term $\mathrm{Prob}(A'(h(X_n), 1^n, |X_n|) = f(X_n))$. Note that

$$\max_{u,v}\{\mathrm{Prob}(f(X_n)=v|h(X_n)=u)\} \geq \mathrm{Prob}(A'(h(X_n), 1^n, |X_n|) = f(X_n)),$$

for every algorithm A'. Hence, for a fixed random variable $\{X_n\}_{n\in\mathbb{N}}$, our requirement seems weaker. However, in the *special case*, where each u satisfies $\mathrm{Prob}(f(X_n) = 0|h(X_n) = u) = \mathrm{Prob}(f(X_n) = 1|h(X_n) = u) = \frac{1}{2}$, the above terms are equal (as A' can easily achieve success probability 1/2 by simply tossing a coin). In view of our results (see subsection 3.3), it suffices to consider only this special case (both in the uniform and non-uniform formulations). It follows that the two formulations are equivalent.

3.2 Indistinguishability of Encryptions

The definition presented here is a uniform-complexity variant of the definition appearing in [18] under the title "polynomial security". We prefer to use the more informative (and cumbersome)

[2] It can be easily shown, that a public-key encryption scheme which is semantically secure must have a probabilistic encryption algorithm. Otherwise, consider a random variable X_n uniformly distributed over $\{0^n, 1^n\}$. This observation may justify the title of [18] (but indeed this seems a rather poor justification).

[3] We show that in the non-uniform context quantifying over all functions h or considering only $h(x) = \lambda$ for all x, yields equivalent notions of security. In light of the results in subsection 3.3 (cf. Remark 9 below), it suffices to consider random variables $\{X_n\}_{n\in\mathbb{N}}$ satisfying for each u, $\mathrm{Prob}(f(X_n)=0|h(X_n)=u) = \mathrm{Prob}(f(X_n)=1|h(X_n)=u) = \frac{1}{2}$. Consider an algorithm A, predicting $f(X_n)$ from $E(X_n)$ and $h(X_n)$ with success probability greater than $\frac{1}{2}+\frac{1}{n^c}$. Consider the choice of u maximizing $\mathrm{Prob}(A(E(X_n),u) = f(X_n))$, and define Y_n such that $\mathrm{Prob}(Y_n=\alpha) = \mathrm{Prob}(X_n = \alpha|h(X_n)=u)$. Note that Y_n is not necessarily polynomial-time computable (even in case X_n is). The non-uniform algorithm, denoted A_u, demonstrating that the scheme is not secure even in the restricted sense, is a modification of A which incorporates u (i.e., $A_u(E(Y_n)) = A(E(Y_n), u)$).

term of "indistinguishability of encryptions". This technical definition states that it is infeasible to find pairs of messages for which an efficient test can distinguish the corresponding encryptions. Loosely speaking, an algorithm A is said to distinguish the random variables R_n and S_n if A "behaves" substantially different in case the input is distributed as R_n and in case the input is distributed as S_n. Without loss of generality, it suffices to ask whether $\text{Prob}(A(R_n) = 1)$ and $\text{Prob}(A(S_n) = 1)$ are substantially different[4].

Definition 5 : *An encryption scheme, (G, E, D), has* indistinguishable encryptions *if for every polynomial-time random variable $\{T_n = X_n Y_n Z_n\}_{n\in\mathbb{N}}$ (with $|X_n| = |Y_n|$), every probabilistic polynomial-time algorithm A, every constant $c > 0$ and all sufficiently large n*

$$|\text{Prob}\left(A(Z_n, E_{G_1(1^n)}(X_n))=1\right) - \text{Prob}\left(A(Z_n, E_{G_1(1^n)}(Y_n))=1\right)| < \frac{1}{n^c}$$

The probability in the above terms is taken over the probability space underlying T_n and the internal coin tosses of algorithms G, E, and A.

Remark 10 : Equivalently, we may require that for every $c > 0$ and all sufficiently large n: $\text{Prob}(T_n \in B_n^{(c)}) < \frac{1}{n^c}$, where

$$B_n^{(c)} \stackrel{\text{def}}{=} \left\{(\alpha, \beta, \gamma) : |\text{Prob}\left(A(\gamma, E_{G_1(1^n)}(\alpha))=1\right) - \text{Prob}\left(A(\gamma, E_{G_1(1^n)}(\beta))=1\right)| > \frac{1}{n^c}\right\}$$

A proof of the equivalence with Definition 5 is given in Appendix B.

Remark 11 : The random variable Z_n models additional information, on the message space, given to algorithm A which tries to distinguish the encryptions (of X_n and Y_n). A *special case* of interest is when $Z_n = X_n \cdot Y_n$. In view of our results (see subsection 3.3), restricting attention to this special case (as done in [18]) is equivalent to the general case.

Remark 12 : In the non-uniform case, the definition is equivalent to requiring that, for all families of polynomial-size circuits $\{C_n\}_{n\in\mathbb{N}}$, all sequences of pairs $\{(\alpha_n, \beta_n)\}_{n\in\mathbb{N}}$ (with $|\alpha_n| = |\beta_n|$), and all $c > 0$ and all sufficiently large n

$$\left|\text{Prob}\left(C_n(E_{G_1(1^n)}(\alpha_n))=1\right) - \text{Prob}\left(C_n(E_{G_1(1^n)}(\beta_n))=1\right)\right| < \frac{1}{n^c}$$

(Namely, in this case the a-priori information can be incorporated into the circuit.)

Remark 13 : The definition presented above corresponds to "private-key" encryption schemes. Again, security definitions for "public-key" encryption schemes can be derived by adding the public-key (i.e. the string $G_1(1^n)$) as an additional input (and possibly allowing $\{X_n\}$ to be computed in probabilistic polynomial-time from the public-key).

[4]The output of A may be interpreted as a "verdict" that the input is taken from the distribution underlying R_n. For a "verdict" to be meaningful it has to be correlated with the actual situation.

3.3 Equivalence of the Security Definitions

The equivalence of the non-uniform versions of the definitions of security has been proved in [18, 23]. Goldwasser and Micali proved that indistinguishability of encryptions implies semantic security [18]. Their proof carries through also to the uniform case. This implication is very important since it seems easier to prove that a proposed encryption scheme has indistinguishable encryptions (than to prove directly that it is semantically secure). Micali, Rackoff and Sloan proved that semantic security implies indistinguishability of encryptions [23], but their proof seems to use non-uniformity in an essential way. Since our variant of semantic security seems stronger than the uniform variant implicit in [18] we present here both directions of the proof of equivalence of Definitions 4 and 5. We state and prove our result for "private-key" encryption scheme. A similar result holds for "public-key" encryption schemes.

Theorem 1 *: An encryption scheme is semantically secure if and only if it has indistinguishable encryptions.*

Let (G, E, D) be an encryption scheme. We formulate a proposition for each of the directions of the above theorem. Each proposition is in fact stronger than the corresponding direction (stated in Theorem 1).

Proposition 1 *: Let (G, E, D) have (the property of) indistinguishable encryptions even in the restricted sense considered in Remark 11 (i.e., for triples $T_n = \{X_n Y_n Z_n\}_{n\in\mathbb{N}}$, where $Z_n = X_n Y_n$). Then (G, E, D) is semantically secure.*

Proposition 2 *: Let (G, E, D) be semantically secure even in the restricted sense considered in Remarks 5, 6 and 9 (i.e., f is polynomial-time computable, quantifiers are "reversed" and the message distribution M_n is such that $f(M_n)$ is equally likely to be 0 or 1 given the value of $h(M_n)$). Then (G, E, D) has indistinguishable encryptions.*

Proof of Proposition 1

We show that if (G, E, D) is not semantically secure then it has distinguishable encryptions (even in the restricted sense mentioned in the hypothesis of the proposition). Namely, for every probabilistic polynomial-time algorithm A, we present a probabilistic polynomial-time algorithm A'. We show that if for some $\{X_n\}_{n\in\mathbb{N}}$, f, and h (as in Definition 4), A guesses $f(X_n)$ from $E(X_n)$ and $h(X_n)$ better than what A' does on input $|X_n|$ and $h(X_n)$, then we can distinguish the encryptions of X_n and $Y_n \stackrel{\text{def}}{=} 1^{|X_n|}$ (using auxiliary input $Z_n = X_n Y_n$).

Let A be a probabilistic polynomial-time algorithm which tries to infer partial information (i.e., the value $f(X_n)$) from the encryption of message X_n (and a-priori information $h(X_n)$). Namely, on input $E(\alpha)$ and $h(\alpha)$, algorithm A tries to guess $f(\alpha)$. We construct a probabilistic polynomial-time algorithm, A', which performs as well without getting the input $E(\alpha)$. Algorithm A' consists of running the algorithm A on input $E(1^{|\alpha|})$ and $h(\alpha)$ (recall that A' gets $|\alpha|$ as input). Indistinguishability of encryptions will be used to prove that A' performs as well as A. Note that the construction of A' does not depend on the functions h and f or on the distribution of messages to be encrypted. We have to show that $(\forall c \exists N \forall n > N)$

$$\text{Prob}\left(A(E_{G_1(1^n)}(X_n), h(X_n), 1^n) = f(X_n)\right) < \text{Prob}\left(A'(|X_n|, h(X_n), 1^n) = f(X_n)\right) + \frac{1}{n^c}$$

Using the definition of A', the above can be rewritten as

$$\text{Prob}\left(A(E_{G_1(1^n)}(X_n), h(X_n), 1^n) = f(X_n)\right) < \text{Prob}\left(A(E_{G_1(1^n)}(1^{|X_n|}), h(X_n), 1^n) = f(X_n)\right) + \frac{1}{n^c}$$

Assuming, to the contradiction that $\exists c > 0$ and infinitely many n's violating the above inequality, we have $\text{Prob}(X_n \in B_n) \geq \frac{1}{2n^c}$, where B_n is the set of strings $\alpha \in \{0,1\}^m$ satisfying

$$\text{Prob}\left(A(E_{G_1(1^n)}(\alpha), h(\alpha), 1^n) = f(\alpha)\right) > \text{Prob}\left(A(E_{G_1(1^n)}(1^n), h(\alpha), 1^n) = f(\alpha)\right) + \frac{1}{2n^c}$$

Clearly, we have $\text{Prob}(X_n \in D_n) \geq \frac{1}{2n^c}$, where D_n is the set of strings $\alpha \in \{0,1\}^m$ satisfying for some v_α

$$\left|\text{Prob}\left(A(E_{G_1(1^n)}(\alpha), h(\alpha), 1^n) = v_\alpha\right) - \text{Prob}\left(A(E_{G_1(1^n)}(1^n), h(\alpha), 1^n) = v_\alpha\right)\right| > \frac{1}{2n^c} \tag{1}$$

In the sequel we show that $\text{Prob}(X_n \in D_n) \geq \frac{1}{2n^c}$ implies that encryptions of different messages can be distinguished, in contradiction to the hypothesis of the proposition. (Hence, contradiction is derived through D_n in which there is no reference to the function f. This may explain the lack of restrictions on the function f.)

We define a random variable $\{Z_n = X_n \cdot 1^{|X_n|}\}$, and construct an algorithm A_2 which (when given auxiliary information $(X_n, 1^m)$) distinguishes the encryptions of X_n and 1^m, where $m = |X_n|$. The algorithm can be shown to perform well (i.e. distinguish encryptions) when X_n is in D_n.

Description of algorithm A_2
On input $\alpha, 1^m$ and $E_e(\gamma)$ (where e is in the range of $G_1(1^n)$ and $\gamma \in \{\alpha, 1^m\}$), algorithm A_2 proceeds in two phases. Loosely speaking, in the first phase the algorithm checks whether α is in D_n and in case the answer is affirmative finds a "witness" (i.e., v_α satisfying Eq. 1) for membership of $\alpha \in D_n$. In the second phase the algorithm "guesses" the identity of γ by whether or not $A(E_e(\gamma), h(\alpha), 1^n)$ equals the witness v_α found in phase 1. Details follow.

Phase 1: Ignoring $E_e(\gamma)$, algorithm A_2 first gathers statistics on the distribution of the random variables $A(E_{G_1(1^n)}(\alpha), h(\alpha), 1^n)$ and $A(E_{G_1(1^n)}(1^m), h(\alpha), 1^n)$. To this end, A_2 computes $h(\alpha)$ and runs A polynomially many[5] times each time feeding A with a randomly computed $E_{G_1(1^n)}(\alpha)$ (resp., a randomly computed $E_{G_1(1^n)}(1^m)$) and the values $h(\alpha)$ and 1^n. Define

$$p_{\alpha,\beta}(v) \stackrel{\text{def}}{=} \text{Prob}\left(A(E_{G_1(1^n)}(\beta), h(\alpha), 1^n) = v\right) \tag{2}$$

Let $\pi_{\alpha,\beta}(v)$ be a random variable representing the estimator of $p_{\alpha,\beta}(v)$ obtained by polynomially many trials, where the polynomial is determined so that Eq. 3 holds. Note that $p_{\alpha,\beta}(v)$ is a value while $\pi_{\alpha,\beta}(v)$ is a random variable. Fixing α and β, with very high probability (say $1 - 2^{-n}$), the following holds for *every* possible value, v:

$$|p_{\alpha,\beta}(v) - \pi_{\alpha,\beta}(v)| < \frac{1}{16n^{2c}} \tag{3}$$

Phase 1 is completed with an arbitrary choice of a value v^* for which

$$|\pi_{\alpha,\alpha}(v^*) - \pi_{\alpha,1^m}(v^*)| > \frac{3}{8n^c} \tag{4}$$

[5]The polynomial depends on the desired accuracy and can be determined so that Eq. 3 (below) holds.

Such a value is found with very high probability if $\alpha \in D_n$ (since for $\alpha \in D_n$ we have $|p_{\alpha,\alpha} - p_{\alpha,1^m}| > \frac{1}{2n^c}$ and by Eq. 3 the π's are good estimators). In case no such value is found, the algorithm does not continue to the second phase but rather halts outputting 1, *obliviously of* $E_e(\gamma)$.

Phase 2: This phase is performed only if v^* has been found satisfying Eq. 4. Without loss of generality, assume $\pi_{\alpha,\alpha}(v^*) > \pi_{\alpha,1^m}(v^*) + \frac{3}{8n^c}$. Algorithm A_2 now runs $A(E_e(\gamma), h(\alpha), 1^n)$ and outputs 1 iff A has output the value v^* (found in phase 1).

Evaluation of algorithm A_2

We now evaluate the performance of A_2 on input $E_e(\gamma)$, 1^m and α, for γ being either α or 1^m. We consider three cases

Case 1: $\alpha \in D_n$ (i.e., $\exists v$ satisfying $|p_{\alpha,\alpha}(v) - p_{\alpha,1^m}(v)| \geq \frac{1}{2n^c}$). In this case, with very high probability (say $1 - 2^{-n}$), phase 1 is completed successfully and v^* is found satisfying, without loss of generality, $p_{\alpha,\alpha}(v^*) \geq p_{\alpha,1^m}(v^*) + \frac{3}{8n^c}$. Hence, in this case

$$\begin{aligned}
\mathrm{Prob}(A_2(\alpha, 1^m, E_{G_1(1^n)}(\alpha))=1) - \mathrm{Prob}(A_2(\alpha, 1^m, E_{G_1(1^n)}(1^m))=1) &> (1-2^{-n}) \cdot \frac{3}{8n^c} - 2^{-n} \\
&> \frac{3}{8n^c} - \frac{1}{32n^{2c}}
\end{aligned}$$

Case 2: $\alpha \notin D_n$ yet there exists v satisfying $|p_{\alpha,\alpha}(v) - p_{\alpha,1^m}(v)| \geq \frac{1}{8n^{2c}}$. In this case, with very high probability (say $1 - 2^{-n}$), one of the two happens: either the algorithm terminates at the end of phase 1 or for the value v^* found the estimator (i.e., $\pi_{\alpha,\alpha}(v) - \pi_{\alpha,1^m}(v)$) has the same sign as the actual expression (i.e., $p_{\alpha,\alpha}(v) - p_{\alpha,1^m}(v)$). Hence, in this case

$$\begin{aligned}
\mathrm{Prob}(A_2(\alpha, 1^m, E_{G_1(1^n)}(\alpha))=1) - \mathrm{Prob}(A_2(\alpha, 1^m, E_{G_1(1^n)}(1^m))=1) &> -2^{-n} \\
&> -\frac{1}{32n^{2c}}
\end{aligned}$$

Case 3: For every value v, $|p_{\alpha,\alpha}(v) - p_{\alpha,1^m}(v)| < \frac{1}{8n^{2c}}$. In this case, no matter whether the estimator computed in phase 1 is correct or not, algorithm A_2 behaves essentially the same on $E(\alpha)$ and $E(1^m)$. Namely,

$$\mathrm{Prob}(A_2(\alpha, 1^m, E_{G_1(1^n)}(\alpha))=1) - \mathrm{Prob}(A_2(\alpha, 1^m, E_{G_1(1^n)}(1^m))=1) \geq -\frac{1}{8n^{2c}}$$

We now analyze the performance of algorithm A_2. By $\mathrm{acc}_{A_2}(z,t)$ we denote the event "$A_2(z, E_{G_1(1^n)}(t)) = 1$" (i.e., A_2 "accepts" input $z, E(t)$). Using the definition of A_1 and Q_n we bound the difference $\mathrm{Prob}(\mathrm{acc}_{A_2}(Z_n, X_n)) - \mathrm{Prob}(\mathrm{acc}_{A_2}(Z_n, 1^m))$ by

$$\begin{aligned}
\sum_{\alpha \in \{0,1\}^m} \mathrm{Prob}(X_n = \alpha) \quad &\cdot \quad (\mathrm{Prob}(\mathrm{acc}_{A_2}(\alpha 1^m, \alpha)) - \mathrm{Prob}(\mathrm{acc}_{A_2}(\alpha 1^m, 1^m))) \\
&\geq \mathrm{Prob}(\text{case } 1) \cdot \left(\frac{3}{8n^c} - \frac{1}{32n^{2c}} \right) \\
&\quad -\mathrm{Prob}(\text{case } 2) \cdot \frac{1}{32n^{2c}} \\
&\quad -\mathrm{Prob}(\text{case } 3) \cdot \frac{1}{8n^{2c}}
\end{aligned}$$

$$
\begin{aligned}
&> \frac{1}{2n^c} \cdot \frac{3}{8n^c} - \frac{1}{32n^{2n}} - \frac{1}{8n^{2n}} \\
&= \frac{1}{32n^{2n}}
\end{aligned}
$$

Hence, algorithm A_2 distinguishes encryptions of the "halves" (i.e. X_n and 1^m) of the polynomial-time random variable Z_n. This completes the proof of the proposition. □

Proof of Proposition 2

We now show that if (G, E, D) has distinguishable encryptions then it is not semantically secure (not even in the restricted sense mentioned in the hypothesis of the proposition). Namely, we assume that there exists a polynomial-time random variable $\{T_n = X_n Y_n Z_n\}_{n \in \mathbb{N}}$, a probabilistic polynomial-time algorithm A, and a constant $c > 0$ such that for infinitely many $n \in \mathbb{N}$, the probability that $T_n \in B_n^{(c)}$ (where $B_n^{(c)}$ is defined as in Remark 10) is greater than $\frac{1}{n^c}$. Recall that

$$
B_n^{(c)} \stackrel{\text{def}}{=} \left\{ (\alpha, \beta, \gamma) : \left| \text{Prob}\left(A(\gamma, E_{G_1(1^n)}(\alpha)) = 1\right) - \text{Prob}\left(A(\gamma, E_{G_1(1^n)}(\beta)) = 1\right) \right| > \frac{1}{n^c} \right\} \tag{5}
$$

We assume, for simplicity, that $|X_n| = |Z_n| = n$ (recall that in general $|X_n| = |Y_n|$). Define a random variable $\{Q_n\}_{n \in \mathbb{N}}$ such that $Q_n = 0^n Z_n X_n Y_n$ with probability $\frac{1}{2}$ and $Q_n = 1^n Z_n Y_n X_n$ otherwise. Note that the difference between the two cases is in the first block (either 0^n or 1^n) and in the order of the two last blocks (i.e. either $X_n Y_n$ or $Y_n X_n$). Clearly, this random variable is polynomial-time. Define $f : \{0,1\}^{4n} \mapsto \{0,1\}$ such that $f(\sigma^n \delta_1 \delta_2 \delta_3) = \sigma$ (i.e., f returns the first bit of its argument). Define $h : \{0,1\}^{4n} \mapsto \{0,1\}^{3n}$ such that $h(0^n \delta_1 \delta_2 \delta_3) = \delta_1 \delta_2 \delta_3$ and $h(1^n \delta_1 \delta_2 \delta_3) = \delta_1 \delta_3 \delta_2$, where $|\delta_1| = |\delta_2| = |\delta_3| = n$ (i.e. h returns the string resulting by omitting the first block of its argument if this block equals 0^n and omits the first block and switches the order of the last two blocks of its argument if the first block equals 1^n). Thus, the random variable $h(Q_n) = Z_n X_n Y_n$ is independent of the random variable $f(Q_n)$. Note that both f and h are polynomial-time computable and are independent of the algorithm A.

We now construct a probabilistic polynomial-time algorithm A_1 which guesses $f(Q_n)$ from $h(Q_n)$ and $E(Q_n)$. The success probability of algorithm A_1 will be shown to be non-negligibly greater than $1/2$ (hence contradicting semantic security even in the special case mentioned in the hypothesis of the proposition).

Description of algorithm A_1

Let δ be either of the form $0^n \gamma \alpha \beta$ or of the form $1^n \gamma \beta \alpha$. On input $h(\delta) = \gamma \alpha \beta$ and $E_e(\delta)$, algorithm A_1 works in two phases. Loosely speaking, in the first phase the algorithm tries to estimate the difference between $\text{Prob}(A(\gamma, E_{G_1(1^n)}(\alpha)) = 1)$ and $\text{Prob}(A(\gamma, E_{G_1(1^n)}(\beta)) = 1)$, and record its sign. In the second phase the algorithm "guesses" σ by computing $a \stackrel{\text{def}}{=} A(E_e(\delta_3), \gamma\alpha\beta, 1^n)$, where $\delta = \sigma^n \delta_1 \delta_2 \delta_3$, and outputs a if the sign of the difference (computed in phase 1) is positive and $\overline{a}$ otherwise. As Conventions 2 and 3 are used, $E_e(\delta_3)$ can be obtained from $E_e(\delta)$. Details follow.

Phase 1: Ignoring $E_e(\delta)$, algorithm A_1 tries to estimate the difference

$$
\Delta(\gamma, \alpha, \beta) \stackrel{\text{def}}{=} \text{Prob}\left(A(\gamma, E_{G_1(1^n)}(\alpha)) = 1\right) - \text{Prob}\left(A(\gamma\ E_{G_1(1^n)}(\beta)) = 1\right) \tag{6}
$$

The estimate is computed by sampling polynomially many times so that with very high probability (say $1 - 2^{-n}$) the estimate does not differ from the real value by more than $\frac{1}{8n^{2c}}$.

Phase 2: Without loss of generality, assume that the estimator of Eq. 6 (computed in phase 1) is positive. Algorithm A_1 takes the fourth block of $E_e(\delta)$ and feeds it together with γ to algorithm A. Note that the fourth block of $E_e(1^n\gamma\beta\alpha)$ equals $E_e(\alpha)$, while the fourth block of $E_e(0^n\gamma\alpha\beta)$ equals $E_e(\beta)$. Algorithm A_1 outputs 1 if A outputs 1, and outputs 0 otherwise.

Evaluation of algorithm A_1

We now analyze the performance of algorithm A_1. By $\mathrm{succ}_{A_1}(Q_n)$ we denote the event "$A_1(E_{G_1(1^n)}(Q_n), h(Q_n), 1^n) = f(Q_n)$" (i.e., A_1 successfully guesses the value of $f(Q_n)$ given $E(Q_n)$ and $h(Q_n)$). By the definition of A_1 and Q_n it follows that

$$\begin{aligned}
\mathrm{Prob}\left(\mathrm{succ}_{A_1}(Q_n)|h(Q_n)=\gamma\alpha\beta\right) &= \mathrm{Prob}(f(Q_n)=1)\cdot\mathrm{Prob}\left(\mathrm{succ}_{A_1}(Q_n)|h(Q_n)=\gamma\alpha\beta\wedge f(Q_n)=1\right)\\
&+ \mathrm{Prob}(f(Q_n)=0)\cdot\mathrm{Prob}\left(\mathrm{succ}_{A_1}(Q_n)|h(Q_n)=\gamma\alpha\beta\wedge f(Q_n)=0\right)\\
&= \frac{1}{2}\cdot\mathrm{Prob}\left(A(\gamma, E_{G_1(1^n)}(\alpha))=1\right)\\
&+ \frac{1}{2}\cdot\mathrm{Prob}\left(A(\gamma, E_{G_1(1^n)}(\beta))=0\right)\\
&= \frac{1}{2}+\frac{\Delta(\gamma,\alpha,\beta)}{2}
\end{aligned}$$

In analyzing the performance of algorithm A_1, we consider three cases. We assume, without loss of generality, that $\Delta(\gamma,\alpha,\beta)\geq 0$.

Case 1: $(\alpha,\beta,\gamma)\in B_n^{(c)}$ (i.e., $\Delta(\gamma,\alpha,\beta)\geq\frac{1}{n^c}$). In this case, with very high probability (say $1-2^{-n}$), the estimator has the correct sign. Hence, in this case

$$\begin{aligned}
\mathrm{Prob}\left(\mathrm{succ}_{A_1}(Q_n)|h(Q_n)=\gamma\alpha\beta\right) &\geq \mathrm{Prob}(\text{estimator is correct})\cdot\left(\frac{1}{2}+\frac{1}{2n^c}\right)\\
&> \frac{1}{2}+\frac{1}{2n^c}-\frac{1}{2^n}\\
&> \frac{1}{2}+\frac{1}{4n^c}
\end{aligned}$$

Case 2: $\frac{1}{4n^{2c}}\leq\Delta(\gamma,\alpha,\beta)<\frac{1}{n^c}$. Also in this case, with very high probability (say $1-2^{-n}$), the estimator has the correct sign. Hence, in this case

$$\begin{aligned}
\mathrm{Prob}\left(\mathrm{succ}_{A_1}(Q_n)|h(Q_n)=\gamma\alpha\beta\right) &\geq \mathrm{Prob}(\text{estimator is correct})\cdot\left(\frac{1}{2}+\frac{1}{8n^{2c}}\right)\\
&> \frac{1}{2}
\end{aligned}$$

Case 3: $\Delta(\gamma,\alpha,\beta)<\frac{1}{4n^{2c}}$. In this case, no matter whether the estimator has the correct sign or not, the outcome of A_1 is very close to an unbiased guess. Namely

$$\mathrm{Prob}\left(\mathrm{succ}_{A_1}(Q_n)|h(Q_n)=\gamma\alpha\beta\right)\geq\frac{1}{2}-\frac{1}{8n^{2c}}$$

Combining the three cases, we get

$$
\begin{aligned}
\mathrm{Prob}\left(\mathrm{succ}_{A_1}(Q_n)\right) &= \mathrm{Prob}(\text{case } 1) \cdot \mathrm{Prob}\left(\mathrm{succ}_{A_1} | \text{case } 1\right) \\
&+ \mathrm{Prob}(\text{case } 2) \cdot \mathrm{Prob}\left(\mathrm{succ}_{A_1} | \text{case } 2\right) \\
&+ \mathrm{Prob}(\text{case } 3) \cdot \mathrm{Prob}\left(\mathrm{succ}_{A_1} | \text{case } 3\right) \\
&\geq \frac{1}{n^c} \cdot \left(\frac{1}{2} + \frac{1}{4n^c}\right) + \left(1 - \frac{1}{n^c}\right) \cdot \left(\frac{1}{2} - \frac{1}{8n^{2c}}\right) \\
&> \frac{1}{2} + \frac{1}{8n^{2c}}
\end{aligned}
$$

Contradiction follows, and this completes the proof of the proposition. □

Remark 14 : The a-priori information represented by the function h, Convention 2 (regarding "block encryption"), and Convention 3 (regarding "length uniformity"), are essential to the proof presented above.

3.4 Constructions of Secure Encryption Schemes

In this subsection we merely point out that the known methods for constructing secure private-key and public-key encryption schemes remain valid in the uniform setting. In particular,

- Using any (uniform strong) one-way function one can construct pseudorandom generators [13, 21, 20], which in turn can be used to produce (uniformly) secure private-key encryption schemes (cf. [4]).
- Using any (uniform strong) trapdoor one-way permutation one can construct (uniformly) secure public-key encryption schemes [18, 29, 5].

3.5 Commitment Schemes

Commitment schemes are a basic ingredient in many cryptographic protocols. The security of these schemes, which is analogous to the security of encryption schemes, is usually defined in terms of non-uniform complexity. Here we provide a uniform treatment of their security. This treatment will be used, in the next section, in the constructions of *uniform* zero-knowledge proof systems.

Loosely speaking, secure commitment schemes are two-party protocols, proceeding in two *phases* by which one party, called *commiter*, commits itself to a value. After the *first phase*, the commiter is committed to a value which is yet unknown (in a strong sense) to the other party. This value (and it only) can be revealed by the commiter in the *second phase*. Without loss of generality, the second phase may consists of the commiter sending its initial input (the committed value) and the outcome of the coins it (i.e. the commiter) used in the first phase. This leads to the following definition, which uses conventions presented in subsection 2.2.

Definition 6 *: A* (uniformly) secure bit commitment scheme *is a pair of probabilistic polynomial-time interactive machines, denoted (C, R) (for* commiter *and* receiver*), satisfying:*

1. Input Specification*: The common input is an integer n presented in unary (serving as the security parameter). The private input to the commiter is a bit v.*

2. Secrecy*: For every probabilistic polynomial-time machine R^* interacting with C, the random variables $R^*_{C(0,1^n)}(1^n)$ and $R^*_{C(1,1^n)}(1^n)$ are polynomially-indistinguishable. Namely, the receiver (even when deviating arbitrarily from the protocol) cannot distinguish a commitment to* 0 *from a commitment to* 1.

3. Nonambiguousity*: We call (a "receiver's history") $\gamma = (r, \overline{m})$ a* possible σ-commitment *if there exists a string s such that $\overline{m}$ describes the messages received by R when it uses local coins r and interacts with machine C which uses local coins s and input $(\sigma, 1^n)$. We call γ* ambiguous *if it is both a possible* 0*-commitment and a possible* 1*-commitment. It is required that for all but a negligible fraction of the $r \in \{0,1\}^{\mathrm{poly}(n)}$ there is no $\overline{m}$ such that $(r, \overline{m})$ is ambiguous.*

Remark 15 : The formulation of the secrecy requirement in the above definition is analogous to Definition 5 ("indistinguishability of encryptions")[6]. An equivalent formulation analogous to Definition 4 ("semantic security") can be presented, but is less useful in typical applications of commitment schemes. The nonambiguousity requirement can be relaxed by requiring that it is infeasible, when interacting with R (which uses random coins r), to find $\overline{m}$ such that $(r, \overline{m})$ is ambiguous. We choose to present a more conservative (and simpler) formulation since it can be achieved as well using the same complexity assumptions (cf. [24]).

Remark 16 : The secrecy requirement refers explicitly to the situation at the end of the first phase. On the other hand, the nonambiguousity requirement implicitly assumes (without loss of generality) that the second phase takes the following form: (1) the commiter C sends its initial private input, v, and the random coins, s, it has used in the first phase; (2) the receiver R verifies that v and s (together with r the coin used by R in the first phase) indeed yield the messages R has received in the first phase. Verification is done in polynomial-time (by running the programs C and R).

Remark 17 : In the above definition, secrecy is with respect to probabilistic polynomial-time machines, while nonambiguousity is absolute (i.e. even with respect to non-uniform machines C^* with no time bounds). A dual definition, requiring information theoretic secrecy and infeasibility of creating ambiguousities, is presented in [8, 10, 6].

The following results establish sufficient conditions (which are also necessary) for the existence of secure bit commitments.

Theorem (Naor [24]): Assuming the existence of (uniformly) pseudorandom generators, there exist (uniformly) secure bit commitment schemes.

Theorem (Impagliazzo, Levin and Luby [21] and Hastad [20]): Pseudorandom generators exist if and only if one-way functions exist.

[6] Here, there is no need to provide R^* with polynomial-time generated auxiliary input, since R^* can generate it by himself. In the context of Definition 5, the auxiliary input Z_n (e.g. $Z_n = X_n Y_n$) may be dependent on $X_n Y_n$ in a way which does not provide an efficient algorithm for generating Z_n given, for example, that $E(X_n) = \beta$ (though there is an efficient sampling algorithm for $X_n Z_n$). Here X_n is identically 1, so Z_n is statistically independent of X_n and the problem does not arise.

4 Uniformly Zero-Knowledge Proof Systems

In this section we provide a uniform complexity treatment of zero-knowledge. Using the conventions presented in subsection 2.2, we define uniformly zero-knowledge proof systems. Intuitively, these are interactive proof for which it is infeasible to find an instance on which the verifier gains knowledge from the interaction with the prover. More generally, we provide a definition of protocols which are (uniformly) zero-knowledge on a restricted set of inputs (i.e., in the case of interactive proofs the common input is restricted to the language). Next, we show that uniformly zero-knowledge protocols are closed under sequential composition. Finally, we present uniformly zero-knowledge proof systems for every language in $\mathbb{NP}$ (and even $\mathbb{MA}$).

4.1 Definition of Uniformly Zero-Knowledge Protocols

In the following definition we decouple the notion of "zero-knowledge" from the (traditional) setting of language recognition.

Definition 7 (zero-knowledge interactive machine): *Let $S \subseteq \{0,1\}^* \times \{0,1\}^*$. A probabilistic polynomial-time interactive machine A is* uniformly zero-knowledge *over S if, for every probabilistic polynomial-time interactive machine B^*, there exists a probabilistic polynomial-time machine M^* (called the* simulator*), such that for every polynomial-time random variable $\{T_n = X_n Y_n Z_n\}_{n\in\mathbb{N}}$ with $X_n Y_n$ ranging over S, the random variables $B^*_{A(X_n,Y_n)}(X_n, Z_n)$ and $M^*(X_n, Z_n)$ are polynomially indistinguishable.*

Incorporating the definition of polynomially indistinguishable random variable, the above requires that (for every machine B^* there exists a machine M^* such that) for every probabilistic polynomial-time algorithm D (for all $c > 0$ and all sufficiently large $n \in \mathbb{N}$)

$$\left|\text{Prob}\left(D(B^*_{A(X_n,Y_n)}(X_n, Z_n))=1\right) - \text{Prob}\left(D(M^*(X_n, Z_n))=1\right)\right| < \frac{1}{n^c}$$

In the above definition, X_n represents the common input, Y_n is private input to A whereas Z_n is private input to B. In the context of interactive proofs, A is the prover (which may use auxiliary input Y_n, e.g. an NP-witness for X_n being in an NP-language), and B is the verifier (which has auxiliary input Z_n). Note that Z_n may depend on X_n, and that it may not be feasible to generate the random variable "Z_n conditioned on $X_n = x$", even for a randomly chosen x. Hence, Definition 7 guarantees that no matter what B a-priori "knows", he is not going to "gain knowledge" from the interaction with A (see parenthetical subsection below).

Remark 18 : An equivalent formulation of Definition 7 follows: for every probabilistic polynomial-time interactive machine B^*, there exists a probabilistic polynomial-time machine M^*, such that for every polynomial-time random variable $\{T_n = X_n Y_n Z_n\}_{n\in\mathbb{N}}$, for every probabilistic polynomial-time algorithm D, for every $c > 0$ and all sufficiently large $n \in \mathbb{N}$, $\text{Prob}(T_n \in B_n^{(c)}) < \frac{1}{n^c}$, where $(x, y, z) \in B_n^{(c)}$ if $(x, y) \in S$ and

$$\left|\text{Prob}\left(D(B^*_{A(x,y)}(x, z))=1\right) - \text{Prob}\left(D(M^*(x, z))=1\right)\right| > \frac{1}{n^c} \tag{7}$$

Intuitively, it is infeasible to find a triple $(x, y, z) \in S \times \{0,1\}^*$ satisfying Eq. 7. (The equivalence to Definition 7 is proven in Appendix B.)

Definition 8 (zero-knowledge protocol): *A probabilistic polynomial-time protocol* $\Pi = (A, B)$ *is* uniformly zero-knowledge *over* $T \subseteq \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^*$ *if* A *is uniformly zero-knowledge over* $T_{\{1,2\}}$ *and* B *is uniformly zero-knowledge over* $T_{\{1,3\}}$, *where* $T_{\{i,j\}}$ *is the projection of the set of triples* T *on coordinates* i *and* j *(i.e., for* $\{1,2,3\} = \{i,j,k\}$, *we have* $(a_i, a_j) \in T_{\{i,j\}}$ *if there exists* $a_k \in \{0,1\}^*$ *s.t.* $(a_1, a_2, a_3) \in T$*).*

Parenthetical subsection: classification of zero-knowledge formulations

As many variants of the notion of zero-knowledge have been proposed in the literature (cf. [19, 25, 28]), a classification attempt is indeed called for. We distinguish three parameters:

1. Existence or absence of *a-priori information* for the potential knowledge-receiver (i.e., machine B, or in the context of interactive proof, the verifier). Such information is captured by an auxiliary input given to the potential knowledge-receiver (in addition to the common input). In Definition 7, this auxiliary input is denoted by Z_n.

2. *Uniform or non-uniform* formalization. Definition 7 is the first (totally) uniform formalization of zero-knowledge. The original definition of zero-knowledge (as appearing in early versions of [19]) is "semi-uniform": it uses uniform machines (esp., in the roles of B^* and D) but quantifies over all common inputs (i.e., x). In the non-uniform formalization all machines are modelled by non-uniform polynomial-size circuits, and the auxiliary inputs are arbitrary strings of polynomial length (i.e., $\{Z_n\}_{n \in \mathbb{N}}$ may be an arbitrary sequence of distributions each concentrated on a single string).

3. *Universal (or Black-Box) simulator.* Definition 7 only requires that *for every* B^* there *exists* a machine M^* simulating (A, B^*) conversations. A stronger requirement, met in all known cases, is that there exists a *universal* machine simulating (A, B^*) conversations by using B^* as a *black-box* [25, 17, 12].

Following are some remarks concerning the relations among these parameters:

- Consider the definitions in which a-priori information is given to the potential "knowledge-receiver" in form of an auxiliary input. Both in the uniform and non-uniform formalizations, the order of quantifiers is of the form "for every B^* (interacting with A) there is a simulator M^* such that for every $\{T_n\}_{n \in \mathbb{N}}$ (the output of the simulator is polynomially indistinguishable from the output of B^* after interacting with A)". Namely, the simulator must perform well for any admissible choice of the random variable $\{T_n\}_{n \in \mathbb{N}}$ (and in particular for every admissible choice of auxiliary input Z_n). Thus, the effect of a-priori information (i.e. auxiliary input) is *not* captured by the non-uniform formalization. Namely, saying that "for every non-uniform polynomial-size circuit B^* there exists a non-uniform polynomial-size circuit M^* such that for every x, y, z we have $M^*(x,z)$ indistinguishable from $B^*_{A(x,y)}(x,z)$" is not the same as saying "for every non-uniform polynomial-size circuit B^* there exists a non-uniform polynomial-size circuit M^* such that for every x we have $M^*(x)$ indistinguishable from $B^*_{A(x)}(x)$". Intuitively, having auxiliary input external to the machine or having it wired into the machine, changes the power of the machine. Hence, using a non-uniform formalization without introducing an auxiliary input to the machines (and in particular to B^*) is not satisfactory.[7]

[7] In fact, even when allowing auxiliary-inputs, the above non-uniform formalization is not satisfactory. Unfortunately, this point (justified below) was not made in the original version where the focus was to explain that the effect of auxiliary-input is *not* captured by the non-uniform formalization. As to demonstrating the inadequacy of the

- Formulating zero-knowledge protocols in a way which allows a-priori information to the parties is crucial to all known cryptographic applications of zero-knowledge (cf. [25, 17]). This point is demonstrated by considering the sequential composition of zero-knowledge protocols. As we shall see in the next subsection, the formulation of zero-knowledge with respect to auxiliary input allows to prove a sequential composition lemma (even for *non-constant* number of repetitions). It is *not* known how to prove an analogous result for the non-uniform formalization of zero-knowledge without auxiliary inputs (and no universal simulator). Hence, it seems that the first two parameters (i.e., items (1) and (2) above) are "orthogonal".

- The existence of a universal simulator implies the robustness against auxiliary input (in both uniform and non-uniform formalizations). Namely, if A is uniformly (resp., non-uniformly) zero-knowledge in the stronger sense described in item (3) above, then A is uniformly (resp., non-uniformly) zero-knowledge with respect to auxiliary input [25, 17].

4.2 Sequential Composition of Uniformly Zero-Knowledge Protocols

A desired property of zero-knowledge protocols is that their sequential composition is zero-knowledge as well. It has been showed that non-uniform zero-knowledge with auxiliary input is preserved under sequential composition [25, 17]. On the other hand, the "semi-uniform" formalization[8] with no auxiliary input is *not* preserved under sequential composition [12]. Here we demonstrate that the auxiliary input is the only essential ingredient in proving preservation of zero-knowledge under sequential composition. Namely

Lemma 1 (Sequential Composition Lemma): *Let A be a probabilistic polynomial-time interactive machine which is uniformly zero-knowledge (over S), and $Q(\cdot)$ be a polynomial. Let A_Q be an interactive machine that on input (x, y) proceeds in $Q(|x|)$ phases, each of them consisting of running A on input (x, y). Then A_Q is uniformly zero-knowledge (over S).*

At first glance, the above formalization seems as a special case of the general case in which A is executed on different inputs at each phase. This impression is changed once one realizes that the parties may act "differently" (and/or on different parts of their inputs) in the various phases. These changes can be monitored by the other party (e.g., the prescribed B).

Proof (following ideas of [25, 17]): Let B^* be an *arbitrary* probabilistic polynomial-time interactive machine. There exists a probabilistic polynomial-time B^{**} (a minor modification of B^*) so that the interaction of $A_Q(x, y)$ with $B^*(x, z)$ (on common input x) can be partitioned into $Q(|x|)$ phases so that at the i^{th} phase $A(x, y)$ interacts with $B^{**}(x, z_{i-1})$, where the string z_0 equals z and the string z_i is the output of $B^{**}(x, z_{i-1})$. Since B^{**} is a probabilistic polynomial-time machine interacting with A on probabilistic polynomial-time generated inputs and since A is uniformly zero-knowledge, these conversations can be generated by a simulator, denoted M^{**}. For sake of simplicity, assume that B^{**} as well as M^{**} also output the common input x.

We construct a probabilistic polynomial-time machine M^* (supposed to simulate the interaction of A_Q with B^*) as follows. On input (x, z), machine M^* sets $\overline{z}_0 = z$ and proceeds in $Q(|x|)$ phases.

non-uniform formulation, consider the protocol in which A sends the result of applying a very hard predicate (e.g., an uncomputible one) to the length of the input. Intuitively, such a protocol is not zero-knowledge since B (viewed as a real party) gets something he cannot compute by himself. However, this is not captured by a non-uniform view of the party B. Specifically, a small non-uniform can certainly simulate this exchange (by incorporating the relevant bit). Formally speaking, the above demonstrate that the non-uniform formulation of zero-knowledge does not cover the original "semi-uniform" formulation of [19]. *Text added in Aug. 1998.*

[8] Namely, uniform B^* and M^* but quantifying over all possible inputs x

In the $i^{\rm th}$ phase, machine M^* computes $\overline{z}_i = M^{**}(x, \overline{z}_{i-1})$. After $Q(|x|)$ phases were completed, machine M^* stops outputting $\overline{z}_{Q(|x|)}$.

We now show that the simulator M^*, constructed above, indeed produces output which is polynomially indistinguishable from the output of B^* (after interacting with A_Q). Let $\{T_n\}_{n\in\mathbb{N}}$ be a polynomial-time random variable with X_nY_n ranging over S, and let $m=|X_n|$. To show that $B^*_{A_Q(X_n,Y_n)}(X_n,Z_n)$ and $M^*(X_n,Z_n)$ are indistinguishable we use a "hybrid" argument. Consider the following hybrids:

$$H_i(X_n,Y_n,Z_n) \stackrel{\rm def}{=} M^{**}_{Q(m)-i}(X_n, Z_n^{(i)}) \tag{8}$$

where

$$Z_n^{(i)} \stackrel{\rm def}{=} B^{**}_{A(X_n,Y_n)}(X_n, Z_n^{(i-1)}) \quad \text{and} \quad Z_n^{(0)} \stackrel{\rm def}{=} (X_n, Z_n) \tag{9}$$

$$M^{**}_k(x,z) \stackrel{\rm def}{=} M^{**}(x, M^{**}_{k-1}(x,z)) \quad \text{and} \quad M^{**}_0(x,z) \stackrel{\rm def}{=} (x,z) \tag{10}$$

Namely, H_k is the distribution obtained by letting B^{**} interact with A for k phases and then iterating M^{**} on the output for the remaining $Q(m)-k$ phases. Clearly,

$$H_{Q(m)}(X_n,Y_n,Z_n) = B^*_{A_Q(X_n,Y_n)}(X_n,Z_n) \tag{11}$$

whereas

$$H_0(X_n,Y_n,Z_n) = M^*(X_n,Z_n) \tag{12}$$

All that is required to complete the proof is to show that every two adjacent hybrids are polynomially indistinguishable (as this would imply that the extreme hybrids, $H_{Q(m)}$ and H_0, are indistinguishable too). To show that $H_i(X_n,Y_n,Z_n)$ and $H_{i-1}(X_n,Y_n,Z_n)$ are computationally indistinguishable, we note that

$$H_i(X_n,Y_n,Z_n) = M^{**}_{Q(m)-i}(B^{**}_{A(X_n,Y_n)}(Z_n^{(i-1)})) \tag{13}$$

whereas

$$H_{i-1}(X_n,Y_n,Z_n) = M^{**}_{Q(m)-i}(M^{**}(Z_n^{(i-1)})) \tag{14}$$

Hence, if H_i and H_{i-1} are polynomially distinguishable on the random variable $\{T_n\}_{n\in\mathbb{N}}$ then B^{**}_A and M^{**} are polynomially distinguishable on the random variable $\{T_n^{(i-1)} = X_nY_nZ_n^{(i-1)}\}_{n\in\mathbb{N}}$ (incorporate $M^{**}_{Q(m)-i}$ into the distinguisher). Using the definition of $Z_n^{(i)}$, and the fact that $\{T_n\}_{n\in\mathbb{N}}$ is polynomial-time[9], we conclude that $T_n^{(i-1)}$ is polynomial-time. Contradiction to the hypothesis that M^{**} simulates B^{**}_A follows. Hence, the hybrids are indeed polynomially indistinguishable and the lemma follows. □

4.3 Uniformly Zero-Knowledge Proof Systems for NP and MA

Interactive proof systems were introduced as protocols (games) between infinitely powerful provers and probabilistic polynomial-time verifiers [19, 1]. In practical applications it is often the case that the prover is also a probabilistic polynomial-time interactive machine and that its "computational advantage" over the verifier is in having a-priori knowledge. Clearly, every language in $\mathbf{NP}$ has an interactive proof in which the prover is a probabilistic polynomial-time machine which gets an NP-witness as an auxiliary input. Goldreich, Micali and Wigderson showed that the existence of non-uniformly secure bit commitment schemes implies that every language in $\mathbf{NP}$ has a

[9]We rely on the fact that $\{X_nY_nZ_n\}_{n\in\mathbb{N}}$, not merely $\{Z_n\}_{n\in\mathbb{N}}$, is polynomial-time

(non-uniformly) zero-knowledge interactive proof in which the prover is a probabilistic polynomial-time machine which gets an NP-witness as an auxiliary input [15]. The fact that the prover is probabilistic polynomial-time is crucial for the cryptographic applicability of these zero-knowledge proofs. As our intention is to deal only with settings in which all objects are generated via probabilistic polynomial-time means, we restrict our attention to interactive proofs with probabilistic polynomial-time provers (see Definition 9 below).

Remark: When we talk of restricting the prover to probabilistic polynomial-time we mean only that the (prescribed) prover can prove valid theorems efficiently (i.e. we refer only to the completeness condition). The soundness of the proof system does not rely on an intractability assumption concerning the party which provides the evidence. This notion should not be confused with the notion of an *interactive argument* (introduced in [8, 10, 6] see also [9]) where the soundness condition relies on the assumption that the prover is probabilistic polynomial-time. Though zero-knowledge arguments suffice for the practical purposes we consider, we stick to the more conservative formulation of interactive proof systems (as introduced in [19]), a formulation which we are able to meet as well.

Definition 9 (polynomial-time interactive proofs): *A probabilistic polynomial-time protocol* (P, V) *is called an* interactive proof system for a language $L \subseteq \{0,1\}^*$ *if for all* $c > 0$ *and all sufficiently large* x *the following two conditions hold*

- Completeness: *If* $x \in L$ *then there exists* y *such that* $(\forall z)$

$$\text{Prob}\left(V_{P(x,y)}(x,z) = 1\right) > 1 - \frac{1}{|x|^c}$$

- Soundness: *If* $x \notin L$ *then for all* y $(\forall z)$ *and for all interactive machines* P^* *(not necessarily probabilistic polynomial-time ones)*

$$\text{Prob}\left(V_{P^*(x,y)}(x,z) = 1\right) < \frac{1}{|x|^c}$$

A language L *having such a protocol* (P, V) *is said to have a* probabilistic polynomial-time interactive proof system*.*

By Convention 5, both prover and verifier (i.e., machines P and V) have running time polynomial in the common input x. Hence, without loss of generality, y mentioned in the completeness condition satisfies $|y| = \text{poly}(|x|)$.

Remark 19 : Probabilistic polynomial-time interactive proofs exist exactly for the languages in the class **MA** defined in [1][10]. The class **MA** seem a conservative extension of **NP** (in contrast to the class **IP** defined in [19] and recently shown to equal **PSPACE** [26]). However, we are not interested here in the complexity theoretic aspects of interactive proofs but rather in their utility in practice.

[10] The class **MA** consists of all languages L such that there is an interactive proof for L in which the prover sends the first and only message. The verifier in this proof system is allowed to toss coins after receiving the prover's message, and is allowed to make errors with probability bounded away from $\frac{1}{2}$. The probabilistic relaxation of the verifier's decision is the only additional power that this proof system has over the NP-proof system.

Remark 20 : Definition 9 has some "non-uniform" flavour; namely, the quantification over all x's rather than requiring that it is infeasible to find x's for which either the completeness (resp., the soundness) condition is violated. Nevertheless, in view of Theorem 2, we see no essential reason to present more liberal definitions here.

Definition 10 (zero-knowledge interactive proof): *Let* $\Pi = (P, V)$ *be a polynomial-time interactive proof system for* L*. The protocol* Π *is called a* uniformly zero-knowledge proof *for* L *if there exists* $R_\Pi \subseteq \{0,1\}^* \times \{0,1\}^*$ *such that*

1. *For every* $c > 0$ *and all sufficiently large* $x \in L$ *there exists a* y *such that* $(x, y) \in R_\Pi$ *and (for every* z*),* $\text{Prob}(V_{P(x,y)}(x, z) = 1) > 1 - \frac{1}{|x|^c}$
2. Π *is (uniformly) zero-knowledge over* $R_\Pi \times \{0,1\}^*$.

Assuming the existence of one-way functions, which in turn imply the existence of uniformly secure bit commitment (cf. [21, 20, 24]), we derive the main result of this subsection: every language having a probabilistic polynomial-time interactive proof system, also has one which is uniformly zero-knowledge.

Theorem 2 *Assuming the existence of uniformly secure bit commitment schemes, every language in* $\mathbf{MA}$ *(*$\supseteq \mathbf{NP}$*) has a uniformly zero-knowledge (probabilistic polynomial-time) proof system.*

We prove Theorem 2 in two steps. We start by proving the claim of Theorem 2 only for languages in $\mathbf{NP}$ (see Proposition 3 below). This is done by showing that the (probabilistic polynomial-time) interactive proof for Graph Colorability (i.e. $G3C$) presented in [15] is uniformly zero-knowledge[11]. The proof, which carefully adapts the ideas of [15] to the uniform setting, uses a uniform intractability assumption (instead of the non-uniform assumption used in [15]). The proof of Theorem 2 is completed by using a zero-knowledge proof system for $\mathbf{NP}$ to prove membership in languages in $\mathbf{MA}$ (see Proposition 4 below).

The interactive proof for Graph Colorability presented in [15] consists of polynomially many sequential applications of the same atomic protocol. Each execution of the atomic protocol adds "stochastic confidence" towards believing the claim (this statement is given precise meaning in Definition 11 below). Demonstrating that the atomic protocol is uniformly zero-knowledge, and using the Sequential Composition Lemma we complete the first step of the proof of Theorem 2.

Definition 11 (weak interactive proofs): *A probabilistic polynomial-time protocol* (P, V) *is called a* weak interactive proof system *for a language* $L \subseteq \{0,1\}^*$ *if there exist two polynomial-time computable functions* $p, q : \mathbf{N} \mapsto \mathbf{R}$ *such that the following three conditions hold*

- Weak completeness: *If* $x \in L$ *then there exists* y *such that (for all* z*)*

$$\text{Prob}(V_{P(x,y)}(x, z) = 1) \geq p(|x|)$$

- Weak soundness: *If* $x \notin L$ *then for all* y *(*$\forall z$*) and for all interactive machines* P^*

$$\text{Prob}(V_{P^*(x,y)}(x, z) = 1) \leq q(|x|)$$

[11] We do not know whether the [15] interactive proof can be proven (non-uniformly) zero-knowledge assuming only a uniform intractability assumption. This holds also for the "semi-uniform" formulation of zero-knowledge in which non-uniformity is present only in the universal quantification over all $x \in G3C$. The problem is that the commitment scheme used may not be non-uniformly secure and might be broken using the input x as an auxiliary (non-uniform) input.

- Gap: *There exists $c > 0$ such that all sufficiently large n, we have $p(n) > q(n) + \frac{1}{n^c}$.*

Proposition 3 *: If there exists a uniformly secure bit commitment scheme then there exists a uniformly zero-knowledge weak interactive proof for Graph 3-Colorability (G3C).*

Proposition 4 *: Assume that there exists a uniformly secure bit commitment scheme and that every language in* $\mathbf{NP}$ *has a uniformly zero-knowledge proof system. Then every language in* $\mathbf{MA}$ *has a uniformly zero-knowledge proof system.*

We conclude this section with the proofs of Proposition 3 and 4. As stated above, the proof of Proposition 3, carefully adapts the ideas of [15] to the uniform setting. In addition we believe that, in its details, the proof appearing here is more elegant than the one appearing in [15]. The proof of Proposition 4, carefully adapts the ideas of [22] to the uniform setting. It is worthwhile mentioning that the alternative ideas of [2] which build on [14] fail here.

Proof of Proposition 3

The proof follows ideas of [15], but there are some modifications in the constructions which are due to the fact that we cannot use non-uniformity here. The protocol we prove to be a uniformly zero-knowledge weak interactive for Graph 3-Colorability is exactly the basic step in the zero-knowledge proof system of [15]. For sake of self-containment, we repeat the protocol here.

Protocol 1
Inputs: The common input is a 3-colorable graph, denoted $G(V, E)$. The auxiliary input to the prescribed prover is a legal 3-colouring of the vertices, denote $\phi: V \mapsto \{1, 2, 3\}$.
Conventions: Let $n = |V|$, $m = |E|$, and Sym_3 be the symmetric group over $\{1, 2, 3\}$. For simplicity, let $V = \{1, 2, ..., n\}$. By $a \in_R A$ we mean that a is chosen uniformly in the set A. Let $C(v)$ be a (probabilistic) commitment to value v and $C_r(v)$ be the commitment to v when the committer uses r as its coin tosses. For simplicity, we assume that the commitment takes place in one round of communication (the argument can easily be extended to the general case).

P1) The prover chooses a random permutation of the 3-colouring, colours the graph using this 3-colouring, and commits to these colours. More specifically, the prover chooses a permutation $\pi \in_R \mathrm{Sym}_3$, selects uniformly $r_1, ..., r_n \in \{0, 1\}^n$, computes $c_i = C_{r_i}(\pi(\phi(i)))$, and sends $c_1, ..., c_n$ to the verifier.

V1) The verifier chooses at random an edge $e \in E$ and sends it to the prover.

P2) If $e = (u, v) \in E$, then the prover sends $(\pi(\phi(u)), r_u)$ and $(\pi(\phi(v)), r_v)$ to the verifier. If $e \notin E$ the prover selects at random $(u, v) \in E$ and acts as above.

V2) Receiving (a, s) and (b, t), the verifier checks whether the $a \neq b$, $a, b \in \{1, 2, 3\}$, $c_u = C_s(a)$ and $c_v = C_t(b)$. The verifier outputs 1 iff all conditions are satisfied.

It is easy to see that Protocol 1 constitutes a weak interactive proof system (with $p(m) = 1$ and $q(m) = 1 - \frac{1}{m}$) for Graph 3-Colorability. To prove that Protocol 1 is uniformly zero-knowledge, we first present for every probabilistic polynomial-time interactive machine V^*, a probabilistic polynomial-time simulator M^*. We then prove that for all polynomial-time random variables $\{X_n, Y_n, Z_n\}_{n \in \mathbb{N}}$ (with X_n, Y_n ranging over pairs (G, ϕ) where G is a simple graph on n vertices and ϕ is a 3-colouring of its vertices) the random variables $V^*_{P(X_n, Y_n)}(X_n, Z_n)$ and $M^*(X_n, Z_n)$

are polynomially indistinguishable. For sake of simplicity we assume, without loss of generality, that V^*, (after interacting with P) outputs the contents of its input tape, random tape and both communication tapes.

Following is a detailed description of M^*, which uses V^* as a subroutine. On input a graph G and auxiliary information z, machine M^* starts by choosing a random tape $r \in_R \{0,1\}^q$ for V^*, where $q = \text{poly}(|G|)$ is a bound on the running time of the interactive machine V^* on input G. Machine M_{V^*} places r on its record tape and repeats the following two steps no more than m^2 times.

S1) Machine M^* picks an edge $(u,v) \in_R E$ and a pair of integers $(a,b) \in_R \{(i,j): i \neq j \in \{1,2,3\}\}$. Machine M^* chooses random r_i's ($r_i \in_R \{0,1\}^n$) and computes $c_i = C_{r_i}(\gamma_i)$ for each $i \in V$, where $\gamma_i = 0$ for $i \in V - \{u,v\}$, $\gamma_u = a$ and $\gamma_v = b$.

S2) Machine M^* sets $e = V^*(G, z, r; (c_1, ..., c_n))$. (Namely, e is the message that V^* sends on input graph G, auxiliary input z and random tape r, after receiving message $(c_1, ..., c_n)$.) Without loss of generality, $e \in E$ (otherwise the simulator selects $e \in_R E$). We consider two cases:
Case 1: $e = (u,v)$ ("lucky for M^*").
Machine M^* stops outputting $(G, z, r, (c_1, ..., c_n), (u,v), (a, r_u, b, r_v))$.
Case 2: $e \neq (u,v)$ ("unlucky for M^*").
Machine M^* is going to repeat steps (S1) and (S2).

If all m^2 repetitions were completed (without M^* halting) the machine M^* halts outputting $\perp$.

We now have to prove the validity of the construction. Clearly the simulator M^* is probabilistic polynomial-time. It is left to prove that the output distribution produced by M^* on polynomial-time generated inputs is polynomially-indistinguishable from the distribution over V^*'s tapes when interacting with P on the same input. There is clearly a difference between these probability distributions, but, as stated in the Indistinguishability Lemma (below), this difference cannot be "detected" in probabilistic polynomial-time.

Notations: Let $\sigma_1, ..., \sigma_n \in \{0,1,2,3\}$. Then $C(\sigma_1 \cdots \sigma_n) \stackrel{\text{def}}{=} C(\sigma_1) \cdots C(\sigma_n)$.

Indistinguishability Lemma: Let S_n be the set of pairs (G, ϕ), where G is a simple graph on n vertices and ϕ is a 3-colouring of its vertices. For every polynomial-time random variable $\{X_n Y_n Z_n\}_{n \in \mathbb{N}}$ (with X_n, Y_n ranging over S_n), the (polynomial-time) random variables $M^*(X_n, Z_n)$ and $V^*_{P(X_n,Y_n)}(X_n, Z_n)$ are polynomially-indistinguishable.

Proof: The proof is by contradiction. We assume that the two random variables can be told apart by a polynomial-time algorithm, D, and derive a contradiction to the uniform security of the commitment scheme C. The reader should note that algorithm D receives as input a text of the form $(C(a_1), ..., C(a_n)), a_u, r_u, a_v, r_v$, where r_u and r_v are the coin tosses used in the commitments to a_u and a_v, respectively. The u and v may be determined by the entire sequence. This creates difficulties that need to be resolved with some care. Details follow.

By the contradiction hypothesis we have a polynomial-time random variable $\{X_n Y_n Z_n\}_{n \in \mathbb{N}}$ and a probabilistic polynomial-time algorithm D such that (for some $c > 0$ and infinitely many $n \in \mathbb{N}$) the following inequality holds

$$\left| \text{Prob}\left(D(V^*_{P(X_n,Y_n)}(X_n, Z_n)) = 1\right) - \text{Prob}\left(D(M^*(X_n, Z_n)) = 1\right) \right| > \frac{1}{n^c} \tag{15}$$

In the sequel we fix the random variable $\{X_n Y_n Z_n\}_{n \in \mathbb{N}}$ and sometimes omit it from the notation. Let g be a graph on n vertices (and m edges), let ϕ be a 3-colouring of the vertices of g, and z an arbitrary string. We define the following random variables:

1) The random variable $\xi_{PV}(g,\phi,z)$ is defined by the following randomized process: select uniformly an edge (u,v) in the edge-set of g, a permutation $\pi \in_R \mathrm{Sym}_3$ and $r_1,...,r_n \in_R \{0,1\}^n$; and set $\overline{c} \leftarrow C_{r_1}(\pi(\phi(1)))\cdots C_{r_n}(\pi(\phi(n)))$. If $V^*(g,z,\overline{c}) = (u,v)$ then set $\xi_{PV}(g,\phi,z)$ to $(g,z,\overline{c},(u,v),(r_u,\pi(\phi(u))),(r_v,\pi(\phi(v))))$ else set $\xi_{PV}(g,\phi,z)$ to $\perp$. (Note that conditioned on $\xi_{PV}(g,\phi,z)$ not being $\perp$ the random variable $\xi_{PV}(g,\phi,z)$ has the same distribution as $V^*_{P(g,\phi)}(g,z)$.)

2) The random variable $\xi_M(g,\phi,z)$ is defined by the following randomized process: select uniformly an edge (u,v) in the edge-set of g, a pair $a \neq b \in_R \{1,2,3\}$ and $r_1,...,r_n \in_R \{0,1\}^n$; and set $\overline{c} \leftarrow C_{r_1}(\sigma_1)\cdots C_{r_n}(\sigma_n)$, where $\sigma_u = a$, $\sigma_v = b$ and $\sigma_w = 0$ for all $w \in \{1,2,...,n\} - \{u,v\}$. If $V^*(g,z,\overline{c}) = (u,v)$ then set $\xi_M(g,\phi,z)$ to $(g,z,\overline{c},(u,v),(r_u,a),(r_v,b))$ else set $\xi_M(g,\phi,z)$ to $\perp$.

We shall reduce the analysis of the relation between the random variables V^*_P and M^* (which is the focus of the lemma) to the analysis of the relation between the random variables ξ_{PV} and ξ_M. First we show that with probability $\approx \frac{1}{m}$ each of the variables ξ_{PV} and ξ_M is not assigned $\perp$.

Claim 3.1:

1. For every g,ϕ,z

$$\mathrm{Prob}(\xi_{PV}(g,\phi,z) \neq \perp) = \frac{1}{m}$$

2. For every sufficiently large n

$$\left|\mathrm{Prob}(\xi_M(X_n,Y_n,Z_n) \neq \perp) - \frac{1}{m}\right| < \frac{1}{m^3}$$

Part (2) of the claim does not play a role in our argument and in fact it is implied by the rest of the analysis. We state and prove it here only in order to appeal to the reader's intuition.

Proof: Part (1) is immediate by definition of ξ_{PV}. Part (2) is proved using the uniform security of the commitment scheme: If the probability that V^* asks to reveal the colours of u and v when seeing $C(0^{u-1}a0^{v-u-1}b0^{n-v})$ on its communication tape differs substantially from the probability that V^* asks so when seeing $C(0^{r-1}a0^{s-r-1}b0^{n-s})$ on its communication tape, for $(r,s) \in_R E$, then these commitments can be told apart (using the auxiliary information (u,v,a,b)). The claim follow. $\diamond$

For every integer t, we now define another two pairs of random variables.

3) Let $\xi^t_{PV}(g,\phi,z)$ denote the random variable obtained by taking a sequence of t independent copies of $\xi_{PV}(g,\phi,z)$. Similarly, define $\xi^t_M(g,\phi,z)$ as the sequence consisting of t independent copies of $\xi_M(g,\phi,z)$.

4) Define a function ρ which maps a sequence into the first element which is not $\perp$ and if not such element exists then the value is $\perp$ (i.e. $\rho(\alpha_1,\alpha_2,...,\alpha_t) = \alpha_i$, iff $\alpha_j = \perp$ for all $j < i$ and either $\alpha_i \neq \perp$ or $i = t$). In the sequel, we will consider the random variables $\rho(\xi^t_{PV})$ and $\rho(\xi^t_M)$.

It is also easy to see that

Claim 3.2:

1. $\rho(\xi^{m^2}_{PV}(g,\phi,z))$ is statistically close to $V^*_{P(g,\phi)}(g,z)$. Namely, the statistical difference between the two random variables is smaller than 2^{-m}.

2. $\rho(\xi^{m^2}_M(g,\phi,z))$ has the same distribution as $M^*(g,z)$.

Proof: Part (1) follows by part (1) of Claim 3.1 and by noting that the conditional distribution of $\xi_{PV}(g,\phi,z))$, given it is not $\perp$, is identical to $V^*_{P(g,\phi)}(g,z)$. Part (2) is immediate by the definitions of $M^*(g,z)$ and $\xi_M^{m^2}(g,\phi,z)$. $\diamond$

Using the contradiction hypothesis (i.e., Eq. 15) and Claim 3.2, we get

Claim 3.3: There exists a probabilistic polynomial-time algorithm D' satisfying for infinitely many $n \in \mathbb{N}$

$$|\text{Prob}\,(D'(X_nY_nZ_n, \xi_{PV}(X_n,Y_n,Z_n))=1) - \text{Prob}\,(D'(X_nY_nZ_n, \xi_M(X_n,Y_n,Z_n))=1)| > \frac{1}{2\cdot m^2\cdot n^c}$$

Proof: Combining Eq. 15 and Claim 3.2 we conclude that algorithm D distinguishes $\rho(\xi_{PV}^{m^2}(X_n,Y_n,Z_n))$ and $\rho(\xi_M^{m^2}(X_n,Y_n,Z_n))$ with gap $\frac{1}{n^c} - \frac{1}{2^m} > \frac{1}{2n^c}$. The claim follows by a standard hybrid argument: The i^{th} hybrid, $h_i(g,\phi,z)$, consists of i independent copies of $\xi_{PV}(g,\phi,z)$ followed by m^2-i independent copies of $\xi_M(g,\phi,z)$. Clearly, $h_{m^2}(X_n,Y_n,Z_n) = \xi_{PV}^{m^2}(X_n,Y_n,Z_n)$ whereas $h_0(X_n,Y_n,Z_n) = \xi_M^{m^2}(X_n,Y_n,Z_n)$. Hence, there exists an i (a random one will do) so that

$$|\text{Prob}\,(D(\rho(h_i(X_n,Y_n,Z_n)))=1) - \text{Prob}\,(D(\rho(h_{i-1}(X_n,Y_n,Z_n)))=1)| > \frac{1}{2n^c\cdot m^2}$$

Incorporating the polynomial-time computable processes ξ_{PV} and ξ_M and the function ρ into the distinguisher D, results in a distinguisher D' which justifies the claim: on input (g,ϕ,z) and a string α, the distinguisher D' selects $i \in_R \{1,...,m^2\}$, and outputs $D(\rho(\beta_1\beta_2\cdots\beta_{m^2}))$, where $\beta_j = \xi_{PV}(g,\phi,z)$ for $j < i$, $\beta_i = \alpha$, and where $\beta_j = \xi_M(g,\phi,z)$ for $j > i$. $\diamond$

We now use algorithm D' to construct a polynomial-time algorithm D'' distinguishing commitments to string of the form $1^n2^n3^n$ from commitments to 0^{3n}, $n \in \mathbb{N}$.

Claim 3.4: There exists a probabilistic polynomial-time D'' such that for infinitely many $n \in \mathbb{N}$

$$\left|\text{Prob}\,(D''(C(1^n2^n3^n))=1) - \text{Prob}\,(D''(C(0^{3n}))=1)\right| > \frac{1}{2m^2n^c}$$

Proof: On input $\alpha_1\alpha_2\cdots\alpha_{3n}$, where each $\alpha_i \in \{0,1,2,3\}$, algorithm D'' acts as follows. D'' generates $(g,\phi,z) \leftarrow X_nY_nZ_n$, picks (u,v) uniformly in the edge-set of the graph g, picks $\pi \in_R \text{Sym}_3$ and $r,s \in_R \{0,1\}^n$, and computes $\overline{c} = c_1\cdots c_n$, where $c_u = C_r(\pi(\phi(u)))$, $c_v = C_s(\pi(\phi(v)))$ and $c_w = \alpha_{(\pi(\phi(w))-1)\cdot n+w}$ for $w \in \{1,2,...,n\} - \{u,v\}$. Algorithm D'' runs V^* on input g,z placing $\overline{c}$ on the communication tape of V^*.

Before continuing with the description of D'' let us examine the string $\overline{c}$ constructed by the algorithm D''. Independently of the input, c_u and c_v are generated by D'' itself so that they are in the range of $C(\pi(\phi(u)))$ and $C(\pi(\phi(u)))$, respectively. The other c_w's are taken from the input. In particular, c_w is taken from the $\pi(\phi(w))^{\text{th}}$ (n-element) block of the input. Hence, if $\alpha_1\alpha_2\cdots\alpha_{3n}$ is in the range of $C(1^n2^n3^n)$ then c_w is in the range of $C(\pi(\phi(w)))$, whereas if $\alpha_1\alpha_2\cdots\alpha_{3n}$ is in the range of $C(0^{3n})$ then c_w is in the range of $C(0)$.

Continuing with the description of algorithm D'', if $V^*(g,z,\overline{c})$ equals (u,v) then algorithm D'' outputs $D'(g,\phi,z,\overline{c},(u,v),(r,\pi(\phi(u))),(s,\pi(\phi(v))))$, else D'' outputs $D'(g,\phi,z,\perp)$.

The reader can easily verify that

$$\begin{aligned} D''(C(0^{3n})) &= D'(X_nY_nZ_n, \xi_M(X_n,Y_n,Z_n)) \\ D''(C(1^n2^n3^n)) &= D'(X_nY_nZ_n, \xi_{PV}(X_n,Y_n,Z_n)) \end{aligned}$$

Using Claim 3.3, our claim follows. $\diamond$

Claim 3.4 constitutes a contradiction to the uniform security of the commitment scheme C, and the Indistinguishability Lemma follows. $\Box$

This completes the proof of Proposition 3. Note that the fact that the random variable $\{X_n Y_n Z_n\}_{n\in\mathbb{N}}$ is polynomial-time is crucial to the proof of Claim 3.4. This claim is the heart of the proof that the simulator produces conversations which are polynomial indistinguishable from the real conversations.

Remark 21 :(due to Erez Petrank): A more efficient simulator can be constructed as follows. On input a graph g and auxiliary input z, the simulator selects $r \in \{0,1\}^{\mathrm{poly}(|g|)}$ and repeats the following steps m times.

S1) Machine M^* chooses independently and uniformly $\gamma_1, \gamma_2, ..., \gamma_n \in_R \{1,2,3\}$. Machine M^* chooses random r_i's ($r_i \in_R \{0,1\}^n$) and computes $c_i = C_{r_i}(\gamma_i)$ for each $i \in V$.

S2) Machine M^* sets $(u,v) = V^*(g,z,r;(c_1,...,c_n))$ (w.l.o.g., $(u,v) \in E$). If $\gamma_u \neq \gamma_v$ ("lucky for M^*"), machine M^* stops outputting $(g,z,r,(c_1,...,c_n),(u,v),(\gamma_u,r_u,\gamma_v,r_v))$; otherwise M^* is going to repeat steps (S1) and (S2).

If all m repetitions were completed (without M^* halting) then the machine M^* halts outputting $\perp$. Note that, with probability greater than $1 - \frac{1}{m^{O(1)}}$, m repetitions suffice to produce a conversation here, whereas in the simulator used before m repeatinions will produce a conversation with probability at most $(1-\frac{1}{m})^m < \frac{1}{2}$.

Proof of Proposition 4

The proof follows ideas of Impagliazzo and Yung [22], which proved that under non-uniform assumptions all languages in $\mathbf{IP}$ have zero-knowledge proof systems. Their proof is in fact a transformation of a given interactive proof into a (non-uniformly) zero-knowledge proof for the same language. Here, we use the same transformation, but the proof that the transformation produces a (uniformly) zero-knowledge system is different as we cannot use non-uniformity in the reductions. The ideas in the alternative transformation of [2] (also demonstrating that "all $\mathbf{IP}$ is in zero-knowledge") are not applicable here, since they require first to transform $\mathbf{MA}$ to "$\mathbf{MA}$ with no error on yes-instances". However, the transformation of $\mathbf{MA}$ to "$\mathbf{MA}$ with no error on yes-instances" presented in [14] requires powerful provers which are not probabilistic polynomial-time machines with probabilistic polynomial-time generated auxiliary input.

Let $L \in \mathbf{MA}$ and R_L be the probabilistic polynomial-time witness-checking algorithm for L, guaranteed by the definition of $\mathbf{MA}$. Without loss of generality, we assume that algorithm R_L has exponentially vanishing error probability. The (zero-knowledge) interactive proof system for L proceeds as follows.

Protocol 2
Inputs: $x \in L$ is common input. The prover gets $y \in \{0,1\}^{\mathrm{poly}(|x|)}$, satisfying $\mathrm{Prob}(R_L(x,y)=1) > 1 - 2^{-|x|}$, as auxiliary input.
Conventions: $n = |x|$. Let $m = \mathrm{poly}(n)$ be a bound on the length of y and the number of coin-tosses for R_L. By $R_L(x,y,r)$ we denote the output (i.e., either 0 or 1) of algorithm R_L on input (x,y) and coin tosses r. For conventions regarding the commitment scheme C, see the proof of Proposition 3.

P1) In this step, the prover commits itself to y (commitment c_2 below) and starts a subprotocol of "coin-tossing-into-the-well" [3] by committing itself to a randomly chosen string (commitment c_1 below). Namely, the prover chooses $r' \in_R \{0,1\}^m$, $r_1, r_2 \in_R \{0,1\}^{nm}$, computes the commitments $c_1 = C_{r_1}(r')$ and $c_2 = C_{r_2}(y)$, and sends c_1, c_2 to the verifier.

V1) The verifier answers with a uniformly chosen $r'' \in \{0,1\}^m$.

P2) The prover answers with r' and r_1 (in order to prove that r' is indeed the value committed to in step P1).

V2) The verifier verifies that r' is indeed the value committed to in step P1, by comparing $C_{r_1}(r')$ and c_1. The verifier continues only if $C_{r_1}(r') = c_1$.

PV) Each of the parties compute $r \leftarrow r' \oplus r''$ (this completes the "coin-tossing-into-the-well"). The prover evaluates $R_L(x, y, r)$ and continues only if its value is 1 (halting in the unlikely case that $R_L(x, y, r) = 0$). Using a zero-knowledge proof system, denoted (P_{NP}, V_{NP}), the prover proves to the verifier the following NP-statement concerning (x, r, c_2).

$$\exists r_2, y \text{ such that } c_2 = C_{r_2}(y) \wedge R_L(x, y, r) = 1 \tag{16}$$

We stress that the prover "knows" r_2 (chosen by him in step P1) and y (given as auxiliary input), and hence can perform his part in the proof in polynomial-time.

The fact that Protocol 2 constitutes an probabilistic polynomial-time interactive proof for L, follows from the fact that r is uniformly distributed as long as V follows the protocol and the fact that (P_{NP}, V_{NP}) is an interactive proof for the statement in Eq. 16. To show that the prover is uniformly zero-knowledge, we construct the following simulator (for the conversations of verifier V^* with the above prover). On input x and z the simulator, denoted M^*, proceeds as follows

S1) Chooses r' uniformly in $\{0,1\}^m$, and produce using coin tosses r_1 and r_2, respectively, the commitments $c_1 = C_{r_1}(r')$ and $c_2 = C_{r_2}(1^m)$.

S2) Produces ("the verifier's answer") $r'' \leftarrow V^*(x, z, c_1, c_2)$, and sets $r \leftarrow r' \oplus r''$. Sets z' to be the contents of the work tape of V^* at this stage.

S3) Using the simulator M^*_{NP}, guaranteed for the zero-knowledge subprotocol, M^* outputs $text \leftarrow M^*_{NP}((x, r, c_2), z')$, where (x, r, c_2) is the common input to the NP-protocol and z' is the auxiliary input of the verifier.

Remark: Typically, M^* runs the simulator M^*_{NP} on a false statement (as 1^m, "hidden" in c_2, is unlikely to be a witness). Yet the security of the commitment scheme yields that M^*_{NP} will not be able to "tell the difference".

Clearly, the simulator M^* works in polynomial-time. To analyze the quality of the texts produced by the simulator, we consider two modifications of the simulator M^*. The first machine, denoted M^*_1, gets inputs x, y, z (note that y is not given to M^*) and acts as M^* except that in step (S1) it computes $c_2 = C_{r_2}(y)$ (i.e. as the prover does rather then setting $c_2 = C_{r_2}(1^m)$ as M^* does). The second machine, denoted M^*_2, gets inputs x, y, z and acts as M^*_1 except that it executes step (S3) if and only if $R_L(x, y, r) = 1$ (otherwise it outputs $\perp$). (Note that both M^* and M^*_1, unlike the real prover, produce a text even if $R_L(x, y, r) = 0$.) Clearly, *Claim 4.1*: All three machines (i.e. M^*, M_1^* and M_2^*) are probabilistic polynomial-time. $\diamond$

We now prove, for any polynomial-time random variable $\{X_n Y_n Z_n\}_{n\in\mathbb{N}}$, the following three claims:

Claim 4.2: The random variables $M_2^*(X_n, Y_n, Z_n)$ and $V^*_{P(x,y)}(x,z)$ are polynomially indistinguishable.

Proof: Both random variables, $V^*_{P(x,y)}(x,z)$ and $M_2^*(x,y,z)$, have the form $((x,r,c_2,z'), text)$, where the (x,r,c_2,z') part is identically distributed. The only difference between these variables is in the $text$ part produced from (x,r,c_2,z'). In $V^*_{P(x,y)}(x,z)$ the text is produced by the interaction between the prover $P_{NP}((x,r,c_2),(r_2,y))$ and the verifier $V^*_{NP}(x,z')$, whereas in $M_2^*(x,y,z)$ the text is the output of $M^*_{NP}((x,r,c_2),z')$. Note that (x',y',z'), where $x' = (x,r,c_2)$, $y' = (r_2,y)$ and z' is as above, are computed in probabilistic polynomial-time from (x,y,z). The Claim follows from the zero-knowledge property of the proof system (P_{NP}, V_{NP}) (on the input triple (X'_n, Y'_n, Z'_n)). $\diamond$

Claim 4.3: The random variables $M_1^*(X_n, Y_n, Z_n)$ and $M_2^*(X_n, Y_n, Z_n)$ are polynomially indistinguishable.

Proof Sketch: We will show that the probability that $M_2^*(X_n, Y_n, Z_n)$ does not execute step (S3) is negligible, and the claim will follow. The proof boils down to showing that V^* given $x, z, C(y), C(r')$, where $(x,y,z) \leftarrow X_n Y_n Z_n$ and r' is uniformly chosen, is unlikely to choose r'' such that $R_L(x,y,r'\oplus r'') \neq 1$.

Namely, we wish to show that, for every probabilistic polynomial-time algorithm A, for every $c > 0$ and all sufficiently large n, $\text{Prob}(R_L(x,y,r' \oplus A(x,z,C(y),C(r'))) < \frac{1}{n^c}$. Recall that by definition of y, the probability that $R_L(x,y,r) \neq 1$ is smaller than 2^{-n}, where $r \in_R \{0,1\}^m$. Hence, for any random variable r'' independent of r', we have $\text{Prob}(R_L(x,y,r' \oplus r'') \neq 1) < 2^{-n}$. In particular, this holds for $r'' = A(x,z,C(y),C(1^m))$. Hence, there is only a negligible fraction of $r' \in \{0,1\}^m$ for which $Prob(R_L(x,y,r' \oplus A(x,z,C(y),C(1^m)) \neq 1) > \frac{1}{n^{O(1)}}$ (here r' is fixed and the probability is taken only over the coin tosses of algorithm A). It follows, that there is only a negligible fraction of $r' \in \{0,1\}^m$ for which $Prob(R_L(x,y,r' \oplus A(x,z,C(y),C(r')) \neq 1) > \frac{1}{n^{O(1)}}$, otherwise this yields an efficient method for finding pairs $(1^m, r')$ such that $C(1^m)$ and $C(r')$ can be distinguished (in the presence of auxiliary input x,y,z,r'), in contradiction to the security of the commitment scheme C. The claim follows. $\diamond$

Claim 4.4: The random variables $M^*(X_n, Z_n)$ and $M_1^*(X_n, Y_n, Z_n)$ are polynomially indistinguishable.

Proof Sketch: Otherwise, one can transform the identical continuations of these programs (i.e. steps (S2) and (S3) of M^* and M_1^*) into a distinguisher of $C(Y_n)$ from $C(1^m)$, in contradiction to the security of the commitment scheme C. $\diamond$

Combining Claims 4.1 through 4.4, Proposition 4 follows. $\square$

5 Discussion

Yao [30] and Goldreich, Micali and Wigderson [16] presented methods for automatically constructing (two-party and multi-party) fault-tolerant protocols for any computable game. Here we only point out that if the inputs to the game are generated by probabilistic procedures of complexity comparable to that of the game, then the constructions of [30, 16] can be carried out using uniform complexity assumptions. In particular, we will use uniformly secure public-key encryption and uniformly secure zero-knowledge proof systems for languages in $\mathcal{NP}$.

Acknowledgements

I wish to thank Shafi Goldwasser for discussions concerning probabilistic polynomial-time interactive proofs, Silvio Micali for discussions concerning the definition of security, and Russell Impagliazzo for explanations of his proof (with Yung) that secure bit commitment imply that all $\mathcal{IP}$ languages have zero-knowledge proof systems.

Special thanks to the referee who has supplied me with many extremely valuable comments. I liked some of his/her comments so much that I could not resist the temptation of incorporated them into the paper without even changing the wording. I'm also grateful to Ran Canetti for pointing out an error in an earlier version of this work, and to Erez Petrank for Remark 21.

References

[1] Babai, L., "Trading Group Theory for Randomness", *Proc. 17th STOC*, 1985, pp. 421-429.

[2] Ben-Or, M., O. Goldreich, S. Goldwasser, J. Hastad, J. Kilian, S. Micali, and P. Rogaway, "Everything Provable is Provable in Zero-Knowledge", *Advances in Cryptology - Crypto 88*, Lecture Notes in Computer Science, Vol. 403, Springer-Verlag, New York, 1990, pp. 37-56.

[3] Blum, M., "Coin Flipping by Phone", *IEEE Spring COMPCOM*, pp. 133-137, February 1982. See also *SIGACT News*, Vol. 15, No. 1, 1983.

[4] Blum, M., and Micali, S., "How to Generate Cryptographically Strong Sequences of Pseudo-Random Bits", *SIAM Jour. on Computing*, Vol. 13, 1984, pp. 850-864. Extended abstract in *FOCS 82*.

[5] Blum, M., and S. Goldwasser, "An Efficient Probabilistic Public-Key Encryption Scheme which hides all partial information", *Advances in Cryptology: Proc. of Crypto 84*, ed. B Blakely, Springer Verlag Lecture Notes in Computer Science, Vol. 196, pp. 289-302.

[6] Brassard, G., D. Chaum, and C. Crepeau, "Minimum Disclosure Proofs of knowledge", *JCSS*, Vol. 37, No. 2, 1988, pp. 156–189.

[7] Brassard, G., and C. Crepeau, "Zero-Knowledge Simulation of Boolean Circuits", *Advances in Cryptology - Crypto86 (proceedings)*, A.M. Odlyzko (ed.), Springer-Verlag, Lecture Notes in Computer Science, Vol. 263, pp. 223-233, 1987.

[8] Brassard, G., and C. Crepeau, "Non-Transitive Transfer of Confidence: A Perfect Zero-Knowledge Interactive Protocol for SAT and Beyond", *Proc. 27th FOCS*, pp. 188-195, 1986.

[9] Brassard, G., C. Crepeau, and M. Yung, "Everything in NP can be argued in perfect zero-knowledge in a bounded number of rounds", *Proc. of the 16th ICALP*, July 1989.

[10] Chaum, D., "Demonstrating that a Public Predicate can be Satisfied Without Revealing Any Information About How", *Advances in Cryptology - Crypto86 (proceedings)*, A.M. Odlyzko (ed.), Springer-Verlag, Lecture Notes in Computer Science, Vol. 263, pp. 195-199, 1987.

[11] Chor, B., and E. Kushilevitz, "Secret Sharing Over Infinite Domains", *Crypto89*, 1989.

[12] Goldreich, O., and H. Krawczyk, "On Sequential and Parallel Composition of Zero-Knowledge Protocols", *17th International Colloquium on Automata Languages and Programming*, Lecture Notes in Computer Science, Vol. 443, Springer-Verlag, New York, 1990, pp. 268–282.

[13] Goldreich O., and L.A. Levin, "Hard-core Predicates for any One-Way Function". *Proc. of the 21th ACM Symp. on Theory of Computing (STOC)*, pp. 25-32, 1989.

[14] Goldreich, O., Y. Mansour, and M. Sipser "Interactive Proof Systems: Provers that Never Fail and Random Selection", *28th FOCS*, pp. 449-461, 1987.

[15] Goldreich, O., S. Micali, and A. Wigderson, "Proofs that Yield Nothing But their Validity and a Methodology of Cryptographic Protocol Design", To appear in *JACM*, 1991.

[16] Goldreich, O., S. Micali, and A. Wigderson, "How to Play any Mental Game or A Completeness Theorem for Protocols with Honest Majority", *Proc. of 19th STOC*, pp. 218-229, 1987.

[17] Goldreich, O., and Y. Oren, "Definitions and Properties of Zero-Knowledge Proof Systems", TR-610, Computer Science Dept., Technion, Haifa, Israel. Submitted to *Jour. of Cryptology*.

[18] Goldwasser, S., and S. Micali, "Probabilistic Encryption", *JCSS*, Vol. 28, No. 2, 1984, pp. 270-299.

[19] Goldwasser, S., S. Micali, and C. Rackoff, "The Knowledge Complexity of Interactive Proof Systems", *SIAM J. on Comput.*, Vol. 18, No. 1, 1989, pp. 186-208.

[20] Hastad, J., "Pseudo-Random Generators under Uniform Assumptions", *22nd STOC*, 1990, pp. 395–404.

[21] Impagliazzo, R., L.A. Levin, and M. Luby, "Pseudorandom Generation from One-Way Functions", *21st STOC*, pp. 12–24, 1989.

[22] Impagliazzo, R., and M. Yung, "Direct Minimum-Knowledge Computations", *Advances in Cryptology - Crypto87 (proceedings)*, C. Pomerance (ed.), Springer-Verlag, Lecture Notes in Computer Science, Vol. 293, 1987, pp. 40-51.

[23] Micali, S., C. Rackoff, and B. Sloan, "The Notion of Security for Probabilistic Cryptosystems", *SIAM Jour. of Computing*, 1988, Vol. 17, pp. 412-426.

[24] M. Naor, "Bit Commitment Using Pseudorandomness", *Advances in Cryptology - Crypto 89*, Lecture Notes in Computer Science, Vol. 435, Springer-Verlag, New York, 1990, pp. 128–137.

[25] Oren, Y., "On the Cunning Power of Cheating Verifiers: Some Observations about Zero-Knowledge Proofs", *Proc. 28th FOCS*, pp. 462-471, 1987.

[26] Shamir, A., "IP = PSPACE", *Proc. 31st FOCS*, pp. 11–15, 1990.

[27] Shannon, C.E., "Communication Theory of Secrecy Systems", *Bell Sys. Tech. J.*, 28, pp. 656-715, 1949.

[28] Tompa, M., and H. Woll, "Random Self-Reducibility and Zero-Knowledge Interactive Proofs of Possession of Information", *Proc. 28th FOCS*, pp. 472-482, 1987.

[29] Yao, A.C., "Theory and Applications of Trapdoor Functions", *Proc. of the 23rd IEEE Symp. on Foundation of Computer Science*, 1982, pp. 80-91.

[30] Yao, A.C., "How to Generate and Exchange Secrets", *Proc. 27th FOCS*, pp. 162-167, 1986.

Appendix A: Encryption schemes cannot hide the plaintext length

Let (G, E, D) be an encryption scheme which does not necessarily satisfy the length conventions of Section 2. In particular, the encryption algorithm E is defined for every possible key and every message (not necessarily of the same length). Furthermore, there is no restriction about the distribution of the length of the ciphertext produced by E (except that the length of the ciphertext must be, of course, polynomial in the length of the inputs to the E).

Let e be an encryption key in the range of $G(1^n)$. Consider the random variables $E_e(1^m)$ and $E_e(1^{m+1})$, for some m polynomial in $|e|$. If the encryption hides also the length of the cleartext, then these two random variables must be polynomially indistinguishable. Considering $m = |e|, ..., P(2|e|)+1$, where $P(\cdot)$ is a polynomial bounding the running time of E, we conclude that $E_e(1^{|e|})$ is polynomially indistinguishable from $E_e(1^{P(2|e|)+2})$. Since $\mathrm{Prob}(|E_e(1^{|e|})| \leq P(2|e|)) = 1$ it follows that $\mathrm{Prob}(|E_e(1^{P(2|e|)+2})| \leq P(2|e|)) > \frac{2}{3}$. On the other hand, using the fact that the encryption is uniquely decipherable, one can easily see that for at most half of the strings $\alpha \in \{0,1\}^{P(2|e|)+2}$ we have $\mathrm{Prob}(|E_e(\alpha)| \leq P(2|e|) + 2) > \frac{1}{3}$. Hence, we can easily find $\alpha \in \{0,1\}^{P(2|e|)+2}$ such that $E_e(\alpha)$ and $E_e(1^{P(2|e|)+2})$ are distinguishable (by merely measuring their length!) in polynomial time.

The reader should not be confused by the following suggestion of an encryption schemes which seems to hide the length of the cleartext. The suggested encryption, when using key e, first pads the message to length $P(|e|)$, where $P(\cdot)$ is some fixed polynomial, and then applies some standard encryption scheme to the resulting (padded) string. The problem is how to encrypt strings which are longer than $P(|e|)$. The only way to solve the problem is to assume that, when using key e, one is never asked to encrypt messages of length greater than $P(|e|)$. In other words, the solution boils down to postulating that the message space contains only strings of length $\leq P(|e|)$, and treating all messages as if they have length $P(|e|)$. Hence, the problem of leaking the length of the message is "solved" by assuming that there is nothing to leak! In any case, we note that assuming that one knows an a-priori bound on the length of the messages to be encrypted (or on the number of messages to be encrypted) may severely restrict the applications.

Futher investigations of related questions appear in [11].

Appendix B: Proof of Remarks 10 and 18

Text added in July 1998.

Proof of Remark 10

The easy direction: Suppose first that the condition in Definition 5 is violated. Specifically, suppose that for some A and $\{T_n\}$ as in the definition,

$$|\text{Prob}\,(A(Z_n, E_{G_1(1^n)}(X_n))=1) - \text{Prob}\,(A(Z_n, E_{G_1(1^n)}(Y_n))=1)\,| \;\geq\; \epsilon(n) \tag{17}$$

Then,

$$\sum_{(\alpha,\beta,\gamma)} \text{Prob}\,(T_n = (\alpha,\beta,\gamma)) \cdot \left|\text{Prob}\,(A(\gamma, E_{G_1(1^n)}(\alpha))=1) - \text{Prob}\,(A(\gamma, E_{G_1(1^n)}(\beta))=1)\right| \;\geq\; \epsilon(n) \tag{18}$$

It follows that there exist sets B_n so that $\text{Prob}\,(T_n \in B_n) \;\geq\; \frac{\epsilon(n)}{2}$, and for every $(\alpha,\beta,\gamma) \in B_n$

$$\left|\text{Prob}\,(A(\gamma, E_{G_1(1^n)}(\alpha))=1) - \text{Prob}\,(A(\gamma, E_{G_1(1^n)}(\beta))=1)\right| \;\geq\; \frac{\epsilon(n)}{2}$$

The opposite direction: On the other hand, suppose that there exists A, $\{T_n\}$ and B_n so that both $\text{Prob}\,(T_n \in B_n) \geq \epsilon(n)$, and for every $(\alpha,\beta,\gamma) \in B_n$,

$$\left|\text{Prob}\,(A(\gamma, E_{G_1(1^n)}(\alpha))=1) - \text{Prob}\,(A(\gamma, E_{G_1(1^n)}(\beta))=1)\right| \geq \epsilon(n) \tag{19}$$

Note that if we sum the probability differences (rather than the absolute value of these differences), over all (α,β,γ), then the result may be small (in absolute value) and even zero. Thus, to derive violation of the condition in Definition 5, we may need to modify the random valiables $\{T_n\}$. Ideally, we would like to do the following, where Π is a random process defined by $\Pi(\gamma,\delta) \stackrel{\text{def}}{=} A(\gamma, E_{G_1(1^n)}(\delta))$.

1. Obtain a sample, denoted (α,β,γ), of T_n.
2. Test whether $\text{Prob}(\Pi(\gamma,\alpha)=1) \geq \text{Prob}(\Pi(\gamma,\beta)=1)$.
3. If the above inequality holds then output (α,β,γ), otherwise output (β,α,γ).

The problem, off course, is that we cannot efficiently implement Step (2). Still a good enough approximation would do. For example, using $\text{poly}(1/\epsilon(n))$ invocations of Π, one may approximate $\text{Prob}(\Pi(\gamma,\delta)=1)$ upto an $\frac{\epsilon(n)^2}{4}$ additive error with failure probability less than $\frac{\epsilon(n)^2}{8}$. This yields a modified polynomial-time random variable, $\{T'_n = X'_n Y'_n Z'_n\}$, so that

$$|\text{Prob}\,(\Pi(Z'_n, X'_n)=1) - \text{Prob}\,(\Pi(Z'_n, Y'_n)=1)\,| \;\geq\; \epsilon(n)\cdot\epsilon(n) - \frac{3\epsilon(n)^2}{4}$$

An alternative (and somewhat more elegant) construction of a modified T_n follows:

1. Obtain a sample, denoted (α,β,γ), of T_n.
2. Invoke Π once on (γ,α) (resp., (γ,β)), obtaining $\sigma \leftarrow \Pi(\gamma,\alpha)$ and $\tau \leftarrow \Pi(\gamma,\beta)$.
    (a) If $\sigma = \tau$ then output (α,β,γ) with probability 1/2 and (β,α,γ) otherwise (also w.p. 1/2).

(b) If $\sigma = 1$ and $\tau = 0$ then output (α, β, γ).

(c) If $\sigma = 0$ and $\tau = 1$ then output (β, α, γ).

Let $T_n' = (X_n', Y_n', Z_n')$ denote the output.

Then, conditioned on Case 2a occurring, we have

$$\begin{aligned}\text{Prob}\,(\Pi(Z_n', X_n') = 1) &= \text{Prob}\,(\Pi(Z_n', Y_n') = 1) \\ &= \frac{1}{2} \cdot (\text{Prob}\,(\Pi(Z_n, X_n) = 1) + \text{Prob}\,(\Pi(Z_n, Y_n) = 1))\end{aligned}$$

Conditioning on Case 2b occurring and $t \stackrel{\text{def}}{=} (\alpha, \beta, \gamma)$ being obtained at Step 1, we have

$$\text{Prob}\,(\Pi(Z_n', X_n') = 1 | T_n = t) - \text{Prob}\,(\Pi(Z_n', Y_n') = 1 | T_n = t) \;=\; p - q$$

where $p \stackrel{\text{def}}{=} \text{Prob}(\sigma = 1 | T_n = t)$ and $q \stackrel{\text{def}}{=} \text{Prob}(\tau = 1 | T_n = t)$. Likewise, when Case 2c occurs, the difference is $q - p$. Also, given (α, β, γ) being obtained at Step 1, Case 2b occurs with probability $p \cdot (1 - q)$ and Case 2b occurs with probability $q \cdot (1 - p)$. Thus, we have

$$\begin{aligned}\text{Prob}\,(\Pi(Z_n', X_n') = 1 | T_n = t) \;-\; &\text{Prob}\,(\Pi(Z_n', Y_n') = 1 | T_n = t) \\ &= (p \cdot (1 - q)) \cdot (p - q) + (q \cdot (1 - p)) \cdot (q - p) \\ &= (p - q) \cdot (p \cdot (1 - q) - q \cdot (1 - p)) \\ &= (p - q)^2\end{aligned}$$

Thus,

$$\text{Prob}(\Pi(Z_n', X_n') = 1) - \text{Prob}(\Pi(Z_n', Y_n') = 1) \;\geq\; \epsilon(n) \cdot \epsilon(n)^2$$

Proof of Remark 18

The easy direction is proved as above. For the opposite direction we employ a third alternative to the proof presented above (for Remark 10): That is, rather than switching X_n and Y_n when necessary, we trim T_n whenever the desired relation between the two random variables is not satisfied. Let us carry this idea in greater detail in the current context.

Suppose that for A, S, B^*, M^* and $\{T_n\}$ as in Remark 18 it holds that $\text{Prob}(T_n \in B_n) > \epsilon(n)$, with $B_n \subseteq S \times \{0,1\}^*$, and for every $(x, y, z) \in B_n$

$$\left| \text{Prob}\left(D(B^*_{A(x,y)}(x, z)) = 1\right) - \text{Prob}\,(D(M^*(x, z)) = 1) \right| > \epsilon(n) \tag{20}$$

Suppose, without loss of generality (by possiblly redefining the algorithm D so that $D(t) \stackrel{\text{def}}{=} 1 - D(t)$), that for infinitely many n's we have

$$\text{Prob}\left(D(B^*_{A(x,y)}(x, z)) = 1\right) > \text{Prob}\,(D(M^*(x, z)) = 1) + \epsilon(n) \tag{21}$$

We construct a new random variable, T_n', by "trimming" $T_n = X_n Y_n Z_n$ as follows.

- Let $\ell \stackrel{\text{def}}{=} O(\log(1/\epsilon(n)))$.
- Repeat the following for at most $\ell/\epsilon(n)$ many times:

  1. Obtain a sample, denoted (x, y, z), of T_n.

2. Using poly$(1/\epsilon(n))$ many invocations, estimate the following probabilities upto an $\epsilon(n)/8$ deviation (with failure probability bounded by poly$(\epsilon(n))$)

$$\begin{aligned} p &\stackrel{\text{def}}{=} \text{Prob}\left(D(B^*_{A(x,y)}(x,z))=1\right) \\ q &\stackrel{\text{def}}{=} \text{Prob}\left(D(M^*(x,z))=1\right) \end{aligned}$$

Let $\tilde{p}$ and $\tilde{q}$ denote the obtained estimates.

3. If $\tilde{p} > \tilde{q} + \frac{3\epsilon(n)}{4}$ then output (x, y, z) and halt. Otherwise, proceed to the next iteration.

- If all iterations were completed without output then output a dummy triplet (e.g., T_n itself).

Note that with probability at least $1 - \epsilon(n)/8$ at least one of the iterations obtains a sample in B_n, and with probability at least $1 - \epsilon(n)/8$ all estimates are within $\epsilon(n)/8$ of the correct value. Thus, with probability at least $1 - \epsilon(n)/4$ we output a triplet (x, y, z) so that

$$\text{Prob}\left(D(B^*_{A(x,y)}(x,z))=1\right) > \text{Prob}\left(D(M^*(x,z))=1\right) + \frac{\epsilon(n)}{2}$$

and so $T'_n = X'_n Y'_n Z'_n$ satisfies

$$\text{Prob}\left(D(B^*_{A(X'_n,Y'_n)}(X'_n,Z'_n))=1\right) > \text{Prob}\left(D(M^*(X'_n,Z'_n))=1\right) + \frac{\epsilon(n)}{4}$$