Technical Report CS97-529, Department of Computer Science and Engineering, University of California at San Diego, March 1997.

A Note on Negligible Functions

Mihir Bellare*

Dept. of Computer Science & Engineering
University of California at San Diego
9500 Gilman Drive
La Jolla, California 92093, USA.

March 1997

Abstract

The notion of a negligible function is used in theoretical cryptography to formalize the notion of a function asymptotically "too small to matter." We claim the issue that really arises is what it might mean for a *sequence* of functions to be "negligible." We consider (and define) two such notions, and prove them equivalent.

Roughly, this enables us to say that any cryptographic primitive has a *specific* associated "security level." In particular we can say this for any one-way function. We can also reconcile different definitions of negligible error arguments and computational proofs of knowledge that have appeared in the literature.

Although there are some cryptographic consequences, the main result is something purely about negligible functions.

*E-Mail: `mihir@cs.ucsd.edu`. URL: `http://www-cse.ucsd.edu/users/mihir`. Supported in part by NSF CAREER Award CCR-9624439 and a Packard Foundation Fellowship in Science and Engineering.

Contents

1 Introduction

Recall that a function $g\colon \mathsf{Z}_+ \to \mathbf{R}$ is *negligible* if it vanishes faster than the reciprocal of any polynomial. (That is, for every $c \in \mathsf{Z}_+$ there is an integer n_c such that $g(n) \le n^{-c}$ for all $n \ge n_c$.) The notion of a negligible function is used in theoretical cryptography to formalize the notion of something that (asymptotically) doesn't matter. The "something" is usually the success probability of an adversary, measured as a function of a security parameter n.

In this note we will be looking not at a single function g, but at a collection $\mathcal{F}$ of functions, and asking what is an appropriate notion of "negligibility" of this collection. We claim this issue of "negligibility of a collection of functions," as opposed to negligibility of a single function, is what actually arises in definitions in theoretical cryptography (although it is not looked at like this), and there are a couple of ways it could be formalized. One corresponds to our standard definitions of security of cryptographic primitives. But the other also seems natural and meaningful, and in some contexts (such as error probabilities of protocols) perhaps more so than than the standard notion. Roughly, the difference is in whether the primitive has a single "security level," or a different one for each adversary.

We will relate the two notions. It turns out the underlying question has nothing to do with cryptography: it is just about negligible functions. We formulate two notions of "negligibility of a collection of functions" and prove an equivalence. This appears to be the first theorem that is solely about negligible functions.

Roughly, what this implies is that to any cryptographic primitive we can associate a single function which is its "security level," rather than having a different security level for each adversary. In particular we apply this to negligible error arguments and computational proofs of knowledge with negligible knowledge error. Here we reconcile two different definitions that have appeared in the literature.

Arguments and proofs of knowledge were our main motivation, because here is seems the alternative form of the definition is more natural. But the issue actually arises for any cryptographic primitive. To illustrate the issue let us look at a simple primitive, namely a one-way function.

1.1 The issue for one-way functions

Let $f\colon \{0,1\}^* \to \{0,1\}^*$ be a polynomial time computable, length preserving function. An *inverter* for f is a polynomial time algorithm I.[1] To any inverter I we associate its success probability in inverting f. This is a function Inv_I of the security parameter n, defined by

$$\mathsf{Inv}_I(n) \;=\; \Pr\left[\, f(x') = y \;:\; x \stackrel{R}{\leftarrow} \{0,1\}^n \;;\; y \leftarrow f(x) \;;\; x' \leftarrow I(y) \,\right] \;.$$

The standard notion of f being one-way is the following.

Definition A. *We say f is one-way if for every inverter I the function Inv_I is negligible.*

This says that for each fixed inverter I, the associated success probability function is negligible. But the success probabilities for different inverters may vary widely, following different rates of approach to zero.

There is another way we might consider saying f is one-way. To describe this it is first convenient to adopt the following shorthand: for two functions $g_1, g_2\colon \mathsf{Z}_+ \to \mathbf{R}$ we say that g_1 is *eventually less than* g_2, written $g_1 \le_{\mathrm{ev}} g_2$, if there is an integer k such that $g_1(n) \le g_2(n)$ for all $n \ge k$. Now,

[1] In some cases it is a probabilistic, polynomial time algorithm, in other cases a polynomial sized family of circuits. The results hold for both cases, so we use the generic term "polynomial time algorithm."

we say f is one-way if there is a single negligible function δ that is a witness to the fact that the success probability of any inverter eventually gets small:

> DEFINITION B. *We say f is one-way if there is a negligible function δ such that $\mathsf{Inv}_I \leq_{\text{ev}} \delta$ for every inverter I.*

In other words, for each inverter I there is an integer k_I such that $\mathsf{Inv}_I(n) \leq \delta(n)$ for all $n \geq k_I$. Now, the success probabilities of different inverters have a common rate of growth: they must all eventually drop below the single negligible function δ. When this happens we call $s(n) = 1/\delta(n)$ the "security level."[2]

IS THERE A SINGLE SECURITY LEVEL? It is not hard to see that Definition B implies Definition A. But Definition B might seem to be asking for something stronger. It says there is a specific function, say $\delta(n) = 2^{-n^{1/5}}$, such that no matter what is the (polynomial time) inversion algorithm, it cannot achieve a success better than this (except for finitely many values of n). It is not a priori clear that for a one-way function (in the sense of Definition A) such a function δ exists. Why can't different inverters have different rates, all negligible, but so that for any particular negligible δ, there is some inverter who does better than δ infinitely often?

More concretely, take a one-way function based, say, on the hardness of the discrete logarithm problem. Then Definition B is saying that there is some specific δ such that computing discrete logarithms can never be done (in polynomial time) with success probability greater than δ. Is this a stronger requirement than just asking that any polynomial time algorithm have negligible success probability?

A QUANTIFIER BASED VIEW. Another way of viewing the definitions is that the order of quantification is different. The definitions ask, respectively, that:

$\forall$ inverters I $\;\exists$ negligible δ_I such that $\mathsf{Inv}_I \leq_{\text{ev}} \delta_I$
$\exists$ negligible δ such that $\forall$ inverters I : $\mathsf{Inv}_I \leq_{\text{ev}} \delta$

Thus the question is whether the order of quantification matters.

THE CONTRA-POSITIVE. Yet another way to see the difference is by taking the contra-positives of the definitions, as we must do in proving theorems based on the assumption that f is one-way. The contra-positive of Definition A is familiar: it says that if f is not one-way then there is an inverter I and a constant c such that $\mathsf{Inv}_I(n) > n^{-c}$ for infinitely many n. That is, there is some inverter whose success probability is not negligible. But the contra-positive of Definition B yields no such inverter. It says that for *every* negligible δ there is an inverter I_δ such that $\mathsf{Inv}_{I_\delta}(n) > \delta(n)$ for infinitely many n. But it does not directly say that there is some inverter achieving non-negligible success.

EQUIVALENCE. However it turns out the above two definitions are equivalent. This means that given a one-way function f according to the standard Definition A, there exists a *specific* negligible function δ so that Definition B is met. In other words, every one-way function does have a "specific" associated security level. Or, put another way, the order of the quantifiers does not matter.

[2] In asking for a single "security level", this definition is in the style of the definitions of Levin [Le] and Luby [Lu]. The latter however measure the quality of inverters via their time to success probability ratios. It seems Definition B is sort of a simplified, special case of their notions in which one looks only at polynomial time adversaries and security of the form $1/\delta$ for a negligible function δ.

1.2 Negligibility of function sequences

It turns out the actual question has nothing to do with one-way functions, or even with cryptography. It is just about negligible functions. Let us first formulate the question, then relate it to the above.

Let $\mathcal{F} = \{F_i\}_{i \in \mathsf{Z}_+}$ be a sequence of functions, all mapping Z_+ to $\mathbf{R}$. We want to ask what it can mean to say that this sequence is "negligible." The first formulation is simple: just say each function, taken individually, is negligible. Formally, we say $\mathcal{F}$ is *pointwise negligible* if F_i is negligible for each $i \in \mathsf{Z}_+$. The second is to ask that the sequence is "uniformly" negligible in that the different functions conform to some common limit point. Formally, $\mathcal{F}$ is *uniformly negligible* if there is a negligible function δ (called a limit point) such that $F_i \leq_{\mathrm{ev}} \delta$ for all $i \in \mathsf{Z}_+$. That is, each F_i drops below δ for large enough n. (The terminology here is by some sort of rough analogy with the notions of pointwise and uniform convergence of sequences of functions in real analysis.)

It is quite easy to see that if sequence $\mathcal{F}$ is uniformly negligible then it is also pointwise negligible. But if it pointwise negligible, is it uniformly negligible? That is, will there be a single limit point? Theorem 3.2 constructs such a limit point to say that the answer is yes: the two notions of negligible sequences are equivalent.

We stress that when we say "sequence" we mean a countable collection of functions. The result is not true for an uncountable collection.

1.3 Relation to cryptographic notions

Relation to one-way functions. Now, how does this relate to the issue for one-way functions? Let $\mathcal{I} = \{I_i\}_{i \in \mathsf{Z}_+}$ be an enumeration of all inverters.[3] For each $i \in \mathsf{Z}_+$ define the function F_i by $F_i(n) = \mathsf{Inv}_{I_i}(n)$. Let $\mathcal{F} = \{F_i\}_{i \in \mathsf{Z}_+}$. Then observe that Definition A is asking that the sequence $\mathcal{F}$ is pointwise negligible and Definition B is asking that the sequence $\mathcal{F}$ is uniformly negligible. So by the above the two definitions are equivalent.

More generally. Now that we see this, it is clear the same is true for pretty much any cryptographic primitive. The (asymptotic) definition of security for any primitive has the following form. To any "adversary" A and any value of the security parameter n there will be associated a success probability $\mathsf{Success}_A(n)$, under some experiment. (For now, an adversary is a uniform algorithm.) The primitive will be said to be "secure" if for each adversary A the function $\mathsf{Success}_A$ is negligible.

To put this in the framework we have been looking at, let $\mathcal{A} = \{A_i\}_{i \in \mathsf{Z}_+}$ be an enumeration of all adversaries in question. Let $F_i(n) = \mathsf{Success}_{A_i}(n)$. Then we see that the definition indicated above is asking that the sequence of functions $\mathcal{F} = \{F_i\}_{i \in \mathsf{Z}_+}$ is pointwise negligible. It seems equally reasonable, however, to ask that the sequence of functions be uniformly negligible. This says there exists a particular negligible function δ such that the success probability $\mathsf{Success}_A(\cdot)$ of any adversary A is eventually less than δ. Here, a specific security level is associated to the primitive, to which all adversaries must eventually conform. This might seem different, but Theorem 3.2 says the two notions are equivalent. In particular it says that it is always possible to find such a specific security level for any primitive even if the definition does not explicitly ask for it.

In some sense it means we can make a stronger statement about the security of standard primitives than we might think. Roughly, we say there is always a single function that captures the "security" of a primitive.

[3] For the moment, assume an inverter is a uniform (probabilistic, polynomial time) algorithm, so that the number of inverters is countable. For the non-uniform case, where there are uncountably many inverters, we have to do more work. See Section 1.4 and the body of the paper.

Does it matter which way we look at it? For a one-way function, it seemed to us of some interest that there is a single "security level" associated to the function, but we do not see any particular advantage to actually formulating the definition in the new way. However, a setting where the second type of formulation seems more natural is in arguments and computational proofs of knowledge.

ERROR PROBABILITIES IN PROTOCOLS. It seems to us natural that the "error probability" in a protocol is a (single) function associated to the verifier. A definition of "negligible error arguments" based on this view is given in [BJY]. Earlier, however, other definitions had appeared which did not have this view of error probability in the case of negligible error [Go, NOVY]. Applying the above however we can show that the two formulations are nonetheless equivalent. See Section 4.2. Similarly, we relate two notions of computational proofs of knowledge with negligible knowledge error suggested in [BeGo]. See Section 4.3.

1.4 Non-uniform adversaries and uncountability

As we indicated above, we wish to associate a function to each adversary, this function being the adversary's success probability, and consider the "negligibility" of the ensuing collection of functions. When the class of adversaries includes only uniform algorithms, the number of adversaries, and hence of functions, is countable, and the result mentioned in Section 1.2 applies. When the class of adversaries is, say, all *non-uniform* polynomial time algorithms, it becomes uncountable, so that we have uncountably many functions in our collection. In this case, we cannot *directly* apply our main result. However, we will see that it is still possible to apply the equivalence, and get the desired results, by considering the "best possible" non-uniform adversaries: this will "reduce" the uncountable case to the countable one.

Since this reduction always works, the countable case (ie. sequences of functions) is in fact the central one. Nonetheless, to cover both kinds of adversaries, the treatment in Section 2 is general, applying to either countable or uncountable collections. We prove in Section 3 the equivalence in the countable case, and also a characterization, for the uncountable case, that enables us to reduce the latter to the former.

2 Definitions and elementary facts

Let $\mathsf{Z}_+ = \{1, 2, 3 \ldots\}$ be the set of positive integers, and $\mathbf{R}$ the reals. Unless otherwise indicated, a function maps Z_+ to $\mathbf{R}$. We sometimes regard a function as a "point" in the space of all functions and refer to it this way. An "integer" means a positive integer, ie. an element of Z_+.

EVENTUALLY LESS THAN AND NEGLIGIBILITY. We begin with a useful shorthand:

Definition 2.1 *If f, g are functions we say that f is* eventually less than g*, written $f \leq_{\text{ev}} g$, if there is an integer k such that $f(n) \leq g(n)$ for all $n \geq k$.*

It is useful to note this relation is transitive:

Proposition 2.2 *The relation $\leq_{\text{ev}}$ is transitive: if $f_1 \leq_{\text{ev}} f_2$ and $f_2 \leq_{\text{ev}} f_3$ then $f_1 \leq_{\text{ev}} f_3$.*

Proof: The first assumption means there is an integer k_1 such that $f_1(n) \leq f_2(n)$ for all $n \geq k_1$. The second assumption means there is an integer k_2 such that $f_2(n) \leq f_3(n)$ for all $n \geq k_2$. Let $k = \max(k_1, k_2)$. Then $f_1(n) \leq f_2(n) \leq f_3(n)$ for all $n \geq k$. So $f_1 \leq_{\text{ev}} f_3$. ▮

Recall that a function f is negligible if for every integer c there is an integer n_c such that $f(n) \leq n^{-c}$ for all $n \geq n_c$. With the above shorthand, another way to say it is:

Definition 2.3 *A function f is* negligible *if $f \leq_{\text{ev}} (\cdot)^{-c}$ for every integer c.*

Here $(\cdot)^{-c}$ stands for the function $n \mapsto n^{-c}$. It is useful to note the following:

Proposition 2.4 *A function f is negligible if and only if there is a negligible function g such that $f \leq_{\text{ev}} g$.*

Proof: If f is negligible then the other condition is satisfied by setting $g = f$. Conversely assume there is a negligible g such that $f \leq_{\text{ev}} g$. We want to show f is negligible. So let $c \in \mathsf{Z}_+$. Then $f \leq_{\text{ev}} g$ (by assumption) and $g \leq_{\text{ev}} (\cdot)^c$ (because g is negligible) so by Proposition 2.2 we have $f \leq_{\text{ev}} (\cdot)^c$. So f is negligible by Definition 2.3. ▮

Negligible collections of functions. Let $\mathcal{F}$ be a collection of functions. (This might be a countable or an uncountable collection. A countable collection is also referred to as a sequence.) We say that such a collection is pointwise negligible if each function, taken individually, is negligible.

Definition 2.5 *Let $\mathcal{F}$ be a collection of functions. We say that $\mathcal{F}$ is* pointwise negligible *if F is negligible for each $F \in \mathcal{F}$.*

This means that for each $F \in \mathcal{F}$ and each integer c there is some number $k(F, c)$, depending on both F and c, such that $F(n) \leq n^{-c}$ whenever $n \geq k(F, c)$. In other words, the different functions could go down at different rates, and although each is eventually below any inverse polynomial, the time at which this happens depends both on the function and the value of c defining the inverse polynomial.

The notion we define next is stronger, in that it asks that there be a single negligible function δ that is a "witness" to the fact that the functions in the collection eventually become small. All functions must eventually drop below δ.

Definition 2.6 *Let $\mathcal{F}$ be a collection of functions. We say $\mathcal{F}$ is* uniformly negligible *if there is a negligible function δ such that $F \leq_{\text{ev}} \delta$ for all $F \in \mathcal{F}$.*

In other words, the collection $\mathcal{F}$ is uniformly negligible if there is a negligible function δ such that for each $F \in \mathcal{F}$ there is an integer $k(F)$ such that $F(n) \leq \delta(n)$ for all $n \geq k(F)$. Notice that the point at which F drops below δ is allowed to depend on F and may vary a lot from function to function in the collection.

Limit points. In what follows it is useful to have the following shorthand:

Definition 2.7 *Let $\mathcal{F}$ be a collection of functions and let δ be a function. We say that δ is a* limit point *of $\mathcal{F}$ if $F \leq_{\text{ev}} \delta$ for each $F \in \mathcal{F}$.*

With this shorthand, we can say:

Proposition 2.8 *A collection of functions $\mathcal{F}$ is uniformly negligible if and only if it has a negligible limit point.*

Proof: Follows from Definition 2.6 and Definition 2.7. ▮

Note that limit points are not unique.

QUANTIFIER BASED VIEW. Formulating the two notions of negligible sequences via quantifiers may help us see more clearly how they differ. The statements that $\mathcal{F}$ is pointwise negligible, or is uniformly negligible, can be written, respectively, as:

$\forall F \in \mathcal{F}\ \exists$ negligible δ_F such that $F \leq_{\rm ev} \delta_F$ [4]
$\exists$ negligible δ such that $\forall F \in \mathcal{F} : F \leq_{\rm ev} \delta$.

The question is whether swapping the order of quantifiers in this way makes a difference.

3 Relations between the two notions of negligible collections

It is easy to see that uniform negligibility implies pointwise negligibility. This is true regardless of whether the collection is countable or uncountable.

Proposition 3.1 *If $\mathcal{F}$ is uniformly negligible then it is pointwise negligible.*

Proof: By assumption $\mathcal{F}$ is uniformly negligible. By Proposition 2.8 there exists a negligible function δ which is a limit function for $\mathcal{F}$. Let $F \in \mathcal{F}$. We know that $F \leq_{\rm ev} \delta$ since δ is a limit point of $\mathcal{F}$, and we know that δ is negligible, so F is negligible by Proposition 2.4. So $\mathcal{F}$ is pointwise negligible as per Definition 2.5. ∎

The question we want to look at is whether the notions are equivalent. The important case is when the collection is a sequence, ie. is countable.

3.1 Equivalence of the two notions of negligibility of function sequences

Here we show that the two notions of negligibility of a sequence (ie. a countable collection) of functions are equivalent. In other words, if we know that each function in a sequence is individually negligible, then it is possible to define a single negligible function δ which is a witness to the negligibility of each of these functions.

Theorem 3.2 *Let $\mathcal{F}$ be a sequence of functions. Then $\mathcal{F}$ is pointwise negligible if and only if it is uniformly negligible.*

We know from Proposition 3.1 that if $\mathcal{F}$ is uniformly negligible then it is pointwise negligible. The other direction, which is the following lemma, is more interesting.

Lemma 3.3 *If $\mathcal{F} = \{F_i\}_{i \in \mathsf{Z}_+}$ is pointwise negligible then it is uniformly negligible.*

The assumption is that each F_i is a negligible function. We claim that $\mathcal{F}$ is uniformly negligible. To show this we will define a negligible limit point δ for $\mathcal{F}$. Before doing so, it may help to see why a tempting easier construction does not work:

Remark 3.4 (Why not the max?) Perhaps the first thought would be to set

$$\delta(n) \;=\; \max_{1 \leq i \leq n} F_i(n)\ . \tag{1}$$

Certainly $F_i \leq_{\rm ev} \delta$ for each $i \in \mathsf{Z}_+$. But it is not hard to see that δ need not be negligible. For example let ν be a negligible function and set $F_i(j) = 1$ if $j \leq i$ and $F_i(j) = \nu(j)$ if $j > i$. The collection $\{F_i\}_{i \in \mathsf{Z}_+}$ is pointwise negligible, but the function δ of Equation (1) is the constant function 1 which is definitely not negligible.

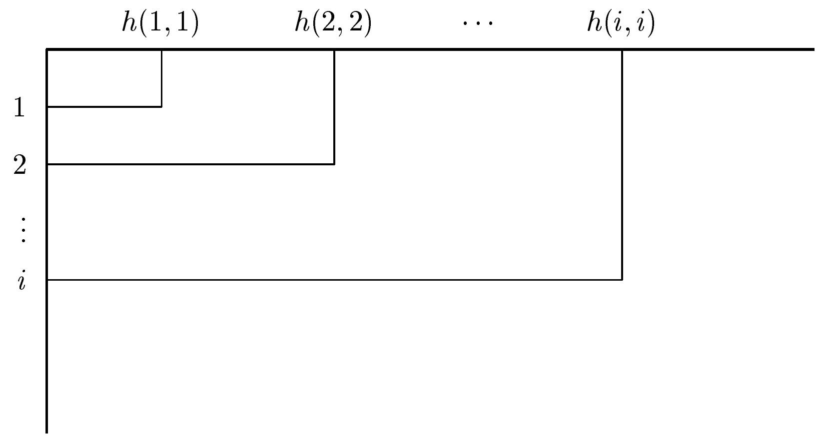

Figure 1: Entry (i,n) of this table is $F_i(n)$. To each row number i we associate a column number $h(i,i)$ such that all entries to the right of the corresponding rectangle (meaning stay above the bottom edge!) are bounded by n^{-i}.

Proof of Lemma 3.2: Let us first sketch the idea and then give the construction and proof.

Imagine a table with rows indexed by the values of $i = 1, 2, \ldots$; columns indexed by the values of $n = 1, 2, \ldots$; and entry (i,n) of the table containing $F_i(n)$. We know that for any c, the entries in each row eventually drop below n^{-c}. But where it happens differs from row to row. We define δ by a sort of diagonalization, in a sequence of "stages." In stage (m,c) we will consider only the first m functions in the list, namely $F_1, \ldots, F_m$. We will find a value $h(m,c)$, such that all these functions are less than $(\cdot)^{-c}$ for $n \geq h(m,c)$, by "moving out" as much as is necessary for all m functions to fall below our target. We view this as defining a sequence of rectangles, each finite, but so that the sequence eventually covers the entire table. Now this function h defines a new table, namely one in which the row m column c entry is $h(m,c)$. We look at its diagonal, namely the points of the form $h(j,j)$. We will use these points to define δ. Namely for each n we define $\delta(n)$ to maximize the functions in the rectangle with column number "closest" to n. Let us now give the construction and proof in more detail. For a vague illustration, see Figure 1.

For any integer c we know that $F_i \leq_{\rm ev} (\cdot)^{-c}$. Let $g(i,c) \in \mathsf{Z}_+$ be such that $F_i(n) \leq n^{-c}$ for all $n \geq g(i,c)$. Now define

$$h(m,c) \;=\; \max_{1 \leq i \leq m} g(i,c) \; . \tag{2}$$

Namely we are considering the first m functions, and seeing how large n must be so that the values of all these functions fall below n^{-c}. We can see that being above $h(m,c)$ does suffice:

Claim 1. For $n \geq h(m,c)$ we have

$$\max_{1 \leq i \leq m} F_i(n) \;\leq\; n^{-c} \; .$$

Proof. If $n \geq h(m,c)$ then Equation (2) implies $n \geq g(i,c)$ for all $i = 1, \ldots, m$. The definition of g then implies that $F_i(n) \leq n^{-c}$ for all $i = 1, \ldots, m$, which is what we have claimed. □

We now look at the "diagonal" of the h function, namely values $h(j,j)$. Observe that these values form a non-decreasing sequence,

$$h(1,1) \;\leq\; h(2,2) \;\leq\; h(3,3) \;\ldots$$

[4] To see this is the same as Definition 2.5, use Proposition 2.4.

We define $\delta(n)$ according to where n falls with respect to this sequence. More precisely, let

$$\textsc{Level}(n) = \text{ the largest integer } j \text{ such that } h(j,j) \leq n\ .$$

Now define

$$\delta(n) \;=\; \max_{1 \leq i \leq \textsc{Level}(n)} F_i(n)\ . \tag{3}$$

Claim 2. The function δ is a limit point of the sequence $\mathcal{F} = \{F_i\}_{i \in \mathsf{Z}_+}$.

Proof. Let $i \in \mathsf{Z}_+$. As per Definition 2.7 we need to show there is an integer n_i such that $F_i(n) \leq \delta(n)$ for all $n \geq n_i$. We set $n_i = h(i,i)$ and claim this works. Indeed, suppose $n \geq h(i,i)$. From the definition of $\textsc{Level}(n)$ it follows that $i \leq \textsc{Level}(n)$. From Equation (3) it follows that $F_i(n) \leq \delta(n)$, as desired. □

Claim 3. The function δ is negligible.

Proof. We need to show δ meets Definition 2.3. So let $c \in \mathsf{Z}_+$. We need to show that there is an integer n_c such that $\delta(n) \leq n^{-c}$ for all $n \geq n_c$. We set $n_c = h(c,c)$ and claim this works. To see this, assume $n \geq h(c,c)$. We want to show $\delta(n) \leq n^{-c}$. Set $m = \textsc{Level}(n)$. Now from Equation (3) we have

$$\delta(n) \;=\; \max_{1 \leq i \leq m} F_i(n)\ . \tag{4}$$

Now by definition of $m = \textsc{Level}(n)$ we know that $n \geq h(m,m)$. So by Claim 1 we have

$$\max_{1 \leq i \leq m} F_i(n) \;\leq\; n^{-m}\ . \tag{5}$$

But we know $n \geq h(c,c)$ and $m = \textsc{Level}(n)$, so by definition of $\textsc{Level}(n)$ it must be that $c \leq m$. This means $n^{-m} \leq n^{-c}$. Now combine this with Equation (4) and Equation (5) to get $\delta(n) \leq n^{-c}$, as desired. □

Claims 2 and 3 together say that δ is a negligible limit point for the sequence $\mathcal{F} = \{F_i\}_{i \in \mathsf{Z}_+}$, and hence $\mathcal{F}$ is uniformly negligible by Proposition 2.8. ▮

Remark 3.5 (An extra property) The limit point δ constructed in Theorem 3.2 has some properties over and above those required to be a negligible limit point. In particular, it is a non-increasing function, meaning $\delta(m) \leq \delta(n)$ for $m \leq n$.

3.2 The uncountable case

Above, the collection of functions considered is countable. What happens when we consider an uncountable collection? We know that Proposition 3.1 applies even then. But the converse fails.

Proposition 3.6 *There is an uncountable collection of functions $\mathcal{F}$ which is pointwise negligible but not uniformly negligible.*

Proof: Simply let $\mathcal{F}$ be the set of *all* negligible functions. (Since we are considering functions mapping Z_+ to $\mathbf{R}$, notice $\mathcal{F}$ is indeed an uncountable collection.) Obviously $\mathcal{F}$ is pointwise negligible. But it is not uniformly negligible. To see this, let g be any negligible function. It cannot be a limit point, because the function $f = 2g$ is negligible, hence in $\mathcal{F}$, but f is not eventually less than g. Thus no negligible function can be a limit point for $\mathcal{F}$, so that $\mathcal{F}$ has no negligible limit point. ▮

This does not mean that *all* uncountable collections of pointwise negligible functions fail to be uniformly negligible. Here is a simple characterization of collections where the equivalence holds.

Definition 3.7 *Let $\mathcal{F}, \mathcal{M}$ be collections of functions. We say that $\mathcal{F}$ is* majored by *$\mathcal{M}$ (or $\mathcal{M}$* majors *$\mathcal{F}$) if for every $F \in \mathcal{F}$ there is an $M \in \mathcal{M}$ such that $F \leq_{\text{ev}} M$.*

The following characterization holds for any collection $\mathcal{F}$, but the interesting case is when $\mathcal{F}$ is uncountable. The key point below is that the collection that majors $\mathcal{F}$ is required to be a sequence, ie. countable.

Theorem 3.8 *Let $\mathcal{F}$ be a collection of functions. Then $\mathcal{F}$ is uniformly negligible if and only if it is majored by some pointwise negligible sequence of functions.*

Proof: First assume $\mathcal{F}$ is uniformly negligible. By Proposition 2.8 it has a negligible limit point δ. We set $\mathcal{M} = \{\delta\}$. This is a countable collection (with a slight abuse of terminology, we include finite in countable!). It is also pointwise negligible and majors $\mathcal{F}$. So $\mathcal{F}$ is indeed majored by some pointwise negligible sequence.

Conversely suppose $\mathcal{M}$ is a pointwise negligible sequence of functions that majors $\mathcal{F}$. Since $\mathcal{M}$ is a sequence (ie. countable) it is uniformly negligible by Theorem 3.2. By Proposition 2.8 $\mathcal{M}$ has a negligible limit point δ. Now if $F \in \mathcal{F}$ then by Definition 3.7 there is some $M \in \mathcal{M}$ such that $F \leq_{\text{ev}} M$. But $M \leq_{\text{ev}} \delta$ because $M \in \mathcal{M}$ and δ is a limit point for $\mathcal{M}$. So $F \leq_{\text{ev}} \delta$ by Proposition 2.2. So δ is also a limit point for $\mathcal{F}$. So $\mathcal{F}$ is uniformly negligible. ∎

Recall that we want to make the functions in the collection correspond to success probabilities of adversaries. We have discussed in Section 1.4 how the countability or uncountability of the collection is a question of whether uniform or non-uniform adversaries are being considered. Although it is not possible to directly apply Theorem 3.2 in the latter case, we will see that it is possible to apply Theorem 3.8, and get the desired results.

4 Application to cryptographic notions

We discussed in Section 1.3 how the above relates to cryptographic definitions. Let us look at this in more detail. We first summarize the implications for one-way functions and then move on to arguments and proofs of knowledge.

4.1 One-way functions

Let $f\colon \{0,1\}^* \to \{0,1\}^*$ be a polynomial time computable, length preserving function. Refer to Section 1.1 where we defined inverters and the success probability function Inv_I associated to any inverter I. As we indicated in Section 1.1, we always fix some class of inverters. We denote this class by $\mathcal{I}$. It is either the class of all probabilistic, polynomial time algorithms (the uniform case) or the class of all polynomial sized circuit families (the non-uniform case). It does not matter which for the results, but the proof of Theorem 4.3 is different for the two settings.

THE INVERSION PROBABILITY FUNCTION COLLECTION. Let us define the collection of functions

$$\mathcal{F} \;=\; \{\mathsf{Inv}_I\}_{I \in \mathcal{I}} \;. \tag{6}$$

This collection captures the success probabilities of all inverters. We will see how the different notions of one-wayness relate to different notions of negligibility of this collection. Notice that in the uniform case $\mathcal{I}$ (and hence $\mathcal{F}$) is countable, while in the non-uniform case, these sets are uncountable.

NOTIONS OF ONE-WAYNESS. The standard notion of f being one-way is re-stated below:

Definition 4.1 *We say f is one-way if for every inverter $I \in \mathcal{I}$ the function Inv_I is negligible.*

In other words, taken individually, the success probability of each inverter eventually becomes less than the reciprocal of any polynomial. Notice that this definition is saying that f is one-way if and only if the collection $\mathcal{F}$ is pointwise negligible.

The stronger formulation is to ask that there is a single negligible function δ below which all success probabilities eventually drop, namely

Definition 4.2 *We say f is one-way if there is a negligible function δ such that $\mathsf{Inv}_I \leq_{\mathrm{ev}} \delta$ for every inverter $I \in \mathcal{I}$.*

Notice that f meets Definition 4.2 if and only if the collection of functions $\mathcal{F}$ is uniformly negligible. Thus, the two notions of one-way functions are arising out of the two possible ways to define negligible collections of functions.

EQUIVALENCE. The following theorem says the two are the same.

Theorem 4.3 *Definitions 4.1 and 4.2 of a one-way function are equivalent.*

Proof: We have already observed that f meets Definition 4.1 if and only if the collection of functions $\mathcal{F}$ defined in Equation (6) is pointwise negligible, and f meets Definition 4.2 if and only if $\mathcal{F}$ is uniformly negligible. However we claim that $\mathcal{F}$ is pointwise negligible if and only if it is uniformly negligible. This says Definitions 4.1 and 4.2 of a one-way function are equivalent.

To prove the claim, let us first take the case where the class $\mathcal{I}$ of inverters considered is the class of uniform (probabilistic) polynomial time algorithms. There are (only) countably many of these. Thus the collection of functions $\mathcal{F}$ is countable. So Theorem 3.2 says $\mathcal{F}$ is pointwise negligible if and only if it is uniformly negligible, as desired.

Now, consider the case where an inverter is a non-uniform polynomial time algorithm. (That is, a polynomial sized circuit family.) The difficulty is that there are uncountably many such inverters. That is, $\mathcal{I}$, and hence $\mathcal{F}$, is now uncountable, so we cannot directly apply Theorem 3.2.

However Proposition 3.1 (which applies even to uncountable collections) says that if $\mathcal{F}$ is uniformly negligible then it is pointwise negligible. Our concern is thus the other direction. Assume $\mathcal{F}$ is pointwise negligible. We know from Proposition 3.6 that not all uncountable pointwise negligible collections are uniformly negligible. But we still claim that our $\mathcal{F}$ is. To prove this we will use Theorem 3.8. To do so, we must exhibit a countable pointwise negligible collection $\mathcal{M}$ that majors $\mathcal{F}$.

This is done by considering only the "best" circuits of any size, as follows. For any n and any circuit C with n inputs let

$$\mathsf{Inv}(C) \;=\; \Pr\left[\, f(x') = y \;:\; x \stackrel{R}{\leftarrow} \{0,1\}^n \;;\; y \leftarrow f(x) \;;\; x' \leftarrow C(y) \,\right]$$

be the inverting probability of this circuit. Now, for each n, k we can fix a circuit $\mathrm{BestCircuit}_{n,k}$ having n inputs and size at most k, such that

$$\mathsf{Inv}(\mathrm{BestCircuit}_{n,k}) \;\geq\; \max_{C} \mathsf{Inv}(C) \;,$$

the maximum here taken over all circuits C with n inputs and size at most k. We call $\text{BestCircuit}_{n,k}$ the *best* size k circuit on n inputs. Now let $p_1, p_2, \ldots$ be an enumeration of all polynomials and let I_i be the circuit family in which the n-th circuit is $\text{BestCircuit}_{n,p_i(n)}$, for each $i, n \in \mathsf{Z}_+$. Let $M_i(n) = \mathsf{Inv}_{I_i}(n)$ and let $\mathcal{M} = \{M_i\}_{i \in \mathsf{Z}_+}$.

We claim $\mathcal{M}$ is countable, pointwise negligible, and majors $\mathcal{F}$. Given these, it follows from Theorem 3.8 that $\mathcal{F}$ is uniformly negligible, and our proof is complete. So let us check that $\mathcal{M}$ has the claimed properties.

That $\mathcal{M}$ is countable is clear. Notice that $I_i \in \mathcal{I}$ for each $i \in \mathsf{Z}_+$ and hence $\mathcal{M} \subseteq \mathcal{F}$. Since the latter is (by assumption) pointwise negligible, so is the former. Now let p be a polynomial and I an inverter which is a circuit family of size p. Notice that for each n we have $\mathsf{Inv}_I(n) \leq \mathsf{Inv}_{I_i}(n)$ where i is such that $p = p_i$. Thus $\mathsf{Inv}_I \leq_{\text{ev}} \mathsf{Inv}_{I_i} = M_i$. So $\mathcal{M}$ majors $\mathcal{F}$. This concludes the proof. ▌

4.2 Negligible error arguments

A couple of definitions of negligible error arguments have appeared in the literature. We show how they correspond to the two different views of negligibility of sequences of functions (even though this is not how they were viewed before), and then use Theorem 3.2 to show they are equivalent.

Preliminaries. We consider a two-party protocol in which a prover attempts to convince a (probabilistic) polynomial time verifier V that their common input x belongs to some underlying language L. We will be looking at the notion of an argument (also called a computationally convincing proof) [BrCr, BCC], so that the prover is a polynomial time algorithm.

Fix a verifier V and a language L for all that follows. Let $\mathsf{Acc}_P(x)$ denote the probability that V accepts in a conversation with P on common input x. We are only interested in what happens when $x \notin L$. (That is, we are interested only in soundness, not in completeness or zero-knowledge.) Associate to each polynomial time prover P the function Acc_P defined by

$$\mathsf{Acc}_P(n) \;=\; \max_{x \notin L \,:\, |x|=n} \text{Accept}_P(x) \;.$$

(We adopt the convention that the function has value 0 at a point n for which there is no $x \notin L$ of length n.)

When $x \notin L$, a polynomial time prover must have low probability of convincing the verifier to accept. We let $\mathcal{P}$ be the class of all provers considered. Again, there is the uniform case, in which $\mathcal{P}$ is the class of all probabilistic, polynomial time algorithms, and the non-uniform case, where $\mathcal{P}$ is the class of all polynomial sized circuit families. Which of these we consider does not matter for the definitions and results, but the two cases will be treated differently in the proof of Theorem 4.7.

The prover success probability function collection. Let us define the collection of functions

$$\mathcal{F} \;=\; \{\mathsf{Acc}_P\}_{P \in \mathcal{P}} \;. \tag{7}$$

This collection captures the success probabilities of all provers. We will see how the different notions of negligible error arguments relate to different notions of negligibility of this collection. As before, in the uniform case $\mathcal{P}$ (and hence $\mathcal{F}$) is countable, while in the non-uniform case, these sets are uncountable.

Error probability of an argument. As we said above, the protocol must have computational soundness: when $x \notin L$, a polynomial time prover must be unable to make the verifier accept, except with low probability. The latter is called the "error probability."

In defining this error probability, however, one must be a little careful.[5] A definition for the case of error 1/3 is given by Goldreich [Go, Section 6.8.1]. Extending this, the following general definition of an argument with error probability a function ϵ is given in [BJY]:

Definition 4.4 [BJY] *Verifier V is computationally sound for L, with error-probability ϵ, if $\mathsf{Acc}_P \leq_{\rm ev} \epsilon$ for every prover $P \in \mathcal{P}$.*

Namely, the acceptance probability of any prover eventually drops below the error probability ϵ.

NEGLIGIBLE ERROR ARGUMENTS. Here we are interested in arguments of negligible error probability. How can we define this? In the light of having Definition 4.8, which is a general notion of an argument with error probability ϵ, it is natural to view negligible error as a special case. Namely, an argument has negligible error if it has error ϵ for some negligible function ϵ. This is the view adopted in [BJY]. The definition amounts to the following:

Definition 4.5 [BJY] *Verifier V is computationally sound for L, with negligible error-probability, if there is a negligible function ϵ such that $\mathsf{Acc}_P \leq_{\rm ev} \epsilon$ for every prover $P \in \mathcal{P}$.*

However, previous works had given a different definition of negligible error arguments. The notion proposed by Goldreich [Go, Section 6.8.1] and Naor et. al. [NOVY] is the following:

Definition 4.6 [Go, NOVY] *Verifier V is computationally sound for L, with negligible error-probability, if for every prover $P \in \mathcal{P}$ the function Acc_P is negligible.*

As a notion of security this seems satisfactory. But notice that this notion does not have the view of an error probability being a single function associated to the *verifier*. That is, I imagine the notion of error probability to be that given a verifier one can say "the system has error probability ϵ", where ϵ is some specific function. But in the notion of Definition 4.6, there is no such function. That is, to each prover is associated a different negligible function that bounds its success, but there is no single "error probability."

Indeed, Definition 4.6 is very specific to *negligible* error. It does not yield a general notion of error probability of an argument, or conform to such a notion. On the other hand, Definition 4.5 enables us to pinpoint a particular function as the error-probability and is in line with the general notion of error probability given in Definition 4.4.

EQUIVALENCE. However, it turns out the two notions are equivalent. In other words, even though Definition 4.6 did not have an explicit single function which one could call the error probability, such a function does exist. (And the proof of Lemma 3.3 shows how to construct it.)

Theorem 4.7 *Definitions 4.5 and 4.6 of computational soundness with negligible error-probability are equivalent.*

Proof: We observe that V meets Definition 4.5 if and only if the collection of functions $\mathcal{F}$ defined in Equation (7) is uniformly negligible, and V meets Definition 4.6 if and only if $\mathcal{F}$ is pointwise negligible. However we claim that $\mathcal{F}$ is pointwise negligible if and only if it is uniformly negligible. This says Definitions 4.5 and 4.6 of computational soundness with negligible error-probability are equivalent.

[5] In an interactive proof setting [GMR] we would say the error probability is ϵ if for all provers P and all $x \notin L$ the probability that P convinces V to accept x is at most $\epsilon(|x|)$. Replacing "all provers" by "all polynomial time provers" here does NOT yield a meaningful notion of error probability for an argument, because for any fixed x there may in fact be a prover who succeeds with probability more than $\epsilon(|x|)$, by solving the underlying computational problem. Hence the use of "eventually less than" in Definition 4.8.

To prove the claim, first take the case where the class $\mathcal{P}$ of provers considered is the class of uniform (probabilistic) polynomial time algorithms. Then $\mathcal{F}$ is countable and we can conclude by applying Theorem 3.2.

Now take the case where a prover P is a polynomial sized circuit family. (We allow a different circuit P_x for each $x \notin L$.) So $\mathcal{P}$, and hence $\mathcal{F}$ is now uncountable. If $\mathcal{F}$ is uniformly negligible then it is still pointwise negligible by Proposition 3.1. Now assume $\mathcal{F}$ is pointwise negligible. We want to apply Theorem 3.8 to say that $\mathcal{F}$ is uniformly negligible. To do so we must find a countable, pointwise negligible collection $\mathcal{M}$ that majors $\mathcal{F}$.

For any $x \notin L$ and any circuit C let $\mathrm{Accept}(C, x)$ denote the probability that V accepts on input x when C plays the role of the prover. For each $x \notin L$ and each integer k we can fix a size k circuit $\mathrm{BestProver}_{x,k}$ such that

$$\mathrm{Accept}(\mathrm{BestProver}_{x,k}) \; = \; \max_C \mathrm{Accept}(C, x) \; ,$$

the maximum here taken over all circuits of size at most k. Let $p_1, p_2, \ldots$ be an enumeration of all polynomials and let P^i be the circuit family in which $P^i_x = \mathrm{BestProver}_{x,p_i(|x|)}$ for all $i \in \mathsf{Z}_+$ and $x \notin L$. Let

$$M_i(n) \; = \; \max_{x \notin L : |x| = n} \mathrm{Accept}(\mathrm{BestProver}_{x,p_i(n)}) \; = \; \mathsf{Acc}_{P^i}(n) \; .$$

Let $\mathcal{M} = \{M_i\}_{i \in \mathsf{Z}_+}$.

We claim that $\mathcal{M}$ is countable, pointwise negligible, and majors $\mathcal{F}$. So $\mathcal{F}$ is uniformly negligible by Theorem 3.8, as desired. It remains to verify that $\mathcal{M}$ has the claimed properties. But the reasoning is analogous to that in the proof of Theorem 4.3 so we skip it. ▌

Thus the definitions of negligible error arguments of [Go, NOVY] and [BJY] are equivalent.

4.3 Negligible knowledge error

We are in the same setting of polynomially bounded parties as above, but the question now pertains to $x \in L$. We want to discuss the knowledge error of a computational proof of knowledge of a witness for the membership of $x \in L$.

Preliminaries. First, we need to say what are witnesses. Let $\rho(\cdot, \cdot)$ be a binary relation. We say that ρ is an NP-relation if it is polynomial time computable and, moreover, there exists a polynomial p such that $\rho(x, w) = 1$ implies $|w| \leq p(|x|)$. For any $x \in \{0,1\}^*$ we let $\rho(x) = \{ w \in \{0,1\}^* : \rho(x, w) = 1 \}$ denote the witness set of x. We let $L_\rho = \{ x \in \{0,1\}^* : \rho(x) \neq \emptyset \}$ denote the language defined by ρ. Note that a language L is in NP iff there exists an NP-relation ρ such that $L = L_\rho$.

Fix a verifier V and an NP-relation ρ defining $L = L_\rho$. As before let $\mathrm{Accept}_P(x)$ denote the probability that V accepts in a conversation with P on common input x. This time however we are interested in $x \in L$. (However, we are still only interested in the security. We do not state the completeness condition, which would say there is some prover who given a witness can always convince the verifier to accept in polynomial time.)

Below P_x denotes prover P with common input fixed to x. As in Section 4.2 let $\mathcal{P}$ denote the class of all provers being considered.

Computational knowledge error. The following general notion of a computational proof of knowledge with knowledge error a function κ is given in [BeGo].[6]

Definition 4.8 [BeGo] *We say that verifier V defines a computational proof of knowledge for NP-relation ρ, with knowledge-error κ, if there is an expected polynomial time oracle algorithm E (the extractor) such that for every prover $P \in \mathcal{P}$ there is a constant n_P such that*

$$\Pr[\, E^{P_x}(x) \in \rho(x)\,] \;\geq\; \mathrm{Accept}_P(x) - \kappa(|x|)$$

for all $x \in L_\rho$ which have length at least n_P.

In other words, if E has oracle access to P then it can output a witness for membership of x in L_ρ with a probability only slightly less than the probability that P would convince V to accept x.

Negligible knowledge error. Now, it seems natural to say that a computational proof of knowledge with negligible knowledge error is one that has knowledge error κ, in the above sense, for some negligible function κ. In other words:

Definition 4.9 [BeGo] *We say that verifier V defines a computational proof of knowledge for NP-relation ρ, with negligible knowledge-error, if there is an expected polynomial time oracle algorithm E (the extractor) and a negligible function κ such that for every prover $P \in \mathcal{P}$ there is a constant n_P such that*

$$\Pr[\, E^{P_x}(x) \in \rho(x)\,] \;\geq\; \mathrm{Accept}_P(x) - \kappa(|x|)$$

for all $x \in L_\rho$ which have length at least n_P.

An alternative is to allow the "knowledge error" to depend on the prover:

Definition 4.10 [BeGo] *We say that verifier V defines a computational proof of knowledge for NP-relation ρ, with negligible knowledge-error, if there is an expected polynomial time oracle algorithm E (the extractor) such that for each prover $P \in \mathcal{P}$ there is a negligible function κ_P such that*

$$\Pr[\, E^{P_x}(x) \in \rho(x)\,] \;\geq\; \mathrm{Accept}_P(x) - \kappa_P(|x|)$$

for all $x \in L_\rho$.

It is suggested in [BeGo] that Definition 4.9 is preferable to Definition 4.10. This does seem the case because, like (soundness) error, it is more natural to think of knowledge error as a single function associated to the verifier. And Definition 4.9 is consistent with a general notion of knowledge error given in Definition 4.8.

Equivalence. But it turns out the two definitions are equivalent anyway:

Theorem 4.11 *Definitions 4.9 and 4.10 of a computational proof of knowledge with negligible knowledge error are equivalent.*

Proof: In this case it may be a little less evident than before how the issue corresponds to negligibility of some collection of functions. We first claim something stronger than the theorem statement, namely that not only are the notions equivalent, but the extractor is the same in both cases. So view the extractor E as now fixed. Let $L = L_\rho$. For each prover $P \in \mathcal{P}$ define the function

$$F_P(n) \;=\; \max_{x \in L : |x| = n} \left(\mathrm{Accept}_P(x) - \Pr[\, E^{P_x}(x) \in \rho(x)\,] \right) .$$

We consider the collection of functions $\mathcal{F} = \{F_P\}_{P \in \mathcal{P}}$.

[6] They have two (equivalent) formulations of any of their definitions. The one we are using is what they call the "alternative form of validity."

Now we proceed as usual (cf. the proof of Theorem 4.7). First, observe that Definition 4.9 is met if and only if $\mathcal{F}$ is uniformly negligible, while Definition 4.10 is met if and only if $\mathcal{F}$ is pointwise negligible. We claim however that $\mathcal{F}$ is uniformly negligible if and only if it is pointwise negligible. To show this, we split the proof into two cases, the uniform and the non-uniform. For the first, we use Theorem 3.2. For the second, by defining the "best provers", we use Theorem 3.8. The details are omitted. ∎

ALTERNATIVE FORM. As we mentioned in a previous footnote, the proof of knowledge definitions of [BeGo] are formulated in two styles. To be concrete, take Definition 4.8. The other formulation asks that the extractor always succeed as long at $\mathrm{Accept}_P(x) > \kappa(|x|)$, but is allowed an expected running time of $n^{O(1)}/(\mathrm{Accept}_P(x) - \kappa(|x|))$. They show that the two formulations are equivalent. Definition 4.10 and Definition 4.9 have similar analogues.

However, we note that with the notion of negligible knowledge error of Definition 4.10, the equivalence seems hard to prove. The way we would prove it is by using the fact that Definitions 4.10 and 4.9 are equivalent. This indicates another "application" of this equivalence.

Acknowledgments

Thanks to Oded Goldreich for pointing out that Theorem 3.2 cannot be directly applied in the case of non-uniform adversaries because there are uncountably many of these, and suggesting how to overcome this difficulty. His solution is used in the proofs of the non-uniform cases. Thanks to Phillip Rogaway and Oded Goldreich for (independently) pointing to the connection with the notion of security of [Le, Lu].

References

[BeGo] M. BELLARE AND O. GOLDREICH. On Defining Proofs of Knowledge. *Advances in Cryptology – Crypto 92 Proceedings*, Lecture Notes in Computer Science Vol. 740, E. Brickell ed., Springer-Verlag, 1992.

[BJY] M. BELLARE, M. JAKOBSSON AND M. YUNG. Round-optimal zero-knowledge arguments based on any one-way function. *Advances in Cryptology – Eurocrypt 97 Proceedings*, Lecture Notes in Computer Science Vol. ??, W. Fumy ed., Springer-Verlag, 1997. Available at `http://www-cse.ucsd.edu/users/mihir`.

[BrCr] G. BRASSARD AND C. CRÉPEAU. Non-transitive Transfer of Confidence: A perfect Zero-knowledge Interactive protocol for SAT and Beyond. *Proceedings of the 27th Symposium on Foundations of Computer Science*, IEEE, 1986.

[BCC] G. BRASSARD, D. CHAUM AND C. CRÉPEAU. Minimum Disclosure Proofs of Knowledge. *J. Computer and System Sciences*, Vol. 37, 1988, pp. 156–189.

[Go] O. GOLDREICH. Foundations of cryptography: Fragments of a book. Weizmann Institute of Science, February 1995.

[GMR] S. GOLDWASSER, S. MICALI AND C. RACKOFF. The knowledge complexity of interactive proof systems. *SIAM J. on Computing*, Vol. 18, No. 1, pp. 186–208, February 1989.

[Le] L. LEVIN. One-way functions and pseudorandom generators. *Combinatorica*, Vol. 7, No. 4, 1987, pp. 357–363.

[Lu] M. Luby. Pseudorandomness and cryptographic applications. Princeton Computer Science Notes, Princeton University Press, 1996.

[NOVY] M. Naor, R. Ostrovsky, R. Venkatasan, M. Yung. Perfect zero knowledge arguments for NP can be based on general complexity assumptions. *Advances in Cryptology – Crypto 92 Proceedings*, Lecture Notes in Computer Science Vol. 740, E. Brickell ed., Springer-Verlag, 1992.