Query-Complexity Amplification for Random Oracles*

Grégory Demay[1], Peter Gaži[†,2], Ueli Maurer[1], and Björn Tackmann[†,3]

[1]Department of Computer Science, ETH Zürich, Switzerland
{gregory.demay,maurer}@inf.ethz.ch
[2]Institute of Science and Technology, Austria
peter.gazi@ist.ac.at
[3]Computer Science & Engineering, University of California San Diego
btackmann@eng.ucsd.edu

April 13, 2015

Abstract

Increasing the computational complexity of evaluating a hash function, both for the honest users as well as for an adversary, is a useful technique employed for example in password-based cryptographic schemes to impede brute-force attacks, and also in so-called proofs of work (used in protocols like Bitcoin) to show that a certain amount of computation was performed by a legitimate user. A natural approach to adjust the complexity of a hash function is to iterate it c times, for some parameter c, in the hope that any query to the scheme requires c evaluations of the underlying hash function. However, results by Dodis et al. (Crypto 2012) imply that plain iteration falls short of achieving this goal, and designing schemes which provably have such a desirable property remained an open problem.

This paper formalizes explicitly what it means for a given scheme to amplify the query complexity of a hash function. In the random oracle model, the goal of a secure query-complexity amplifier (QCA) scheme is captured as transforming, in the sense of indifferentiability, a random oracle allowing R queries (for the adversary) into one provably allowing only $r < R$ queries. Turned around, this means that making r queries to the scheme requires at least R queries to the actual random oracle. Second, a new scheme, called collision-free iteration, is proposed and proven to achieve c-fold QCA for both the honest parties and the adversary, for any fixed parameter c.

Keywords: hash functions, random oracle, indifferentiability, moderately hard functions

*A preliminary version of this paper appears in the proceedings of ICITS 2015. This is the full version.
†Work partially done while author was at ETH Zürich.

Contents

1 Introduction

1.1 Motivation of This Work

Moderately hard hashing. Hash functions are one of the most basic and widely used building blocks in practically deployed cryptographic protocols. Their use in different contexts puts diverse requirements on their properties. There is a vast body of literature exploring various desirable properties of cryptographic hash functions such as collision resistance, (second-) preimage resistance, indifferentiability from a random oracle, and several others.

A seemingly orthogonal property of a hash function is its efficiency — quantified by the amount of computational resources that are required to evaluate it. Naturally, the typical design goal is to provide hash functions that are as efficient as possible, while still maintaining the desired security requirements mentioned above. As a result of the long-term design effort with this motivation, the currently standardized and used cryptographic hash functions such as SHA-1, SHA-2 [SHA12] and SHA-3 [SHA14] are extremely efficient: for example, a software implementation of SHA-2 can process data at (very roughly) about 100 MB/s on a typical PC.

However, in several application scenarios the efficiency of the hash function actually has serious *security* implications, and these motivate design efforts going in the opposite direction. Namely, sometimes hash functions are used to perform computation by the honest parties that would need to be repeated on a significantly higher scale by an adversary trying to compromise the security of the system. One example of such a setting is any non-interactive password-based scheme where the hash function is used to, say, derive a key from this password. Here, increasing the complexity of the hash-function evaluation, while slightly increasing the computational burden for the honest user, also significantly increases the cost of a brute-force and password-guessing (dictionary) attack. Another setting that could benefit from an adjustable complexity of a hash function is a proof of work [DN93] where a legitimate protocol participant shows that he performed a certain amount of computation. This concept was proposed, among other uses, as a countermeasure against denial-of-service attacks or junk mail. Similar ideas are used in the now widely used Bitcoin system [Nak08] and other cryptocurrencies basing their security on proofs of work.

The common denominator of all the settings mentioned above is that it would be desirable to employ hash functions that are, loosely speaking, *moderately hard* to compute [Nao03]. While the occasional evaluation of such a function by an honest user needs to still remain feasible, at the same time the scaling resulting from a brute-force attack must be prohibitive for any adversary.

Complexity amplification. Since designing new cryptographic hash functions from scratch is a long and intricate process (e.g., the SHA-3 competition spanned over almost 5 years), to answer the above-described demand it would be preferable to give *generic schemes* that would instead turn an existing hash function h into a new function H with moderately increased evaluation complexity. A natural first candidate for such a scheme is the simple *c-iteration* (or plain iteration), i.e., letting

$$H(\cdot) := h^c(\cdot) := \underbrace{h(\ldots h(\cdot)\ldots)}_{c \text{ times}}$$

for some integer $c > 1$.

Indeed, many password-hashing schemes are based on some form of iteration. Historically, the earliest implementations of crypt(3) used several iterations of (a variant of) the block cipher DES to hash users' passwords on Unix systems [MT79], the more recent bcrypt [PM99] iterates the block cipher Blowfish instead. Iteration is also used in the password-based key derivation function PBKDF2 standardized in PKCS#5 [Kal00] and recommended by NIST [TBBC10].

However, when it comes to assessing the security of any such generic scheme for increasing evaluation complexity (for example to justify the choice of plain iteration), it turns out that

merely *defining* the security requirement formally is a surprisingly subtle task. This is especially true if one asks for a composable definition that then allows every scheme secure under this definition to be plugged into any possible application, so that proving a scheme secure according to this single definition immediately implies that it can be used in, e.g., key derivation, proofs of work, or other applications. One of our main contributions will be to give such a composable definition by modeling the underlying hash function h as a random oracle and exploiting the well-established notion of indifferentiability. Before inspecting it in greater detail, let us first mention a surprising observation given in recent work that is very relevant in our context.

The caveats of plain iteration. Dodis et al. [DRST12] studied the structural differences between a random oracle and its second iterate: more precisely, they investigated the indifferentiability of the 2-iteration of a random oracle from a plain random oracle. Interestingly, they showed that such indifferentiability *does* hold, but only with poor parameters. Namely (and very roughly), any simulator in this indifferentiability statement, if asked r queries during the distinguishing experiment, would itself have to issue a large number of queries $\Omega(\ell r)$ to the underlying random oracle in order to succeed in simulation, where ℓ denotes (an upper bound on) the number of honest queries. (We show in Section 4 that the result extends to higher-order iterates.) On a high level, this large number of simulator queries means that if one uses the c-iterate of a hash function in some application, then the *concrete* security statement obtained through the composition theorem of indifferentiability is weaker than intuitively expected. Therefore, any strong security guarantee could only be obtained through an ad-hoc security analysis depending on the particular scenario considered, as done by Bellare et al. [BRT12].

Let us recall an example of Dodis et al. [DRST12] to illustrate this last point. In the hash-then-sign paradigm, a signature scheme SS_n signing n-bit messages and a hash function $h : \{0,1\}^* \to \{0,1\}^n$ are combined into a signature scheme $\mathrm{SS}_*(h)$ for arbitrary length messages by signing the hash $h(m)$ of the message instead of the message m itself. Forging a signature for the extended scheme $\mathrm{SS}_*(h)$ requires either to find a collision for the hash function h or to find a forgery for the original fixed-length signature scheme SS_n. If the hash function h is modelled as a random oracle, then its second iterate h^2 is indifferentiable from h [DRST12, Thm. 2], and the composition theorem of indifferentiability [MRH04, RSS11] implies that the security of $\mathrm{SS}_*(h^2)$ can be reduced to that of $\mathrm{SS}_*(h)$. However, such a reductionist argument, which is standard in any composable cryptographic framework such as indifferentiability, consists of obtaining an adversary against $\mathrm{SS}_*(h)$ from an adversary against $\mathrm{SS}_*(h^2)$ which *additionally* performs the job of the simulator given in the indifferentiability statement. Due to the blow-up in simulator queries mentioned above, this concretely means that one relates an adversary trying to forge a signature for $\mathrm{SS}_*(h^2)$ with at most ℓ signing queries and r random oracle queries, to an adversary trying to forge a signature for $\mathrm{SS}_*(h)$ also with ℓ signing queries, but with $\ell \cdot r$ random oracle queries. Thus, although $\mathrm{SS}_*(h)$ is secure as long as the collision probability $(\ell + r)^2/2^n$ is sufficiently small (assuming that the original length-restricted signature scheme SS_n is secure within ℓ queries), the security of $\mathrm{SS}_*(h^2)$ derived through composition depends instead on the much higher collision probability $(\ell \cdot r)^2/2^n$, representing a quadratic decrease of security.

1.2 Contributions of This Paper

In this work, we develop a new formal framework for treating the amplification of the evaluation complexity for random oracles (which are often used to model hash functions in practical scenarios). We first develop a security definition that tightly captures how well a given scheme increases the computational burden for an adversary in evaluating the function. Being based on indifferentiability, our definition is naturally composable and hence guarantees the desired universal applicability of any scheme meeting it. Secondly, guided by the observations of Dodis et al. [DRST12] about the second iterate, we show that plain iteration, regardless of the num-

ber of iterations employed, fails to achieve the amplification of the hash-function complexity in the above sense. In response, we develop a modification of the plain-iteration scheme, called collision-free iteration, that does provably and generically achieve the desired amplification. Let us now discuss the details of each of these contributions.

Composable security for hash-complexity amplification. Employing the random oracle model (ROM) [BR93], we model hash functions as random oracles. A random oracle can be viewed as a resource that is available to all parties in a given setting, and allows each of them to evaluate the oracle by querying it—this corresponds to the party internally computing the output of the hash function. A restriction on the computational resources of the adversary hence naturally translates to a restriction on the number of queries it is allowed to ask the random oracle. In a typical security proof in the ROM, one establishes that the scheme in question is secure unless the ROM-adversary performs a huge number of queries to the random oracle. This then suggests that the adversary against the real implementation has to evaluate the hash function on a prohibitive number of inputs. Following this intuition, we model the increase in evaluation complexity of a hash function by a decrease in the number of queries that the adversary is allowed to issue to the random oracle (before its computational resources are exhausted).

As a starting point, we make explicit the number of queries that such an oracle allows to each party: for two integers L and R, a random oracle that allows up to L queries at the left (honest user's) interface and up to R queries at the right (adversary's) interface formalizes the guarantee that the honest user has sufficient resources to evaluate the hash function (at least) L times, whereas the resources of the adversary are bounded to (at most) R evaluations. Naturally, a desirable guarantee for the honest user is that the number L is large enough to execute higher-level protocols, whereas the number R must be small enough to prevent the adversary from attacking those protocols with significant probability. The goal of a protocol for the *amplification of query complexity* is hence to reduce the number R, while at the same time not affecting the number L more than necessary.

Following the paradigm of constructive cryptography [MR11], we understand a cryptographic protocol or scheme as a way to *construct*, in a well-defined sense, a certain desired resource from one or more assumed resources. In the context of query-complexity amplification (QCA), this means that the goal is to construct, from a random oracle that allows the adversary to do some number R of queries, a random oracle that allows the adversary only a smaller number $r < R$ of queries. Intuitively, such a construction means that an adversary with the same computational resources can evaluate the random oracle less often, which will generally reduce his success in attacking higher-level protocols.

This constructive way of stating security definitions comes with a natural notion of composition. Denoting the statement that a protocol π constructs the desired resource $\mathbf{S}$ from the assumed resource $\mathbf{R}$ as $\mathbf{R} \xrightarrow{\pi} \mathbf{S}$, any two such construction steps that "syntactically" match can be composed: If we consider another protocol ψ that assumes the resource $\mathbf{S}$ and constructs a resource $\mathbf{T}$, the composition theorem immediately implies that

$$\mathbf{R} \xrightarrow{\pi} \mathbf{S} \quad \wedge \quad \mathbf{S} \xrightarrow{\psi} \mathbf{T} \quad \Longrightarrow \quad \mathbf{R} \xrightarrow{\psi \circ \pi} \mathbf{T},$$

where $\psi \circ \pi$ denotes the composed protocol. For example, let π in the above represent a protocol for hash-complexity amplification that is capable of transforming a random oracle $\mathbf{R}$ that can be evaluated R times within the adversary's resources into a ("much harder") random oracle $\mathbf{S}$ that the adversary can only evaluate $r \ll R$ times. Then, for *any* higher-level construction ψ of some useful resource $\mathbf{T}$ that uses an underlying random oracle $\mathbf{S}$ and guarantees that $\mathbf{T}$ will be secure as long as the adversary is not capable of evaluating S more than r times, we can instead start from the oracle $\mathbf{R}$ and amplify its complexity using π before using it to construct $\mathbf{T}$. The security will not be compromised by this as long as the adversary cannot evaluate $\mathbf{R}$

more than R times; and this guarantee then heuristically translates into the setting where we use an efficient hash function instead of $\mathbf{R}$.

Finally, while aiming for a formalization of hash-complexity amplification, we also arrive at a new formalism of parameterized construction statements, as detailed in Section 3. We believe that this formalism will be useful also in many other settings, such as secure communication as discussed in [Tac14], and consider it an additional contribution of this paper of independent interest.

A scheme for hash-complexity amplification. As our second contribution, we present a simple scheme, called collision-free iteration, that achieves query-complexity amplification in the sense of our new definition discussed above.

One would naturally expect that the c-iteration of a random oracle for some $c \geq 2$ would lead to a reduction of adversary queries from R to R/c, at the cost of simultaneously reducing the honest party's queries from L to L/c. However, we show in Section 4 that c-iteration, much like the second iterate studied by Dodis et al. [DRST12], suffers from the blow-up in the number of simulator queries and therefore falls short of achieving this goal.

We show that modifying the c-iterate of a random oracle by a proper encoding of the queries will indeed lead to the desired (and expected) result. The high-level idea is to make sure that each query will access a distinct part of the random oracle and hence the "shifted chains" of queries that caused problems for the plain iteration will not occur. In greater detail, collision-free iteration works almost like the plain iteration, but each query to the underlying function $h(\cdot)$ during the computation of $H(x)$ is prefixed by a prefix-free encoding $\lfloor x \rceil$ of the original query x, as well as the sequence number within the iterative process. Formally, we define $W_0(x)$ to be the empty string and

$$W_j(x) := h(\lfloor x \rceil \,\|\langle j \rangle \,\|\, W_{j-1}(x)) \;\text{ for all } j \in \{1, \ldots, c\}\ ,$$

where $\lfloor \cdot \rceil$ and $\langle \cdot \rangle$ denote a prefix-free encoding and an injective encoding of an integer over $\lceil \log c \rceil$ bits, respectively. Finally, we simply let $H(x) := W_c(x)$. We prove in Section 5 that this construction reduces the number of adversary queries from R to R/c, at the cost of simultaneously reducing the honest party's queries from L to L/c.

Towards proving optimality. In Section 6 we study whether this simultaneous reduction of the honest-party queries is inherent to any query-complexity amplification scheme. Based on the observation that the adversary can always choose to evaluate the honest scheme, we can show that our construction, which reduces the adversary's queries exactly as much as the honest party's queries, is optimal with respect to a natural, albeit restricted, class of simulators.

We aimed for simplicity in the design of our construction and did not tailor it to minimize query lengths. In particular, extending the length of each subquery by the length of $\lfloor x \rceil$ is most likely not necessary. We leave the question of improving the lengths of the honest-user queries open for future work.

2 Preliminaries

2.1 Basic Notation

We denote sets by calligraphic letters or capital Greek letters (e.g., $\mathcal{X}$, Σ). Throughout this paper, we consider only discrete random variables. A discrete random variable will be denoted by an upper-case letter X, its range by the corresponding calligraphic letter $\mathcal{X}$, and a realization of the random variable X will be denoted by the corresponding lower-case letter x. Unless stated otherwise, $X \xleftarrow{\$} \mathcal{X}$ denotes a random variable X selected independently and uniformly at random in $\mathcal{X}$. A tuple of n random variables $(X_1, \ldots, X_n)$ is denoted by X^n. Similarly, x^n

denotes a tuple of n values $(x_1, \ldots, x_n)$. The set of bit strings of finite length is denoted $\{0,1\}^*$ and λ denotes the empty bit string.

2.2 Random Systems

Many cryptographic primitives like block ciphers, MAC schemes, random functions, etc., can be described as $(\mathcal{X}, \mathcal{Y})$-random systems [Mau02] taking inputs $X_1, X_2, \ldots \in \mathcal{X}$ and generating for each input X_k an output $Y_k \in \mathcal{Y}$. In full generality, such an output Y_k depends probabilistically on all the previous inputs X^k as well as all the previous outputs Y^{k-1}. For an $(\mathcal{X}, \mathcal{Y})$-random system $\mathbf{S}$, such a dependency is captured by a (possibly infinite) sequence of functions $\mathsf{p}^{\mathbf{S}}_{Y_k|X^kY^{k-1}} : \mathcal{Y} \times \mathcal{X}^k \times \mathcal{Y}^{k-1} \to [0,1]$ such that for all choices of the arguments x^k and y^{k-1} the sum of the function values over the choices of y_k equals 1, and where the superscript indicates the considered system. Random systems are usually denoted by upper-case boldface letters such as $\mathbf{R}$ or $\mathbf{S}$. An $(\mathcal{X}, \mathcal{Y})$-random system $\mathbf{S}$ considered in isolation does not define a random experiment since the distribution of the inputs to the system $\mathbf{S}$ is not defined. For this reason, the function $\mathsf{p}^{\mathbf{S}}_{Y_k|X^kY^{k-1}}$, which is called a *conditional* probability distribution, is denoted by a lower-case letter p instead of an upper-case letter P, which we use for probability distributions in a fully specified random experiment.

A random system $\mathbf{S}$ can alternatively be described by the sequence of conditional distributions $\mathsf{p}^{\mathbf{S}}_{Y^k|X^k}$, where $\mathsf{p}^{\mathbf{S}}_{Y^k|X^k} := \prod_{j=1}^{k} \mathsf{p}^{\mathbf{S}}_{Y_j|X^jY^{j-1}}$. Note that the conditional distribution $\mathsf{p}^{\mathbf{S}}_{Y^k|X^k}$ contains the conditional distribution $\mathsf{p}^{\mathbf{S}}_{Y^j|X^j}$ for all $j < k$ and hence the above description of a system is redundant. The conditional distribution $\mathsf{p}^{\mathbf{S}}_{Y^k|X^k}$ must satisfy a consistency condition which ensures that Y_j does not depend on $X_{j+1}, \ldots, X_k$. Two random systems $\mathbf{R}$ and $\mathbf{S}$ are said to be *equivalent*, denoted $\mathbf{R} \equiv \mathbf{S}$, if they behave identically, i.e., $\mathsf{p}^{\mathbf{R}}_{Y^k|X^k} = \mathsf{p}^{\mathbf{S}}_{Y^k|X^k}$, for all $k \geq 1$.

Distinguishers and a distance measure on random systems. A natural notion of similarity for random systems can be based on the concept of *distinguishers*. Intuitively, a distinguisher can be viewed as a system that connects to a random system, interacts with this system, and at some point outputs a single bit. In the case of $(\mathcal{X}, \mathcal{Y})$-random systems, a distinguisher $\mathbf{D}$ that makes some arbitrary but fixed number $q \in \mathbb{N}$ of queries corresponds to a finite $(\mathcal{Y}, \mathcal{X})$-random system which is one query ahead [MPR07], i.e., distributions $\mathsf{p}^{\mathbf{D}}_{X_i|Y^{i-1}X^{i-1}}$ for $i \in \{1, \ldots, q\}$, and an additional distribution $\mathsf{p}^{\mathbf{D}}_{Z|Y^qX^q}$. The distinguisher interacts with an $(\mathcal{X}, \mathcal{Y})$-random system $\mathbf{R}$ by providing inputs $X_1, X_2, \ldots \in \mathcal{X}$ to $\mathbf{R}$ and by receiving its corresponding outputs $Y_1, Y_2, \ldots \in \mathcal{Y}$. Connecting a distinguisher $\mathbf{D}$ to an $(\mathcal{X}, \mathcal{Y})$-random system $\mathbf{R}$ defines a binary random variable (the output bit Z of the distinguisher), denoted $\mathbf{DR}$. For two $(\mathcal{X}, \mathcal{Y})$-random systems $\mathbf{R}$ and $\mathbf{S}$, the distinguishing advantage of a distinguisher $\mathbf{D}$ in telling apart $\mathbf{R}$ from $\mathbf{S}$ is then defined as

$$\Delta^{\mathbf{D}}(\mathbf{R}, \mathbf{S}) \quad := \quad \left|\mathsf{P}(\mathbf{DR} = 1) - \mathsf{P}(\mathbf{DS} = 1)\right| \ .$$

For a class $\mathcal{D}$ of distinguishers, we define $\Delta^{\mathcal{D}}(\mathbf{R}, \mathbf{S}) := \sup_{\mathbf{D} \in \mathcal{D}} \Delta^{\mathbf{D}}(\mathbf{R}, \mathbf{S})$. (The only classes we are interested in are the class of *all* distinguishers, in which case we omit the superscript and write $\Delta(\mathbf{R}, \mathbf{S})$, and the class NA of all *non-adaptive* distinguishers.)

Games. A central tool in deriving an indistinguishability proof between two systems is to characterize both systems as being equivalent until a certain condition arises [Mau02, BR06]. Thus, being able to distinguish both systems requires to provoke this condition, and one is then interested in upper-bounding the probability of this event. Interacting with a random system in order to provoke a certain condition is naturally modeled by defining an additional *monotone binary output (MBO)* on the original system, where the binary output is monotone in the sense that it is initially set to 0 and that, once it has turned to 1, it can not turn back to 0. An

$(\mathcal{X}, \mathcal{Y} \times \{0,1\})$-system where the second output component is monotone is often indicated by using a system symbol with a hat, such as $\widehat{\mathbf{R}}$.

For an $(\mathcal{X}, \mathcal{Y} \times \{0,1\})$-system $\widehat{\mathbf{R}}$ with an MBO, we consider two particular $(\mathcal{X}, \mathcal{Y})$-systems which are derived from $\widehat{\mathbf{R}}$, following Maurer et al. [MPR07]:

1. $\widehat{\mathbf{R}}^{-}$ is the $(\mathcal{X}, \mathcal{Y})$-system obtained from $\widehat{\mathbf{R}}$ by ignoring the MBO, we usually refer to this system as $\mathbf{R}$ (i.e., we simply omit the hat);
2. $\widehat{\mathbf{R}}^{\dashv}$ is the $(\mathcal{X}, \mathcal{Y} \cup \{\diamond\})$-system which masks the $\mathcal{Y}$-output to a dummy symbol $\diamond \notin \mathcal{Y}$ as soon as the MBO turns 1, and in addition, it does not output the MBO itself.[1]

We will alternatively refer to an $(\mathcal{X}, \mathcal{Y} \times \{0,1\})$-random system $\widehat{\mathbf{R}}$ with an MBO as an $(\mathcal{X}, \mathcal{Y})$-*game*, in particular if we are interested in the probability with which the MBO can be provoked. More formally, we are then interested in the probability that some $(\mathcal{X}, \mathcal{Y})$-game winner $\mathbf{W}$ (which, like a distinguisher, can be viewed as a finite $(\mathcal{Y}, \mathcal{X})$-random system that is one query ahead) provokes the MBO of a game $\widehat{\mathbf{R}}$ to be 1. As in a distinguishing experiment, the game winner $\mathbf{W}$ and the game $\widehat{\mathbf{R}}$ define a binary random variable, the value of the MBO of $\widehat{\mathbf{R}}$ after $\mathbf{W}$ stops, which we denote as $\mathbf{W}\widehat{\mathbf{R}}$. Hence, the *winning probability of* $\mathbf{W}$ *in the game* $\widehat{\mathbf{R}}$ is defined as

$$\Gamma^{\mathbf{W}}(\widehat{\mathbf{R}}) \quad := \quad \Pr[\mathbf{W}\widehat{\mathbf{R}} = 1] \ .$$

Similarly to $\Delta^{\mathcal{D}}$, the supremum of $\Gamma^{\mathbf{D}}(\widehat{\mathbf{R}})$ over $\mathcal{D}$ is denoted $\Gamma^{\mathcal{D}}(\widehat{\mathbf{R}})$.

Restricted systems and games. The concept of a blocked system $\widehat{\mathbf{R}}^{\dashv}$, derived from a given system $\widehat{\mathbf{R}}$ with MBO, is particularly useful if $\widehat{\mathbf{R}}$ is in turn derived from some underlying system $\mathbf{R}$ (i.e., $\widehat{\mathbf{R}}^{-} = \mathbf{R}$, where $\mathbf{R}$ is of interest to us) by adding an MBO representing some *restriction* on $\mathbf{R}$ (e.g., an upper bound on the number of queries than can be made to this system). In this case, the *restricted distinguishing advantage* of a distinguisher $\mathbf{D}$ in distinguishing the two systems with MBO $\widehat{\mathbf{R}}$ and $\widehat{\mathbf{S}}$ is defined as

$$\widehat{\Delta}^{\mathbf{D}}(\widehat{\mathbf{R}}, \widehat{\mathbf{S}}) \quad := \quad \Delta^{\mathbf{D}}(\widehat{\mathbf{R}}^{\dashv}, \widehat{\mathbf{S}}^{\dashv}) \ . \tag{1}$$

The concept of restricting a system via an additional MBO can also be applied to the case of games and game winning. In such a case, we consider a system restricted by some MBO $A_1, A_2, \ldots$ with an additional MBO $B_1, B_2, \ldots$ specifying when the game is won. Formally, this is an $(\mathcal{X}, \mathcal{Y} \times \{0,1\} \times \{0,1\})$-random system $\mathbf{R}$, where the outputs are triples (Y_j, A_j, B_j) and the latter two components are monotone. Then, we can consider the task of winning the restricted game, i.e., provoking the event modelled by the MBO $B_1, B_2, \ldots$ *before* violating the restriction modelled by the MBO $A_1, A_2, \ldots$, as the task of winning the game with the MBO $C_1, C_2, \ldots$ with $C_j = C_{j-1} \vee (\neg A_j \wedge B_j)$. Denoting the system with the single MBO $C_1, C_2, \ldots$ as $\mathbf{R}^{<}$, we define the restricted game-winning advantage as

$$\widehat{\Gamma}^{\mathbf{W}}(\mathbf{R}) \quad := \quad \Gamma^{\mathbf{W}}(\mathbf{R}^{<}) \ .$$

Conditional equivalence. The notion of *conditional equivalence* has been introduced by Maurer [Mau02, Mau13] and is a useful tool in deriving indistinguishability proofs. An $(\mathcal{X}, \mathcal{Y})$-game $\widehat{\mathbf{R}}$ with MBO $B_1, B_2, \ldots$ is said to be *conditionally equivalent* to an $(\mathcal{X}, \mathcal{Y})$-random system $\mathbf{S}$, denoted $\widehat{\mathbf{R}} \models \mathbf{S}$, if $\mathsf{p}^{\widehat{\mathbf{R}}}_{Y^j|X^jB_j=0} = \mathsf{p}^{\mathbf{S}}_{Y^j|X^j}$, for all $j \geq 1$ and for all arguments for which $\mathsf{p}^{\widehat{\mathbf{R}}}_{Y^j|X^jB_j=0}$ is defined. If a game $\widehat{\mathbf{R}}$ is conditionally equivalent to a system $\mathbf{S}$, then the distinguishing advantage between the systems $\mathbf{R}$ and $\mathbf{S}$ is upper bounded by the probability of winning the game $\widehat{\mathbf{R}}$ in a *non-adaptive* manner, a statement which was first presented by Maurer [Mau02] and was studied more extensively later by Jetchev et al. [JÖS12] and Maurer [Mau13].

[1] This definition deviates from the one used by Maurer et al. [MPR07], where the MBO is still output by $\widehat{\mathbf{R}}^{\dashv}$. The difference between the definitions is irrelevant because the output is $\diamond$ if and only if the MBO is 1.

2.3 Two-Interface Systems and Converters

Two-interface systems. Systems that can be accessed by multiple parties can be viewed as systems with multiple interfaces and formalized as random systems by making the interface identifier an explicit part of the input (or output) of the system. In this work, we focus on systems with two interfaces, which we naturally label by elements of the set $\mathcal{I} := \{\mathsf{left}, \mathsf{right}\}$.

We restrict our considerations to the particular class of two-interface systems that only produce an output (from some set $\mathcal{Y}$) in response to an input (from $\mathcal{X}$) and on the same interface where the input was received, and hence we omit the interface label from the output. Then, such a two-interface system $\mathbf{S}$ takes as input a pair $(I_k, X_k) \in \mathcal{I} \times \mathcal{X}$, where the k^{th} query X_k was input at the I_k-interface, and produces as output $Y_k \in \mathcal{Y}$, where it is understood that the response Y_k of the system $\mathbf{S}$ is output at the same interface I_k that the query X_k was input. In other words, a two-interface system corresponds (due to our restrictions) to an $(\mathcal{I} \times \mathcal{X}, \mathcal{Y})$-random system and can be described by a sequence of conditional probability distributions $\mathsf{p}^{\mathbf{S}}_{Y_k|I^k X^k Y^{k-1}}$, $k \geq 1$. Moreover, we will usually consider two-interface systems which have an additional MBO, this is defined exactly as above and will be used to restrict the access of the distinguisher as in equation (1).

In this work, we focus on variants of the arbitrary input-length random oracle $\mathbf{RO}$ with output length n, which we understand as two-interface systems with one interface for the honest party and one interface for the adversary, and which are thus formally seen as $(\mathcal{I} \times \{0,1\}^*, \{0,1\}^n)$-random systems.

Converters. Strategies employed locally by a party are modeled by a *converter*[2], which can also be viewed as a system with two interfaces: an *inside* interface and an *outside* interface, denoted by in and out, respectively. In this view, the inside interface is attached to the i-interface of a resource and models how the scheme makes use of this resource, where $i \in \mathcal{I}$, while the outside interface of the converter becomes the i-interface of the composite system and models how the scheme can be used in applications and higher-level protocols.

We consider that a converter is always invoked by queries $X_1, X_2, \ldots \in \mathcal{X}$ at the out-interface. For each such query, it (adaptively) makes zero or more[3] queries $X'_1, \ldots, X'_{j_1}$ (resp., $X'_{j_1+1}, \ldots, X'_{j_2}$ etc.) at the inside interface, i.e., to the two-interface system whose i-interface is attached to the in-interface of the converter. After having received the corresponding answers $Y'_1, \ldots, Y'_{j_1}$ (resp., $Y'_{j_1+1}, \ldots, Y'_{j_2}$ etc.), it finally produces an output $Y_1 \in \mathcal{Y}$ (resp., Y_2 etc.) at the out-interface. As it is always clear at which interface the input to the converter is obtained (it is the same interface where the converter produced the last output), it need not be explicitly specified. Finally, we will usually consider converters which have an additional MBO, also for the purpose of restricting the distinguisher's access. Summarizing the above, such a converter can be formalized as a $(\mathcal{X} \cup \mathcal{Y}, (\{\mathsf{out}\} \times \mathcal{Y}) \cup (\{\mathsf{in}\} \times \mathcal{X})) \times \{0,1\})$-random system.

Attaching a converter to the i-interface of a two-interface system with label set $\mathcal{I}$, where $i \in \mathcal{I}$, results in a two-interface system that can be described as follows.[4] Inputs to interfaces $i' \neq i$ are processed by the system as before. Whenever an input is given to the i-interface of the combined system, the converter is evaluated on this input. If the output of the converter (without the MBO) is of the form (in, x) for some $x \in \mathcal{X}$, the resource is evaluated on (i, x) and provides an output $y \in \mathcal{Y}$ (and an MBO). Then, the converter is evaluated on y. This process continues until the output of the converter is of the form (out, y') for some $y' \in \mathcal{Y}$, and this value y' is then considered the output of the composed system. This process leads to a well-defined random system because the number of inside queries is bounded for each query to the random

[2]We use the term converter here although it is only fully appropriate once we consider the object within a cryptographic algebra [MR11].

[3]We assume that, for each converter, there is some (constant) upper bound on the number of inside queries it makes per outside query.

[4]The described process can be written as a closed formula to formally obtain a random system.

system. The MBO of the overall system is defined to be the disjunction of the MBOs of the two-interface system and the converter.

Converters are denoted by lower-case Greek letters (e.g., α, π, σ) or by sans-serif fonts (e.g., amp_c). The set of all converters is denoted as Σ. To denote the composition of converters and two-interface systems, we will understand the left and the right side of the symbol $\mathbf{R}$ as representing the left- and right-interface of the system $\mathbf{R}$, respectively. Hence, attaching a converter π to the left-interface of a two-interface system $\mathbf{R}$ results in a two-interface system $\pi\mathbf{R}$ while attaching a converter σ to the right-interface of a two-interface system $\mathbf{S}$ results in a two-interface system $\mathbf{S}\sigma$.

2.4 Indifferentiability

Indifferentiability was introduced by Maurer et al. [MRH04] as a generalization of indistinguishability for settings where some access to the internal state of the considered resources is available publicly, within reach of any potential adversary. In such a scenario, the left-interface of a two-interface system $\mathbf{R}$ models interaction with honest users and is referred to as the "private" interface, while the right-interface formalizes adversarial access and is referred to as the "public" interface. For a protocol $\pi \in \Sigma$ and $\varepsilon \in [0, 1]$, the system $\pi\mathbf{R}$ is said to be (strongly) ε-indifferentiable from the system $\mathbf{S}$ if there exists a converter $\sigma \in \Sigma$ such that $\Delta^{\mathbf{D}}(\pi\mathbf{R}, \mathbf{S}\sigma) \leq \varepsilon$ for all distinguishers $\mathbf{D} \in \mathcal{D}$. We usually refer to the converter σ as the *simulator*. Indifferentiability has been widely applied, especially in the context of hash functions [CDMP05, BDPVA08] and reductions among idealized primitives [HKT11].

3 Parameterized Constructions and QCA

As outlined in Section 1, we formalize query-complexity amplification as a construction of random oracles which only allow for a limited number of queries from random oracles which allow more queries, both at the (honest user's) left and at the (adversary's) right interface. That is, we consider a random oracle as a resource, and the "quality" of a certain QCA scheme will be captured by the translation of restrictions (in the numbers of queries) that it achieves at both the honest and the adversarial interface. In this section we formalize the above intuition.

Query-restricted systems. We are interested in two-interface systems that only allow a certain number of queries that can be made to their left- or right-interface.[5] This is formalized by extending the considered system $\mathbf{R}$ with an MBO that captures when the system is exhausted. Notationally, for some integers $L, R \in \mathbb{N}$, we denote by $\mathbf{S}^{|R}$ the system $\mathbf{S}$ with an MBO that becomes 1 as soon as more than R queries have been made at the right-interface of the system $\mathbf{S}$, and similarly ${}^{L|}\mathbf{S}$ denotes the system $\mathbf{S}$ with an MBO that becomes 1 as soon as more than L queries have been made at the left-interface of $\mathbf{S}$. If a system has both types of restrictions, we consider the MBO which is the disjunction of the two individual MBOs described above, i.e., ${}^{L|}\mathbf{S}^{|R}$ denotes the restricted system allowing at most L queries at the left-interface *and* at most R queries at the right-interface. We use the same notation for restricting the number of queries at the outside interface of a converter (i.e., we write ${}^{L|}\alpha$ for $\alpha \in \Sigma$ and $L \in \mathbb{N}$), and it is easy to see that for a converter α and a system $\mathbf{S}$ we have ${}^{L|}(\alpha\mathbf{S}) \equiv ({}^{L|}\alpha)\mathbf{S}$ and hence we typically drop the parentheses.

Parameterized families of construction statements. We recall the definition of a construction statement for the case where there is only a single (external) adversary as described

[5] In contrast to most other definitional approaches, we restrict the number of queries in a distinguishing experiment by restricting the *system*, not the *distinguisher*.

originally by Maurer et al. [MR11, Mau12, MT10].[6] This construction notion, specialized to two-interface resources, is equivalent to (strong) indifferentiability as described in Section 2.4.[7] The described construction notion is composable if the pseudo-metric on the set of resources (i.e., the distinguishing advantage) is non-expanding. We defer the simple proof that $\widehat{\Delta}(\cdot,\cdot)$ is non-expanding to Appendix A.

Definition 1. *A protocol* $\pi \in \Sigma$ constructs a restricted resource $\mathbf{S}$ from an assumed restricted resource $\mathbf{R}$ relative to a simulator $\sigma \in \Sigma$ and within $\varepsilon \in [0,1]$, *denoted* $\mathbf{R} \xrightarrow{(\pi,\sigma,\varepsilon)} \mathbf{S}$, *if*

$$\mathbf{R} \xrightarrow{(\pi,\sigma,\varepsilon)} \mathbf{S} \qquad :\Longleftrightarrow \qquad \widehat{\Delta}\left(\pi\mathbf{R}, \mathbf{S}\sigma\right) \;\leq\; \varepsilon \;\; .$$

In the distinguishing advantage $\widehat{\Delta}(\cdot,\cdot)$ that we consider, the outputs of a system are blocked once the MBO of the system becomes 1. In the particular case of query-restricted systems this means that the distinguisher does not obtain further outputs from the system once the specified number of queries is exhausted.

We extend the "arrow notation" from Definition 1 to the case where we consider parameterized families of construction statements, where we require that all of the individual statements must hold. More formally, given a space $\mathcal{K}$ of parameters, a family of protocols $\boldsymbol{\pi} := \{\pi_k\}_{k\in\mathcal{K}}$ constructs a family of restricted resources $\{\mathbf{S}_k\}_{k\in\mathcal{K}}$ from an assumed family of restricted resources $\{\mathbf{R}_k\}_{k\in\mathcal{K}}$, relative to a family of simulators $\boldsymbol{\sigma} := \{\sigma_k\}_{k\in\mathcal{K}}$ and within $\boldsymbol{\varepsilon} : \mathcal{K} \to [0,1]$, denoted $\{\mathbf{R}_k\}_{k\in\mathcal{K}} \xrightarrow{(\boldsymbol{\pi},\boldsymbol{\sigma},\boldsymbol{\varepsilon})} \{\mathbf{S}_k\}_{k\in\mathcal{K}}$, if

$$\{\mathbf{R}_k\}_{k\in\mathcal{K}} \xrightarrow{(\boldsymbol{\pi},\boldsymbol{\sigma},\boldsymbol{\varepsilon})} \{\mathbf{S}_k\}_{k\in\mathcal{K}} \qquad :\Longleftrightarrow \qquad \forall k \in \mathcal{K} : \; \mathbf{R}_k \xrightarrow{(\pi_k,\sigma_k,\boldsymbol{\varepsilon}(k))} \mathbf{S}_k \;\; .$$

Uniform protocols. A family of converters $\boldsymbol{\alpha} = \{\alpha_k\}_{k\in\mathcal{K}}$ is said to be *uniform* if all the converters in the family are identical without their MBO, i.e., $\alpha_k^- = \alpha_{k'}^-$, for all $k, k' \in \mathcal{K}$. Thus, in a uniform parameterized family of converters, the parameter can only influence the MBO of each converter in the family and can therefore only influence the end of a random experiment (and not the values of the random variables). The reason to consider uniform families of converters is that (semantically) a protocol shall not depend on the number of queries that are made to it, since the restriction is a parameter of the environment in which the protocol is used (and not of the protocol itself). We often denote uniform families of converters only by a symbol that denotes a single converter which has no specified MBO, with the implicit understanding that for each single instance of the construction statement, the converter is amended by an MBO that formalizes the suitable restriction of queries.

Query-complexity amplifiers. The construction notion in Definition 1 induces a definition of ε-security for protocols, with respect to a given simulator, if one considers a specific assumed resource $\mathbf{R}$ and a specific desired resource $\mathbf{S}$. In our case, both resources $\mathbf{R}$ and $\mathbf{S}$ will be variants of the random oracle $\mathbf{RO}$.

Definition 2. *Consider two functions* $\varphi : \mathbb{N}\times\mathbb{N} \to \mathbb{N}\times\mathbb{N}$ *and* $\boldsymbol{\varepsilon} : \mathbb{N}\times\mathbb{N} \to [0,1]$. *Then, a uniform family of protocols* $\{\pi_{L,R}\}_{L,R\in\mathbb{N}}$, *where* $\pi_{L,R}^- = \pi$ *for all* $L, R \in \mathbb{N}$ *and for some* deterministic *and* stateless *protocol* $\pi \in \Sigma$, *is said to be a* $(\varphi,\boldsymbol{\varepsilon})$-query-complexity amplifier, *with respect to a family of simulators* $\boldsymbol{\sigma} := \{\sigma_{L,R}\}_{L,R\in\mathbb{N}}$, *if*

$$\left\{{}^{L}|\mathbf{RO}^{|R}\right\}_{L,R\in\mathbb{N}} \xrightarrow{(\pi,\boldsymbol{\sigma},\boldsymbol{\varepsilon})} \left\{{}^{\ell}|\mathbf{RO}^{|r}\right\}_{L,R\in\mathbb{N}},$$

where $(\ell, r) := \varphi(L, R)$ *and* $r < R$, *for all* $L, R \in \mathbb{N}$.

[6] The exact form we describe here, which considers the simulator to be an explicit parameter of the construction, has appeared in the work of Coretti et al. [CMTV15]. However, we formalize the definition only for the information-theoretic case where ε is a constant.

[7] The statement that $\pi\mathbf{R}$ is indifferentiable from $\mathbf{S}$ corresponds to the statement that π constructs $\mathbf{S}$ from $\mathbf{R}$.

(a) Assumed world. **(b)** Desired world.

Figure 1. A (φ, ε)-query-complexity amplifier π: For any number of queries L, R, the resource on the left is within $\varepsilon(L, R)$ from the resource on the right and the simulator $\sigma_{L,R}$ does at most $r < R$ inner queries, where $(\ell, r) := \varphi(L, R)$.

Schemes for query-complexity amplification are often used in contexts where they are evaluated independently by several parties. Requiring such schemes to be deterministic and stateless[8] ensures that the results remain consistent for all parties[9]. According to Definition 2, proving that a protocol π is a (φ, ε)-query-complexity amplifier requires in particular to show that the system π $^{L|}\mathbf{RO}^{|R}$ is within $\varepsilon(L, R)$ from the system $^{\ell|}\mathbf{RO}^{|r}$ $\sigma_{L,R}$, and where $(\ell, r) := \varphi(L, R)$ quantifies the exact amplification achieved for all $L, R \in \mathbb{N}$. Both resources are depicted in Fig. 1.

4 The Caveats of Plain Iterated Hashing

We show in this section that the protocol consisting of iterating c times a random oracle, denoted iter_c, is *not* a query-complexity amplifier, for *any* number $c \geq 2$ of iteration. To do so, we generalize some of the results of Dodis et al. [DRST12], who specifically focused on the case $c = 2$, to deal with a higher number of iterations. The next theorem shows that if one assumes a random oracle with only 2 adversarial queries, then the random oracle constructed by the c-iteration protocol iter_c must allow at least ℓ adversarial queries, where ℓ roughly corresponds to the number of honest queries in the constructed random oracle. For example, this implies that the c-iteration protocol iter_c cannot construct the random oracle $^{4|}\mathbf{RO}^{|1}$ from $^{4c|}\mathbf{RO}^{|2}$ (unless the distinguishing advantage becomes trivial), and therefore iter_c is not a query-complexity amplifier according to Definition 2.

To give some intuition behind this result, consider the c-iteration of a random oracle iter_c **RO** and a chain $\left(y^{(0)}, y^{(c)}, \ldots, y^{(c\ell)}\right)$ of ℓ hashes, where $y^{(cj)}$ denotes the output of the c-iteration protocol iter_c when queried on the previous chain element $y^{(c(j-1))}$. The key observation here is that $y^{(c\ell+1)}$, the output of the random oracle **RO** when queried on the last chain element $y^{(c\ell)}$, forms the end of another chain of ℓ hashes starting with $y^{(1)}$, the output of **RO** when queried on the first element $y^{(0)}$ of the previous chain, *and* that both chains do not have any element in common (with overwhelming probability). In contrast, such shifted chains of queries cannot occur in the system **RO** σ, unless the simulator σ does at least ℓ inner queries to its underlying random oracle.

Note that if the assumed random oracle in Theorem 3 had more adversarial queries, say R instead of 2, then one could force the simulator to make in total in the order of $\Omega(\ell R)$ queries to the underlying random oracle by "hiding" the query on the last chain element $y^{(c\ell)}$ among $R - 2$ random queries. A similar technique was used in [DRST12, Th. 1].

[8] A converter is said to be stateless if it does not keep a state between answering outer queries, i.e., its behavior for a particular outer query depends only on the query itself and the ongoing interaction at the inside interface. We refer to [DGHM13, Def. 1] for a more formal treatment.

[9] Definition 2 is a slight departure from the corresponding definition in the proceedings version of this work [DGMT15], where it was only mentioned afterwards that the protocols of interest are deterministic and stateless. The reason for this change is that of course lazy-sampling, which is a probabilistic and stateful protocol, could then trivially and perfectly achieve any query-complexity amplification.

Algorithm 1: Distinguisher $\mathbf{D}_{\ell,n}$

$y^{(0)} \xleftarrow{\$} \{0,1\}^n$
for $j = 1$ **to** ℓ **do**
$\quad y^{(cj)} :=$ **result of querying** $y^{(c(j-1))}$ at the left-interface
$y^{(c\ell+1)} :=$ **result of querying** $y^{(c\ell)}$ at the right-interface
$y^{(1)} :=$ **result of querying** $y^{(0)}$ at the right-interface
$\tilde{y}^{(1)} := y^{(1)}$
for $j = 1$ **to** ℓ **do**
$\quad \tilde{y}^{(cj+1)} :=$ **result of querying** $\tilde{y}^{(c(j-1)+1)}$ at the left-interface
$\mathcal{C} := \left\{y^{(c)}, y^{(2c)}, \ldots, y^{(c\ell)}\right\}$ and $\tilde{\mathcal{C}} := \left\{\tilde{y}^{(1)}, \tilde{y}^{(c+1)}, \ldots, \tilde{y}^{(c\ell+1)}\right\}$
return $y^{(c\ell+1)} = \tilde{y}^{(c\ell+1)} \wedge \mathcal{C} \cap \tilde{\mathcal{C}} = \emptyset$

Theorem 3. *The protocol* iter_c*, consisting of iterating* c *times a random oracle, where* $c \geq 2$*, is such that for any number* ℓ *of queries and any simulator* σ*,*

$$ {}^{2c\ell|}\mathbf{RO}^{|2} \xrightarrow{(\mathsf{iter}_c, \sigma, \varepsilon)} {}^{2\ell|}\mathbf{RO}^{|r} \qquad \Longrightarrow \qquad r \geq \ell \quad \vee \quad \varepsilon \geq 1 - \mu, $$

where $\mu := 2^{-n} \cdot f(c,\ell)$ *and* $f(c,\ell) := \frac{1}{1-3\ell \cdot 2^{-n}} + \frac{1}{2}(3\ell)^2 + 2(c\ell+1)^2$.

Proof. Let us assume that $r < \ell$ since otherwise the proof is finished. In order to have shorter notations within the proof, let us denote by $\mathbf{R}$ and $\mathbf{S}$ the systems $\mathsf{iter}_c\ {}^{2c\ell|}\mathbf{RO}^{|2}$ and ${}^{2\ell|}\mathbf{RO}^{|r}\ \sigma$, respectively, for some simulator σ. Then, we give a distinguisher $\mathbf{D}_{\ell,n}$, described in Alg. 1, and show that it achieves the desired restricted distinguishing advantage, i.e., $\widehat{\Delta}^{\mathbf{D}_{\ell,n}}(\mathbf{R},\mathbf{S}) \geq 1-\mu$. Intuitively, the distinguisher $\mathbf{D}_{\ell,n}$ is based on the aforementioned shifted chains of queries. In greater details, the distinguisher $\mathbf{D}_{\ell,n}$ first prepares a ℓ-chain $\left(y^{(0)}, y^{(c)}, \ldots, y^{(c\ell)}\right)$ of hashes, starting at a random n-bit string $y^{(0)}$, by querying ℓ times the left-interface of $\mathbf{R}$ or $\mathbf{S}$. Note that when interacting with the system $\mathbf{R}$, the chain element $y^{(cj)}$ is indeed the c-iterate of the random oracle when queried on the previous chain element $y^{(c(j-1))}$, for all $j \in \{1,\ldots,\ell\}$. Then, the distinguisher $\mathbf{D}_{\ell,n}$ tries to "shift" the obtained chain by querying successively $y^{(c\ell)}$ and $y^{(0)}$ at the right-interface to obtain $y^{(c\ell+1)}$ and $y^{(1)}$, respectively. When interacting with $\mathbf{R}$, the values $y^{(c\ell+1)}$ and $y^{(1)}$ are simply the answers of the random oracle when queried on $y^{(c\ell)}$ and $y^{(0)}$, respectively. Finally, the distinguisher $\mathbf{D}_{\ell,n}$ checks by doing ℓ queries at the left-interface that $y^{(c\ell+1)}$ is indeed the end of a ℓ-chain of hashes starting with $y^{(1)}$ and that the shifted chain does not have any element in common with the chain initially prepared. Overall, the distinguisher $\mathbf{D}_{\ell,n}$ does 2ℓ queries at the left-interface and only 2 queries at the right-interface.

To lower bound the restricted distinguishing advantage of the distinguisher $\mathbf{D}_{\ell,n}$, it will be convenient to consider a variation of the random oracle, denoted $\mathbf{T}$, which answers to the first 2^n fresh queries by a n-bit string chosen uniformly at random but without replacement, i.e., on the first 2^n fresh queries $\mathbf{T}$ corresponds to a random injective function. Trivially, $\mathbf{RO}$ and $\mathbf{T}$ are indistinguishable unless a collision occurs in $\mathbf{RO}$, i.e.,

$$ \widehat{\Delta}\left({}^{L|}\mathbf{RO}^{|R}, {}^{L|}\mathbf{T}^{|R}\right) \leq p_{\mathrm{coll}}(L+R, 2^n), \tag{2} $$

for all $L, R \in \mathbb{N}$, and where $p_{\mathrm{coll}}(q,t)$ denotes the probability that there exists a collision among q values distributed independently and uniformly at random over a set of t elements. Let $\mathbf{R}'$ and $\mathbf{S}'$ denote the systems $\mathbf{R}$ and $\mathbf{S}$, respectively, where the random oracle $\mathbf{RO}$ was replaced by its variation $\mathbf{T}$, i.e., $\mathbf{R}' := \mathsf{iter}_c\ {}^{2c\ell|}\mathbf{T}^{|2}$ and $\mathbf{S}' := {}^{2\ell|}\mathbf{T}^{|r}\ \sigma$. Note that (2) implies that $\widehat{\Delta}(\mathbf{R}',\mathbf{R}) \leq p_{\mathrm{coll}}(2c\ell+2, 2^n)$, and similarly $\widehat{\Delta}(\mathbf{S}',\mathbf{S}) \leq p_{\mathrm{coll}}(2\ell+r, 2^n)$. It suffices thus to lower bound the distinguishing advantage of $\mathbf{D}_{\ell,n}$ when interacting with $\mathbf{R}'$ or $\mathbf{S}'$ since

$$ \widehat{\Delta}(\mathbf{R},\mathbf{S}) \geq \widehat{\Delta}(\mathbf{R}',\mathbf{S}') - \left(p_{\mathrm{coll}}(2c\ell+2, 2^n) + p_{\mathrm{coll}}(2\ell+r, 2^n)\right) \ . $$

In the random experiment defined by the distinguisher $\mathbf{D}_{\ell,n}$ interacting with the system $\mathbf{R}'$, we always have $y^{(c\ell+1)} = \tilde{y}^{(c\ell+1)}$. Since no collision can occur in $\mathbf{R}'$, the two chains $\mathcal{C}$ and $\tilde{\mathcal{C}}$, which both contain n-bit strings coming only from the system $\mathbf{R}'$, do not collide, i.e., $\mathcal{C} \cap \tilde{\mathcal{C}} = \emptyset$. Thus, $\mathbf{D}_{\ell,n}$ always outputs 1 when interacting with $\mathbf{R}'$.

In contrast, in the random experiment defined by the distinguisher $\mathbf{D}_{\ell,n}$ interacting with the system $\mathbf{S}'$, $\mathbf{D}_{\ell,n}$ outputs 1 if the simulator σ in $\mathbf{S}'$ outputs the start and the end of a ℓ-chain $\left(\tilde{y}^{(1)}, \tilde{y}^{(c+1)}, \ldots, \tilde{y}^{(c(\ell-1)+1)}, \tilde{y}^{(c\ell+1)}\right)$ of hashes later queried by $\mathbf{D}_{\ell,n}$, i.e., the simulator σ must output $y^{(c\ell+1)}$ and $y^{(1)}$ such that $\tilde{y}^{(1)} = y^{(1)}$ and $\tilde{y}^{(c\ell+1)} = y^{(c\ell+1)}$. The simulator σ does at most r queries to the underlying variant of the random oracle $\mathbf{T}$ and therefore σ receives at most r responses from $\mathbf{T}$, where by assumption $r \leq \ell - 1$. Thus, at least one of the chain element $\tilde{y}^{(cj+1)}$ that the distinguisher $\mathbf{D}_{\ell,n}$ obtained was never received by the simulator σ, for some $j \in \{1, \ldots, \ell\}$. If $j = \ell$, then the simulator σ never queried the second to last chain element $\tilde{y}^{(c(\ell-1)+1)}$ and therefore the last chain element $\tilde{y}^{(c\ell+1)}$ is uniformly distributed over a set of size at least $2^n - 3\ell$, since at most $2\ell + r \leq 3\ell$ queries were done to $\mathbf{T}$, so that the probability that $y^{(c\ell+1)} = \tilde{y}^{(c\ell+1)}$ is at most $1/(2^n - 3\ell)$. Similarly, if $j < \ell$ and the simulator σ did receive all the next chain elements $\tilde{y}^{(c(j+1)+1)}, \ldots, \tilde{y}^{(c\ell+1)}$, then $\tilde{y}^{(cj+1)}$ is uniformly distributed over a set of size at least $2^n - 3\ell$, and the probability that it equals the value input by the simulator σ when it received the next chain element $\tilde{y}^{(c(j+1)+1)}$ is therefore at most $1/(2^n - 3\ell)$. Thus, when interacting with $\mathbf{S}'$, the distinguisher $\mathbf{D}_{\ell,n}$ outputs 1 with probability at most $1/(2^n - 3\ell)$, and overall we have

$$\widehat{\Delta}(\mathbf{R}', \mathbf{S}') \;\geq\; 1 - \frac{1}{2^n - 3\ell} \;.$$

The proof is finished by combining the two previous equations and by using the standard argument that $p_{\text{coll}}(q, t) \leq q^2/(2t)$. □

A vulnerable application. There are concrete applications where the fact that the plain iteration protocol iter_c fails to be a query-complexity amplifier is problematic. One example of such a vulnerable application is the setting of mutual proofs of work, introduced by Dodis et al. [DRST12], which is secure if a monolithic random oracle $\mathbf{RO}$ is employed, but becomes insecure if the c-iterate iter_c $\mathbf{RO}$ is used instead, for any $c \geq 2$. This fact was already known for the special case $c = 2$ [DRST12] and it is easy to show, with the same kind of arguments as used to prove Theorem 3, that it generalizes to higher iteration counts.

Recall that in mutual proofs of work, two parties aim at proving to each other that they did a certain amount of computation. In the protocol proposed by Dodis et al. [DRST12], both parties exchange in the first round a nonce and then compute a chain of hashes of a certain length (chosen by the computing party) starting with the received nonce. In the second round, both parties exchange the length and the last element of their computed chain. Then, each party checks that the other party actually did the claimed amount of computation by first computing a chain of hashes of the asserted length starting with the nonce that was originally sent, and second, by checking that both computed chains do not have any common element.

Note that such a scheme is insecure if the parties use iter_c $\mathbf{RO}$ to compute their chain of hashes. Indeed, a malicious party, similarly to the distinguisher $\mathbf{D}_{\ell,n}$ in Alg. 1, can simply "shift" the chain of hashes computed by the honest party and needs therefore only two hash evaluations to compute the beginning and the end of a valid chain of hashes (which with overwhelming probability has no common element with the chain computed by the honest party). In contrast, this protocol for mutual proofs of work is secure if the parties use a query-complexity amplifier, such as the collision-free iteration protocol amp_c presented in the next section, to compute their chain of hashes.

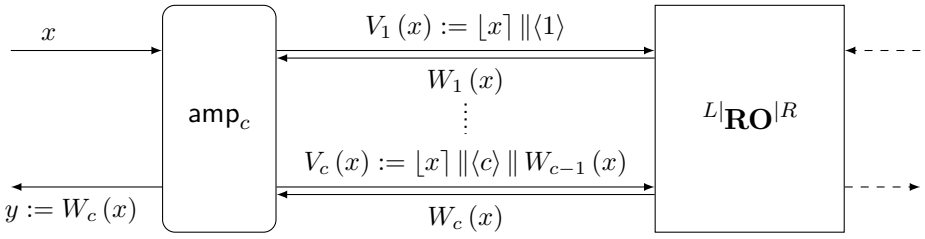

Figure 2. Protocol amp_c for amplifying the complexity of a random oracle by a factor c. A prefix-free encoding of a bit-string x is denoted by $\lfloor x \rceil$, and an encoding of an integer j over $\lceil \log_2 c \rceil$-bit strings is denoted by $\langle j \rangle$.

5 Complexity Amplification via Collision-Free Iteration

The main result of this section is to present the collision-free iteration protocol, denoted amp_c, for amplifying the query complexity of a random oracle by a constant factor c, for some fixed parameter $c \in \mathbb{N}$. We present the (uniform) protocol amp_c and the corresponding (uniform) simulator sim in Section 5.1 and prove the actual construction stated below in Section 5.2.

Theorem 4. *The collision-free iteration protocol* amp_c *is an* $\left((L,R) \mapsto \left(\lfloor \frac{L}{c} \rfloor, \lfloor \frac{R}{c} \rfloor\right), \boldsymbol{\delta}\right)$*-query-complexity amplifier with respect to the simulator* sim*, i.e.,*

$$\left\{ {}^{L}|\mathbf{RO}|^{R} \right\}_{L,R \in \mathbb{N}} \xrightarrow{(\mathsf{amp}_c, \mathsf{sim}, \boldsymbol{\delta})} \left\{ {}^{\lfloor \frac{L}{c} \rfloor}|\mathbf{RO}|^{\lfloor \frac{R}{c} \rfloor} \right\}_{L,R \in \mathbb{N}},$$

where $\boldsymbol{\delta}(L,R) := R \cdot 2^{-n}$ *and* n *is the output length of the random oracle* $\mathbf{RO}$*, for all* $L, R \in \mathbb{N}$*. The protocol* amp_c *and the simulator* sim *are described in Fig. 2 and Alg. 2, respectively.*

Notice that the upper bound $\boldsymbol{\delta}$ on the distinguishing advantage in the previous theorem is independent of the number L of queries made to the left-interface and also of the factor c, which also corresponds to the number of iterations in the protocol amp_c given in Fig. 2 above. Throughout this section, we will denote by ℓ and r the two integers corresponding to $\lfloor \frac{L}{c} \rfloor$ and $\lfloor \frac{R}{c} \rfloor$, respectively, for all $L, R \in \mathbb{N}$.

5.1 The Protocol and the Simulator

Protocol amp_c. Consider the collision-free iteration protocol amp_c attached to the left-interface of a random oracle as described in Fig. 2. When queried on an input $x \in \{0,1\}^*$ (at its outside interface), the protocol amp_c does c queries $V_1(x), \ldots, V_c(x)$ to the random oracle, where the query $V_j(x)$ contains the answer of the random oracle on the previous query $V_{j-1}(x)$. In addition, amp_c uses prefixing to ensure that there is no collision among the queries asked, i.e., $V_j(x) \neq V_{j'}(x')$ whenever $(j,x) \neq (j',x')$. Namely, amp_c prefixes each query $V_j(x)$ with a prefix-free encoding $\lfloor x \rceil$ of x and with an iteration counter $\langle j \rangle$ where $\langle \cdot \rangle : \{1, \ldots, c\} \to \{0,1\}^{\lceil \log_2 c \rceil}$ denotes an arbitrary *injective* function from $\{1, \ldots, c\}$ to the set of $\lceil \log_2 c \rceil$-bit strings. The former guarantees no overlap between the queries for two different inputs x and x', while the second prevents collisions within the sequence of queries for the same input x. More generally, letting $W_0(x)$ be the empty bit string and $W_j(x)$ be the inner response of the connected resource to the inner query $V_j(x)$, we then define iteratively

$$\begin{aligned} V_j(x) &:= \lfloor x \rceil \,\|\langle j \rangle \,\|\, W_{j-1}(x) \\ W_j(x) &:= \textbf{result of querying } V_j(x) \text{ at the } \mathsf{in}\text{-interface}, \end{aligned}$$

for all $j \in \{1, \ldots, c\}$. Finally, we simply let $W_c(x)$, the response of the connected resource to the final query, be the output of the protocol.

PREFIX-FREE ENCODINGS. A prefix-free encoding function $\lfloor\cdot\rceil : \{0,1\}^* \to \{0,1\}^*$ is a function ensuring that $\lfloor\tilde{x}\rceil$ is not a prefix of $\lfloor x\rceil$ whenever $x \neq \tilde{x}$. We also assume it can be easily decided whether a bit string $y \in \{0,1\}^*$ is in the range of $\lfloor\cdot\rceil$, and in that case the (unique) pre-image of y can be efficiently recovered. Our results are independent of which such prefix-free encoding function is used. A simple example of such a prefix-free encoding is the function $\lfloor\ \rceil : \{0,1\}^* \to \{0,1\}^*;\ (b_1,\ldots,b_n) \mapsto (1,b_1,1,b_2,\ldots,1,b_n,0)$. Many other (more efficient) examples exist, such as those described by Coron et al. [CDMP05].

Simulator sim. Before describing the behavior of the simulator sim defined in Alg. 2, let us first characterize more precisely the different types of queries we will consider. A query v is said to be well-formed, denoted $\texttt{isWellFormed}(v)$, if it contains the prefixes as used by the protocol amp_c, i.e., $v \in \mathcal{V} \subseteq \{0,1\}^*$, where $\mathcal{V} := \bigcup_{\substack{x\in\{0,1\}^* \\ j\in\{1,\ldots,c\}}} \mathcal{V}_j(x)$, with

$$\mathcal{V}_1(x) := \{\lfloor x\rceil \,\|\langle 1\rangle\} \quad \text{and} \quad \mathcal{V}_j(x) := \{\lfloor x\rceil \,\|\langle j\rangle \,\|\, z :\ z \in \{0,1\}^n\} \quad \text{for} \quad j \geq 2\ .$$

An element of $\mathcal{V}_j(x)$ will be called a well-formed query of level j with prefix x. We denote by $\texttt{parse}(\cdot) :\ \mathcal{V} \to \{0,1\}^* \times \{1,\ldots,c\}$ the function which, given a well-formed query v, returns the pair (x,j) corresponding to the prefix and level associated with this query, respectively. Given an arbitrary subset of well-formed queries $\tilde{\mathcal{V}} \subseteq \mathcal{V}$, a prefix $x \in \{0,1\}^*$ is declared to be "fresh", denoted $\texttt{isPrefixFresh}(x,\tilde{\mathcal{V}})$, if it was never encountered, i.e.,

$$\texttt{isPrefixFresh}(x,\tilde{\mathcal{V}}) \quad :\Longleftrightarrow \quad \forall v \in \tilde{\mathcal{V}}\ \forall j \in \{1,\ldots,c\} :\ (x,j) \neq \texttt{parse}(v)\ .$$

The simulator sim works as follows: whenever it receives a well-formed query $v \in \{0,1\}^*$ of some level $j \in \{1,\ldots,c\}$ with a "fresh" prefix $x \in \{0,1\}^*$, it emulates the behavior of the protocol amp_c on input x by generating a "fake" chain of queries $\tilde{V}_1(x)$, $\tilde{W}_1(x)$, $\ldots$, $\tilde{V}_{c-1}(x)$, $\tilde{W}_{c-1}(x)$, $\tilde{V}_c(x)$, where the emulated answers $\tilde{W}_k(x)$ are simply uniform n-bit strings *locally* sampled by the simulator. Then, the simulator sim returns the answer of the random oracle ${}^{\ell|}\mathbf{RO}^{|r}$ when queried on the prefix x, only if the outer query v matches the last chain element $\tilde{V}_c(x)$ and all previous chain elements $\tilde{V}_1(x),\ldots,\tilde{V}_{c-1}(x)$ were already queried. On the other hand, if the query v matches one of the lower-level chain elements, i.e., $v = \tilde{V}_j(x)$ with $j < c$ and all previous chain elements were already queried, then the simulator sim replies with the answer $\tilde{W}_j(x)$ that was already chosen earlier (when generating the chain for the prefix x). In the (unlikely) case where a distinguisher happens to have guessed the value of $\tilde{V}_j(x)$, i.e., $v = \tilde{V}_j(x)$ but the previous chain element $\tilde{V}_{j-1}(x)$ was never queried, the simulator sim gives up on simulation by outputting the all zero bit string 0^n and setting internally the event hit to 1 in order to prevent any further inner query to the random oracle. Finally, if the query v considered is not well-formed, then the simulator sim replies with a fresh uniform n-bit string. We refer to Alg. 2 for a precise description of the simulator sim. Note that it maintains a state over all invocations, keeping track of the set $\tilde{\mathcal{V}}$ of well-formed queries received, the values $\tilde{V}_j(x)$ and $\tilde{W}_j(x)$ corresponding to the locally generated chains of queries, and the mapping g to be able to reply consistently to any repeated query.

5.2 Indistinguishability Proof

Recall that the statement of Theorem 4 considers a construction between an assumed random oracle ${}^{L|}\mathbf{RO}^{|R}$ and a desired random oracle ${}^{\ell|}\mathbf{RO}^{|r}$, for all integers L, R. If the number of queries that can be made to the left-interface of the desired random oracle is limited to ℓ, then so should also be restricted the number of queries that can be made to the outside interface of the protocol. Similarly, restricting the assumed random oracle to at most R queries at its right-interface implies the same restriction on the number of queries that can be made to the outside interface of the simulator. Thus, we will prove Theorem 4 for the uniform family of protocols

Algorithm 2: Simulator sim

$g(v) := \lambda$, for all $v \in \{0,1\}^*$ // λ denotes the empty bit string
$\tilde{\mathcal{V}} = \emptyset$ and $\mathsf{hit} := 0$
on input $v \in \{0,1\}^*$ at the out-interface
  if $g(v) = \lambda$ **then** // v was never queried before
    if $\mathtt{isWellFormed}(v)$ **then**
      $(x, j) := \mathtt{parse}(v)$
      if $\mathtt{isPrefixFresh}(x, \tilde{\mathcal{V}})$ **then** // generate a "fake" chain of queries
        $(\tilde{V}_1(x), \tilde{W}_1(x), \ldots, \tilde{V}_{c-1}(x), \tilde{W}_{c-1}(x)) := \mathtt{GenerateChain}(x, c-1)$
        $\tilde{V}_c(x) := \lfloor x \rceil \,\|\langle c \rangle\| \, \tilde{W}_{c-1}(x)$
      if $v = \tilde{V}_j(x)$ **then**
        if $j > 1 \ \wedge \ g(\tilde{V}_{j-1}(x)) = \lambda$ **then** // previous chain element was not queried
          $\mathsf{hit} := 1$
          $\tilde{Y} := 0^n$
        else if $j = c \ \wedge \ \mathsf{hit} = 0$ **then**
          $\tilde{Y} :=$ **result of querying** x at the in-interface
        else if $j = c \ \wedge \ \mathsf{hit} = 1$ **then**
          $\tilde{Y} := 0^n$
        else $\tilde{Y} := \tilde{W}_j(x)$
      else $\tilde{Y} \xleftarrow{\$} \{0,1\}^n$
      $\tilde{\mathcal{V}} \leftarrow \tilde{\mathcal{V}} \cup \{v\}$
    else $\tilde{Y} \xleftarrow{\$} \{0,1\}^n$
    $g(v) := \tilde{Y}$
  output: $g(v)$ at the out-interface

Procedure $\mathtt{GenerateChain}(x, m)$
  $\tilde{W}_0(x) := \lambda$
  for $j = 1$ **to** m **do**
    $\tilde{V}_k(x) := \lfloor x \rceil \,\|\langle k \rangle\| \, \tilde{W}_{k-1}(x)$
    $\tilde{W}_k(x) \xleftarrow{\$} \{0,1\}^n$
  return $(\tilde{V}_1(x), \tilde{W}_1(x), \ldots, \tilde{V}_m(x), \tilde{W}_m(x))$

$\left\{{}^{\ell|}\mathsf{amp}_c\right\}_{L,R\in\mathbb{N}}$ and for the uniform family of simulators $\left\{\mathsf{sim}^{|R}\right\}_{L,R\in\mathbb{N}}$. We therefore need to upper bound the distinguishing advantage between the query-restricted systems ${}^{\ell|}\mathsf{amp}_c \ {}^{L|}\mathbf{RO}^{|R}$ and ${}^{\ell|}\mathbf{RO}^{|r} \ \mathsf{sim}^{|R}$, for all $L, R \in \mathbb{N}$. The idea for upper bounding this distinguishing advantage is to first transform the system $\mathbf{RO}\ \mathsf{sim}$ into a game $\boxed{\mathbf{RO}\ \mathsf{sim}}$, where the latter is defined to be won if the event hit is provoked in the simulator sim described in Alg. 2; and second, to show that this game $\boxed{\mathbf{RO}\ \mathsf{sim}}$ is conditionally equivalent to the system $\mathsf{amp}_c\ \mathbf{RO}$. Before proving the corresponding conditional equivalence statement in Lemma 5 below, we start by describing informally how it implies Theorem 4. A more formal treatment appears in the full version.

Query complexity. The protocol amp_c makes exactly c inner queries for every query it receives at its outside interface. Consequently, the protocol amp_c does in total at most L inner queries if it is queried at most ℓ times at its outside interface. The simulator sim makes a query x at its inside interface only if it receives a chain of c (distinct) queries $\tilde{V}_1(x), \ldots, \tilde{V}_c(x)$. The simulator sim keeps in memory the previous interaction, so that when such a chain of c queries is received, at most one query is made to the inside interface of sim. Furthermore, the prefix scheme employed prevents any form of collision among the queries so that making multiple, say

k, such chains of c queries requires at least $k \cdot c$ queries. Hence, any tuple of R queries contains at most r such chains of c queries, and thus the simulator does in total at most r inner queries if it is queried at most R times at its outside interface. Thus, the protocol amp_c and the simulator sim are such that

$$\left[{}^{\ell|}\mathsf{amp}_c\ \mathbf{RO}^{|R}\right]^{\dashv} \equiv \left[{}^{\ell|}\mathsf{amp}_c\ {}^{L|}\mathbf{RO}^{|R}\right]^{\dashv} \qquad \text{and} \qquad \left[{}^{\ell|}\mathbf{RO}\ \mathsf{sim}^{|R}\right]^{\dashv} \equiv \left[{}^{\ell|}\mathbf{RO}^{|r}\ \mathsf{sim}^{|R}\right]^{\dashv} .$$

It is therefore sufficient to upper bound the restricted distinguishing advantage between the query-restricted systems ${}^{\ell|}\mathsf{amp}_c\ \mathbf{RO}^{|R}$ and ${}^{\ell|}\mathbf{RO}\ \mathsf{sim}^{|R}$. To do so, we consider two games $\ulcorner\mathbf{RO}\ \mathsf{sim}\urcorner$ and $\ulcorner{}^{\ell|}\mathbf{RO}\ \mathsf{sim}^{|R}\urcorner$, where both games are won if and only if the event hit in the simulator sim is provoked. We show in Lemma 5 below that the game $\ulcorner\mathbf{RO}\ \mathsf{sim}\urcorner$ is conditionally equivalent to the system $\mathsf{amp}_c\ \mathbf{RO}$. Then, Lemma 9 in Appendix B implies that the restricted game $\ulcorner{}^{\ell|}\mathbf{RO}\ \mathsf{sim}^{|R}\urcorner$ is conditionally equivalent to the restricted system ${}^{\ell|}\mathsf{amp}_c\ \mathbf{RO}^{|R}$, the intuitive reason being that the added MBO, corresponding to a restriction on the number of queries, is simply a deterministic function of the inputs. Similarly to [Mau13, Th. 3], we show in Lemma 8 in Appendix B that this conditional equivalence statement between query-restricted systems implies that the restricted distinguishing advantage between ${}^{\ell|}\mathsf{amp}_c\ \mathbf{RO}^{|R}$ and ${}^{\ell|}\mathbf{RO}\ \mathsf{sim}^{|R}$ is upper bounded by the probability for non-adaptive game winners to win the query-restricted game $\ulcorner{}^{\ell|}\mathbf{RO}\ \mathsf{sim}^{|R}\urcorner$, where the latter is shown in Lemma 6 below to be at most $R \cdot 2^{-n}$. Overall, we thus have for all integers $L, R \in \mathbb{N}$ that

$$\begin{aligned}\widehat{\Delta}\left({}^{\ell|}\mathsf{amp}_c\ {}^{L|}\mathbf{RO}^{|R}, {}^{\ell|}\mathbf{RO}^{|r}\ \mathsf{sim}^{|R}\right) &= \widehat{\Delta}\left({}^{\ell|}\mathsf{amp}_c\ \mathbf{RO}^{|R}, {}^{\ell|}\mathbf{RO}\ \mathsf{sim}^{|R}\right) \\ &\leq \widehat{\Gamma}^{\mathsf{NA}}\left(\ulcorner{}^{\ell|}\mathbf{RO}\ \mathsf{sim}^{|R}\urcorner\right) \\ &\leq R \cdot 2^{-n} .\end{aligned}$$

The proof of the following lemma, together with Lemma 6 below, complete the proof of Theorem 4.

Lemma 5. *Consider the protocol* amp_c *and the simulator* sim *defined in Fig. 2 and Alg. 2, respectively. Let* $\ulcorner\mathbf{RO}\ \mathsf{sim}\urcorner$ *denote the game which is won if and only if the event* hit *in* sim *is provoked. Then,*

$$\ulcorner\mathbf{RO}\ \mathsf{sim}\urcorner \models \mathsf{amp}_c\ \mathbf{RO} .$$

Proof. Let us denote by $\mathbf{R}$ and $\widehat{\mathbf{S}}$ the systems $\mathsf{amp}_c\ \mathbf{RO}$ and $\ulcorner\mathbf{RO}\ \mathsf{sim}\urcorner$, respectively (where $\widehat{\mathbf{S}}$ is actually a game). We need to show that $\widehat{\mathbf{S}} \models \mathbf{R}$. We are going to argue that as long as the game $\widehat{\mathbf{S}}$ is not won, the probability distribution of the response to any possible query is the same in both $\mathbf{R}$ and $\widehat{\mathbf{S}}^{\dashv}$. Both systems reply consistently to any repeated queries, let us hence without loss of generality only consider fresh queries. To analyze the sampling process of responses, note that we can see both $\mathbf{R}$ and $\widehat{\mathbf{S}}^{\dashv}$ as generating the responses to all possible queries in advance (according to distributions described below) and then using the pre-generated responses to answer all actual queries. To describe these distributions, let us denote by $\mathit{Left}(x)$ and $\mathit{Right}(v)$ the responses of the system in question (either $\mathbf{R}$ or $\widehat{\mathbf{S}}^{\dashv}$) to queries (left, x) and (right, v), respectively. The (inefficient) sampling processes for the systems $\mathbf{R}$ and $\widehat{\mathbf{S}}^{\dashv}$ (as long as the game is not won) are described in Alg. 3 and 4, respectively. It is now easy to see that these two sampling processes result in the same distribution of all the random variables $\mathit{Left}(x)$ and $\mathit{Right}(v)$.

□

Algorithm 3: Sampling for $\mathbf{R}$

1. **foreach** $x \in \{0,1\}^*$ **do**
   $U_1, \ldots, U_c \xleftarrow{\$} \{0,1\}^n$
   $\mathit{Right}(\lfloor x \rceil \,\|\langle 1\rangle) := U_1$
   $\mathit{Right}(\lfloor x \rceil \,\|\langle j\rangle \,\|\, U_{j-1}) := U_j$, for all $j \in \{1, \ldots, c\}$
   $\mathit{Left}(x) := U_c$
2. Sample all remaining values
   $\mathit{Right}(v) \xleftarrow{\$} \{0,1\}^n$

Algorithm 4: Sampling for $\widehat{\mathbf{S}}^{\dashv}$

1. **foreach** $x \in \{0,1\}^*$ **do**
   $\mathit{Left}(x) \xleftarrow{\$} \{0,1\}^n$
   $U_1, \ldots, U_{c-1} \xleftarrow{\$} \{0,1\}^n$
   $\mathit{Right}(\lfloor x \rceil \,\|\langle 1\rangle) := U_1$
   $\mathit{Right}(\lfloor x \rceil \,\|\langle j\rangle \,\|\, U_{j-1}) := U_j$, for all $j \in \{1, \ldots, c-1\}$
   $\mathit{Right}(\lfloor x \rceil \,\|\langle c\rangle \,\|\, U_{c-1}) := \mathit{Left}(x)$
2. Sample all remaining values
   $\mathit{Right}(v) \xleftarrow{\$} \{0,1\}^n$

Lemma 6. *Let* $\overline{\lceil L|\mathbf{RO}\ \mathsf{sim}^{|R}\rceil}$ *denote the game formed by the restricted system* $^{L|}\mathbf{RO}\ \mathsf{sim}^{|R}$ *and which is won if and only if the event* hit *in the simulator* sim *is provoked. Then,*

$$\widehat{\Gamma}^{\mathsf{NA}}\left(\overline{\lceil L|\mathbf{RO}\ \mathsf{sim}^{|R}\rceil}\right) \;\leq\; \boldsymbol{\delta}(L,R)\,,$$

for all $L, R \in \mathbb{N}$ *and for* $\boldsymbol{\delta}$ *defined as in Theorem 4.*

Proof. Let $L, R \in \mathbb{N}$ and consider a non-adaptive game winner $\mathbf{W}$ trying to win the restricted game $\overline{\lceil L|\mathbf{RO}\ \mathsf{sim}^{|R}\rceil}$. The game winner $\mathbf{W}$ wins the game only if it provokes the event hit in the simulator sim within R (well-formed) queries. Any well-formed query has a certain prefix x and level j, where $x \in \{0,1\}^*$ and $j \in \{1, \ldots, c\}$, and the probability for such a query to win the game is therefore at most 2^{-n} since it requires to guess the value of $\tilde{W}_{j-1}(x)$, an independent and uniformly distributed n-bit string. By applying the union bound it follows that

$$\widehat{\Gamma}^{\mathsf{NA}}\left(\overline{\lceil L|\mathbf{RO}\ \mathsf{sim}^{|R}\rceil}\right) \;\leq\; R \cdot 2^{-n} \;=\; \boldsymbol{\delta}(L,R)\,,$$

for all $L, R \in \mathbb{N}$. □

6 Towards Optimality

Theorem 4 gives a simple protocol to amplify the query-complexity of a random oracle by a constant factor c, at the cost of simultaneously reducing the number of honest-party queries by the same factor. Such a decrease is of course undesired, and the goal of this section is to study whether such a reduction of the honest-party queries is inherent to any query-complexity amplification scheme. To do so, we consider an arbitrary (deterministic) protocol π and an arbitrary simulator σ, and show in Lemma 7 that in order for the construction $^{L|}\mathbf{RO}^{|R} \xrightarrow{(\pi,\sigma,\varepsilon)} {}^{\ell|}\mathbf{RO}^{|r}$ to be achievable, the composed converter $\sigma \circ \pi$ must do (with high probability) at least as many inner queries as many outer queries it receives. This in particular implies that if the simulator σ considered is such that for every r-tuple of outer queries it does at most r/c inner queries, for some positive constant c, then the protocol π must do at least $c \cdot \ell$ inner queries for every ℓ-tuple of outer queries it receives. Therefore, the protocol amp_c given in Theorem 4 is in this sense optimal (if one restricts oneself to such a class of simulators). In the following, we denote by $\mathcal{Q}_\alpha\left(x^k\right)$ the random variable corresponding to the *sequence* of inner queries made by α when queried at its outside interface on $x_1, \ldots, x_k$ in a given random experiment (e.g., a distinguisher interacting with $\mathbf{RO}\alpha$). The number of distinct elements in such a tuple, which is also a random variable, will be denoted by $\left|\mathcal{Q}_\alpha\left(x^k\right)\right|$.

Algorithm 5: Distinguisher $\mathbf{D}_{x^k}$ for an $(\mathcal{I} \times \{0,1\}^*, \{0,1\}^n \cup \{\diamond\})$-system $\mathbf{S}$

$Z := 0$
for $j = 1$ **to** k **do** $y_j :=$**result of querying** x_j at the left-interface of $\mathbf{S}$
for $j = 1$ **to** k **do** $\tilde{y}_j :=$`EmulateProtocol(right,`x_j`)` `// Emulate` π `at the right-interface`
if $\exists j \in \{1, \ldots, k\} : \ y_j \neq \diamond \ \wedge \ y_j \neq \tilde{y}_j$ **then** $Z := 1$
output: Z

Procedure `EmulateProtocol(`i, x`)`
  $(i', x') :=$**emulate result of querying** π at the out-interface on x
  while $i' =$ in **do**
    $y' :=$ **result of querying** x' at the i-interface of $\mathbf{S}$
    $(i', x') :=$**emulate result of querying** π at the in-interface on y'
  return x'

Lemma 7. *Consider four integers $L, R, \ell, r \in \mathbb{N}$, where $R \geq L > 0$ and $\ell > 0$. Consider a deterministic protocol $\pi \in \Sigma$ and a simulator $\sigma \in \Sigma$. Then, for any integer $k \leq \ell$ and for any sequence of k distinct bit strings $x^k = (x_1, \ldots, x_k) \in (\{0,1\}^*)^k$,*

$$\widehat{\Delta}^{\mathbf{D}_{x^k}}\left(\pi \ {}^{L|}\mathbf{RO}^{|R}, \ {}^{\ell|}\mathbf{RO}^{|r} \ \sigma\right) \quad \geq \quad (1 - 2^{-n}) \cdot \mathsf{P}\left(\left|\mathcal{Q}_\sigma\left(\mathcal{Q}_\pi\left(x^k\right)\right)\right| < k\right),$$

where the distinguisher $\mathbf{D}_{x^k}$ is defined in Alg. 5 and the probability is taken in the random experiment defined by $\mathbf{D}_{x^k}$ interacting with the system $\left[{}^{\ell|}\mathbf{RO}^{|r} \ \sigma\right]^{\dashv}$.

Proof. Consider an integer k such that $k \leq \ell$ and a sequence of k distinct bit strings $x_1, \ldots, x_k$, where $x_j \in \{0,1\}^*$. Consider the distinguisher $\mathbf{D}_{x^k}$ described in Alg. 5. In the $\mathbf{D}_{x^k}\mathbf{S}$ random experiment, where $\mathbf{S}$ is one of the two systems $\left[\pi \ {}^{L|}\mathbf{RO}^{|R}\right]^{\dashv}$ or $\left[{}^{\ell|}\mathbf{RO}^{|r} \ \sigma\right]^{\dashv}$, the distinguisher $\mathbf{D}_{x^k}$ emulates the protocol π at the right-interface of $\mathbf{S}$. The distinguisher $\mathbf{D}_{x^k}$ first queries the left-interface of $\mathbf{S}$ on the sequence of inputs x^k, obtaining the corresponding sequence of answers y^k, and queries also the right-interface of $\mathbf{S}$ (through the emulated protocol π) on the same sequence of inputs x^k to obtain the sequence of responses $\tilde{y}^k$. The distinguisher outputs its decision bit Z which is 1 if and only the answers obtained are not consistent, i.e., $y_j \neq \diamond$ and $y_j \neq \tilde{y}_j$ for some $j \in \{1, \ldots, k\}$.

Consider first the random experiment where the distinguisher $\mathbf{D}_{x^k}$ interacts with the system $\left[\pi \ {}^{L|}\mathbf{RO}^{|R}\right]^{\dashv}$. If the answer y_j obtained at the left-interface during the j^{th} query is such that $y_j \neq \diamond$, then the protocol π did at most L inner queries to the left-interface of the random oracle ${}^{L|}\mathbf{RO}^{|R}$. Since the protocol π is assumed to be deterministic, it follows therefore that when queried on the same sequence of inputs x^k, the emulated protocol π at the right-interface of the random oracle ${}^{L|}\mathbf{RO}^{|R}$ did also at most $L \leq R$ queries and thus $y_j = \tilde{y}_j$ whenever $y_j \neq \diamond$, for all $j \in \{1, \ldots, k\}$. Thus, the decision bit Z output by $\mathbf{D}_{x^k}$ in such a random experiment can never be 1.

Consider now the random experiment where the distinguisher $\mathbf{D}_{x^k}$ interacts instead with the system $\left[{}^{\ell|}\mathbf{RO}^{|r} \ \sigma\right]^{\dashv}$. The probability that the decision bit Z output by $\mathbf{D}_{x^k}$ is 1 in such a random experiment is trivially lower bounded by that of the joint event $Z = 1$ *and* $\left|\mathcal{Q}_\sigma\left(\mathcal{Q}_\pi\left(x^k\right)\right)\right| < k$. Since the distinguisher $\mathbf{D}_{x^k}$ first does $k \leq \ell$ queries to the left-interface of the random oracle ${}^{\ell|}\mathbf{RO}^{|r}$, the answers y^k to such queries are such that $y_j \neq \diamond$, for all $j \in \{1, \ldots, k\}$. Given the event $\left|\mathcal{Q}_\sigma\left(\mathcal{Q}_\pi\left(x^k\right)\right)\right| < k$, at least one of the inputs x_j was never queried by the simulator σ and thus the corresponding answer $\tilde{y}_j$ will differ from y_j with probability at least $1 - 2^{-n}$. $\square$

Acknowledgments. The authors wish to thank Philip Rogaway and the anonymous reviewers for their valuable comments.

Grégory Demay is supported by the Zurich Information Security and Privacy Center (ZISC). Peter Gaži is supported by the European Research Council under an ERC Starting Grant (259668-PSPC). Björn Tackmann is supported by the Swiss National Science Foundation (SNF).

References

[BDPVA08] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. On the Indifferentiability of the Sponge Construction. In Nigel Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 181–197. Springer Berlin Heidelberg, 2008.

[BR93] Mihir Bellare and Phillip Rogaway. Random Oracles Are Practical: A Paradigm for Designing Efficient Protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, CCS '93, pages 62–73, New York, NY, USA, 1993. ACM.

[BR06] Mihir Bellare and Phillip Rogaway. The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 409–426. Springer Berlin Heidelberg, 2006.

[BRT12] Mihir Bellare, Thomas Ristenpart, and Stefano Tessaro. Multi-instance Security and Its Application to Password-Based Cryptography. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 312–329. Springer Berlin Heidelberg, 2012.

[CDMP05] Jean-Sébastien Coron, Yevgeniy Dodis, Cécile Malinaud, and Prashant Puniya. Merkle-Damgård Revisited: How to Construct a Hash Function. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 430–448. Springer Berlin Heidelberg, 2005.

[CMTV15] Sandro Coretti, Ueli Maurer, Björn Tackmann, and Daniele Venturi. From Single-Bit to Multi-Bit Public-Key Encryption via Non-Malleable Codes. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *Theory of Cryptography*, volume 9014 of *Lecture Notes in Computer Science*. Springer, 2015.

[DGHM13] Grégory Demay, Peter Gaži, Martin Hirt, and Ueli Maurer. Resource-Restricted Indifferentiability. In Thomas Johansson and PhongQ. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 664–683. Springer Berlin Heidelberg, May 2013.

[DGMT15] Grégory Demay, Peter Gaži, Ueli Maurer, and Björn Tackmann. Query-Complexity Amplification for Random Oracles. In Anja Lehmann and Stefan Wolf, editors, *ICITS 2015*, volume 9063 of *Lecture Notes in Computer Science*, pages 159–180. Springer Berlin Heidelberg, May 2015.

[DN93] Cynthia Dwork and Moni Naor. Pricing via Processing or Combatting Junk Mail. In ErnestF. Brickell, editor, *Advances in Cryptology — CRYPTO' 92*, volume 740 of *Lecture Notes in Computer Science*, pages 139–147. Springer Berlin Heidelberg, 1993.

[DRST12] Yevgeniy Dodis, Thomas Ristenpart, John Steinberger, and Stefano Tessaro. To Hash or Not to Hash Again? (In)Differentiability Results for H^2 and HMAC. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 348–366. Springer Berlin Heidelberg, 2012.

[HKT11] Thomas Holenstein, Robin Künzler, and Stefano Tessaro. The Equivalence of the Random Oracle Model and the Ideal Cipher Model, Revisited. In *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing*, STOC '11, pages 89–98, New York, NY, USA, 2011. ACM.

[JÖS12] Dimitar Jetchev, Onur Özen, and Martijn Stam. Understanding Adaptivity: Random Systems Revisited. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology – ASIACRYPT 2012*, volume 7658 of *Lecture Notes in Computer Science*, pages 313–330. Springer Berlin Heidelberg, 2012.

[Kal00] B. Kaliski. PKCS #5: Password-Based Cryptography Specification Version 2.0. RFC 2898, RFC Editor, September 2000.

[Mau02] Ueli Maurer. Indistinguishability of Random Systems. In LarsR. Knudsen, editor, *Advances in Cryptology — EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 110–132. Springer Berlin Heidelberg, 2002.

[Mau12] Ueli Maurer. Constructive Cryptography – A New Paradigm for Security Definitions and Proofs. In Sebastian Mödersheim and Catuscia Palamidessi, editors, *Theory of Security and Applications*, volume 6993 of *Lecture Notes in Computer Science*, pages 33–56. Springer Berlin Heidelberg, 2012.

[Mau13] U. Maurer. Conditional Equivalence of Random Systems and Indistinguishability Proofs. In *2013 IEEE International Symposium on Information Theory Proceedings (ISIT)*, pages 3150–3154, July 2013.

[MPR07] Ueli Maurer, Krzysztof Pietrzak, and Renato Renner. Indistinguishability Amplification. In Alfred Menezes, editor, *Advances in Cryptology - CRYPTO 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 130–149. Springer Berlin Heidelberg, 2007.

[MR11] Ueli Maurer and Renato Renner. Abstract Cryptography. In Bernard Chazelle, editor, *The Second Symposium in Innovations in Computer Science, ICS 2011*, pages 1–21. Tsinghua University Press, January 2011.

[MRH04] Ueli Maurer, Renato Renner, and Clemens Holenstein. Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology. In Moni Naor, editor, *Theory of Cryptography*, volume 2951 of *Lecture Notes in Computer Science*, pages 21–39. Springer Berlin Heidelberg, 2004.

[MT79] Robert Morris and Ken Thompson. Password Security: A Case History. *Commun. ACM*, 22(11):594–597, November 1979.

[MT10] Ueli Maurer and Björn Tackmann. On the Soundness of Authenticate-then-encrypt: Formalizing the Malleability of Symmetric Encryption. In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, CCS '10, pages 505–515, New York, NY, USA, 2010. ACM.

[Nak08] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008.

[Nao03] Moni Naor. Moderately Hard Functions: From Complexity to Spam Fighting. In ParitoshK. Pandya and Jaikumar Radhakrishnan, editors, *FST TCS 2003: Foundations of Software Technology and Theoretical Computer Science*, volume 2914 of *Lecture Notes in Computer Science*, pages 434–442. Springer Berlin Heidelberg, 2003.

[PM99] Niels Provos and David Mazieres. A Future-Adaptable Password Scheme. In *USENIX Annual Technical Conference, FREENIX Track*, pages 81–91, 1999.

[RSS11] Thomas Ristenpart, Hovav Shacham, and Thomas Shrimpton. Careful with Composition: Limitations of the Indifferentiability Framework. In KennethG. Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 487–506. Springer Berlin Heidelberg, 2011.

[SHA12] Secure Hash Standard. National Institute of Standards and Technology, NIST FIPS PUB 180-4, U.S. Department of Commerce, 2012.

[SHA14] SHA-3 Standard. National Institute of Standards and Technology (NIST), Draft FIPS Publication 202, U.S. Department of Commerce, April 2014.

[Tac14] Björn Tackmann. *A Theory of Secure Communication*. PhD thesis, ETH Zürich, August 2014.

[TBBC10] Meltem Sönmez Turan, Elaine Barker, William Burr, and Lily Chen. Recommendation for Password-Based Key Derivation. NIST Special Publication 800-132, National Institute of Standards and Technology, December 2010.

A Properties of the Distinguishing Metric

For the composability of the construction notion stated in Definition 1 we need the metric $\widehat{\Delta}(\cdot,\cdot)$ to be non-expanding under the application of converters, i.e. $\widehat{\Delta}(\alpha\mathbf{R},\alpha\mathbf{S}) \leq \widehat{\Delta}(\mathbf{R},\mathbf{S})$ [Mau12, Definition 2 (2)]. This holds, since

$$\begin{aligned}\widehat{\Delta}(\alpha\mathbf{R},\alpha\mathbf{S}) &= \sup_{\mathbf{D}\in\mathcal{D}} \left|\Pr(\mathbf{D}[\alpha\mathbf{R}]^{\dashv}=1) - \Pr(\mathbf{D}[\alpha\mathbf{S}]^{\dashv}=1)\right| \\ &\leq \sup_{\mathbf{D}\in\mathcal{D}} \left|\Pr(\mathbf{D}\mathbf{R}^{\dashv}=1) - \Pr(\mathbf{D}\mathbf{S}^{\dashv}=1)\right| = \widehat{\Delta}(\mathbf{R},\mathbf{S}),\end{aligned}$$

where the inequality holds since the distinguisher can emulate the behavior of α and ignore the outputs once the MBO of α becomes 1.

B Conditional Equivalence and Query-Restricted Systems

We show in the next lemma how the notion of conditional equivalence helps upper-bounding the distinguishing advantage between query-restricted systems. Before doing so, we first recall the definition of a particular non-adaptive distinguisher introduced in [Mau13] and that will be relevant to the statement of Lemma 8. Given an arbitrary distinguisher $\mathbf{D}$ for $(\mathcal{I}\times\mathcal{X},\mathcal{Y})$-systems as well as an $(\mathcal{I}\times\mathcal{X},\mathcal{Y})$-system $\mathbf{S}$, we denote by $[\![\mathbf{DS}]\!]$ the *non-adaptive* distinguisher which interacts with some $(\mathcal{I}\times\mathcal{X},\mathcal{Y}')$-system $\mathbf{R}$ as follows, for some set $\mathcal{Y}'$. At the j^{th} step the distinguisher $\mathbf{D}$ issues a query $(I_j,X_j)\in\mathcal{I}\times\mathcal{X}$ to the system $\mathbf{S}$ and receives Y_j as answer; then the distinguisher $[\![\mathbf{DS}]\!]$ issues exactly the same query (I_j,X_j) to the system $\mathbf{R}$ but completely discards its answer, for all $j\geq 1$. We refer to [Mau13, Fig. 1] for a more pictorial description of the distinguisher $[\![\mathbf{DS}]\!]$.

Lemma 8. *Consider two $(\mathcal{I}\times\mathcal{X},\mathcal{Y}\times\{0,1\})$-systems ${}^{L}|\mathbf{R}|^{R}$ and ${}^{L}|\mathbf{S}|^{R}$, both restricted to L and R queries at their left- and right-interface, respectively, for some $L,R\in\mathbb{N}$. Let $\ulcorner {}^{L}|\mathbf{R}|^{R}\urcorner$ denote the game obtained from the system ${}^{L}|\mathbf{R}|^{R}$ adjoined with an arbitrary MBO. Then, for any distinguisher $\mathbf{D}$,*

$$\ulcorner {}^{L}|\mathbf{R}|^{R}\urcorner \equiv {}^{L}|\mathbf{S}|^{R} \quad\Longrightarrow\quad \widehat{\Delta}^{\mathbf{D}}\left({}^{L}|\mathbf{R}|^{R},{}^{L}|\mathbf{S}|^{R}\right) \leq \widehat{\Gamma}^{[\![\mathbf{D}[{}^{L}|\mathbf{S}|^{R}]^{\dashv}]\!]}\left(\ulcorner {}^{L}|\mathbf{R}|^{R}\urcorner\right),$$

and in particular,

$$\widehat{\Delta}\left({}^{L}|\mathbf{R}|^{R},{}^{L}|\mathbf{S}|^{R}\right) \leq \widehat{\Gamma}^{\mathsf{NA}}\left(\ulcorner {}^{L}|\mathbf{R}|^{R}\urcorner\right).$$

Proof. The proof basically follows by seeing the MBO indicating the restriction on the number of queries as part of the system's output and by applying [Mau13, Th. 3]. More precisely, let us consider an arbitrary distinguisher $\mathbf{D}$ for $(\mathcal{I}\times\mathcal{X},\mathcal{Y}\cup\{\diamond\})$-systems trying to tell apart $\left[{}^{L}|\mathbf{R}|^{R}\right]^{\dashv}$ from $\left[{}^{L}|\mathbf{S}|^{R}\right]^{\dashv}$. Let $\mathbf{D}'$ be a distinguisher for $(\mathcal{I}\times\mathcal{X},\mathcal{Y}\times\{0,1\})$-systems that works as follows: it simulates $\mathbf{D}$ but whenever it receives a response $(y,a)\in\mathcal{Y}\times\{0,1\}$ to any of its queries it forwards instead y' to the distinguisher $\mathbf{D}$, where $y'=y$ if $a=0$, and otherwise $y'=\diamond$. Once the MBO is 1, i.e., $a=1$ and $\mathbf{D}'$ thus forwarded the default value $\diamond$ to $\mathbf{D}$, the distinguisher $\mathbf{D}'$ does not query further the system it was interacting with, but instead replies to any additional query made by $\mathbf{D}$ directly by the dummy symbol $\diamond$. By definition of the distinguisher $\mathbf{D}'$ and of the restricted distinguishing avantage $\widehat{\Delta}^{\mathbf{D}}$ in (1) it follows that

$$\widehat{\Delta}^{\mathbf{D}}\left({}^{L}|\mathbf{R}|^{R},{}^{L}|\mathbf{S}|^{R}\right) = \Delta^{\mathbf{D}'}\left({}^{L}|\mathbf{R}|^{R},{}^{L}|\mathbf{S}|^{R}\right).$$

Since the game $\ulcorner {}^{L}|\mathbf{R}|^{R}\urcorner$ is conditionally equivalent to the system ${}^{L}|\mathbf{S}|^{R}$ by assumption, [Mau13, Th. 3] implies that

$$\Delta^{\mathbf{D}'}\left({}^{L}|\mathbf{R}|^{R},{}^{L}|\mathbf{S}|^{R}\right) \leq \Gamma^{[\![\mathbf{D}'\,{}^{L}|\mathbf{S}|^{R}]\!]}\left(\ulcorner {}^{L}|\mathbf{R}|^{R}\urcorner\right).$$

Note that the distinguisher $\mathbf{D}'$ stops interacting with the system ${}^{L|}\mathbf{S}^{|R}$ once its restriction on the number of queries has been violated. In the random experiment formed by the distinguisher $[\![\mathbf{D}'^{L|}\mathbf{S}^{|R}]\!]$ interacting with the game $\overline{{}^{L|}\mathbf{R}^{|R}}$, both systems $\overline{{}^{L|}\mathbf{R}^{|R}}$ and ${}^{L|}\mathbf{S}^{|R}$ receive the same inputs, and consequently the value of the restriction, which is a deterministic function of the inputs, is the same in both systems. Thus, the distinguisher $[\![\mathbf{D}'^{L|}\mathbf{S}^{|R}]\!]$ stops interacting with the game $\overline{{}^{L|}\mathbf{R}^{|R}}$ as soon as the restriction of the latter has been violated, and since the distinguisher $[\![\mathbf{D}'^{L|}\mathbf{S}^{|R}]\!]$ does not see the output of the game $\overline{{}^{L|}\mathbf{R}^{|R}}$ it follows that

$$\Gamma^{[\![\mathbf{D}'^{L|}\mathbf{S}^{|R}]\!]}\left(\overline{{}^{L|}\mathbf{R}^{|R}}\right) \;=\; \widehat{\Gamma}^{[\![\mathbf{D}'^{L|}\mathbf{S}^{|R}]\!]}\left(\overline{{}^{L|}\mathbf{R}^{|R}}\right) \;.$$

Finally, the distinguisher $\mathbf{D}'$ modifies the outputs of the system ${}^{L|}\mathbf{S}^{|R}$ to exactly correspond to what the system $\left[{}^{L|}\mathbf{S}^{|R}\right]^{\dashv}$ would have output when interacting with $\mathbf{D}$. Since all the queries issued by the distinguisher $[\![\mathbf{D}'^{L|}\mathbf{S}^{|R}]\!]$ originate from the original distinguisher $\mathbf{D}$, it follows that

$$\widehat{\Gamma}^{[\![\mathbf{D}'^{L|}\mathbf{S}^{|R}]\!]}\left(\overline{{}^{L|}\mathbf{R}^{|R}}\right) \;=\; \widehat{\Gamma}^{[\![\mathbf{D}\left[{}^{L|}\mathbf{S}^{|R}\right]^{\dashv}]\!]}\left(\overline{{}^{L|}\mathbf{R}^{|R}}\right) \;.$$

The previous equations and the observation that the distinguisher $[\![\mathbf{D}\left[{}^{L|}\mathbf{S}^{|R}\right]^{\dashv}]\!]$ is non-adaptive lead to the desired result. □

Lemma 8 requires a conditional equivalence statement between query-restricted systems in order to upper bound the corresponding restricted disitnguishing advantage. The next lemma shows that one could equivalently consider a conditional equivalence between unrestricted systems.

Lemma 9. *Consider and $(\mathcal{I}\times\mathcal{X},\mathcal{Y})$-system $\mathbf{S}$ and an $(\mathcal{I}\times\mathcal{X},\mathcal{Y})$-game $\widehat{\mathbf{R}}$ outputting pairs (Y_j,B_j). Let $\overline{{}^{L|}\mathbf{R}^{|R}}$ denote the $(\mathcal{I}\times\mathcal{X},\mathcal{Y}\times\{0,1\})$-game outputting triples (Y_j,A_j,B_j), where the MBO $A_1,A_2,\ldots$ models the restriction on the number of queries, for some $L,R\in\mathbb{N}$. Then,*

$$\widehat{\mathbf{R}} \;\models\; \mathbf{S} \quad\Longleftrightarrow\quad \overline{{}^{L|}\mathbf{R}^{|R}} \;\models\; {}^{L|}\mathbf{S}^{|R} \;.$$

Proof. Note that $\overline{{}^{L|}\mathbf{R}^{|R}}$ is an $(\mathcal{I}\times\mathcal{X},\mathcal{Y}\times\{0,1\}\times\{0,1\})$-system outputting triples (Y_j,A_j,B_j), where the MBO $A_1,A_2,\ldots$ models the restriction on the number of queries and the MBO $B_1,B_2,\ldots$ specifies when the game $\widehat{\mathbf{R}}$ is won. Such an MBO $A_1,A_2,\ldots$ is a (binary) deterministic function of the inputs and it thus follows that if the game $\widehat{\mathbf{R}}$ is conditionally equivalent to the system $\mathbf{S}$, then so is the game $\overline{{}^{L|}\mathbf{R}^{|R}}$ to the system ${}^{L|}\mathbf{S}^{|R}$.

The other direction of the equivalence trivially follows from the fact that each of the two conditional probability distrubtions involved in the left conditional equivalence statement $\widehat{\mathbf{R}} \models \mathbf{S}$ is a marginal distribution of the corresponding one involved in the right conditional equivalence statement $\overline{{}^{L|}\mathbf{R}^{|R}} \models {}^{L|}\mathbf{S}^{|R}$. □