Hardness Preserving Constructions of Pseudorandom Functions

Abhishek Jain[1,⋆], Krzysztof Pietrzak[2,⋆⋆], and Aris Tentes[3,⋆⋆⋆]

[1] UCLA
abhishek@cs.ucla.edu
[2] IST Austria
pietrzak@ist.ac.at
[3] New York University
tentes@cs.nyu.edu

Abstract. We show a hardness-preserving construction of a PRF from any length doubling PRG which improves upon known constructions whenever we can put a non-trivial upper bound q on the number of queries to the PRF. Our construction requires only $O(\log q)$ invocations to the underlying PRG with each query. In comparison, the number of invocations by the best previous hardness-preserving construction (GGM using Levin's trick) is logarithmic in the *hardness* of the PRG.

For example, starting from an exponentially secure PRG $\{0,1\}^n \mapsto \{0,1\}^{2n}$, we get a PRF which is exponentially secure if queried at most $q = \exp(\sqrt{n})$ times and where each invocation of the PRF requires $\Theta(\sqrt{n})$ queries to the underlying PRG. This is much less than the $\Theta(n)$ required by known constructions.

1 Introduction

In 1984, the notion of *pseudorandom functions* was introduced in the seminal work of Goldreich, Goldwasser and Micali [10]. Informally speaking, a pseudorandom function (PRF) is a keyed function $\mathsf{F} : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$, such that no efficient oracle aided adversary can distinguish whether the oracle implements a uniformly random function, or is instantiated with $\mathsf{F}(k,.)$ for a random key $k \leftarrow \{0,1\}^n$. PRFs can be used to realize a shared random function, which has found many applications in cryptography [9,7,8,2,16,15,12].

Goldreich et al. [10] gave the first construction of a PRF from any length-doubling pseudorandom generator $\mathsf{G} : \{0,1\}^n \to \{0,1\}^{2n}$; this is known as the GGM construction. In this work, we revisit this classical result. Although we will state the security of all constructions considered in a precise quantitative way, it helps to think in asymptotic terms to see the qualitative differences between constructions. In the discussion below, we will therefore think of n as a parameter

⋆ Research conducted while at CWI Amsterdam.
⋆⋆ Supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC Starting Grant (259668-PSPC).
⋆⋆⋆ Research conducted while at CWI Amsterdam and IST Austria.

R. Cramer (Ed.): TCC 2012, LNCS 7194, pp. 369–382, 2012.

(and assume the PRG G is defined for all input lengths $n \in \mathbb{N}$, and not just say $n = 128$). Moreover, for concreteness we assume that G is exponentially hard, that is, for some constant $c > 0$ and all sufficiently large n, no adversary of size 2^{cn} can distinguish $G(U_n)$ from U_{2n} (where U_n denotes a variable with uniform distribution over $\{0,1\}^n$) with advantage more than 2^{-cn}. We will also refer to this as "G having cn bits of security".

The GGM construction $\mathsf{GGM}_\mathsf{G} : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$ is **hardness preserving**, which means that if the underlying PRG G has cn bits of security, it has $c'n$ bits of security for some $0 < c' < c$. The domain size $\{0,1\}^m$ can be arbitrary, but the **efficiency** of the construction depends crucially on m as every invocation of GGM_G requires m calls to the underlying PRG G.

Levin [13] proposed a modified construction which improves efficiency for long inputs: first hash the long m-bit input to a short u-bit string using a universal hash function $h : \{0,1\}^m \to \{0,1\}^u$, and only then use the GGM construction on this short u-bit string. The smaller a u we choose, the better the efficiency. If we just want to achieve security against polynomial size adversaries, then a super-logarithmic $u = \omega(\log n)$ will do. But if we care about exponential security and want this construction to be hardness preserving, then we must choose a $u = \Omega(n)$ that is linear in n. Thus, the best hardness-preserving construction of a PRF F^G from a length-doubling PRG G requires $\Theta(n)$ invocations of G for every query to F (unless the domain $m = o(n)$ is *sublinear*, then we can use the basic GGM construction.) In this work we ask if one can improve upon this construction in terms of efficiency. We believe that in this generality, the answer actually is no, and state this explicitly as a conjecture. But our main result is a new construction which dramatically improves efficiency in many practical settings, namely, whenever we can put a bound on the number of queries the adversary can make.

In the discussion above, we didn't treat the number of queries an adversary can make as a parameter. Of course, the size of the adversary considered is an upper bound on the number of queries, but in many practical settings, the number of outputs an adversary can see is tiny compared to its computational resources.

For example consider an adversary of size 2^{cn} who can make only $q = 2^{\sqrt{n}} \ll 2^{cn}$ queries to the PRF. If the domain of the PRF is small, $m = \Theta(\sqrt{n})$, then using GGM we get a hardness-preserving construction with efficiency $\Theta(\sqrt{n})$ (where efficiency is measured by the number of queries to G per invocation of the PRF.) If we want a larger domain $m = \omega(\sqrt{n})$, then the efficiency drops to $m = \omega(\sqrt{n})$. We can get efficiency $\Theta(n)$ regardless of how large m is by using Levin's trick, but cannot go below that without sacrificing hardness preservation.

In this paper we give a hardness-preserving construction which, for any input length m, achieves efficiency $\Theta(\sqrt{n})$. The construction works also for other settings of the parameters. In particular, for $q = 2^{n^\epsilon}$ (note that above we considered the case $\epsilon = 1/2$) we get a construction with efficiency $\Theta(\log q) = \Theta(n^\epsilon)$. Actually, this is only true for $\epsilon \geq 1/2$; whether there exists a hardness-preserving black-box construction with efficiency $\Theta(\log q)$ for $q = 2^{n^\epsilon}$ where $\epsilon < 1/2$ is an interesting open question.

Other Applications. Although we described our result as an improved reduction from PRFs to PRGs, the main idea is more general. Viewing it differently, ours is a new technique for extending the domain of a PRF. If we apply our technique to PRFs with an input domain of length ℓ bits, Levin's trick would require roughly a domain of size ℓ^2 to achieve a comparable quality of hardness preservation.

This technique can be used to give more efficient constructions in other settings, for example to the work of Naor and Reingold [18] who construct PRFs computable by low depth circuits from so called pseudorandom synthesizer (PRS), which is an object stronger than a PRG, but weaker than a full blown PRF. Very briefly, [18] gives a hardness-preserving construction of a PRF from PRS which can be computed making $\Theta(n)$ queries to the PRS in depth $\Theta(\log n)$ (GGM also makes $\Theta(n)$ queries, but sequentially, i.e. has depth $\Theta(n)$; on the other hand, GGM only needs a PRG, not a PRS as building block). Our domain extension technique can also be used to improve on the Naor-Reingold construction, and improves efficiency from $\Theta(n)$ to $\Theta(\log q) = \Theta(n^\epsilon)$ whenever one can put an upper bound $q = 2^{n^\epsilon}$ $(\epsilon \geq 1/2)$ on the number of adversarial queries.

Subsequent to [18], several number-theoretic constructions of PRFs have been proposed, inspired by the PRS based construction and GGM [19,20,14,5,1]. In particular, in [19], Naor and Reingold gave an efficient construction of a PRF from the DDH assumption that requires only n multiplications and one exponentiation instead of the n exponentiations required for GGM or the PRS based construction. This is achieved by exploiting particular properties of the underlying assumptions like the self reducibility of DDH. Our technique does not seem to be directly applicable to improve upon these constructions [19].

The Construction. Before we describe our construction in more detail, it is instructive to see why the universal hash-function $h : \{0,1\}^m \to \{0,1\}^u$ used for Levin's trick must have range $u = \Omega(n)$ to be hardness-preserving. Consider any two queries x_i and x_j made by the adversary. If we have a collision $h(x_i) = h(x_j)$ for the initial hashing, then the outputs $\mathsf{GGM}_\mathsf{G}(k, h(x_i)) = \mathsf{GGM}_\mathsf{G}(k, h(x_j))$ of the PRF will also collide. To get exponential security, we need this collision probability to be exponentially small. The probability for such a collision depends on the range u and is $\mathsf{Pr}_h[h(x_i) = h(x_j)] = 2^{-u}$. So we must choose $u = \Theta(n)$ to make this term exponentially small.

Similar to Levin's trick, we also use a hash function $h : \{0,1\}^m \to \{0,1\}^t$ to hash the input down to $t = 3 \log q$ bits (Recall that q is an upper bound on the queries to the PRF, so if say $q = 2^{\sqrt{n}}$, then $t = 3\sqrt{n}$.) As discussed earlier, the collision probability with such a short output length will not be exponentially small. However, we can prove something weaker, namely, if h is t-wise independent, then the probability that we have a $t+1$-wise collision (i.e. any $t+1$ of the q inputs hash down to the same value.) is exponentially small.

Next, the hashed value $x'_i = h(x_i)$ is used as input to the standard GGM PRF to compute $x''_i := \mathsf{GGM}_\mathsf{G}(k, x'_i)$. Note, however, that we can't simply set x''_i as the output of our PRF because several of the inputs $x_1, \ldots, x_q$ can be mapped by h to the same x', and thus also the same x'', which would not look random at all.

We solve this problem by using $x_i'' = \mathsf{GGM}_{\mathsf{G}}(k, h(x_i))$ to sample a t-wise independent hash function h_i. The final output $z_i := h_i(x_i)$ is then computed by hashing the original input x_i using this h_i. Note that with very high probability, for every i, at most $t' \leq t$ different $x_{i_1}, \ldots, x_{i_{t'}}$ will map to the same t-wise independent h_i. Thus, the corresponding outputs $h_i(x_{i_1}), \ldots, h_i(x_{i_{t'}})$ will be random.

The invocation of the GGM construction and the sampling of h_i from x_i'' can both be done with $\Theta(t)$ invocations of G, thus we get an overall efficiency of $\Theta(\sqrt{n})$.

2 Preliminaries

Variables, Sets and Sampling. By lowercase letters we denote values and bit strings, by uppercase letters we denote random variables and by uppercase calligraphic letters we denote sets. Specifically, by U_m we denote the random variable which takes values uniformly at random from the set of bit strings of length m and by $\mathcal{R}_{m,n}$ the set of all functions $F : \{0,1\}^m \mapsto \{0,1\}^n$. If $\mathcal{X}$ is a set, then by $\mathcal{X}^t$ we denote the t'th direct product of $\mathcal{X}$, i.e., $(\mathcal{X}_1, \ldots, \mathcal{X}_t)$ of t identical copies of $\mathcal{X}$. If X is a random variable, then by $X^{(t)}$ we denote the random variable which consists of t independent copies of X. By $x \leftarrow X$ we denote the fact that x was chosen according to the random variable X and analogously by $x \leftarrow \mathcal{X}$, that x was chosen uniformly at random from set $\mathcal{X}$.

Computational/Statisical Indistinguishability. For random variables X_0, X_1 distributed over some set $\mathcal{X}$, we write $X_0 \sim X_1$ to denote that they are identically distributed, we write $X_0 \sim_\delta X_1$ to denote that they have statistical distance δ, i.e. $\frac{1}{2}\sum_{x\in\mathcal{X}} |\mathsf{Pr}_{X_0}[x] - \mathsf{Pr}_{X_1}[x]| \leq \delta$, and $X_0 \sim_{(\delta,s)} X_1$ to denote that they are (δ, s) indistinguishable, i.e. for all distinguishers D of size at most $|D| \leq s$ we have $\sum_{x\in\mathcal{X}} |\mathsf{Pr}_{X_0}[D(x) \to 1] - \mathsf{Pr}_{X_1}[D(x) \to 1]| \leq \delta$. In informal discussions we will also use $\sim_s$ to denote statistical closeness (i.e. $\sim_\delta$ for some "small" δ) and $\sim_c$ to denote computational indistinguishability (i.e. $\sim_{(\delta,s)}$ for some "large" s and "small" δ.)

3 Definitions

We will need two information theoretic notions of hash functions, namely, δ-universal and t-wise independent hash functions. Informally, a hash function is t-wise independent if its output is uniform on any t distinct inputs. A function is δ-universal if any two inputs collide with probability at most δ.

Definition 1 (almost universal hash function). *For $\ell, m, n \in \mathbb{Z}$, a function $h : \{0,1\}^\ell \times \{0,1\}^m \to \{0,1\}^n$ is δ-almost universal if for any $x \neq x' \in \{0,1\}^m$*

$$\mathsf{Pr}_{k\leftarrow\{0,1\}^\ell}[h_k(x) = h_k(x')] \leq \delta$$

Universal hash functions were studied in [6,21], who also gave explicit constructions.

Proposition 1. *For any m, n there exists a 2^{-n+1}-universal hash function with key length $\ell = 4(n + \log m)$. Further, no such function can be δ-universal for $\delta < 2^{-n-1}$.*

Definition 2 (t-wise independent hash function family). *For $\ell, m, n, t \in \mathbb{Z}$, a function $h : \{0,1\}^\ell \times \{0,1\}^m \to \{0,1\}^n$ is t-wise independent, if for every t distinct inputs $x_1, \ldots, x_t \in \{0,1\}^m$ and a random key $k \leftarrow \{0,1\}^\ell$ the outputs are uniform, i.e.*

$$h_k(x_1) \| \ldots \| h_k(x_t) \sim U_n^{(t)}$$

Proposition 2. *For any $t, m, n \leq m$ there exits a t-wise independent hash function with key length $\ell = m \cdot t$.*

Remark 1. Note that 2-wise independence implies 2^{-n}-universality. The reason to consider the notion of δ-universality for $\delta > 2^{-n}$ is that it can be achieved with keys of length linear in the output, as opposed to the input.

Definition 3 (PRG[4,22]). *A length-increasing function $\mathsf{G} : \{0,1\}^n \mapsto \{0,1\}^m$ $(m > n)$ is a (δ, s)-hard pseudorandom generator if*

$$\mathsf{G}(U_n) \sim_{(\delta,s)} U_m$$

We say G has σ **bits of security** *if G is $(2^{-\sigma}, 2^\sigma)$-hard. G is* **exponentially hard** *if it has cn bits of security for some $c > 0$, and G is* **sub-exponentially hard** *if it has cn^ϵ bits of security for some $c > 0, \epsilon > 0$.*

The following lemma, which follows from a standard hybrid argument, will be useful.

Lemma 1. *If $\mathsf{G} : \{0,1\}^n \mapsto \{0,1\}^m$ is a (δ, s)-hard* PRG *of size $|\mathsf{G}| = s'$, then for any $q \in \mathbb{N}$*

$$\mathsf{G}(U_n)^{(q)} \sim_{(q \cdot \delta, s - q \cdot s')} U_m^{(q)}$$

Definition 4 (PRF[10]). *A function $\mathsf{F} : \{0,1\}^\ell \times \{0,1\}^m \to \{0,1\}^n$ is a (q, δ, s)-hard pseudorandom function (PRF) if for every oracle aided distinguisher D^* of size $|D^*| \leq s$ making at most q oracle queries*

$$\left| \mathsf{Pr}_{k \leftarrow \{0,1\}^\ell}[D^{\mathsf{F}(k,.)} \to 1] - \mathsf{Pr}_{f \leftarrow \mathcal{R}_{m,n}}[D^{f(.)} \to 1] \right| \leq \delta$$

F has σ bits of security against q queries if F is $(q, 2^{-\sigma}, 2^\sigma)$ secure.

If q is not explicitly specified, it is unbounded (the size 2^σ of the distinguisher considered is a trivial upper bound on q.)

3.1 The GGM Construction

Goldreich, Goldwasser and Micali [10] gave the first construction of a PRF from any length doubling PRG. We describe their simple construction below.

For a length-doubling function $\mathsf{G} : \{0,1\}^n \to \{0,1\}^{2n}$ and $m \in \mathbb{N}$, let $\mathsf{GGM}_\mathsf{G} : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$ denote the function

$\mathsf{GGM}_\mathsf{G}(k, x) = k_x$ where k_x is recursively defined as $k_\varepsilon = k$ and $k_{a\|0}\|k_{a\|1} := \mathsf{G}(k_a)$

Proposition 3 ([10]). *If* G *is a* $(\delta_\mathsf{G}, s_\mathsf{G})$*-hard PRG, then for any* $m, q \in \mathbb{N}$, $\mathsf{GGM}_\mathsf{G} : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$ *is a* (q, δ, s)*-hard PRF where*

$$\delta = m \cdot q \cdot \delta_\mathsf{G} \qquad s = s_\mathsf{G} - q \cdot m \cdot |\mathsf{G}| \tag{1}$$

3.2 Levin's Trick

One invocation of the GGM construction $\mathsf{GGM}_\mathsf{G} : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$ requires m invocations of the underlying PRG G, so the efficiency of the PRF depends linearly on the input length m. Levin observed that the efficiency can be improved if one first hashes the input using a universal hash function. Using this trick one gets a PRF on long m-bit inputs at the cost of evaluating a PRF on "short" u bit inputs plus the cost of hashing the m-bit string down to u bits.[1]

Proposition 4 (Levin's trick). *Let* $h : \{0,1\}^\ell \times \{0,1\}^m \to \{0,1\}^u$ *be a* δ_h*-universal hash function and* $\mathsf{F} : \{0,1\}^{\ell'} \times \{0,1\}^u \to \{0,1\}^n$ *be a* $(q, \delta_\mathsf{F}, s)$*-hard PRF, then the function* $\mathsf{F}^h : \{0,1\}^{\ell+\ell'} \times \{0,1\}^m \to \{0,1\}^n$ *defined as*

$$\mathsf{F}^h(k_\mathsf{F}||k_h, x) := \mathsf{F}(k_\mathsf{F}, h(k_h, x))$$

is a (q, δ, s)*-hard PRF where*

$$\delta = q^2 \cdot \delta_h + \delta_\mathsf{F} \qquad s = s_\mathsf{F} - q \cdot |h| \tag{2}$$

3.3 Hardness Preserving and Good Constructions

Definition 5 (Hardness Preserving Construction). *A construction* F^* *of a PRF from a PRG is* **hardness preserving** *up to* $q = q(n)$ *queries, if for every constant* $c > 0, \epsilon > 0$ *there is a constant* $c' > 0$ *and* $n' \in \mathbb{Z}$ *such that for all* $n \geq n'$*: if* G *is of polynomial size and has* cn^ϵ *bits of security,* F *has* $c'n^\epsilon$ *bits of security against* q *queries. It is hardness preserving if it is hardness preserving for any* q*.*

If the above holds for every $c' < c$*, we say that it is* **strongly** *hardness preserving.*

[1] As universal hash functions are non-cryptographic primitives, hashing is generally much cheaper than evaluating pseudorandom objects.

Proposition 5. *The GGM construction is hardness preserving, more concretely (1) if* G *has* cn^ϵ *bits of security,* GGM_G *has* $c'n^\epsilon$ *bits of security for any* $c' < c/2$ *(2) GGM for* $q = n^{\epsilon'}$, $\epsilon' < \epsilon$ *queries is strongly hardness preserving.*

Proof. By eq.(1), if G has cn^ϵ bits of security, then the GGM construction has

$$\min\{cn^\epsilon - \log(q) - \log(m), cn^\epsilon - \log|\mathsf{G}| - \log(m)\} \tag{3}$$

bits of security. To see (1), we observe that for any $c' < c/2$ eq.(3) is $c'n^\epsilon$ for sufficiently large n as required (using that m and $|\mathsf{G}|$ are polynomial in n and $q = 2^{c'n}$.) To see (2) observe that for $\log(q) = n^{\epsilon'}$ where $\epsilon' < \epsilon$, the term eq.(3) is $c'n^\epsilon$ for sufficiently large n and every $c' < c$.

Recall that one invocation of GGM requires m invocations of the underlying PRG, where m must be at least $\lceil \log(q) \rceil$. We conjecture that $\Omega(\log(q))$ invocations are necessary for any hardness preserving construction.

Conjecture 1. Any construction[2] $\mathsf{F}^\mathsf{G}(.,.) : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$ that preserves hardness for q queries and has a black-box security proof must make $\Omega(\log q)$ invocation to G per invocation of F^G.

In the appendix we give some intuition as why we believe this conjecture holds. We show that the standard black-box security proof technique as used e.g. for GGM will not work for constructions making $o(\log q)$ invocations.

Definition 6 ((Very) Good Construction). *We call a construction as in Definition 5* **good** *for q queries, if it is hardness preserving up to q queries and each invocation of* F^G *results in* $O(\log q)$ *invocations of* G*. We call it very good, if it is even strongly hardness-preserving.*

Thus, GGM_G is good as long as the domain m is in $O(\log q)$, but not if we need a large domain $m = \omega(\log q)$. Let's look at Levin's construction.

Proposition 6. *The GGM construction with Levin's trick* $\mathsf{GGM}^h_\mathsf{G}$ *(with* $h : \{0,1\}^\ell \times \{0,1\}^m \to \{0,1\}^u$ *as in Proposition 4) is hardness preserving if and only if u is linear in the security of the underlying PRG (e.g. u has to be linear in n if* G *is exponentially hard.)*

Proof. For concreteness, we assume G is exponentially hard, the proof is easily adapted to the general case. The number of queries to G per invocation of $\mathsf{GGM}^h_\mathsf{G}$ is u, where $\{0,1\}^u$ is the range of the δ_h universal hash function. By eq.(2), $\mathsf{GGM}^h_\mathsf{G}$ has security $\delta = q^2 \cdot \delta_h + \delta_{\mathsf{GGM}_\mathsf{G}}$. To preserve exponential hardness, δ must be exponentially small. So also $\delta_h \leq \delta$ must be exponentially small. By Proposition 1 $\delta_h > 2^{-u-1}$, thus u must be linear in n.

[2] We restrict the key length to n bits. This is not much of a restriction, as one can use G to expand the key. If we allow polynomially sized keys directly, then the conjecture would be wrong for polynomial q as the key could just contain the entire function table.

Summing up, the GGM construction is hardness preserving for any q, but only good if the domain is restricted to $m = O(\log q)$ bits. By using Levin's trick, we can get a hardness preserving construction where $u = \Theta(n)$ (if G is exponentially hard), but this will only be a good construction for q queries if q is also exponentially large.

In a practical setting we often know that a potential adversary will never see more than, say $2^{\sqrt{n}}$ outputs of a PRF F^{G}. If we need a large domain for the PRF, and would like the construction to preserve the exponential hardness of the underlying PRG G, then the best we can do is to use GGM with Levin's trick, which will invoke G a linear in n number of times with every query. Can we do better? If Conjecture 1 is true, then one needs $\Theta(\sqrt{n})$ invocations, which is much better than $\Theta(n)$. The main result in this paper is a construction matching this (conjectured) lower bound.

4 Our Construction

Let $\mathsf{G} : \{0,1\}^n \to \{0,1\}^{2n}$ be a length doubling function. For $e \in \mathbb{N}$, we denote with $\mathsf{G}^e : \{0,1\}^n \to \{0,1\}^{en}$ the function that expands an n bit string to a

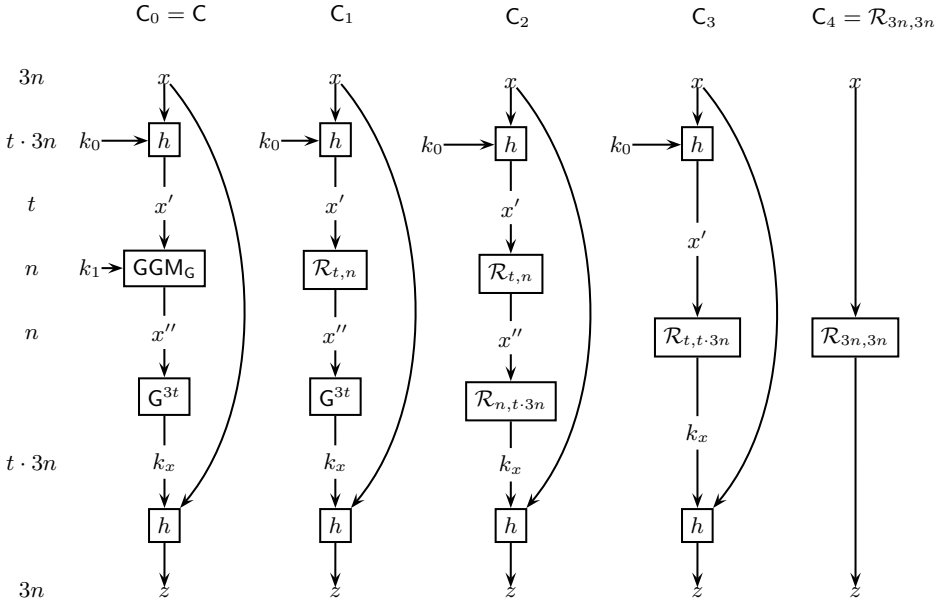

Fig. 1. The leftmost figure illustrates our construction $\mathsf{C}(.,.)$ using key $k = k_0 \| k_1$ on input x. The numbers $3n, t \cdot 3n, \ldots$ on the left indicate the bit-length of the corresponding values $x, k_0, \ldots$. The remaining figures illustrate the games from the proof of Theorem 1. $t = 3\log(q)$ is a parameter which depends on the number of queries q we allow.

en bits string using $e-1$ invocations of G (this can be done sequentially, or in parallel in depth $\lceil \log e \rceil$.) We will use the following simple lemma which follows by a standard hybrid argument.

Lemma 2. *Let G be a (δ, s)-hard PRG, then G^e is a $(e \cdot \delta, s - e \cdot |\mathsf{G}|)$-hard PRG.*

Further, our constructions uses a t-wise independent (cf. Proposition 2) hash function.

$$h : \{0,1\}^{t \cdot 3n} \times \{0,1\}^{3n} \to \{0,1\}^{3n}$$

Our construction $\mathsf{C}^{\mathsf{G}} : \{0,1\}^{t \cdot 3n+n} \times \{0,1\}^{3n} \to \{0,1\}^{3n}$ of a PRF which will be good for large ranges of q, on input a key $k = k_0 \| k_1$ (where $k_0 \in \{0,1\}^{t \cdot 3n}$ and $k_0 \in \{0,1\}^n$) and $x \in \{0,1\}^{3n}$, computes the output as ($X_{|t}$ denotes the t bit prefix of X.)

$$\mathsf{C}(k, x) = h(\mathsf{G}^{3t}(\mathsf{GGM}_{\mathsf{G}}(k_1, h(k_0, x)_{|t})) \ , \ x)$$

Remark 2 (About the domain size and key-length). This construction has a domain of $3n$ bits and key-length $t \cdot 3n + n$. We can use Levin's trick to expand the domain to any m bits, and this will not affect the fact that the construction is good: by eq.(2) we get an additional $q^2 \cdot \delta_h = q^2/2^{3n-1}$ term in the distinguishing advantage, which can be ignored compared to the other terms.

We can also use a short n-bit key (like in plain GGM) and then expand it to a longer $t \cdot 3n + n$ bit key with every invocation (if we use Levin's trick we will need an extra $4(3n + \log m)$ bits.) This also will preserve the fact that the construction is good.

Theorem 1 (Main Theorem). *If G is a $(\delta_{\mathsf{G}}, s_{\mathsf{G}})$-hard PRG, then C^{G} is a (q, δ, s)-secure PRF where*

$$\delta = 4 \cdot q \cdot t \cdot \delta_{\mathsf{G}} + q^2/2^n + q^t/2^{t^2} \qquad s = s_{\mathsf{G}} - q \cdot |\mathsf{C}^{\mathsf{G}}| - q \cdot 3 \cdot t \cdot n \cdot |\mathsf{G}| \qquad (4)$$

Before we prove this Theorem, let's see what it implies for some concrete parameters. Assume G is $(\delta_{\mathsf{G}} = 2^{-cn}, s_{\mathsf{G}} = 2^{cn})$-hard and we want security against $q = 2^{\sqrt{n}}$ queries. If we set $t := 3 \log q$ then the construction is very good (cf. Def. 6):

- It makes $7t = 21 \log(q) = O(\log q)$ invocation to G per query to C^{G}.
- C^{G} is strongly hardness preserving for $q = 2^{\sqrt{n}}$ queries. By eq.(4) we get

$$\delta < 2^{-cn+2+\sqrt{n}+\log(3\sqrt{n})} \qquad s \geq 2^{cn} - 2^{\sqrt{n}} \cdot |\mathsf{C}^{\mathsf{G}}|$$

  If $|\mathsf{C}^{\mathsf{G}}|$ is polynomial in n (which is the only case we care about), then by the above equation, for every $c' < c$, we have $\delta \leq 2^{-c'n}$ and $s \geq 2^{c'n}$ for sufficiently large n as required by Definition 5.

The above argument works for any $q = 2^{n^\epsilon}$ where $0.5 \leq \epsilon < 1$. It also works if q is unbounded (i.e. $\epsilon = 1$), but we only get normal (and not strong) hardness-preservation. The argument fails for $\epsilon < 0.5$, that is whenever $q = 2^{o(\sqrt{n})}$. Technically, the reason is that the $q^t/2^{t^2}$ term in eq.(4) is not exponentially small (as

required for hardness-preservation) when we set $t = O(\log q)$ (as required for a good construction.) It is an interesting open question if an optimal and hardness preserving construction with range $\omega(\log(q))$ exits for any $q = 2^{o\sqrt{n}}$. Summing up, we get the following corollary of Theorem 1

Corollary 1. *For any $0 < \delta \leq 1$ and $\epsilon \in [\delta/2, \delta[$, the construction C^{G} (setting $t := 3\log(q)$) is very good for $q = 2^{n^\epsilon}$ queries for any G with cn^δ bits of security (for any $c > 0$.) It is good for $\epsilon = \delta$.*

Proof (Proof of Theorem 1). Let D^* be any q-query distinguisher of size s. We denote with C_0 our construction C, with $\mathsf{C}_4 = \mathcal{R}_{3n,3n}$ a random function and with $\mathsf{C}_1, \mathsf{C}_2, \mathsf{C}_3$ intermediate constructions as shown in Figure 1. With $p(i)$ we denote the probability that $\mathsf{D}^{\mathsf{C}_i(.)}$ outputs 1 (where e.g. in C_0 the probability is over the choice of $k_0 \| k_1$, in C_1 the probability is over the choice of k_0 and $f \leftarrow \mathcal{R}_{t,n}$, etc.)

Note that the advantage of D^* in breaking C is $\delta = |p(0) - p(4)|$, to prove the theorem we will show that

$$|p(0) - p(4)| = \left| \sum_{i=0}^{3} p(i) - p(i+1) \right| \leq \sum_{i=0}^{3} |p(i) - p(i+1)| \leq 4 \cdot q \cdot t \cdot \delta_{\mathsf{G}} + q^2/2^n + q^t/2^{t^2}$$

The last step follows from the four claims below.

Claim. $|p(0) - p(1)| \leq q \cdot t \cdot \delta_{\mathsf{G}}$

Proof (Proof of Claim). Assume $|p(0) - p(1)| > q \cdot t \cdot \delta_{\mathsf{G}}$. We will construct a distinguisher D_1 for $\mathsf{GGM}_{\mathsf{G}}$ and $\mathcal{R}_{t,n}$, which is of size $s_{\mathsf{G}} - q \cdot t \cdot |\mathsf{G}|$ and has advantage $> q \cdot t \cdot \delta_{\mathsf{G}}$, contradicting Proposition 3. $\mathsf{D}_1^{\mathcal{O}(.)}$ chooses a random $k_0 \in \{0,1\}^{3tn}$, and then runs D^* where it answers its oracle queries by simulating C (using k_0), but replacing the $\mathsf{GGM}_{\mathsf{G}}$ invocation with its oracle $\mathcal{O}(.)$. In the end D_1 outputs the same as D. If $\mathcal{O}(.) = \mathsf{GGM}_G(k_1, .)$ (for some random k_1) then this simulates C_0, and if $\mathcal{O}(.) = \mathcal{R}_{t,n}$ it will simulate C_1. Thus D_1 will distinguish $\mathsf{GGM}_{\mathsf{G}}$ and $\mathcal{R}_{t,n}$ with exactly the same advantage $> q \cdot t \cdot \delta_{\mathsf{G}}$ that D has for C_0 and C_1.

Claim. $|p(1) - p(2)| \leq q \cdot 3 \cdot t \cdot \delta_{\mathsf{G}}$

Proof (Proof of Claim). Assume $|p(1) - p(2)| > q \cdot 3 \cdot t \cdot \delta_{\mathsf{G}}$. We will construct a distinguisher D_2 which is of size $s_{\mathsf{G}} - q \cdot 3 \cdot n \cdot t \cdot |\mathsf{G}|$ who can distinguish q-tuples of samples of U_{3tn} from $\mathsf{G}^{3t}(U_n)$ with advantage $> q \cdot 3 \cdot t \cdot \delta_{\mathsf{G}}$. Using a standard hybrid argument this then gives a distinguisher D_2' who distinguishes a single sample of U_{3tn} from $\mathsf{G}^{3t}(U_n)$ with advantage $> q \cdot 3 \cdot t \cdot \delta_{\mathsf{G}}/q = 3 \cdot t \cdot \delta_{\mathsf{G}}$, contradicting Lemma 2.

D_2 on input $v_1, \ldots, v_q \in \{0,1\}^{3tn}$, runs D^* and answers its oracle queries by simulating C_1, but replacing the output of G^{3t} with the v_i's (using a fresh v_i for every query, except if x' appeared in a previous query, then it uses the same v_i as in this previous query. If the v_i's have distribution $\mathsf{G}^{3t}(U_n)$ this perfectly simulates C_1, and if they have distribution U_{3tn} this simulates C_2. So D_2 has the same distinguishing advantage as D^* has for C_1 and C_2.

The proofs of the final two claims are completely information theoretic.

Claim. $|p(2) - p(3)| \leq q^2/2^n$

Proof (Proof of Claim). We claim that the distinguishing advantage of any (even computationally unbounded) q-query distinguisher for C_2 and C_3 is $\leq q^2/2^n$.

We get C_3 from C_2 by replacing the two nested uniformly random functions $f_2(.) = \mathcal{R}_{n,t\cdot 3n}(\mathcal{R}_{t,n}(.))$ with a single $f_3(.) = \mathcal{R}_{t,t\cdot 3n}(.)$. As each invocation of C_2 results in exactly one invocation of $f_2(.)$, the distinguishing advantage of the best q-query distinguisher for C_2 from C_3 can be upper bounded by the distinguishing advantage of the best such distinguisher for $f_2(.)$ and $f_3(.)$.

Let $\mathcal{E}$ denote the event that the q distinct queries $x'_1, \ldots, x'_q$ to $f_2(.) = \mathcal{R}_{n,t\cdot 3n}(\mathcal{R}_{t,n}(.))$ do not contain a collision on the inner function, i.e. $\mathcal{R}_{t,n}(x'_i) \neq \mathcal{R}_{t,n}(x'_j)$ for all $x'_i \neq x'_j$. Conditioned on $\mathcal{E}$, the outputs of f_2 and f_3 have the same distribution, namely $U^{[q]}_{t\cdot 3n}$. Using this observation, we can bound (using e.g. Theorem 1.(i) in [17] or the "fundamental Lemma" from [3])[3] the distinguishing advantage of any q-query distinguisher for f_2 and f_3 by the probability that one can make the event $\mathcal{E}$ fail. This is the probability that q uniformly random elements from $\{0,1\}^n$ (i.e. the outputs of the inner function) contain a collision. This probability can be upper bounded as $q^2/2^n$.

Claim. $|p(3) - p(4)| \leq q^t/2^{t^2}$

Proof (Proof of Claim). We claim that the distinguishing advantage of any (even computationally unbounded) q-query distinguisher for C_3 and C_4 is $\leq q^t/2^{t^2}$. We will first prove this only for non-adaptive distinguishers. To show security against adaptive adversaries, security against adaptive adversaries will then follow by a result from [17].

Let $x_1, \ldots, x_q$ denote the q distinct queries non-adaptively chosen by D^*. Let $\mathcal{E}$ denote the event which holds if there is no $t+1$-wise collision after the evaluation of the initial hash function h in C_3. That is, there is no subset $\mathcal{I} \subseteq [q]$ of size $|\mathcal{I}| = t+1$ such that $h(k_0, x_i) = h(k_0, x_j)$ for all $i, j \in \mathcal{I}$. Below we show that conditioned on $\mathcal{E}$, the outputs $y_1, \ldots, y_q$ (where $y_i := \mathsf{C}_3(x_i)$) are uniformly random, and thus have the same distribution like the outputs of C_4. Using Theorem 1.(i) in [17] or the "fundamental Lemma" from [3], this means we can upper bound the distinguishing advantage of any non-adaptive distinguisher for C_3 and C_4 by the the probability that the event $\mathcal{E}$ fails to hold. Which means we have a $t+1$ wise collision in q t-wise independent strings over $\{0,1\}^t$. This can be upper bounded as $q^t/2^{t^2}$.[4]

We now show that the outputs of C_3 are uniform conditioned on $\mathcal{E}$. Consider a subset $\mathcal{J} \subseteq [q]$ with $|\mathcal{J}| \leq t$ such that $h(k_0, x_i) = h(k_0, x_j) = a$ for all

[3] Informally, the statement we use is the following: given two systems F and G and an event $\mathcal{E}$ defined for F, if F conditioned on $\mathcal{E}$ behaves exactly as G, then distinguishing F from G is at least as hard as making the event $\mathcal{E}$ fail.

[4] The probability that any t particular strings in $\{0,1\}^t$ collide is exactly $(2^{-t})^{t-1} = 2^{-t^2+t}$, we get the claimed bound by taking the union bound over all $q^t/t!$ possible t-element subsets of the q element set.

$i, j \in \mathcal{J}$. The fact that $\mathcal{R}_{t,3tn}(a)$ is a uniformly random together with the fact that h is a t-wise independent hash function implies that the joint distribution of $(h(a, x_i))_{i\in\mathcal{J}}$ follows the uniform distribution. Now let $\mathcal{J}_1, \ldots, \mathcal{J}_{q'}$ be the subsets of $[q]$ of size at most t, such that for $j \in \mathcal{J}_i$ $h(k_0, x_j) = a_i$ and all a_i's are distinct. The fact that $(\mathcal{R}_{t,3nt}(a_i))_{i\in[q']}$ follows the uniform distribution and $\mathcal{J}_1, \ldots, \mathcal{J}_{q'}$ are of size at most t implies that $(h(\mathcal{R}_{t,3nt}(a_i), x_i))_{i\in[q]}$ follows the uniform distribution as well.

So far, we only established the indistinguishability of C_3 and C_4 against non-adaptive distinguishers. We get the same bound for adaptive distinguishers using Theorem 2 from [17], which (for our special case) states that adaptivity does not help if the outputs of the system (C_3 in our case) are uniform conditioned on the event we want to provoke. Very recently [11] found that the precondition stated in[17] is not sufficient, but one additionally requires that the probability of the event failing is independent of the outputs observed so far. Fortunately in our case (and also for all applications in [17]) this stronger precondition is easily seen to be satisfied.

References

1. Banerjee, A., Peikert, C., Rosen, A.: Pseudorandom functions and lattices. In: EUROCRYPT (2012)
2. Bellare, M., Goldwasser, S.: New Paradigms for Digital Signatures and Message Authentication Based on Non-interactive Zero Knowledge Proofs. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 194–211. Springer, Heidelberg (1990)
3. Bellare, M., Rogaway, P.: The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 409–426. Springer, Heidelberg (2006)
4. Blum, M., Micali, S.: How to generate cryptographically strong sequences of pseudo random bits. In: FOCS, pp. 112–117 (1982)
5. Boneh, D., Montgomery, H.W., Raghunathan, A.: Algebraic pseudorandom functions with improved efficiency from the augmented cascade. In: ACM Conference on Computer and Communications Security, pp. 131–140 (2010)
6. Carter, L., Wegman, M.N.: Universal classes of hash functions. J. Comput. Syst. Sci. 18(2), 143–154 (1979)
7. Goldreich, O.: Two Remarks Concerning the Goldwasser-Micali-Rivest Signature Scheme. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 104–110. Springer, Heidelberg (1987)
8. Goldreich, O.: Towards a theory of software protection and simulation by oblivious rams. In: STOC, pp. 182–194 (1987)
9. Goldreich, O., Goldwasser, S., Micali, S.: On the Cryptographic Applications of Random Functions. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 276–288. Springer, Heidelberg (1985)
10. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions. J. ACM 33(4), 792–807 (1986)
11. Jetchev, D., Özen, O., Stam, M.: Probabilistic analysis of adaptive adversaries revisited. Manuscript in preparation (2011)
12. Katz, J., Lindell, Y.: Introduction to Modern Cryptography

13. Levin, L.A.: One-way functions and pseudorandom generators. Combinatorica 7(4), 357–363 (1987)
14. Lewko, A.B., Waters, B.: Efficient pseudorandom functions from the decisional linear assumption and weaker variants. In: ACM Conference on Computer and Communications Security, pp. 112–120 (2009)
15. Luby, M.: Pseudorandomness and cryptographic applications. Princeton computer science notes. Princeton University Press, Princeton (1996)
16. Luby, M., Rackoff, C.: A Study of Password Security. In: Pomerance, C. (ed.) CRYPTO 1987. LNCS, vol. 293, pp. 392–397. Springer, Heidelberg (1988)
17. Maurer, U.M.: Indistinguishability of Random Systems. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 110–132. Springer, Heidelberg (2002)
18. Naor, M., Reingold, O.: Synthesizers and their application to the parallel construction of psuedo-random functions. In: FOCS, pp. 170–181 (1995)
19. Naor, M., Reingold, O.: Number-theoretic constructions of efficient pseudo-random functions. In: 38th Annual Symposium on Foundations of Computer Science, pp. 458–467. IEEE Computer Society Press (October 1997)
20. Naor, M., Reingold, O., Rosen, A.: Pseudo-random functions and factoring (extended abstract). In: STOC, pp. 11–20 (2000)
21. Wegman, M.N., Carter, L.: New hash functions and their use in authentication and set equality. J. Comput. Syst. Sci. 22(3), 265–279 (1981)
22. Yao, A.C.-C.: Theory and applications of trapdoor functions (extended abstract). In: FOCS, pp. 80–91 (1982)

A Intuition for Conjecture 1

We can think of the GGM construction with domain $\{0,1\}^m$ as a tree, where the outputs are leaves at depth m. More generally, we can think of any construction F^{G} as a directed loop-free graph, which is separated in layers. Each invocations starts at the root which holds the secret key K, and the computation follows a path, crossing layers, where the path within each layer contains at most one invocation of G.

We can define the "entropy" of a layer, as the amount of randomness leaving the layer (assuming G is a uniformly random function.) In the GGM contsruction, the fist layer has $2n$ bits of randomness, namely $\mathsf{G}(K)$, the ith layer has $2^i n$ bits of randomness. In a construction F^{G} contradicting the conjecture, there must be a layer which has significantly more than twice as much randomness as the layer before. To see this note that if each layer at most doubles the randomness, we need $\log(q)$ layers to get the qn bits of randomness. And moreover the last layer must have qn bits of randomness, as for a black-box security proof the only source of randomness is G.

Now, if a layer more than doubles its randomness, it must be the case that in this layer, G is invoked on either (1) inputs that are not uniformly random, or (2) the inputs to G are not independent. In a black-box reduction from F^{G} to G, one considers a series of hybrids $H_1, H_2, \ldots, H_t$, where H_1 is F^{G} and H_t is a random function. One gets from a hybrid H_i to H_{i+1} by replacing some internal value $Y := \mathsf{G}(X)$ with a uniform U_{2n}. If we have an adversary $\mathcal{A}$ who can distinguish H_i from H_{i+1}, we can use it to tell if a random variable Z has

distribution U_{2n} or $\mathsf{G}(U_n)$ by replacing Y with Z in H_i, and using $\mathcal{A}$ to tell if what we get is H_i (which will be the case if $Z = \mathsf{G}(U_n)$) or H_{i+1}.

In the above argument, it is crucial that X has distribution U_n, but if (1) or (2) holds, this will not be the case. It is hard to imagine a black-box technique which works differently than by replacing some internal variables Y with the challenge Z, which as just explained will not work here.