

Trivial hard-core predicates. Some functions have “trivial” hard-core predicates. For example, let f be the function that simply drops the last bit of its input (i.e., $f(x_1 \cdots x_n) = x_1 \cdots x_{n-1}$). It is immediate that it is hard to predict x_n given $f(x) = x_1 \cdots x_{n-1}$ since x_n is independent of the output. However, f is not one-way. When we use hard-core predicates to construct pseudorandom generators, it will become clear why trivial hard-core predicates of this sort are of no use for cryptography.

In contrast, a one-to-one function f that has a hard-core predicate must be one-way (see Exercise 6.10). Intuitively, this is the case because when a function is one-to-one, the value $f(x)$ fully determines x in an information-theoretic sense. Thus, inability to compute $\text{hc}(x)$ from $f(x)$ must be due to some computational limitation in determining x from $f(x)$.

6.2 Overview: From One-Way Functions to Pseudorandom Permutations

The goal of this chapter is to show how to construct pseudorandom generators, functions, and permutations based on any one-way permutation. In this section, we give an overview of these constructions. Details are given in the sections that follow.

A hard-core predicate for any one-way function. The first step is to show that a hard-core predicate exists for any one-way function. Actually, it remains open whether such a statement is true; we will show something slightly weaker that suffices for our purposes. Namely, we will show that given any one-way function f we can construct a *different* one-way function g along with a hard-core predicate for g . That is:

THEOREM 6.6 *Let f be a one-way function. Then there exists (constructively) a one-way function g along with a hard-core predicate gl for g . Furthermore, if f is a permutation then so is g .*

(The hard-core predicate is denoted gl after Goldreich and Levin who proved Theorem 6.6.) Functions g and gl are constructed as follows: set $g(x, r) \stackrel{\text{def}}{=} (f(x), r)$, for $|x| = |r|$, and define

$$\text{gl}(x, r) \stackrel{\text{def}}{=} \bigoplus_{i=1}^n x_i \cdot r_i,$$

where $x = x_1 \cdots x_n$ (and similarly for r). Notice that the function $\text{gl}(x, \cdot)$ outputs the exclusive-or of a *random subset* of the bits of x . This is due to the fact that r can be viewed as selecting a random subset of $\{1, \dots, n\}$ (i.e.,

when $r_i = 1$ the bit x_i is included in the XOR, and otherwise it is not), and r is uniformly distributed. Thus, Theorem 6.6 essentially states that if f is an arbitrary one-way function, then $f(x)$ hides the exclusive-or of a *random subset* of the bits of x .

Pseudorandom generators from one-way permutations. The next step is to show how the hard-core predicate of a one-way *permutation* can be used to construct a pseudorandom generator. (It is known that one-way *functions* suffice for constructing pseudorandom generators, but the proof is extremely complicated and well beyond the scope of this book.) Specifically, we show the following:

THEOREM 6.7 *Let f be a one-way permutation and let hc be a hard-core predicate of f . Then, $G(s) \stackrel{\text{def}}{=} (f(s), \text{hc}(s))$ constitutes a pseudorandom generator with expansion factor $\ell(n) = n + 1$.*

As intuition for why G as defined in the theorem constitutes a pseudorandom generator, note first that the initial n bits of the output of $G(s)$ (i.e., the bits of $f(s)$) are *truly* random when s is chosen uniformly at random, by virtue of the fact that f is a permutation. Next, the fact that hc is a hard-core predicate means that $\text{hc}(s)$ “looks random” — i.e., is *pseudorandom* — even given $f(s)$ (assuming again that s is chosen at random). Putting these observations together we see that the entire output of G is pseudorandom.

Pseudorandom generators with arbitrary expansion. The existence of a pseudorandom generator that stretches its seed by even a single bit (as we have just seen) is already highly non-trivial. But for applications (e.g., for efficient encryption of large messages as in Section 3.4), we need a pseudorandom generator with much larger expansion factor. Fortunately, we can obtain an expansion factor that is essentially as long as we like:

THEOREM 6.8 *Assume that there exists a pseudorandom generator with expansion factor $\ell(n) = n + 1$. Then for any polynomial $p(\cdot)$, there exists a pseudorandom generator with expansion factor $\ell(n) = p(n)$.*

We conclude that pseudorandom generators with (essentially) arbitrary expansion factor can be constructed from any one-way permutation.

Pseudorandom functions and permutations from pseudorandom generators. Pseudorandom generators suffice for obtaining private-key encryption schemes with indistinguishable encryptions in the presence of an eavesdropper. For achieving CPA-secure private-key encryption (not to mention message authentication codes), however, we relied on pseudorandom functions. The following result shows that the latter can be constructed from the former:

THEOREM 6.9 *Assume that there exists a pseudorandom generator with expansion factor $\ell(n) = 2n$. Then there exist pseudorandom functions.*

In fact, we can do even more:

THEOREM 6.10 *Assume that there exist pseudorandom functions. Then there exist strong pseudorandom permutation.*

Combining all the above theorems, as well as the results of Chapters 3 and 4, we have the following corollaries:

COROLLARY 6.11 *Assuming the existence of one-way permutations, there exist pseudorandom generators with any polynomial expansion factor, pseudorandom functions, and strong pseudorandom permutations.*

COROLLARY 6.12 *Assuming the existence of one-way permutations, there exist CCA-secure private-key encryption schemes, and message authentication codes that are existentially unforgeable under an adaptive chosen message attack.*

As noted earlier, it is actually possible to obtain all these results based solely on the existence of one-way functions.

6.3 A Hard-Core Predicate for Any One-Way Function

In this section, we prove Theorem 6.6 by showing the following:

THEOREM 6.13 *Let f be a one-way function and define g by $g(x, r) \stackrel{\text{def}}{=} (f(x), r)$, where $|x| = |r|$. Define $\text{gl}(x, r) \stackrel{\text{def}}{=} \bigoplus_{i=1}^n x_i \cdot r_i$, where $x = x_1 \cdots x_n$ and $r = r_1 \cdots r_n$. Then gl is a hard-core predicate of the function g .*

We now proceed to prove Theorem 6.13. Due to the complexity of the proof, we prove three successively stronger results culminating with what is claimed in the theorem.

6.3.1 A Simple Case

We first show that if there exists a polynomial-time adversary \mathcal{A} that always correctly computes $\text{gl}(x, r)$ given $g(x, r) = (f(x), r)$, then it is possible

to invert f in polynomial time. Given the assumption that f is a one-way function, it follows that no such adversary \mathcal{A} exists.

PROPOSITION 6.14 *Let f and gl be as in Theorem 6.13. If there exists a probabilistic polynomial-time algorithm \mathcal{A} such that*

$$\Pr_{x,r \leftarrow \{0,1\}^n} [\mathcal{A}(f(x), r) = \text{gl}(x, r)] = 1$$

for infinitely-many values of n , then there exists a probabilistic polynomial-time algorithm \mathcal{A}' such that

$$\Pr_{x \leftarrow \{0,1\}^n} [\mathcal{A}'(f(x)) \in f^{-1}(f(x))] = 1$$

for infinitely-many values of n .

PROOF Let \mathcal{A} be as in the proposition. We construct \mathcal{A}' as follows. On input y with $|y| = n$, adversary \mathcal{A}' computes $x_i := \mathcal{A}(y, e^i)$ for $i = 1, \dots, n$, where e^i denotes the n -bit string with 1 in the i th position and 0 everywhere else. Then \mathcal{A}' outputs $x = x_1 \cdots x_n$. Clearly \mathcal{A}' runs in polynomial time.

To analyze the success of \mathcal{A}' in inverting f , fix an n for which

$$\Pr_{x,r \leftarrow \{0,1\}^n} [\mathcal{A}(f(x), r) = \text{gl}(x, r)] = 1 \quad (6.1)$$

and consider the execution of $\mathcal{A}'(y)$. Denote $y = f(\hat{x})$. Then, the value x_i computed by \mathcal{A}' satisfies

$$x_i = \mathcal{A}(f(\hat{x}), e^i) = \bigoplus_{j=1}^n \hat{x}_j \cdot e_j^i = \hat{x}_i,$$

using the definition of $\text{gl}(x, r)$ for the second equality, and the fact that $e_j^i = 0$ for all $j \neq i$ for the third equality. Thus, $x_i = \hat{x}_i$ for all i and so \mathcal{A}' outputs the correct inverse $x = \hat{x}$ with probability 1. \blacksquare

By the assumption that f is one-way, it is impossible for any probabilistic polynomial-time algorithm to invert f with non-negligible probability. Thus, we conclude that there is no probabilistic polynomial-time algorithm that always correctly computes $\text{gl}(x, r)$ from $(f(x), r)$ for infinitely-many values of n . This is a rather weak result that is very far from our ultimate goal of showing that $\text{gl}(x, r)$ cannot be determined with probability significantly better than $1/2$.

6.3.2 A More Involved Case

We now show that it is hard for any polynomial-time algorithm \mathcal{A} to compute $\text{gl}(x, r)$ with probability significantly better than $3/4$. Assuming such

an \mathcal{A} exists, we will once again show that this implies the existence of a polynomial-time \mathcal{A}' that inverts f with non-negligible probability. Notice that the strategy in the proof of Proposition 6.14 fails completely here because it may be that \mathcal{A} *never* succeeds when $r = e^i$ (though it may succeed, say, on all other values of r). Furthermore, in the present case \mathcal{A}' does not know if a particular bit output by \mathcal{A} as a guess for $\mathbf{gl}(x, r)$ is correct or not — the only thing \mathcal{A}' knows is that with probability non-negligibly greater than $3/4$, adversary \mathcal{A} is correct. These issues further complicate the proof.

PROPOSITION 6.15 *Let f and \mathbf{gl} be as in Theorem 6.13. If there exists a probabilistic polynomial-time algorithm \mathcal{A} and a polynomial $p(\cdot)$ such that*

$$\Pr_{x, r \leftarrow \{0,1\}^n} [\mathcal{A}(f(x), r) = \mathbf{gl}(x, r)] \geq \frac{3}{4} + \frac{1}{p(n)}$$

for infinitely-many values of n , then there exists a probabilistic polynomial-time algorithm \mathcal{A}' such that

$$\Pr_{x \leftarrow \{0,1\}^n} [\mathcal{A}'(f(x)) \in f^{-1}(f(x))] \geq \frac{1}{4 \cdot p(n)}$$

for infinitely-many values of n .

PROOF The main observation underlying the proof of this proposition is that for every $r \in \{0,1\}^n$, the values $\mathbf{gl}(x, r \oplus e^i)$ and $\mathbf{gl}(x, r)$ together can be used to derive the i th bit of x . (Recall that e^i denotes the n -bit string with 0s everywhere except the i th position.) This follows from the following calculation:

$$\begin{aligned} & \mathbf{gl}(x, r) \oplus \mathbf{gl}(x, r \oplus e^i) \\ &= \left(\bigoplus_{j=1}^n x_j \cdot r_j \right) \oplus \left(\bigoplus_{j=1}^n x_j \cdot (r_j \oplus e_j^i) \right) = x_i \cdot r_i \oplus (x_i \cdot (r_i \oplus 1)) = x_i, \end{aligned}$$

where the second equality is due to the fact that for all $j \neq i$, the value $x_j \cdot r_j$ appears in both sums and so is canceled out.

The above illustrates that if \mathcal{A} answers correctly on both $(f(x), r)$ and $(f(x), r \oplus e^i)$, then \mathcal{A}' can correctly compute x_i . Unfortunately, \mathcal{A}' does not know when \mathcal{A} answers correctly and when it does not; it only knows that \mathcal{A} answers correctly with “high” probability. For this reason, \mathcal{A}' will use multiple random values of r , using each one to obtain a guess for x_i , and will then take the majority value as its final guess of x_i . As a preliminary step, we therefore show that for many x 's, the probability that \mathcal{A} answers correctly for both $(f(x), r)$ and $(f(x), r \oplus e^i)$, when r is chosen uniformly at random, is sufficiently high. This is proved in the following claims. These claims will

allow us to fix x and then focus solely on the uniform choice of r , which makes the analysis easier.

CLAIM 6.16 *Let n be such that*

$$\Pr_{x,r \leftarrow \{0,1\}^n} [\mathcal{A}(f(x), r) = \mathbf{gl}(x, r)] \geq \frac{3}{4} + \frac{1}{p(n)}.$$

Then there exists a set $S_n \subseteq \{0,1\}^n$ of size at least $\frac{1}{2p(n)} \cdot 2^n$ such that for every $x \in S_n$ it holds that

$$\Pr_{r \leftarrow \{0,1\}^n} [\mathcal{A}(f(x), r) = \mathbf{gl}(x, r)] \geq \frac{3}{4} + \frac{1}{2p(n)}. \quad (6.2)$$

PROOF Set $\varepsilon(n) = 1/p(n)$ and for any $x \in \{0,1\}^n$, let

$$s(x) \stackrel{\text{def}}{=} \Pr_{r \leftarrow \{0,1\}^n} [\mathcal{A}(f(x), r) = \mathbf{gl}(x, r)].$$

Let S_n be the set of all x 's for which $s(x) \geq 3/4 + \varepsilon(n)/2$ (i.e., for which Equation (6.2) holds). If $S_n = \{0,1\}^n$ we are done. Otherwise, we show that $|S_n| \geq \frac{\varepsilon(n)}{2} \cdot 2^n$ using a simple averaging argument. We have:

$$\begin{aligned} & \Pr_{x,r} [\mathcal{A}(f(x), r) = \mathbf{gl}(x, r)] \\ &= \Pr_{x,r} [\mathcal{A}(f(x), r) = \mathbf{gl}(x, r) \mid x \in S_n] \cdot \Pr_x [x \in S_n] \\ & \quad + \Pr_{x,r} [\mathcal{A}(f(x), r) = \mathbf{gl}(x, r) \mid x \notin S_n] \cdot \Pr_x [x \notin S_n] \\ & \leq \Pr_x [x \in S_n] + \Pr_{x,r} [\mathcal{A}(f(x), r) = \mathbf{gl}(x, r) \mid x \notin S_n], \end{aligned}$$

where subscripted variables (i.e., x and/or r) indicate those being chosen at random from $\{0,1\}^n$, while non-subscripted variables are fixed. Therefore:

$$\begin{aligned} & \Pr_x [x \in S_n] \geq \\ & \Pr_{x,r} [\mathcal{A}(f(x), r) = \mathbf{gl}(x, r)] - \Pr_{x,r} [\mathcal{A}(f(x), r) = \mathbf{gl}(x, r) \mid x \notin S_n]. \end{aligned}$$

By definition of S_n , for every $x \notin S_n$, $\Pr_{x,r} [\mathcal{A}(f(x), r) = \mathbf{gl}(x, r)] < 3/4 + \varepsilon(n)/2$. That is, $\Pr_{x,r} [\mathcal{A}(f(x), r) = \mathbf{gl}(x, r) \mid x \notin S_n] < 3/4 + \varepsilon(n)/2$, and so

$$\Pr_x [x \in S_n] \geq \frac{3}{4} + \varepsilon(n) - \left(\frac{3}{4} + \frac{\varepsilon(n)}{2} \right) = \frac{\varepsilon(n)}{2}.$$

This implies that S_n must be of size at least $\frac{\varepsilon(n)}{2} \cdot 2^n$ (because x is uniformly distributed in $\{0,1\}^n$), completing the proof of the claim. \blacksquare

The following, which is the result we need, now follows as an easy corollary.

CLAIM 6.17 *Let n be such that*

$$\Pr_{x, r \leftarrow \{0,1\}^n} [\mathcal{A}(f(x), r) = \mathbf{gl}(x, r)] \geq \frac{3}{4} + \frac{1}{p(n)}.$$

Then there exists a set $S_n \subseteq \{0, 1\}^n$ of size at least $\frac{1}{2p(n)} \cdot 2^n$ such that for every $x \in S_n$ and every i it holds that

$$\Pr_{r \leftarrow \{0,1\}^n} [\mathcal{A}(f(x), r) = \mathbf{gl}(x, r) \wedge \mathcal{A}(f(x), r \oplus e^i) = \mathbf{gl}(x, r \oplus e^i)] \geq \frac{1}{2} + \frac{1}{p(n)}.$$

PROOF Let $\varepsilon(n) = 1/p(n)$, and take S_n to be the set guaranteed by the previous claim. We know that for any $x \in S_n$ we have

$$\Pr_{r \leftarrow \{0,1\}^n} [\mathcal{A}(f(x), r) \neq \mathbf{gl}(x, r)] \leq \frac{1}{4} - \varepsilon(n)/2.$$

Fix any $i \in \{1, \dots, n\}$. If r is uniformly distributed then so is $r \oplus e^i$; this means that

$$\Pr_{r \leftarrow \{0,1\}^n} [\mathcal{A}(f(x), r \oplus e^i) \neq \mathbf{gl}(x, r \oplus e^i)] \leq \frac{1}{4} - \varepsilon(n)/2.$$

We are interested in lower-bounding the probability that \mathcal{A} outputs the correct answer for *both* $\mathbf{gl}(x, r)$ and $\mathbf{gl}(x, r \oplus e^i)$; equivalently, we want to upper-bound the probability that \mathcal{A} fails to output the correct answer in *either* of these cases. Note that r and $r \oplus e^i$ are not independent, and so we cannot just multiply the probabilities of failure. However, we can apply the union bound (see Proposition A.7 in Appendix A) and just sum the probabilities of failure. That is, the probability that \mathcal{A} is *incorrect* on either $\mathbf{gl}(x, r)$ or $\mathbf{gl}(x, r \oplus e^i)$ is at most

$$\left(\frac{1}{4} - \frac{\varepsilon(n)}{2}\right) + \left(\frac{1}{4} - \frac{\varepsilon(n)}{2}\right) = \frac{1}{2} - \varepsilon(n),$$

and so \mathcal{A} is correct on *both* $\mathbf{gl}(x, r)$ and $\mathbf{gl}(x, r \oplus e^i)$ with probability *at least* $1/2 + \varepsilon(n)$. This proves the claim. \blacksquare

For the rest of the proof we set $\varepsilon(n) = 1/p(n)$ and consider only those values of n for which \mathcal{A} succeeds with probability at least $3/4 + \varepsilon(n)$. The claim above states that for an $\varepsilon(n)/2$ fraction of inputs x , the adversary \mathcal{A} answers correctly on both $(f(x), r)$ and $(f(x), r \oplus e^i)$ with probability at least $1/2 + \varepsilon(n)$ over random choice of r , and from now on we will focus only on such values of x . We construct a probabilistic polynomial-time algorithm \mathcal{A}'

that inverts $f(x)$ with probability at least $1/2$ when $x \in S_n$. This suffices to prove the proposition since then

$$\begin{aligned} \Pr_x[\mathcal{A}'(f(x)) \in f^{-1}(f(x))] & \\ & \geq \Pr_x[\mathcal{A}'(f(x)) \in f^{-1}(f(x)) \mid x \in S_n] \cdot \Pr_x[x \in S_n] \\ & \geq \frac{1}{2} \cdot \frac{1}{2p(n)} = \frac{1}{4p(n)}. \end{aligned}$$

Algorithm \mathcal{A}' , given as input an element y , works as follows:

1. For $i = 1, \dots, n$ do:
 - (a) Choose a random $r \leftarrow \{0, 1\}^n$ and compute a “guess” that the value $\bar{x}_i := \mathcal{A}(y, r) \oplus \mathcal{A}(y, r \oplus e^i)$ is the i th bit of the pre-image of y .
 - (b) Repeat the above sufficiently-many times (see further below), and let x_i be the majority of the guesses.
2. Output $x = x_1 \cdots x_n$.

We sketch an analysis of the probability that \mathcal{A}' correctly inverts its given input y . (We allow ourselves to be a bit laconic, since a full proof for the most difficult case is given in the following section.) Fix n to be (one of the infinitely many values) such that $\Pr_{x, r \leftarrow \{0, 1\}^n}[\mathcal{A}(f(x), r) = \text{gl}(x, r)] \geq \frac{3}{4} + \frac{1}{p(n)}$, and assume that \mathcal{A}' 's input $y = f(\hat{x})$ is such that $\hat{x} \in S_n$ (recall that the latter occurs with probability at least $\varepsilon(n)/2$). Fix some i . The previous claim implies that the guess \bar{x}_i is equal to $\text{gl}(\hat{x}, e^i)$ with probability at least $\frac{1}{2} + \varepsilon(n)$. By repeating sufficiently-many times and letting x_i be the majority, \mathcal{A}' can ensure that x_i is equal to $\text{gl}(\hat{x}, e^i)$ with probability at least $1 - \frac{1}{2n}$. We need to ensure that this can be done by taking the majority of only *polynomially-many* guesses; since $\varepsilon(n) = 1/p(n)$ for some polynomial p , this is indeed the case as can be shown using a *Chernoff bound* (a standard bound from probability theory), along with the fact that an independent value of r is chosen in each iteration. We leave a full proof using the Chernoff bound as an exercise.

Summarizing where things stand, we have that for each i the value x_i computed by \mathcal{A}' is incorrect with probability at most $\frac{1}{2n}$. A union bound thus shows that \mathcal{A}' is incorrect for *some* i with probability at most $n \cdot \frac{1}{2n} = \frac{1}{2}$. That is, \mathcal{A}' is correct for all i — and thus correctly inverts y — with probability at least $\frac{1}{2}$. This completes the proof of the proposition. ■

A corollary of Proposition 6.15 is that if f is a one-way function, then the probability of correctly guessing $\text{gl}(x, r)$ when given $(f(x), r)$ is at most negligibly greater than $3/4$. Thus, the bit $\text{gl}(x, r)$ has considerable uncertainty (when considering polynomial-time observers).

6.3.3 The Full Proof

This section is more advanced than the rest of the book, and relies on more involved concepts from probability theory. We include the full proof for completeness, and for more advanced students and courses.

Preliminaries – Probabilistic Sampling

We use some standard results from probability theory that are reviewed quickly here. A 0/1-random variable X_i is one that takes a value in $\{0, 1\}$. The 0/1-random variables X_1, \dots, X_m are *pairwise independent* if for every $i \neq j$ and every $b_i, b_j \in \{0, 1\}$ it holds that

$$\Pr[X_i = b_i \wedge X_j = b_j] = \Pr[X_i = b_i] \cdot \Pr[X_j = b_j].$$

We rely on the following proposition:

PROPOSITION 6.18 *Let $\{X_i\}$ be pairwise-independent, 0/1-random variables with the following property: there exist values $b \in \{0, 1\}$ and $\varepsilon > 0$ such that for all $1 \leq i \leq n$,*

$$\Pr[X_i = b] = \frac{1}{2} + \varepsilon.$$

Consider the process in which m values X_1, \dots, X_m are recorded and X is set to the value that occurs a majority of the time. Then

$$\Pr[X \neq b] \leq \frac{1}{4 \cdot \varepsilon^2 \cdot m}.$$

This proposition is standard. The reader willing to accept the above on faith can proceed directly to the following section; for completeness, we provide a self-contained proof of the proposition here.

Let $\text{Exp}[X]$ denote the expectation of a random variable X . We have:

Markov's Inequality: *Let X be a non-negative random variable and $v > 0$. Then:*

$$\Pr[X \geq v] \leq \text{Exp}[X]/v.$$

PROOF We have

$$\begin{aligned} \text{Exp}[X] &= \sum_{x \geq 0} \Pr[X = x] \cdot x \\ &\geq \sum_{0 \leq x < v} \Pr[X = x] \cdot 0 + \sum_{x \geq v} \Pr[X = x] \cdot v \\ &= \Pr[X \geq v] \cdot v. \end{aligned}$$



Markov's inequality is useful when very little information about X is known. When an upper-bound on the variance of X is known, however, better bounds exist. We will use the following basic facts from probability: $\text{Var}[X] \stackrel{\text{def}}{=} \text{Exp}[(X - \text{Exp}[X])^2]$, $\text{Var}[X] = \text{Exp}[X^2] - \text{Exp}[X]^2$, and $\text{Var}[aX + b] = a^2 \text{Var}[X]$.

Chebyshev's Inequality: *Let X be a random variable and $\delta > 0$. Then:*

$$\Pr[|X - \text{Exp}[X]| \geq \delta] \leq \frac{\text{Var}[X]}{\delta^2}.$$

PROOF Define the non-negative random variable $Y \stackrel{\text{def}}{=} (X - \text{Exp}[X])^2$ and then apply Markov's inequality. That is,

$$\begin{aligned} \Pr[|X - \text{Exp}[X]| \geq \delta] &\leq \Pr[(X - \text{Exp}[X])^2 \geq \delta^2] \\ &\leq \frac{\text{Exp}[(X - \text{Exp}[X])^2]}{\delta^2} \\ &= \frac{\text{Var}[X]}{\delta^2}. \end{aligned}$$

■

If X_1, \dots, X_m are pairwise-independent then $\text{Var}[\sum_{i=1}^m X_i] = \sum_{i=1}^m \text{Var}[X_i]$ (this is due to the fact that $\text{Exp}[X_i \cdot X_j] = \text{Exp}[X_i] \cdot \text{Exp}[X_j]$ when $i \neq j$, using pairwise independence). An important corollary of Chebyshev's inequality follows.

COROLLARY 6.19 *Let X_1, \dots, X_m be pairwise-independent random variables with the same expectation μ and the same variance σ^2 . For every $\varepsilon > 0$,*

$$\Pr\left[\left|\frac{\sum_{i=1}^m X_i}{m} - \mu\right| \geq \varepsilon\right] \leq \frac{\sigma^2}{\varepsilon^2 m}.$$

PROOF By linearity of expectations, $\text{Exp}[\sum_{i=1}^m X_i/m] = \mu$. Applying Chebyshev's inequality to the random variable $\sum_{i=1}^m X_i/m$, we have

$$\Pr\left[\left|\frac{\sum_{i=1}^m X_i}{m} - \mu\right| \geq \varepsilon\right] \leq \frac{\text{Var}\left[\frac{1}{m} \cdot \sum_{i=1}^m X_i\right]}{\varepsilon^2}.$$

Using pairwise independence, it follows that

$$\text{Var}\left[\frac{1}{m} \cdot \sum_{i=1}^m X_i\right] = \frac{1}{m^2} \sum_{i=1}^m \text{Var}[X_i] = \frac{1}{m^2} \sum_{i=1}^m \sigma^2 = \frac{\sigma^2}{m}.$$

The inequality is obtained by combining the above two equations. ■

We now prove Proposition 6.18. Take $b = 1$ in the proposition (by symmetry, this choice is irrelevant); this means $\text{Exp}[X_i] = \frac{1}{2} + \varepsilon$. Let X denote the majority value of the $\{X_i\}$ as in the proposition, and note that $X \neq 1$ only if $\sum_{i=1}^m X_i \leq m/2$. So

$$\begin{aligned} \Pr[X \neq 1] &\leq \Pr\left[\sum_{i=1}^m X_i \leq m/2\right] \\ &= \Pr\left[\frac{\sum_{i=1}^m X_i}{m} - \frac{1}{2} \leq 0\right] \\ &= \Pr\left[\frac{\sum_{i=1}^m X_i}{m} - \left(\frac{1}{2} + \varepsilon\right) \leq -\varepsilon\right] \\ &\leq \Pr\left[\left|\frac{\sum_{i=1}^m X_i}{m} - \left(\frac{1}{2} + \varepsilon\right)\right| \geq \varepsilon\right]. \end{aligned}$$

For a 0/1-random variable X_i , we have $\sigma^2 = \text{Var}[X_i] \leq 1/4$ (this is because in such a case $\text{Exp}[X_i] = \text{Exp}[X_i^2]$ and so $\text{Var}[X_i] = \text{Exp}[X_i](1 - \text{Exp}[X_i])$ which is maximum when $\text{Exp}[X_i] = \frac{1}{2}$). Applying the previous corollary, we conclude that

$$\Pr[X \neq 1] \leq \frac{1}{4\varepsilon^2 m},$$

as claimed.

Proof of Theorem 6.13

We assume familiarity with the simplified proofs in the previous sections, and rely on the ideas developed there. We prove the following proposition, which implies Theorem 6.13:

PROPOSITION 6.20 *Let f and g be as in Theorem 6.13. If there exists a probabilistic polynomial-time algorithm \mathcal{A} and a polynomial $p(\cdot)$ such that*

$$\Pr_{x,r \leftarrow \{0,1\}^n} [\mathcal{A}(f(x), r) = g(x, r)] \geq \frac{1}{2} + \frac{1}{p(n)}$$

for infinitely-many values of n , then there exists a probabilistic polynomial-time adversary \mathcal{A}' and a polynomial $p'(\cdot)$ such that

$$\Pr_{x \leftarrow \{0,1\}^n} [\mathcal{A}'(f(x)) \in f^{-1}(f(x))] \geq \frac{1}{p'(n)}$$

for infinitely-many values of n .

PROOF As in the proof of Proposition 6.15, we set $\varepsilon(n) = 1/p(n)$ and consider only those values of n for which \mathcal{A} succeeds with probability $1/2 + \varepsilon(n)$. The following is analogous to Claim 6.16 and is proved in the same way.

CLAIM 6.21 *Let n be such that*

$$\Pr_{x,r \leftarrow \{0,1\}^n} [\mathcal{A}(f(x), r) = \text{gl}(x, r)] \geq \frac{1}{2} + \varepsilon(n).$$

Then there exists a set $S_n \subseteq \{0, 1\}^n$ of size at least $\frac{\varepsilon(n)}{2} \cdot 2^n$ such that for every $x \in S_n$ it holds that

$$s(x) \stackrel{\text{def}}{=} \Pr_{r \leftarrow \{0,1\}^n} [\mathcal{A}(f(x), r) = \text{gl}(x, r)] \geq \frac{1}{2} + \frac{\varepsilon(n)}{2}. \quad (6.3)$$

If we try to proceed exactly as in the proof of Proposition 6.15, we will run into trouble because an analogue of Claim 6.17 will *not* hold here. Specifically, the best we can claim here is that when $x \in S_n$ it holds that

$$\Pr_{r \leftarrow \{0,1\}^n} [\mathcal{A}(f(x), r) = \text{gl}(x, r) \wedge \mathcal{A}(f(x), r \oplus e^i) = \text{gl}(x, r \oplus e^i)] \geq \frac{1}{p(n)}$$

for any i . This means that if we try to construct an algorithm \mathcal{A}' that guesses x_i by computing $\mathcal{A}(f(x), r) \oplus \mathcal{A}(f(x), r \oplus e^i)$, then all we can claim is that this guess will be correct with probability at least $1/p(n)$, which is not even any better than taking a random guess! (Moreover, we cannot claim that flipping the result gives a good guess with high probability, either.)

Instead, we design \mathcal{A}' so that it computes $\text{gl}(x, r)$ and $\text{gl}(x, r \oplus e^i)$ by invoking \mathcal{A} only once. We do this by having \mathcal{A}' run $\mathcal{A}(x, r \oplus e^i)$, and having \mathcal{A}' simply “guess” the value $\text{gl}(x, r)$ itself. The naive way to do this would be to choose the r 's independently, as before, and to make an independent guess of $\text{gl}(x, r)$ for each value of r . But then the probability that all guesses are correct — which, as we will see, is needed if \mathcal{A}' is to output the correct answer — would be negligible because polynomially-many different r 's are used.

The crucial observation of the present proof is that \mathcal{A}' can generate the r 's in a *pairwise-independent* manner, and make its guesses in a particular way so that with non-negligible probability all of its guesses are correct. Specifically, in order to generate m different values of r , \mathcal{A}' selects $\ell = \lceil \log(m+1) \rceil$ independent and uniformly-distributed strings $s^1, \dots, s^\ell \in \{0, 1\}^n$. Then, for every non-empty subset $I \subseteq \{1, \dots, \ell\}$, algorithm \mathcal{A}' sets $r^I := \oplus_{i \in I} s^i$. Since there are $2^\ell - 1$ non-empty subsets, this defines $2^{\lceil \log(m+1) \rceil} - 1 \geq m$ different strings. Each such string is uniformly distributed when considered in isolation. Moreover, these strings are all pairwise independent. To see this, notice that for every two subsets $I \neq J$ there is an index $j \in I \cup J$ such that $j \notin I \cap J$. Without loss of generality, assume $j \in J$. Then, even conditioned on some known value of r^I , nothing about the value of s^j is revealed and so s^j is still uniformly distributed. Furthermore, since s^j is included in the XOR that defines r^J , we have that r^J is uniformly distributed even conditioned on some known value of r^I .

We now have the following two important observations:

1. Given the correct values of $\mathbf{gl}(x, s^1), \dots, \mathbf{gl}(x, s^\ell)$, it is possible to correctly compute $\mathbf{gl}(x, r^I)$ for every non-empty subset $I \subseteq \{1, \dots, \ell\}$. This is because

$$\mathbf{gl}(x, r^I) = \mathbf{gl}(x, \oplus_{i \in I} s^i) = \oplus_{i \in I} \mathbf{gl}(x, s^i).$$

2. The values $\mathbf{gl}(x, s^1), \dots, \mathbf{gl}(x, s^\ell)$ can all be correctly guessed with probability $1/2^\ell$. This holds because each bit $\mathbf{gl}(x, s^i)$ is guessed correctly with probability $1/2$ and there are ℓ bits. If m is polynomial in the security parameter n , it follows that 2^ℓ is also polynomial in n . Thus, with *non-negligible probability* it is possible to correctly guess all the values $\mathbf{gl}(x, s^1), \dots, \mathbf{gl}(x, s^\ell)$.

Combining the above, we see that this yields a way of obtaining $m = \text{poly}(n)$ pairwise-independent strings $\{r^I\}$ along with *correct* values for $\{\mathbf{gl}(x, r^I)\}$, for all I , with non-negligible probability. These values can then be used to compute x_i in the same way as in the proof of Proposition 6.15. Details follow.

The inversion algorithm \mathcal{A}' . We now provide a full description of an algorithm \mathcal{A}' that receives input y and tries to compute an inverse of y . The algorithm proceeds as follows:

1. Set $n := |y|$ and $\ell := \lceil \log(2n/\varepsilon(n)^2 + 1) \rceil$.
2. Choose $s^1, \dots, s^\ell \leftarrow \{0, 1\}^n$ and $\sigma^1, \dots, \sigma^\ell \leftarrow \{0, 1\}$ uniformly at random.
3. For every non-empty subset $I \subseteq \{1, \dots, \ell\}$, set $r^I := \oplus_{i \in I} s^i$ and compute $\rho^I := \oplus_{i \in I} \sigma^i$.
4. For $i = 1, \dots, n$:

- (a) For every non-empty subset $I \subseteq \{1, \dots, \ell\}$, set

$$x_i^I := \rho^I \oplus \mathcal{A}(y, r^I \oplus e^i).$$

- (b) Set $x_i := \text{majority}_I \{x_i^I\}$ (i.e., take the bit that appeared a majority of the times in the previous step).

5. Output $x = x_1 \cdots x_n$.

Analyzing the success probability of \mathcal{A}' . It remains to compute the probability that \mathcal{A}' successfully outputs $x \in f^{-1}(y)$. Similarly to the proof of Proposition 6.15, we focus only on the case when $y = f(\hat{x})$ for $\hat{x} \in S_n$. Each σ^i can be viewed as a “guess” for the value of $\mathbf{gl}(\hat{x}, s^i)$. As noted earlier, with non-negligible probability all these guesses will be correct; we show that when this occurs then \mathcal{A}' outputs $x = \hat{x}$ with probability at least $1/2$. This will complete the proof.

Assuming $\sigma^i = \text{gl}(\hat{x}, s^i)$ for all i , each ρ^I is equal to the correct value of $\text{gl}(\hat{x}, r^I)$. Fix an index $i \in \{1, \dots, n\}$ and consider the probability that \mathcal{A}' obtains the correct value of x_i . Because $\mathcal{A}(y, r^I \oplus e^i) = \text{gl}(\hat{x}, r^I \oplus e^i)$ with probability at least $\frac{1}{2} + \varepsilon(n)/2$ (this follows from the facts that $\hat{x} \in S_n$ and that $r^I \oplus e^i$, considered in isolation, is a uniformly-distributed string), we know that

$$\Pr[x_i^I = \hat{x}_i] = \Pr[\mathcal{A}(f(\hat{x}), r^I \oplus e^i) = \text{gl}(\hat{x}, r^I \oplus e^i)] = \frac{1}{2} + \frac{\varepsilon(n)}{2}$$

for all I . Moreover, the $\{x_i^I\}_{I \subseteq \{1, \dots, \ell\}}$ are pairwise independent because the $\{r^I\}_{I \subseteq \{1, \dots, \ell\}}$ (and hence the $\{r^I \oplus e^i\}_{I \subseteq \{1, \dots, \ell\}}$) are pairwise independent. Since x_i is defined as the value that occurs a majority of the time among $\{\hat{x}_i^I\}$, we are now in a position to apply Proposition 6.18. Setting $m = 2^\ell - 1$, we have that

$$\begin{aligned} \Pr[x_i \neq \hat{x}_i] &\leq \frac{1}{4 \cdot (\varepsilon(n)/2)^2 \cdot (2^\ell - 1)} \\ &\leq \frac{1}{4 \cdot (\varepsilon(n)/2)^2 \cdot (2n/\varepsilon(n)^2)} \\ &= \frac{1}{2n}. \end{aligned}$$

The above holds for all i , so by applying a union bound we see that the probability that $x_i \neq \hat{x}_i$ for *some* i is at most $1/2$. That is, $x_i = \hat{x}_i$ for *all* i (and hence $x = \hat{x}$) with probability at least $1/2$.

Putting everything together: with probability at least $\varepsilon(n)/2$ it holds that $y = f(\hat{x})$ with $\hat{x} \in S_n$. Independently of this event, the probability that all of the guesses σ_i are correct is at least

$$\frac{1}{2^\ell} \geq \frac{1}{2 \cdot (2n/\varepsilon(n)^2 + 1)} > \frac{\varepsilon(n)^2}{5n}$$

(the last inequality holds for n large enough). Conditioned on both of the above, \mathcal{A}' outputs an inverse of y with probability at least $1/2$. The overall probability with which \mathcal{A}' inverts its input y is therefore at least $\varepsilon(n)^3/20n = 1/20 \cdot n \cdot p(n)^3$ for infinitely-many values of n . Since \mathcal{A}' runs in polynomial-time, this contradicts the one-wayness of f . ■

6.4 Constructing Pseudorandom Generators

We first show how to construct pseudorandom generators that stretch their input by a single bit, under the assumption that one-way *permutations* exist. We then show how to extend this to obtain any polynomial expansion factor.

6.4.1 Pseudorandom Generators with Minimal Expansion

Let f be a one-way permutation and let hc be a hard-core predicate of f (such a predicate exists by Theorem 6.13). The starting point for the construction is the fact that given $f(s)$ for a random s , it is hard to guess the value of $\text{hc}(s)$ with probability that is non-negligibly higher than $1/2$. Thus, intuitively, $\text{hc}(s)$ is a pseudorandom bit. Furthermore, since f is a permutation, $f(s)$ is uniformly distributed (applying a permutation to a uniformly distributed value yields a uniformly distributed value). We therefore conclude that the string $(f(s), \text{hc}(s))$ is pseudorandom and so the algorithm $G(s) = (f(s), \text{hc}(s))$ constitutes a pseudorandom generator.

THEOREM 6.22 *Let f be a one-way permutation, and let hc be a hard-core predicate of f . Then, the algorithm $G(s) = (f(s), \text{hc}(s))$ is a pseudorandom generator with $\ell(n) = n + 1$.*

PROOF Let D be a probabilistic polynomial-time distinguisher, and set

$$\begin{aligned} \varepsilon(n) &\stackrel{\text{def}}{=} \Pr_{s \leftarrow \{0,1\}^n} [D(G(s)) = 1] - \Pr_{r \leftarrow \{0,1\}^{n+1}} [D(r) = 1] \\ &= \Pr_{s \leftarrow \{0,1\}^n} [D(f(s), \text{hc}(s)) = 1] - \Pr_{r \leftarrow \{0,1\}^{n+1}} [D(r) = 1]. \end{aligned}$$

Observe that

$$\begin{aligned} \Pr_{r \leftarrow \{0,1\}^{n+1}} [D(r) = 1] &= \Pr_{r \leftarrow \{0,1\}^n, r' \leftarrow \{0,1\}} [D(r, r') = 1] \\ &= \Pr_{s \leftarrow \{0,1\}^n, r' \leftarrow \{0,1\}} [D(f(s), r') = 1] \\ &= \frac{1}{2} \cdot \Pr_{s \leftarrow \{0,1\}^n} [D(f(s), \text{hc}(s)) = 1] \\ &\quad + \frac{1}{2} \cdot \Pr_{s \leftarrow \{0,1\}^n} [D(f(s), \overline{\text{hc}(s)}) = 1], \end{aligned}$$

using the fact that f is a permutation for the first equality, and the fact that a random bit r' is equal to $\text{hc}(s)$ with probability exactly $1/2$ for the second equality. Thus,

$$\varepsilon(n) = \frac{1}{2} \cdot \left(\Pr_{s \leftarrow \{0,1\}^n} [D(f(s), \text{hc}(s)) = 1] - \Pr_{s \leftarrow \{0,1\}^n} [D(f(s), \overline{\text{hc}(s)}) = 1] \right).$$

Consider the following algorithm \mathcal{A} that is given as input a value $y = f(s)$ and tries to predict the value of $\text{hc}(s)$:

1. Choose $r' \leftarrow \{0, 1\}$ uniformly at random.
2. Run $D(y, r')$. If D outputs 1, output r' ; otherwise output $1 - r'$.

By definition of \mathcal{A} we have

$$\begin{aligned}
 & \Pr_{s \leftarrow \{0,1\}^n} [\mathcal{A}(f(s)) = \text{hc}(s)] \\
 &= \frac{1}{2} \cdot \Pr_{s \leftarrow \{0,1\}^n} [\mathcal{A}(f(s)) = \text{hc}(s) \mid r' = \text{hc}(s)] \\
 &\quad + \frac{1}{2} \cdot \Pr_{s \leftarrow \{0,1\}^n} [\mathcal{A}(f(s)) = \text{hc}(s) \mid r' \neq \text{hc}(s)] \\
 &= \frac{1}{2} \cdot \left(\Pr_{s \leftarrow \{0,1\}^n} [D(f(s), \text{hc}(s)) = 1] + \Pr_{s \leftarrow \{0,1\}^n} [D(f(s), \overline{\text{hc}}(s)) = 0] \right) \\
 &= \frac{1}{2} \cdot \left(\Pr_{s \leftarrow \{0,1\}^n} [D(f(s), \text{hc}(s)) = 1] + (1 - \Pr_{s \leftarrow \{0,1\}^n} [D(f(s), \overline{\text{hc}}(s)) = 1]) \right) \\
 &= \frac{1}{2} + \frac{1}{2} \cdot \left(\Pr_{s \leftarrow \{0,1\}^n} [D(f(s), \text{hc}(s)) = 1] - \Pr_{s \leftarrow \{0,1\}^n} [D(f(s), \overline{\text{hc}}(s)) = 1] \right) \\
 &= \frac{1}{2} + \varepsilon(n).
 \end{aligned}$$

Clearly \mathcal{A} runs in polynomial time. Since hc is a hard-core predicate for f , it follows that there exists a negligible function negl for which $\varepsilon(n) \leq \text{negl}(n)$.

An analogous argument shows that $-\varepsilon(n) \leq \text{negl}(n)$. Taken together, this means that

$$\left| \Pr_{s \leftarrow \{0,1\}^n} [D(G(s)) = 1] - \Pr_{r \leftarrow \{0,1\}^{n+1}} [D(r) = 1] \right| \leq \text{negl}(n),$$

completing the proof that G is a pseudorandom generator. ■

6.4.2 Increasing the Expansion Factor

We now show that the expansion factor of a pseudorandom generator can be increased by any polynomial amount. This means that the previous construction (with expansion factor $\ell(n) = n + 1$) suffices for constructing a pseudorandom generator with arbitrary polynomial expansion factor.

THEOREM 6.23 *If there exists a pseudorandom generator \hat{G} with expansion factor $\hat{\ell}(n) = n + 1$, then for any polynomial $p(n) > n$, there exists a pseudorandom generator G with expansion factor $\ell(n) = p(n)$.*

PROOF The idea behind the construction of G from \hat{G} is as follows. Given an initial seed s of length n , the generator \hat{G} can be used to obtain $n + 1$ pseudorandom bits. One of the $n + 1$ bits may be output, and the remaining n bits can be used once again as a seed for \hat{G} . The reason that these n bits can be used as a seed is because they are pseudorandom, and therefore essentially

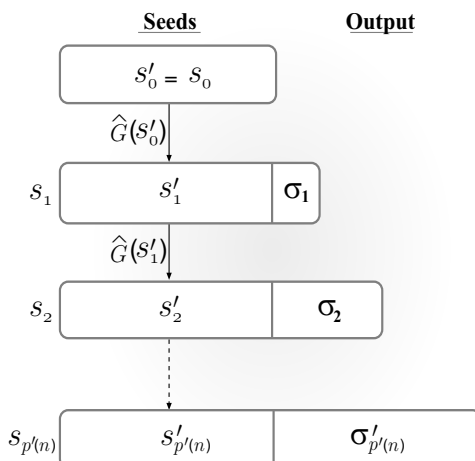


FIGURE 6.1: Increasing the expansion of a pseudorandom generator.

as good as a truly random seed. This procedure can be iteratively applied to output as many bits as desired; see Figure 6.1.

We now formally describe the construction of G . On input $s \in \{0, 1\}^n$:

1. Let $p'(n) = p(n) - n$. Note that this is the amount by which G is supposed to increase the length of its input.
2. Set $s_0 := s$. For $i = 1, \dots, p'(n)$ do:
 - (a) Let s'_{i-1} denote the first n bits of s_{i-1} , and let σ_{i-1} denote the remaining $i - 1$ bits. (When $i = 1$, σ_0 is the empty string.)
 - (b) Set $s_i := (\hat{G}(s'_{i-1}), \sigma_{i-1})$.
3. Output $s_{p'(n)}$.

Before proceeding, note that when $i = 1$, s'_0 is the original seed and in step 2b we have $s_1 = \hat{G}(s'_0)$. Then, when $i = 2$, the string s_1 of length $n + 1$ is split into a prefix of length n , denoted s'_1 , and a suffix of length 1, denoted σ_1 . The string s'_1 is used as the seed to \hat{G} again and the resulting string s_2 is of length $n + 2$ (namely, it is $(\hat{G}(s'_1), \sigma_1)$). Observe that in the next iteration, the last two bits of s_2 become σ_2 (where the first bit of σ_2 is the last bit of $\hat{G}(s'_1)$ and the second bit of σ_2 is σ_1). Thus, in each iteration a single extra bit is generated, and this is incorporated into the “ σ part”. For this reason, the σ_i values grow by one in length in each iteration, as demonstrated in Figure 6.1.

We prove that $G(s)$ is a pseudorandom string of length $p(n)$. We begin by proving this for the simple case of $p(n) = n + 2$.

Define three sequences of distributions $\{H_n^0\}, \{H_n^1\}, \{H_n^2\}$, where each of H_n^1 , H_n^2 , and H_n^3 is a distribution on strings of length $n + 2$. In distribution

H_n^0 , the string $s_0 \leftarrow \{0, 1\}^n$ is chosen uniformly at random and the output is $G(s_0)$. In distribution H_n^1 , the string $s_1 \leftarrow \{0, 1\}^{n+1}$ is chosen uniformly at random and then G is run as above but starting from iteration $i = 2$. That is, parse s_1 as (s'_1, σ_1) with $|s'_1| = n$, and then output $(\hat{G}(s'_1), \sigma_1)$. In distribution H_n^2 , the string $s_2 \leftarrow \{0, 1\}^{n+2}$ is chosen uniformly at random and output. We denote by $s_2 \leftarrow H_n^i$ the choice of the $(n + 2)$ -bit string s_2 according to distribution H_n^i .

We first claim that for any probabilistic polynomial-time distinguisher D there exists a negligible function negl such that

$$\left| \Pr_{s_2 \leftarrow H_n^0} [D(s_2) = 1] - \Pr_{s_2 \leftarrow H_n^1} [D(s_2) = 1] \right| \leq \text{negl}(n). \quad (6.4)$$

To see this, fix some D and consider the polynomial-time distinguisher D' that, on input $w \in \{0, 1\}^{n+1}$, sets $s_1 := w$ and then runs G as above but starting from iteration $i = 2$. This yields a string s_2 , and then D' outputs $D(s_2)$. The following observations are immediate from the syntactic definitions of H_n^0 and H_n^1 :

1. If w is chosen uniformly at random, the distribution on s_2 generated by D' is exactly that of distribution H_n^1 . Thus,

$$\Pr_{w \leftarrow \{0, 1\}^{n+1}} [D'(w) = 1] = \Pr_{s_2 \leftarrow H_n^1} [D(s_2) = 1].$$

2. If $w = \hat{G}(s)$ for $s \leftarrow \{0, 1\}^n$ chosen uniformly at random, the distribution on s_2 generated by D' is exactly that of distribution H_n^0 . I.e.,

$$\Pr_{s \leftarrow \{0, 1\}^n} [D'(\hat{G}(s)) = 1] = \Pr_{s_2 \leftarrow H_n^0} [D(s_2) = 1].$$

Pseudorandomness of \hat{G} implies that there exists a negligible function negl such that

$$\left| \Pr_{s \leftarrow \{0, 1\}^n} [D'(\hat{G}(s)) = 1] - \Pr_{w \leftarrow \{0, 1\}^{n+1}} [D'(w) = 1] \right| \leq \text{negl}(n).$$

Equation (6.4) follows.

We next claim that for any probabilistic polynomial-time distinguisher D there exists a negligible function negl such that

$$\left| \Pr_{s_2 \leftarrow H_n^1} [D(s_2) = 1] - \Pr_{s_2 \leftarrow H_n^2} [D(s_2) = 1] \right| \leq \text{negl}(n). \quad (6.5)$$

The proof is very similar. Consider the polynomial-time distinguisher D' that, on input $w \in \{0, 1\}^{n+1}$, chooses $\sigma_1 \leftarrow \{0, 1\}$ uniformly at random, sets $s_2 := (w, \sigma_1)$, and outputs $D(s_2)$. Notice that if w is chosen uniformly at

random then s_2 is uniformly distributed and so is distributed exactly according to H_n^2 . Thus,

$$\Pr_{w \leftarrow \{0,1\}^{n+1}} [D'(w) = 1] = \Pr_{s_2 \leftarrow H_n^2} [D(s_2) = 1].$$

On the other hand, if $w = \hat{G}(s)$ for $s \leftarrow \{0,1\}^n$ chosen uniformly at random then s_2 is distributed exactly according to H_n^1 and so

$$\Pr_{s \leftarrow \{0,1\}^n} [D'(\hat{G}(s)) = 1] = \Pr_{s_2 \leftarrow H_n^1} [D(s_2) = 1].$$

As before, pseudorandomness of \hat{G} implies Equation (6.5).

Fix some probabilistic polynomial-time distinguisher D . We have

$$\begin{aligned} & \left| \Pr_{s \leftarrow \{0,1\}^n} [D(G(s)) = 1] - \Pr_{r \leftarrow \{0,1\}^{n+2}} [D(r) = 1] \right| & (6.6) \\ &= \left| \Pr_{s_2 \leftarrow H_n^0} [D(s_2) = 1] - \Pr_{s_2 \leftarrow H_n^2} [D(s_2) = 1] \right| \\ &\leq \left| \Pr_{s_2 \leftarrow H_n^0} [D(s_2) = 1] - \Pr_{s_2 \leftarrow H_n^1} [D(s_2) = 1] \right| \\ &\quad + \left| \Pr_{s_2 \leftarrow H_n^1} [D(s_2) = 1] - \Pr_{s_2 \leftarrow H_n^2} [D(s_2) = 1] \right|. \end{aligned}$$

Using Equations (6.4) and (6.5), we conclude that Equation (6.6) is negligible.

The full proof. We now give a proof for arbitrary $p(n)$. The main difference here is a technical one: in the case of $p(n) = n + 2$ we could define and explicitly work with three sequences of distributions $\{H_n^0\}$, $\{H_n^1\}$, and $\{H_n^2\}$. Here, in contrast, we will (in some sense) need to deal with infinitely-many sequences of distributions. Instead of dealing with these explicitly, we deal with them *implicitly* using a common technique known as a *hybrid argument*. (Actually, even the case of $p(n) = 2$ utilized a simple hybrid argument.)

Let $p'(n) = p(n) - n$. For any n and $0 \leq j \leq p'(n)$, let H_n^j be the distribution on strings of length $p(n)$ defined as follows: choose $s_j \leftarrow \{0,1\}^{n+j}$ uniformly at random and then run G starting from iteration $i = j + 1$ and output $s_{p'(n)}$. (When $j = p'(n)$ this means we simply choose $s_{p'(n)} \leftarrow \{0,1\}^{p(n)}$ uniformly at random and output it.) The crucial observation here is that H_n^0 corresponds to outputting $G(s)$ for $s \leftarrow \{0,1\}^n$ chosen uniformly at random, while $H_n^{p'(n)}$ corresponds to outputting a $p(n)$ -bit string chosen uniformly at random. Fixing any polynomial-time distinguisher D , this means that

$$\begin{aligned} \varepsilon(n) &\stackrel{\text{def}}{=} \left| \Pr_{s \leftarrow \{0,1\}^n} [D(G(s)) = 1] - \Pr_{r \leftarrow \{0,1\}^{p(n)}} [D(r) = 1] \right| \\ &= \left| \Pr_{s_{p'(n)} \leftarrow H_n^0} [D(s_{p'(n)}) = 1] - \Pr_{s_{p'(n)} \leftarrow H_n^{p'(n)}} [D(s_{p'(n)}) = 1] \right|. & (6.7) \end{aligned}$$

Our goal is to prove that ε is negligible, implying that G is a pseudorandom generator.

Fix D as above, and consider the distinguisher D' that does the following when given a string $w \in \{0, 1\}^{n+1}$ as input:

1. Choose $j \leftarrow \{1, \dots, p'(n)\}$ uniformly at random.
2. Choose $\sigma_j \leftarrow \{0, 1\}^{j-1}$ uniformly at random.
3. Set $s_j := (w, \sigma_j)$. Then run G starting from iteration $i = j + 1$ and compute $s_{p'(n)}$. Output $D(s_{p'(n)})$.

Analyzing the behavior of D' is more complicated than before, though the underlying ideas are the same. Fix n and say D' chooses $j = J$. If $w \leftarrow \{0, 1\}^{n+1}$ was chosen uniformly at random, then s_J is uniformly distributed, and so the distribution on $s_{p'(n)}$ is exactly that of distribution H_n^J . That is,

$$\Pr_{w \leftarrow \{0,1\}^{n+1}} [D'(w) = 1 \mid j = J] = \Pr_{s_{p'(n)} \leftarrow H_n^J} [D(s_{p'(n)}) = 1].$$

Since each value for j is chosen with equal probability,

$$\begin{aligned} \Pr_{w \leftarrow \{0,1\}^{n+1}} [D'(w) = 1] &= \frac{1}{p'(n)} \cdot \sum_{J=1}^{p'(n)} \Pr_{w \leftarrow \{0,1\}^{n+1}} [D'(w) = 1 \mid j = J] \\ &= \frac{1}{p'(n)} \cdot \sum_{J=1}^{p'(n)} \Pr_{s_{p'(n)} \leftarrow H_n^J} [D(s_{p'(n)}) = 1]. \end{aligned} \quad (6.8)$$

On the other hand, say D' chooses $j = J$ and $w = \hat{G}(s)$ for $s \leftarrow \{0, 1\}^n$ chosen uniformly at random. Mentally setting $s_{J-1} = (s, \sigma_J)$, we see that s_{J-1} is uniformly distributed and so the distribution on $s_{p'(n)}$ is now exactly that of distribution H_n^{J-1} . That is,

$$\Pr_{s \leftarrow \{0,1\}^n} [D'(\hat{G}(s)) = 1 \mid j = J] = \Pr_{s_{p'(n)} \leftarrow H_n^{J-1}} [D(s_{p'(n)}) = 1]$$

and then

$$\begin{aligned} \Pr_{s \leftarrow \{0,1\}^n} [D'(\hat{G}(s)) = 1] &= \frac{1}{p'(n)} \cdot \sum_{J=1}^{p'(n)} \Pr_{s \leftarrow \{0,1\}^n} [D'(\hat{G}(s)) = 1 \mid j = J] \\ &= \frac{1}{p'(n)} \cdot \sum_{J=1}^{p'(n)} \Pr_{s_{p'(n)} \leftarrow H_n^{J-1}} [D(s_{p'(n)}) = 1] \\ &= \frac{1}{p'(n)} \cdot \sum_{J=0}^{p'(n)-1} \Pr_{s_{p'(n)} \leftarrow H_n^J} [D(s_{p'(n)}) = 1]. \end{aligned} \quad (6.9)$$

(Note that the indices of summation have been shifted in the final step.) We can now analyze how well D' distinguishes outputs of \hat{G} from random:

$$\begin{aligned}
& \left| \Pr_{s \leftarrow \{0,1\}^n} [D'(\hat{G}(s)) = 1] - \Pr_{w \leftarrow \{0,1\}^{n+1}} [D'(w) = 1] \right| \\
&= \frac{1}{p'(n)} \cdot \left| \sum_{J=0}^{p'(n)-1} \Pr_{s_{p'(n)} \leftarrow H_n^J} [D(s_{p'(n)}) = 1] - \sum_{J=1}^{p'(n)} \Pr_{s_{p'(n)} \leftarrow H_n^J} [D(s_{p'(n)}) = 1] \right| \\
&= \frac{1}{p'(n)} \cdot \left| \Pr_{s_{p'(n)} \leftarrow H_n^0} [D(s_{p'(n)}) = 1] - \Pr_{s_{p'(n)} \leftarrow H_n^{p'(n)}} [D(s_{p'(n)}) = 1] \right| \\
&= \frac{\varepsilon(n)}{p'(n)},
\end{aligned}$$

relying on Equations (6.8) and (6.9) for the first equality and Equation (6.7) for the final equality. (The second equality is due to the fact that the same terms are included in each sum, except for the first term of the left sum and the last term of the right sum.) Since \hat{G} is a pseudorandom generator, D' runs in polynomial time, and p' is polynomial, we conclude that ε is negligible. ■

The hybrid technique. The *hybrid technique* is used in many proofs of security and is a basic tool for proving indistinguishability when a basic primitive is applied multiple times. The technique works by defining a series of hybrid distributions that bridge between two “extreme distributions”, these being the distributions that we wish to prove indistinguishable (in the proof above, these correspond to the output of G and a random string, respectively). To apply the proof technique, three conditions should hold. First, the extreme hybrids should match the original cases of interest (in the proof above, this means that H_n^0 was equal to the distribution induced by G , while $H_n^{p'(n)}$ was equal to the uniform distribution). Second, it must be possible to translate the capability of distinguishing neighboring hybrids into the capability of breaking some underlying assumption (above, distinguishing H_n^i from H_n^{i+1} was essentially equivalent to distinguishing the output of \hat{G} from random). Finally, the number of hybrids should be polynomial, so the “distinguishing success” is only reduced by a polynomial factor.

An explicit generator with arbitrary expansion factor. Let f be a one-way permutation with hard-core predicate hc . By combining the construction of Theorem 6.22 (that states that $\hat{G}(s) = (f(s), \text{hc}(s))$ is a pseudorandom generator) with the proof of Theorem 6.23, we obtain that

$$G(s) = \left(f^{p'(n)}(s), \text{hc} \left(f^{p'(n)-1}(s) \right), \dots, \text{hc}(s) \right)$$

is a pseudorandom generator with expansion factor $p(n) = n + p'(n)$. This generator is known as the Blum-Micali generator.