

## Scribe for Lecture 12

*Instructor: Arpita Patra**Submitted by: Pranav Nuti*

## 1 Summary of Previous Lecture

In the previous lecture, a pseudo random function was constructed using a pseudo random generator  $G$  with expansion factor  $l(n) = 2n$ . The construction exploited the fact that if  $G(s_1 \dots s_n) = x_1 \dots x_{2n}$ , then  $G(x_1 \dots x_n)$  also looks like a pseudo random string. The usage of hybrids as a proof technique was introduced and a hybrid-based argument was used to prove that the construction was indeed a pseudo random function.

## 2 Goal of Lecture

The Goldreich-Levin theorem states that if there exists a one-way function, then there exists a one-way function with *hard-core predicate*. The Goldreich-Levin theorem can be used to show that the existence of a one-way function implies the existence of a PRG with expansion factor  $n + 1$ .

In this lecture we will recall the notion of a one-way function and then give examples of functions that are possibly one-way. We will then define the notion of a *hard-core predicate* and attempt to gain some intuition regarding functions and their hard-core predicates. Finally we will prove a partial result in the direction of Goldreich and Levin's groundbreaking theorem.

## 3 One-Way Functions

Intuitively, a one-way function is a function that is easy to compute but difficult to invert. Given a *random* value of the function, it should be impossible for an algorithm that runs in polynomial time to find an element which maps to this value.

**Definition.** A function  $f : \{0, 1\}^n \rightarrow \{0, 1\}^n$  is said to be *one-way* if it is polynomially computable and for every PPT algorithm  $\mathcal{A}$ , there is a negligible function  $\text{negl}(n)$  such that

$$\Pr^1[\mathcal{A}(f(x), 1^n) \in f^{-1}(f(x))] \leq \text{negl}(n) \text{ for } x \text{ chosen uniformly from } \{0, 1\}^n$$

As always, we can phrase this definition as a game  $\text{Invert}_{\mathcal{A}, f}(n)$ :

---

<sup>1</sup>The probability is taken over the randomness associated with the choice of  $x$  and any randomness used by  $\mathcal{A}$ .

- A *verifier* picks a value  $y$  at random from  $\{0, 1\}^n$  and gives this value to an adversary  $\mathcal{A}$ .
- $\mathcal{A}$  returns a value  $x$  to the verifier.
- If  $f(x) = y$ , then the game outputs 1, else the game outputs 0.
- $f$  is *one-way* if it is polynomially computable and  $\Pr[\text{Invert}_{\mathcal{A},f}(n) = 1] \leq \text{negl}(n)$ .

Notice that if we were to allow  $\mathcal{A}$  unbounded computational power, then no one-way functions would exist, since  $\mathcal{A}$  could brute-force over the domain to find a pre-image for any given input.

One-way functions are widely believed to exist because polynomial time inverting algorithms have not been found for several functions which have been intensively studied for decades.

**Candidate OWF (Integer Factorisation).** If  $x$  and  $y$  are two equal length primes, then define  $f(x, y) := xy$ .

**Candidate OWF (Subset Sum).** Let  $S = \{x_1, x_2, \dots, x_n\}$  be a set of equal length integers and let  $\mathfrak{S}$  be its power set. Then define  $f : \mathfrak{S} \rightarrow \{0, 1\}^n$  by  $f(J) := \sum_{x \in J} x \pmod{2^n}$ .

However, we have no proof that a one-way function exists. In fact, we expect any proof to be quite hard. If the inversion of a function  $f$  is NP-complete for example, it does not immediately follow that  $f$  is an OWF. For  $f$  to be an OWF, every PPT algorithm must fail to solve *almost every* instance of the inversion problem. But for the inversion problem to be NP-complete, we only require that for every PPT algorithm there is a *single* instance of the problem which it fails to solve. Hence:

- The existence of an OWF immediately yields that  $\text{P} \neq \text{NP}$ , another problem to which a lot of attention has been devoted and no proof has been found.
- It is not known whether  $\text{P} \neq \text{NP}$  yields the existence of an OWF<sup>2</sup>.
- Belief in the existence of an OWF is stronger than than the belief that  $\text{P} \neq \text{NP}$ .

Hence it is perhaps best to understand OWFs by understanding what an OWF is not:

**Non-example.** Define  $f : \{0, 1\}^n \rightarrow \{0, 1\}^n$  by  $f(x_1 \dots x_n) := x_1 x_2 \dots x_{n-1}$ .  $f$  is not an OWF since for the adversary  $\mathcal{A}$  which returns  $x_1 x_2 \dots x_{n-1} \parallel 0$  upon being given the value  $x_1 \dots x_n$ ,  $\Pr[\text{Invert}_{\mathcal{A},f}(n) = 1] = 1$ .

**Non-example.** Suppose  $f : \{0, 1\}^n \rightarrow \{0, 1\}^n$  is polynomially computable and for every PPT algorithm  $\mathcal{A}$ , there is a negligible function  $\text{negl}(n)$  such that

$$\Pr[\mathcal{A}(f(x), 1^n) \in f^{-1}(f(x))] \begin{cases} \leq \text{negl}(n) & \text{if } n \text{ is even} \\ > \frac{1}{n^{100}} & \text{if } n \text{ is odd} \end{cases}$$

---

<sup>2</sup>In class we said that  $\text{P} \neq \text{NP}$  *does not imply* the existence of an OWF. That statement is subtly different and probably incorrect.

Then  $f$  is not an OWF.

In this lecture, we will make use of the following strengthening of the notion of a one-way function:<sup>3</sup> notion:

**Definition.** A one-way function  $f$  is said to be a *one-way permutation* if it is bijective<sup>4</sup>.

The theorems we shall prove later for one-way permutations are also true for one-way functions. We shall use one-way permutations only because they are simpler to work with.

## 4 Hard-Core Predicates

We would like to construct a PRG  $G$  using a one-way function  $f$ . A natural attempt at this is to let  $G(s) = f(s) || r(s)$  where  $r$  is a Boolean function. If this is a PRG, it must be that  $r(x)$  cannot be efficiently determined from  $f(x)$ <sup>5</sup>. If we can find such a  $r$ , we might be able to construct a PRG.

Suppose  $f$  is an OWF. Given  $f(x)$ , it might be possible to determine a large portion of  $x$  even though  $f$  is an OWF:

**Example.** If  $f$  is an OWF, then so is  $g$ , if  $g(x, r) := (f(x), r)$  and  $|x| = |r|$ .

On the other hand, since  $f$  is an OWF, it is difficult to determine  $x$  *completely* given  $f(x)$ . It is therefore natural to expect that there is *something* about  $x$  that  $f(x)$  keeps hidden. Hard-core predicates are a tool which allows us to formalise this expectation.

Intuitively, a hard-core predicate is a single bit of information about  $x$  than every adversary finds difficult to guess when given access to  $f(x)$  alone.

**Definition.** A function  $hc : \{0, 1\}^n \rightarrow \{0, 1\}$  is said to be a *hard-core predicate* for a function  $f$  if it is polynomially computable and for every PPT algorithm  $\mathcal{A}$ , there is a negligible function  $\text{negl}(n)$  such that

$$\Pr^6[\mathcal{A}(f(x), 1^n) = hc(x)] \leq \frac{1}{2} + \text{negl}(n) \text{ for } x \text{ chosen uniformly from } \{0, 1\}^n$$

Notice that the definition makes no mention of  $f$  being a one-way function. In fact lots of non-OWFs have hard-core predicates.

**Example.** Consider the non-OWF  $f : \{0, 1\}^n \rightarrow \{0, 1\}^n$  defined by  $f(x_1 \dots x_n) := x_1 x_2 \dots x_{n-1}$ . Then  $hc(x) := x_n$  is a hard-core predicate for  $f$ .

---

<sup>3</sup>It is not known whether the existence of a one-way function implies the existence of a one-way permutation

<sup>4</sup>Given our definition of an OWF, we do not need to assume that  $f$  is length preserving.

<sup>5</sup>This follows immediately from an equivalent characterization of pseudo random generators. See [2].

<sup>6</sup>Once again, the probability is taken over the randomness associated with the choice of  $x$  and any randomness used by  $\mathcal{A}$ .

On the other hand, if  $f$  is bijective and has a hard-core predicate, then  $f$  must be a one-way permutation (if it were not one-way, one could invert to find the unique pre-image and use it to compute the hard-core predicate).

Finding a hard-core predicate for a OWF is not easy. Once again, consider the following non-example.

**Non-example.** If  $f$  is an OWF, then so is  $g$ , if  $g(x_1 \dots x_n) = f(x_1 \dots x_n) \parallel x_1 \oplus \dots \oplus x_n$ . Hence  $h$  defined by  $h(x_1 \dots x_n) := x_1 \oplus \dots \oplus x_n$  is not a hard-core predicate for  $g$ . In fact, the very same construction allows us to show that given any Boolean function  $h$ , there exists a OWF  $g$  such that  $h$  is not a hard-core predicate for  $g$ .

It is not known whether every one-way function has a hard-core predicate. But we can prove that if there is a one-way function, then there is a one-way function with hard-core predicate. We will prove later that if  $f$  is a one-way function with hard-core predicate  $hc$ , then  $G(s) = f(s) \parallel hc(s)$  is a pseudo random generator. This will allow us to make good progress in showing that the existence of a one-way function is the only assumption we need to construct the wide variety of schemes we considered throughout the course.

## 5 Towards the Goldreich-Levin Theorem

**Theorem (Goldreich-Levin).** Let  $f$  be a one-way permutation and define  $g$  by  $g(x, r) = (f(x), r)$  where  $x = x_1 \dots x_n$  and  $r = r_1 \dots r_n$ . Then the function  $hc(x, r) := r_1 x_1 \oplus \dots \oplus r_n x_n$  is a hard-core predicate for  $g$ .

One can attempt to prove the theorem by proving the contrapositive, i.e., if  $hc$  is not a hard-core predicate, then  $f$  is not an OWP. To show this, one must demonstrate that if there exists a PPT algorithm  $\mathcal{A}$  such that  $\Pr[\mathcal{A}(f(x), r) = hc(x, r)] \geq \frac{1}{2} + \frac{1}{p(n)}$ , then there exists a PPT algorithm  $\mathcal{A}_1$  such that  $\Pr[\mathcal{A}_1(f(x), 1^n) = x] \geq \frac{1}{p_1(n)}$ . We will not be able to prove this. Instead we shall prove the following weaker statement:

**Theorem.** If there exists a PPT algorithm  $\mathcal{A}$  such that  $\Pr[\mathcal{A}(f(x), r) = hc(x, r)] \geq \frac{3}{4} + \frac{1}{p(n)}$ , then there exists a PPT algorithm  $\mathcal{A}_1$  such that  $\Pr[\mathcal{A}_1(f(x), 1^n) = x] \geq \frac{1}{p_1(n)}$ .

**Proof.** First consider the following ‘toy’ problem: If there was an  $\mathcal{A}$  which could *always* compute  $hc(x, r)$  given  $f(x)$  and  $r$ , then could  $f$  possibly be one way?

No. Consider  $\mathcal{A}_1$  which acts as follows: it gets the values of  $f(x)$ ,  $x = x_1 \dots x_n$ , from its verifier and sends  $(f(x), (1, 0, \dots, 0))$  to  $\mathcal{A}$ . Then  $\mathcal{A}$  would compute  $hc(x, r) = x_1$  and send it back to  $\mathcal{A}_1$ . Similarly,  $\mathcal{A}_1$  can determine the other bits of  $x$  and send  $x$  to his verifier, so  $f$  cannot be an OWP.

But we only have an  $\mathcal{A}$  which can determine  $hc(x, r)$  with a certain probability. So what can we do instead? Consider an  $r \in \{0, 1\}^n$  and let  $r^i$  be  $r$  with the  $i^{\text{th}}$  bit flipped. It is straight forward to check that  $x_i = hc(x, r^i) \oplus hc(x, r)$ . Hence given  $hc(x, r^i)$  and  $hc(x, r)$  correctly,  $\mathcal{A}_1$  can determine  $x_i$ .

Consider an  $x$  for which a random choice of  $r$  results in  $\mathcal{A}$  returning  $\text{hc}(x, r)$  and  $\text{hc}(x, r^i)$  ( $\forall i$ ) correctly with a ‘high’ probability. Then  $\mathcal{A}_1$  can compute each  $x_i$  and intuitively, should be able to invert  $f(x)$  with a high probability. If we could show that there are sufficiently many  $x$  for which this is true, then  $\mathcal{A}_1$  can invert  $f$  with a non-negligible probability and we would be done. In this direction, we have the following lemma.

**Lemma.** If there exists a PPT algorithm  $\mathcal{A}$  such that  $\Pr[\mathcal{A}(f(x), r) = \text{hc}(x, r)] \geq \frac{3}{4} + \frac{1}{p(n)}$ , then there exists a set  $S \subset \{0, 1\}^n$  of size at least  $\frac{2^n}{2p(n)}$  such that for every  $x_0$  in  $S$  and every  $i$

$$\Pr[\mathcal{A}(f(x_0), r) = \text{hc}(x_0, r) \text{ and } \mathcal{A}(f(x_0), r^i) = \text{hc}(x_0, r^i)] \geq \frac{1}{2} + \frac{1}{p(n)}$$

where the probability is taken over the random choice of  $r$  and any randomness used by  $\mathcal{A}$ .

**Proof.** Consider the *maximal*<sup>7</sup> set  $S$  such that for every  $x_0$  in  $S$

$$\Pr[\mathcal{A}(f(x_0), r) = \text{hc}(x_0, r)] \geq \frac{3}{4} + \frac{1}{2p(n)}$$

for a random choice of  $r$ . A maximal set exists, since there is at least one such set and the number of sets is finite. Then note the following:

$$\begin{aligned} \Pr[\mathcal{A}(f(x), r) = \text{hc}(x, r)] &= \frac{1}{2^n} \sum_{x_0 \in \{0, 1\}^n} \Pr[\mathcal{A}(f(x_0), r) = \text{hc}(x_0, r)] \\ &= \frac{1}{2^n} \sum_{x_0 \in S \text{ and } x_0 \notin S} \Pr[\mathcal{A}(f(x_0), r) = \text{hc}(x_0, r)] \\ &\leq \frac{|S|}{2^n} + \frac{1}{2^n} \sum_{x_0 \notin S} \frac{3}{4} + \frac{1}{2p(n)} \\ &\leq \frac{|S|}{2^n} + \frac{3}{4} + \frac{1}{2p(n)} \end{aligned}$$

But  $\Pr[\mathcal{A}(f(x), r) = \text{hc}(x, r)] \geq \frac{3}{4} + \frac{1}{p(n)}$ . Hence  $\frac{|S|}{2^n} + \frac{3}{4} + \frac{1}{2p(n)} \geq \frac{3}{4} + \frac{1}{p(n)} \implies |S| \geq \frac{2^n}{2p(n)}$ . So there is a set  $S$  of size at least  $\frac{2^n}{2p(n)}$  such that for every  $x_0$  in  $S$ ,  $\Pr[\mathcal{A}(f(x_0), r) = \text{hc}(x_0, r)] \geq \frac{3}{4} + \frac{1}{2p(n)}$

Now observe the following: for every  $x_0$  in  $S$   $\Pr[\mathcal{A}(f(x_0), r) \neq \text{hc}(x_0, r)] \leq \frac{1}{4} - \frac{1}{2p(n)}$  where the probability is taken over the random choice of  $r$  as well. This means, in particular, that we can replace  $r$  with  $r^i$  in the statement above. It follows that for every  $x_0 \in S$

$$\Pr[\mathcal{A}(f(x_0), r) \neq \text{hc}(x_0, r) \text{ or } \mathcal{A}(f(x_0), r^i) \neq \text{hc}(x_0, r^i)] \leq 2 \left( \frac{1}{4} - \frac{1}{2p(n)} \right)$$

by the union bound. The required result follows immediately from taking the complement of the above event.  $\square$

<sup>7</sup>This was not mentioned in class, but is needed for the proof that follows.

The lemma implies that for every  $x_0 \in S$   $\mathcal{A}_1$  can ask  $\mathcal{A}$  just two questions and determine  $x_i$  with a probability of at least  $\frac{1}{2} + \frac{1}{p(n)}$ . But if  $\mathcal{A}_1$  were to ask more questions (by choosing different  $r$ 's) and then pick the value of  $x_i$  suggested by the *majority* of the answers, then  $\mathcal{A}_1$  could obtain the right answer with a much a higher probability. More formally, we have the following:

**Lemma.**  $\mathcal{A}_1$  can determine  $x_i$  with a probability of at least  $1 - \frac{1}{2n}$  while asking only polynomially many questions.

**Proof.** Suppose  $X_i$ 's,  $i = 1$  to  $i = 2s + 1$ , are indicator random variables with probability of success  $k = \frac{1}{2} + \frac{1}{p(n)}$ . Let  $\delta = 1 - \frac{s}{(2s+1)k}$ . Note that  $\delta > 0$ . Further let  $\mu = \mathbb{E}[\sum X_i] = (2s + 1)k$ . The probability that the majority of them are 0 is equal to the probability that  $\sum X_i \leq s$ . But note the following:

$$\begin{aligned} \Pr \left[ \sum X_i \leq s \right] &= \Pr \left[ \sum X_i \leq (2s + 1)k \cdot \frac{s}{(2s + 1)k} \right] \\ &= \Pr \left[ \sum X_i \leq \mu \cdot \frac{s}{(2s + 1)k} \right] \\ &= \Pr \left[ \sum X_i \leq \mu \cdot (1 - \delta) \right] \\ &\leq e^{-\frac{\mu\delta^2}{2}} \end{aligned}$$

where the last inequality follows from an application of the Chernoff bound. But if  $s > cp(n)^3$  for some constant<sup>8</sup>  $c$ , a straight forward computation shows that  $e^{-\frac{\mu\delta^2}{2}} \leq \frac{1}{2n}$ . We think of the  $X_i$ 's as random variables that are lower bounds for the random variables which indicate the probability of  $\mathcal{A}_1$  determining a bit of  $x$ .  $\mathcal{A}_1$  only needs to ask  $s$  pairs of questions to determine  $x_i$  with a probability of at least  $1 - \frac{1}{2n}$ .  $\square$

Hence there exists  $\mathcal{A}_1$  such that the probability that  $\mathcal{A}_1$  does not determine  $x_i$  correctly is at most  $\frac{1}{2n}$ . By the union bound, the probability that such an  $\mathcal{A}_1$  does not determine  $x_i$  correctly *for some*  $i$  is at most  $n \cdot \frac{1}{2n} = \frac{1}{2}$ . Hence there exists an  $\mathcal{A}_1$  which can determine the inverse of an element in  $S$  with probability at least  $\frac{1}{2}$ .

Since the size of  $S$  is at least  $\frac{2^n}{2p(n)}$ , a random string from  $\{0, 1\}^n$  lies in  $S$  with probability at least  $\frac{1}{2^n} \cdot \frac{2^n}{2p(n)} = \frac{1}{2p(n)}$ . Since  $\mathcal{A}_1$  can invert an element of  $S$  with probability at least  $\frac{1}{2}$ ,  $\mathcal{A}_1$  can invert a random string from  $\{0, 1\}^n$  with probability at least  $\frac{1}{4p(n)} = \frac{1}{p_1(n)}$  as required.  $\square$

A more sophisticated argument involving the use of an algorithm to construct 'nice' sets can be used to show the full Goldreich-Levin theorem. We do not cover it due to a lack of time.

---

<sup>8</sup>This bound is not tight.

## References

- [1] Arpita Patra, *Lecture 12*, <http://drona.csa.iisc.ernet.in/~arpita/Cryptography16.html>
- [2] Goldreich, Oded, *Foundations of Cryptography I: Basic Tools*, Cambridge University Press, 2001. Theorem 3.3.7.