

The role of Pseudo Random Generators in derandomization is discussed in detail in the previous lectures. In this lecture we shall study the conditions under which PRG exists and some related issues.

**Definition:** Let  $f : \Sigma^* \rightarrow \Sigma^*$  such that  $\forall x, |f(x)| = l(|x|)$ , where  $l$  is a polynomial bounded function. Moreover assume that  $|f(x)| > |x|$  for every  $x$ . The function  $f : \Sigma^* \rightarrow \Sigma^*$  is said to be Cryptographic pseudo random generator if the following holds.

- $f$  is polynomial time computable.
- For every polynomials  $p$  and  $q$ , for every  $n$ , the distribution  $f(U_n)$  is  $(p, \frac{1}{q})$  pseudorandom.

Observe that  $f(U_n)$  is a  $p$ -samplable distribution, and  $f(U_n)$  is a distribution over  $\Sigma^{l(n)}$ .

Now let us discuss under what conditions do such functions exist.

**Claim:** If  $P = NP$  then no Cryptographic pseudo random generator exist.

We shall prove this statement by contradiction.

Let  $P=NP$  and assume  $f : \Sigma^* \rightarrow \Sigma^*$  be Cryptographic pseudo random generator. By Definition,  $f$  is polynomial time computable. Therefore  $Range(f) \in NP$ . Since  $NP=P$ ,  $Range(f) \in P$

Thus There is polynomial time deterministic polynomial-time machine  $C$  that accepts  $Range(f)$ . Consider the following polynomial-time algorithm  $M$ : On input  $y$ , run  $C$  on  $y$ . If  $C$  accepts, then output 1, else output 0. Clearly  $M$  outputs value 1 for all  $y \in Range(f)$  and runs in polynomial time.

Thus

$$\Pr_{x \in U_n} [M(f(x)) = 1] = 1$$

and

$$\Pr_{x \in U_{l(n)}} [M(f(x)) = 1] \leq 2^n / 2^{l(n)} \leq 1/2$$

Thus  $f(U_n)$  is not pseudorandom.

Thus if crypto pseudo-random generators exist, then  $P \neq NP$ . To show that PRG's exist, we need an assumption at least as strong as  $P \neq NP$

We believe that NP is not easy, thus hard problems exist. We have seen several where randomness seems to be of help. Thus randomness is useful. Over the next few lectures we show a surprising connection between hardness

and randomness. We will show that at least one of the following statements is false: “NP is not easy”, “randomness is useful”.

For this, we have to define the precise meaning of “easy” and “useful”. One-way to define “non-easiness” is via one-way functions.

**Definition:** A function  $f$  (worst-case) is said to be one-way function if the following holds.

- $f$  is polynomial time computable.
- It does not shrink its inputs too much.
- $f^{-1}$  is not polynomial time computable.

Recall that if  $P \neq NP$  if and only if worst-case one way functions exist. The above definition refers to worst-case hardness. The following definition captures the average-case hardness of inverting  $f$ .

**Definition:** A function  $f : \Sigma^* \rightarrow \Sigma^*$  is one way if  $\forall$  polynomials  $p$  and  $q$ ,  $\forall$  circuit  $C$  of size  $\leq p(n)$

$$\Pr x \in U_n [C[f(x)] \in f^{-1}(f(x))] \leq \frac{1}{q(n)}$$

It is easy to show that if one-way function exist, then NP is not easy on average. However, we do not know whether the converse is true.

**Definition:** A function  $f$  is one way permutation, if  $\forall x, |f(x)| = |x|$ ,  $f$  is one-one, and  $f$  is one way.

Again, if one way permutation function exist, then  $NP \cap CO - NP$  is not easy on average.

We will show that if one-way permutations exist, then crypto-pseudo random generators exist. A crucial ingredient of the proof is Goldreich-Levin theorem.

**Theorem:** (Goldreich-Levin Hardcore Theorem) Let  $f$  be a one way permutation. Define a function  $b$ , called as hard-core bit as

$$b(x, r) = \langle x, r \rangle = \sum x_i r_i \pmod{2} \text{ where } |x| = |r|.$$

If there is a circuit  $C$  such that  $\Pr_{x, r \in U_n} [C[f(r), r] = \langle x, r \rangle] \geq \frac{1}{2} + \epsilon$  then there exist circuit  $D$  of size  $poly(|C|, \frac{1}{\epsilon})$  such that  $\Pr_{x \in U_n} [D[f(x) = x]] \geq \frac{1}{2\epsilon}$

Blum-Micali-Yao Generator: Let  $f$  be one way permutation and  $G$  is defined as  $G(x, r) = f(r) \parallel \langle x, r \rangle$ . Then  $G$  is a crypto PRG.