1 Pseudorandom Generator - PRG

Definition 1.1. A distribution is considered **pseudorandom** if no efficient computation can distinguish it from the true uniform distribution by a non-negligible advantage. Formally, a distribution ensemble $D = \{D_n\}_{n\in N}$ is pseudorandom if for any ppt adversary A, and any negligible parameter ϵ:

$$|Prob_{x\leftarrow D}(A(x) = 1) - Prob_{x\leftarrow U}(A(x) = 1)| \leq \epsilon \tag{1}$$

Where U_n represent the uniform distribution ensemble.
The previous equation can also be written as $D \approx_c U$ which means that the distribution ensemble $D = \{D_n\}_{n\in N}$ is computationally indistinguishable from the uniform distribution ensemble $U = \{U_n\}_{n\in N}$.

Definition 1.2. A **pseudrandom generator** is a deterministic algorithm $G : \{0,1\}^n \rightarrow \{0,1\}^m$ with the following properties:

- Efficiency: G is computable in ppt
- Expansion: $m > n$
- Pseudorandomness: the ensemble $\{G(U_n)\}_{n\in N} \approx_c \{U_m\}_{m\in N}$

1.1 Increasing the expansion factor

Suppose we have a PRG G_1 such that $G_1 : \{0,1\}^n \rightarrow \{0,1\}^{n+1}$ (a generator with expansion factor of 1 Bit), then we can construct a PRG G with arbitrary polynomial expansion factor.
The construction is as follows: we start with a truly uniform input of n bits,U_n, that are used as seed to the first instance of G_1, which by assumption is PRG. Next the output of the first instance of G_1 is divided in two parts: the first n bits are fed into the second instance of G_1 as seed, while the last bit becomes the first bit of the output. If we repeat the process m times we obtain m pseudorandom bits.
It can be shown (using the hybrid argument)that G (the PRG with poly expansion factor constructed using G_1) is also PRG. The sketch of the proof follows.

Proof. Let's consider $m + 1$ intermediate distributions $H_i : 0 \leq i \leq m$ where the first i bits are chosen from the uniform distribution and the last $m - 1$ bits are chosen from the output of G. By construction H_0 is the full output of G while H_m is the truly uniform distribution U_m. H_i and H_{i+1} will be different in only one bit.
We assume that G is not PRG, that is there exist some adversary A that can distinguish between G (H_0) and U_m (H_m) with non negligible advantage ϵ. That means that there must exists an i such that A can distinguish between H_i and H_{i+1}. Then we can construct and adversary $A^{'}$ that uses A to distinguish between G_1 and U_n by at least $\frac{\epsilon}{m}$. □

Theorem 1.3. There exist PRG iff there exist OWF

Proof. $PRG \rightarrow OWF$
Consider a generator $G : \{0,1\}^n \rightarrow \{0,1\}^{2n}$.
We define the following one way function: $f : \{0,1\}^n \rightarrow \{0,1\}^n$ as follows: $f(x,y) = G(x)$ with $|x| = |y|$.
Now we assume by contradiction that there exist and adversary A that can invert f, that is:

$$Prob[f(A(f(U_{2n}))) = f(U_{2n})] > \epsilon \tag{2}$$

Then we can construct and adversary D that can distinguish between $G(U_n)$ and U_{2n}.
D on input $y = f(x)$ output 1 if $f(A(y)) = y$) else output 0. By construction we have that $f(U_{2n}) = G(U_n)$, therefore:

$$Prob[D(G(U_n)) = 1] = Prob[f(A(G(U_n)) = G(U_n)] > \epsilon \tag{3}$$

Also we know that at most 2^n values of the image of f can be mapped to the pre-image, therefore A can invert f only on those values, that is:

$$P(D(U_{2n}) = 1) = P[f(A(U_{2n})) = f(U_{2n})] \leq \frac{1}{2^n} \tag{4}$$

(3) - (4) gives us a contradiction wrt the definition of PRG. □

The other direction of the proof, $OWF \rightarrow PRG$, can be found at `http://en.wikipedia.org/wiki/Pseudorandom_generator_theorem` and it makes use of the definition of hardcore bit.

1.2 Construction of PRG

We can construct PRG from OW Permutations and the use of hard codre bit:

$$G(x) = f(x), B(x) \tag{5}$$

The proof that G is PRG follows from the definition of hard core bit: $f(x), B(x) \approx_c f(x), b$ with $b \leftarrow_R \{0,1\}$

2 Pseudorandom Permutation - PRP and pseudorandom function PRF

A family of functions $P^{a,n}$ is pseudorandom function/permutation if for any ppt adversary A:

$$Prob_{k \in \{0,1\}^a}[A^{P_k} = 1] - Prob_{P \in P^n}[A^P = 1] < \epsilon \tag{6}$$

where P^n is the family of function/permutations $P : \{0,1\}^n \rightarrow \{0,1\}^n$

Basically we have two experiment: in the first one the adversary interacts with an oracle representing a specific f_k from the family $F = \{f_k\}_{k\in\{0,1\}^n}$ giving x and receiving $f_k(x)$ back. In the second experiment the adversary interacts with an oracle that choose a random function from the family. The function/permutation is random if the adversary cannot distinguish between the two experiments.

There is also a stronger definition (SPRF/PRP) where the adversary has access to the oracle to compute the image f_k and the oracle to compute the pre-image f_k^{-1}. This definition takes into account a stronger adversary that has also access to a decryption oracle. A can generates smart cipher texts in order to learn whether or not a real cipher was generated from a random permutation or from a pseudo random permutation.

How to distinguish a random permutation from something totally random? COLLISION

Theorem 2.1. if there exist OWF then there exist SPRP

It can be proven in stages:

1. OWF $\rightarrow$ PRG
2. PRG $\rightarrow$ PRF
3. PRF $\rightarrow$ PRP
4. PRP $\rightarrow$ SPRF

Stage 2) is the **GGM theorem**.

Theorem 2.2. If there exist PRG then there exist PRF

GGM show how to efficiently construct a PRF given a PRG using a binary tree. The proof that the result is a PRF is done by hybrid arguments.

Stage 3) (form PRF to PRP) is done using the **Feistel Transform**. We need three round of the Feistel Transform to obtain a PRP. The theorem that prove it is **Luby and Rackoff '88**