Lecture 18: Pseudorandom Functions
Let \( G_{m,n,k} = \{g_1, g_2, \ldots, g_{2^k}\} \) be a set of functions such that each \( g_i: \{0, 1\}^m \rightarrow \{0, 1\}^n \)

This set of functions \( G_{m,n,k} \) is called a pseudo-random function if the following holds.

Suppose we pick \( g \leftarrow G_{m,n,k} \). Let \( x_1, \ldots, x_t \in \{0, 1\}^m \) be distinct inputs. Given \((x_1, g(x_1)), \ldots, (x_{t-1}, g(x_{t-1}))\) for any computationally bounded party the value \( g(x_t) \) appears to be uniformly random over \( \{0, 1\}^n \)
Before we construct a PRF, let us consider the following secret-key encryption scheme.

1. **Gen()**: Return \( sk = id \leftarrow \{1, \ldots, 2^k\} \)

2. **Enc\(_id\)(m)**: Pick a random \( r \leftarrow \{0, 1\}^m \). Return \((m \oplus g_{id}(r), r)\), where \( m \in \{0, 1\}^n \).

3. **Dec\(_id\)(\(\tilde{c}, \tilde{r}\))**: Return \( \tilde{c} \oplus g_{id}(\tilde{r}) \).

**Features.** Suppose the messages \( m_1, \ldots, m_u \) are encrypted as the cipher-texts \((c_1, r_1), \ldots, (c_u, r_u)\).

- As long as the \( r_1, \ldots, r_u \) are all distinct, each one-time pad \( g_{id}(r_1), \ldots, g_{id}(r_u) \) appear uniform and independent of others to computationally bounded adversaries. So, this encryption scheme is secure against computationally bounded adversaries!

- The probability that any two of the randomness in \( r_1, \ldots, r_u \) are not distinct is very small (We shall prove this later as “Birthday Paradox”)

- This scheme is a “state-less” encryption scheme. Alice and Bob do not need to remember any private state (except the secret-key \( sk \))!
We shall consider the construction of Goldreich-Goldwasser-Micali (GGM) construction.

Let $G : \{0, 1\}^k \rightarrow \{0, 1\}^{2k}$ be a PRG. We define $G(x) = (G_0(x), G_1(x))$, where $G_0, G_1 : \{0, 1\}^k \rightarrow \{0, 1\}^k$.

Let $G' : \{0, 1\}^k \rightarrow \{0, 1\}^n$ be a PRG.

We define $g_{id}(x_1x_2\ldots x_m)$ as follows

$$G'(G_{x_m}(\cdots G_{x_2}(G_{x_1}(id))\cdots))$$
Consider the execution for $x = x_1 x_2 x_3 = 010$. Output $z$ is computed as follows.

- Go Left because $x_1 = 0$
- Go Right because $x_2 = 1$
- Go Left because $x_3 = 0$
We give the pseudocode of algorithms to construct PRG and PRF using a OWP $f : \{0, 1\}^{k/2} \rightarrow \{0, 1\}^{k/2}$

- Suppose $f : \{0, 1\}^{k/2} \rightarrow \{0, 1\}^{k/2}$ is a OWP
- We provide the pseudocode of a PRG $G : \{0, 1\}^k \rightarrow \{0, 1\}^t$, for any integer $t$, using the one-bit extension PRG construction of Goldreich-Levin hardcore predicate construction. Given input $s \in \{0, 1\}^k$, it outputs $G(s)$.

$G(k, t, s)$:

1. Interpret $s = (r, x)$, where $r, x \in \{0, 1\}^{k/2}$
2. Initialize bits = [ ] (i.e., an empty list)
3. Initialize $z = x$
4. For $i = 1$ to $t$:
   1. bits.append($\langle r, z \rangle$), here $\langle \cdot, \cdot \rangle$ is the inner-product
   2. $z = f(z)$
5. Return bits
We provide the pseudocode of the PRF $g_{id}: \{0, 1\}^m \rightarrow \{0, 1\}^n$, where $id \in \{0, 1\}^k$, using the GGM construction. Given input $x \in \{0, 1\}^m$, it outputs $g_{id}(x)$.

\[
g(m, n, k, id, x):
\begin{align*}
1 & \text{ Interpret } x = x_1 x_2 \ldots x_m, \text{ where } x_1, \ldots, x_m \in \{0, 1\} \\
2 & \text{ Initialize } inp = id \\
3 & \text{ For } i = 1 \text{ to } m:
   \begin{align*}
   1 & \text{ Let } y = G(k, 2k, inp) \\
   2 & \text{ If } x_i = 0, \text{ then } inp \text{ is the first } k \text{ bits of } y. \text{ Otherwise (if } \text{ } x_i = 1), \text{ inp } \text{ is the last } k \text{ bits of } y.
   \end{align*}
4 & \text{ Return } G(k, n, inp)
\end{align*}
\]