Last Time

- Proof via Reduction: $f_{\times}$ is a weak OWF
- Amplification: From weak to strong OWFs
Today

- What do OWFs Hide?
- Hard Core Predicate
- Concluding Remarks on OWFs
- Scribe notes volunteers?
What OWFs Hide

- The concept of OWFs is simple and concise.
- But OWFs often not very useful by themselves.
- It only guarantees that $f(x)$ hides $x$ but nothing more!
  - E.g., it may not hide first bit of $x$,
  - Or even first half bits of $x$
  - Or ANY subset of bits
- In fact: if $a(x)$ is some information about $x$, we don’t know if $f(x)$ will hide $a(x)$ for any non-trivial $a(\cdot)$

Is there any non-trivial function of $x$, even 1 bit, that OWFs hide?
A **hard core predicate** for a OWF \( f \)
- is a function over its inputs \( \{x\} \)
- its output is a single bit (called the “hard core bit”)
- it can be easily computed given \( x \)
- but “hard to compute” given only \( f(x) \)

**Intuition:** \( f \) may leak many bits of \( x \) but it does not leak the hard-core bit.

In other words, learning the hardcore bit of \( x \), even given \( f(x) \), is “as hard as” inverting \( f \) itself.

**Think:** What does “hard to compute” mean for a single bit?
- you can always guess the bit with probability 1/2.
Hard Core Predicate: Definition

- Hard-core bit cannot be efficiently “learned” or “predicted” or “computed” with probability $\geq \frac{1}{2} + \mu(|x|)$ even given $f(x)$

**Definition (Hard Core Predicate)**

A predicate $h : \{0, 1\}^* \rightarrow \{0, 1\}$ is a hard-core predicate for $f(\cdot)$ if $h$ is efficiently computable given $x$ and there exists a negligible function $\nu$ s.t. for every non-uniform PPT adversary $A$ and $\forall n \in \mathbb{N}$:

$$\Pr \left[ x \leftarrow \{0, 1\}^n : A(1^n, f(x)) = h(x) \right] \leq \frac{1}{2} + \nu(n).$$
Can we construct hard-core predicates for general OWFs \( f \)?

Define \( \langle x, r \rangle \) to be the **inner product** function mod 2. I.e.,

\[
\langle x, r \rangle = \left( \sum_i x_i r_i \right) \mod 2
\]

Same as taking \( \oplus \) of a random subset of bits of \( x \).

**Theorem (Goldreich-Levin)**

Let \( f \) be a OWF (OWP). Define function

\[
g(x, r) = (f(x), r)
\]

where \( |x| = |r| \). Then \( g \) is a OWF (OWP) and

\[
h(x, r) = \langle x, r \rangle
\]

is a hard-core predicate for \( g \).
Some remarks

- The theorem is not for $f$, but for a different function, $g$.
- Is this useful at all?
  - Indeed, consider the function $g'$:
    \[
    g'(1x) = g'(0x) = f(x).
    \]
  - Clearly, the first bit of $g$'s input is hard core for $g$.
  - It works even if $f$ is not one-way!
- The problem with the above is that it “looses” information about its input. This is not good for applications.
- It “explains” nothing about the inherent hardness of $f$
- Function $g$ in the GL theorem \textit{statistically} does not loose any information that $f$ does not about its input.
- ...and the hard core bit for $g$ is easy to guess if $f$ is not one-way.
Proof of the Goldreich-Levin theorem

- Proof via reduction?
- **Main challenge:** Adversary $A$ for $h$ only outputs 1 bit. Need to build an inverter $B$ for $f$ that outputs $n$ bits.
Warmup Proof (1)

- **Assumption**: Given \( g(x, r) = (f(x), r) \), adversary \( A \) always (i.e., with probability 1) outputs \( h(x, r) \) correctly

- **Inverter \( B \):**
  - Compute \( x_i^* \leftarrow A(f(x), e_i) \) for every \( i \in [n] \) where:
    \[
    e_i = (0, \ldots, 0, 1, \ldots, 0) \quad \text{(}i-1\text{-times)}
    \]
  - Output \( x^* = x_1^* \ldots x_n^* \)
Warmup Proof (2)

- **Assumption:** Given $g(x, r) = (f(x), r)$, for every $x$, adversary $A$ outputs $h(x, r)$ with probability $3/4 + \varepsilon(n)$ over the choices of $r$.

  \[
  \forall x : \Pr_{r}[A(f(x), r) = h(x, r)] \geq \frac{3}{4} + \varepsilon(n).\]

- **Main Problem:** Adversary may not work on “improper” inputs (e.g., $r = e_i$ as in previous case)

- **Main Idea:** Split each query into two queries s.t. each query individually looks random
Warmup Proof (2)

- **Inverter $B$:**
  - Let $a := A(f(x), e_i \oplus r)$ and $b := A(f(x), r)$, for $r \leftarrow \{0, 1\}^n$
  - Compute $c := a \oplus b$ as a guess for $x_i^*$
  - Repeat many times to get many such $c$ and take majority to get $x_i^*$
  - Output $x^* = x^*_1 \ldots x^*_n$

- **Proof** that $B$ inverts $f(x)$:
  - If both $a$ and $b$ are correct, then $c = x_i$ because:
    $$c = a \oplus b = \langle x, e_i \oplus r_i \rangle \oplus \langle x, r \rangle = x \cdot (r + e_i) + x \cdot r \mod 2 = x \cdot e_i = x_i.$$  
  - Claim: $c = x_i$ with probability $1/2 + 2\varepsilon$
  - Proof: by union bound $A$ is wrong about either $a$ or $b$ with at most:
    $$(1/4 - \varepsilon(n)) + (1/4 - \varepsilon(n)) = 1/2 - 2\varepsilon$$
    probability. So $a, b$ are correct w/ prob. $\geq 1/2 + 2\varepsilon$, so is $c$. □

- If you repeat $\frac{2n}{\varepsilon(n)}$ times, by **Chernoff Bound**, majority of $c$ will be correct $x_i^*$ w/ $1 - e^{-n}$ prob.
Full Proof of the GL Theorem

In the next class!

- Goldreich-Levin theorem has been extremely influential even outside cryptography
- Has applications to learning, list-decoding codes, extractors,...
- Great tool to add to your toolkit
Further Remarks

- One-way functions are necessary for most of cryptography
- But often not sufficient for things like key-exchange or public-key encryption.
- *Black-box* separations known [Impagliazzo-Rudich’89]; Open problem: full separations not known
- More examples of one-way functions?
- More than 1 hard core bit?
- Other ways to get hard core bit?
We saw a OWF based on factoring. Are there more candidates?

Many examples based on:

- **Discrete Log**: compute \( x \in G \) from \((g, y, p)\) where \( g \) generates a group \( G \), and \( p = |G| \) is prime, and \( y = g^x \) in \( G \).

- **RSA Problem**: compute \( d \) from \((e, N)\) s.t. \( e \cdot d \equiv 1 \mod \phi(N) \) where \( \phi(N) = |\mathbb{Z}_N^*| \) and \( N \) is product of two large primes.

- **Quadratic Residuosity**: compute square roots of perfect squares modulo \( N \) (Rabin’s function).

**More**: more examples from lattices and LWE problem; such “hardness assumptions” are few and rare.

You actually get a collection of OWFs from the above, not a single OWF. However, collections imply a single OWF as well. (discussed later)

Special hard-core predicates and more than 1 bit based on specific structures of these functions. (For general OWFs, GL can be extended to yield \( \log n \) hard core bits).
On more examples of OWFs

Universal One-way Functions (Levin)
- Suppose somebody tells you that OWFs exist! but they don’t know what that function is.
- Can you use this fact to build an explicit OWF? Explicit = one which you could implement (in principle, on Turing machines).
- Yes! Levin constructs an explicit function which is one-way if there exists any OWF (even if not known explicitly).

OWFs from the famous “P vs NP” problem?
- OWFs whose hardness can be reduced to the validity of $P \neq NP$.
- Unlikely to exist based on current evidence [Goldreich-Goldwasser-Moshkovitz,...]
Proof on the board?