1 More on One-Way Functions

f is a one-way function if $f \in$ poly and:

$$\forall c \text{ and } \forall A^*, \exists n_0 \text{ s.t.} \forall n > n_0, \Pr_{x \leftarrow \$, |x|=n}[A^*(f(x)) \in f^{-1}(f(x))] < \frac{1}{n^c}$$

This means that for a random x from the domain, the probability that any function A^* can successfully find an input to the function x' such that $f(x') = f(x)$, is **negligible**, or decreases slower than any inverse polynomial as $|x| = n$ increases.

The contrapositive of this statement, would be that f is not a one way function if:

$$\exists c \text{ and } \exists A^*, \text{ s.t. for infinitely many } n$$

There exists an A^* that can invert $f(x)$ with probability greater than or equal to $\frac{1}{n^c}$.

In other words, we say that $x \xrightarrow{f} y$ is easy but $x \xleftarrow{\text{hard}} y$

2 Defining Hardcore Bits

We say a function $H : (x) \rightarrow 0/1$ is **Hard Core bit predicate** if, given $f(x)$ for one-way function f, the probability of any adversary predicting $H(x)$ with probability greater than $\frac{1}{2}$ is negligible.

To understand this definition we have the following protocol for one way permutation f:

1. C commits to b by sending $f(x) \oplus b||0^{n-1} = s_1 ... s_n \oplus b||0^{n-1}$. This just xors the first bit of $f(x)$ with b.
2. Decommit by sending x so adversary can xor the commitment to find b.

This protocol assumes that the first bit of $f(x)$ is unpredictable, however, we could have a one-way function that does not change the first bit from $x \rightarrow f(x)$ which would make this commitment protocol predicatable. Since no assumptions can be made about the underlying function properties other than it being one way, a protocol must be designed with the only assumption being that function f is one-way.

2.1 Goldreich and Levin

Goldreich and Levin showed that there exists a hardocore bit for every 1-way function. The desgin for the hardcore bit is as follows.

1. Challenger (C) selects a random $x \overset{\$}{\leftarrow} X$ and $r \overset{\$}{\leftarrow} X$ where $|x| = |r|$. C sends $y = f(x)$ and r to adversary A.
2. Now, C calculates $< x_1x_2...x_n, r >$ or $\sum_{i=1}^{n} x_ir_i$ mod 2 and commits $b \oplus < x, r >$
3. Decommitment is simply sending x.

The rest of the notes are aimed at proving that this protocol is indeed hardcore bit.

2.2 Proving Hardcore

Recall the definition of hardcore means that the adversary cannot predict $b \oplus < x, r >$ with probability greater than or equal to $\frac{1}{2} + \frac{1}{n^c}$. This is equivalent to predicting $< x, r >$.

2.2.1 Proof 1: Perfect A

Assume we have an A that can always guess whether or not $< r, x >$ will be 0/1. Then we can discover x in the following manner:

1. A takes in two inputs, r and $f(x)$. Given theses two values it can guess whether $< r, x >$ will be 0/1 with probability 1 given $f(x), r$.
2. Feed A the input inputs r and r^i where $r \leftarrow \{0,1\}^n$ and r^i is r with the ith bit flipped.
3. This process will return $b_1 =< r, x >$ and $b_2 =< r^i, x >$. By analysis, we can see that if x_i is 1, $b_1 \oplus b_2$ will output 1 (since flipping the ith bit of r will cause the result of the dot product to change), and if x_i is 0, it will output 0. Therefore, we claim that $b_1 \oplus b_2$ gives us x_i with probability 1.
4. If we repeat this for r and r_i for $i = 0...n$, we will get all n bits of x with probability 1.

2.2.2 Proof 2: Weaker A

Now consider an A that can correctly output $< r, x >$ with probability $3/4 + \epsilon(n)$ and gets it wrong with probability $1/4 - \epsilon(n)$. To demonstrate an adversary A^* that can extract x given $(r, f(x))$ and oracle access to A we need the following probability law:

Union Bound: Given events A and B that need not be independent from one another:

$$Pr[A \bigcup B] \leq Pr[A] + Pr[B]$$

In other words, the probability of A **or** B occuring is upper bounded by the sum of the individual probaiblities that each event occurs.

Given this, and the strategy we used earlier, we feed A, which has $\frac{3}{4} + \epsilon(n)$ chance of correctly guessing $< x, r >$, a random p and p^i. With this, the probability that $A(f(x), p) \oplus A(f(x), p^i)$ properlly returns x_i is equal to:

$$Pr[A(f(x), p) =< x, p > \wedge A(f(x), p^i) =< x, p^i >]$$

Or the probability that A outputs the correct bit for each input. This can be rewritten as follows:

$$1 - Pr[A(f(x), p) \neq< x, p > \vee A(f(x), p^i) \neq< x, p^i >]$$

Given the union bound and fact that A outputs the wrong bit with probability $\frac{1}{4} - \epsilon(n)$, we can lower bound the above probability by maximizing the probability that A fails in either case, getting:

$$1 - (\frac{1}{4} - \epsilon(n)) - (\frac{1}{4} - \epsilon(n)) = \frac{1}{2} + 2\epsilon(n)$$

So each run of $A(f(x), p)$ and $A(f(x), p^i)$ succeeds with probability $\frac{1}{2} + 2\epsilon(x)$, which, using the chernoff bound, allows us to find x_i with overwhelming probability through repeated experiments. Therefore, we can do this for each x_i and obtain the full string x with overwhelming probability of success.

2.2.3 Any A with Advantage

We will argue that given any adversary A_b that can guess $< x, r >$ with probability $> \frac{1}{2} + \epsilon(x)$, we can construct A_f which will be able to guess x, and therefore can invert f, with non-negligible probability. First we need to define a **good** x which are x's where:

$$Pr_{p,\omega}[A_B(f(x), p) = B(x, p)] > \frac{1}{2} + \frac{\epsilon(n)}{2}$$

We prove the following claim: At least an $\frac{\epsilon(n)}{2}$ fraction of x's are good. This claim has significance as it demonstrates that a **non-negligible** fraction of x's have advantage when $f(x)$ is fed to A. This means that for a randomly selected x, the algorithm we design has a non-negligible advantage on the input with probaility of at least $\frac{\epsilon(n)}{2}$. Since our A_f needs only a non-negligible advantage, it is okay if in only works on a non-negligible fraction of the x's.

Proof. To prove this, we will find an *upper bound* to the advantage of A_b over all (x, p, ω) tuples, where ω is the distribution of coinflips of A. For sake of contradiction, suppose that there are less than a $\frac{\epsilon(n)}{2}$ fraction of x's that are good. Since we claim that A_b's advantage is greater than a $\epsilon(n)$, we know that:

$$Pr_{x,p,\omega}[A_b(f(x), p) = B(x, p)] > \frac{1}{2} + \epsilon(n)$$

(Where $B(x, p)$ is the dot product of the two values). We can thus rewrite the probability of A_b predicting x as the sum of the following probabilities:

$$= Pr_{x,p,\omega}[A_b(f(x), p) = B(x, p) \mid x \text{ is good.}] \cdot Pr_x[x \text{ is good}] + \\ Pr_{x,p,\omega}[A_b(f(x), p) = B(x, p) | x \text{ is not good.}] \cdot Pr_x[x \text{ is not good.}]$$

In other words, this is the probability that A_b predicts x from $f(x)$ over all possible random p's and coinflips ω. To maximize this value, we will maximize all possibilities giving us the following:

- $Pr_{x,p,\omega}[A_b(f(x), p) = B(x, p) | \; x \text{ is good}] = 1$
- $Pr_x[x \text{ is good}] = \frac{\epsilon(x)}{2}$ (Since we assume that less than a $\epsilon(n)/2$ fraction of x's are good).
- $Pr_{x,p,\omega}[A_b(f(x), p) = B(x, p) | \; x \text{ is not good}] = \frac{1}{2} + \frac{\epsilon(n)}{2}$ (By definition of x being good, the probability that A_b can guess b on a not good x is upper bounded by $\frac{1}{2} + \frac{\epsilon(n)}{2}$
- $Pr_x[x \text{ is not good}] = 1$ (This cannot be true since $Pr_x[x \text{ is good}]$ is not 0, however, as we are looking for an upperbound we can set it to the maximum possible value, being 1)

Therefore, given our assumption, the maximum probability that the above sum could come to is:

$$
\begin{aligned}
&= 1 \cdot \frac{\epsilon(n)}{2} + (\frac{1}{2} + \frac{\epsilon(n)}{2}) \cdot 1 \\
&= \frac{1}{2} + \epsilon(n)
\end{aligned}
$$

However, recall that A_b has an advantage **greater** that $\frac{1}{2} + \epsilon(n)$. Since the maximum advantage that A_b can have under our assumption yields one that is not greater than this, we have a contradiction. Therefore, there must be at least an $\epsilon(n)/2$ fraction of x's that are good. ■