1 Interactive Proofs vs. Argument Schemes in Zero-Knowledge

1.1 Computational/Perfect Security and Binding

We have the following definition of **Hiding**, and **Binding** in commitment schemes.

1. **Hiding:** How well, for commitment $commit(b)$, the scheme hides b from the receiver R. That is, how difficult it is for R to determine the value of b given $commit(b)$.

2. **Binding:** How difficult, for commitment $commit(b)$, it is for the Commiter C to decommit to a values $b' \neq b$ after sending $commit(b)$.

1.1.1 Statistically Binding, Computationally Hiding

Recall the scheme where the Commiter C sends commitment:

$$commit(b) = (b \oplus < x, r >, f(x), r >$$

With one way permutation f and decommits with x. Firstly, since f is a permutation, only one x can decommit the $f(x)$ value so the commitment is **perfectly binding**. Furthermore, we demonstrated that there is no PPT adversary that can sucessfully predict $< x, r >$ with non-negligible probability, and therefore the scheme is **computationally binding**.

1.1.2 Computationally Binding, Statistically Hiding

First we define the notion of a **Claw-Free Pair** (f_1, f_2) which are a pair of 1-way permutations s.t. no polytime machine can find x_0, x_1 such that $f_(x_0) = f_(x_1)$.

With this, we can construct the following commitment scheme that is **Perfectly hiding** and **Computationally Binding**:

1. Commiter C selects random $x \xleftarrow{\$} X$ and commitment $f_b(x)$.

2. Decommit by sending x, where receiver can calculate the decommitment by testing whether $f_1/f_0 = f_b(x)$.

This scheme is clearly computationally binding since a PPT commiter cannot find two values x_0, x_1 such that $f_0(x_0) = f_1(x_1)$. Furthermore, since it relies on the fact that C is weak, the receiver R cannot distinguish whether $f_b(x)$ was derived from $f_0(x_0)$ or $f_1(x_1)$, and therefore cannot determine the commitment.

This type of commitment is called a Zero Knowledge Argument opposed to a proof since an infinitely powerful prover could cheat. However, in practice it is very useful as even a infinitely powerful *eavesdropper* cannot successfuly decommit C's commitment before C does.

1.1.3 Impossibiliy of Perfect Hiding and Binding

It is impossible for a scheme to be both perfectly hiding and binding. Consider a commitment scheme between an infinitely powerful commiter C and receiver R that achieves both perfect hiding and binding. Perfect binding implies that once C chooses b, $decommit(commit(b)) = b$. Since the receiver is infinitely powerful, it can enumerate through all possible inputs of the commitment scheme until it finds one that generates the same value as $commit(b)$, and can therefore determine b, demonstrating that the scheme cannot be perfectly hiding.

2 Pseudo Random Generators

A pseudo random generator G takes small seed s and outputs string $G(s)$ that is of length $Q(|s|)$ for some polynomial Q. The notion of **Indistinguishability** means that two strings $r \xleftarrow{\$} \{0,1\}^{Q(|x|)}$ and $G(s)$ with $s \xleftarrow{\$} \{0,1\}$ are Indistinguishabile by any PPT adversary A. We have a new notion of **Next-Bit Security** as follows:

2.1 Next-Bit Security of a PRG [Blum, Micali]]

A PRG is **Next-Bit secure** if, given a stream of polynomially many bits generated from $PRG(s) = b_0 b_1 ... b_{poly}$, no PPT adversary, given $b_0, ..., b_{i-1}$ can predict b_i with probability $\frac{1}{2} + \epsilon(poly)$.

2.1.1 Next-Bit secure PRG

We define the PRG as follows given 1 way permutation f:

1. G selects a random r and seed x_0 to generate a string of length $m+2n$.

2. Firstly, for we generate m bits in the following manner. For $i = 1, ..., m$, $x_i = f(x_{i-1})$ and $d_i =< x_i, r >$. Then the PRG outputs $x_m|r|d_m, d_{m-1}, ..., d_1$.

To demonstrate that this scheme is secure, we must argue that that any adversary that can determine b_i given $b_1, ..., b_{i-1}$ with non-negligible advantage can sucessfully invert f with non-negligible probability. Firstly, recall that if an adversary A exists that can successfully predict $< x, r >$ given $(f(x), r)$, then we can construct an A' that can successfuly invert f with non-negligible probability. Thus, we will show that the existence of an adversary that has advantage over guessing b_i for some i has an advantage in guessing $(f(x), r)$. The adversary A, given oracle access to A_G, which predicts b_i, does the following:

1. A receives $(f(x'), r)$ and has to output $< x', r >$ with non-negligible probability, firstly, it generates the string as done above using r. Then, for random x_i, it replaces it with $f(x')$ and computes the remainder of the bits according the the algorithm.

2. Now, it sequentially passing the bits $b_1, ..., b_{m+2n}$ to A_G until A_G outputs b_i. If i was the location where the initial x was swapped for $f(x')$, then A outputs A_G's guess.

Demonstrating Advantage:

Firstly, observe that the last $2n$ bits of the PRG are completely random. Since f is a permutation $Pr[x_m = x_m] = \frac{1}{2^n}$. Furthermore, since r is also random it cannot predict any b_i for $i = 1, ..., 2n$ with probability greater than 1/2.

For the last m bits of G, there must be some i where A_G can predict b_i with non-negligible probability. If that location is exactly where we replaced $< x_i, r >$ with $< x', r >$, then the A's advantage of guessing $< x', r >$ is $\frac{1}{m} \cdot \epsilon(n)$ where $\epsilon(n)$ is A_G's advantage of predicting the next bit of the PRG, which is still a non-negligible advantage.

To demonstrate why we can simply replace a random $< x_i, r >$ with $< x', r >$, consider any honestly generated sequence $x_m|r|b_1, ..., b_m$. Then A_G must have an advantage on guessing some b_i. Since $b_i =< x_i, r >$, we can guarantee that A has a non-negligible advantage on $(f(x_i), r)$ since we can construct the first $2n + i$ bits as described above.

2.2 Equivalence of Indistinguishability and Next-Bit Security

We need to show that **Indistinguishability** $\iff$ **Next Bit Security**. Direction that **Indistinguishability** $\implies$ **Next-Bit Security**. To demonstrate this we show that if we have an adversary that can guess the next bit of $G(s)$, then there exists adversary that can distinguish $G(s)$ and random string r. This is simply done by feeding the adversary A the bits of $G(s)/r$ and outputting 1 if it guesses a bit correctly and 0 if it gets it wrong.

2.2.1 Next-Bit Security $\implies$ Indistinguishability

We demonstrate that if there exists an adversary A that can distinguish $G(s)$ from r with non-negligible probability, then it can guess the next-bit of a stream of either r or $G(s)$ with non-negligible probability. We once again use the hybrid argument as follows, since:

$$|Pr[A(G(s)) = 1] - Pr[A(r) = 1]| = \epsilon(n)$$

There must be some i where, $X_i = PRG(S)_1, ..., PRG(S)_i, r_{i+1}, ..., r_n$:

$$|Pr[A(X_{i-1}) = 1] - Pr[A(X_i) = 1]| \geq \frac{\epsilon(n)}{m}$$

We design A', which guesses bit i from $G(S)$ as follows.

1. Receive the first $i-1$ bits and construct $X = b_1, ...b_{i-1}$.
2. Guess bit b_i as b' and generate $n-i$ random bits r.
3. If $A(b_1, ..., b_{i-1}, b', r_{i+1}, ...r_n) = 1$ output guess b', else $1 - b'$

Analysis: To argue that it guesses b_i with non-negligible probability, we first simplify $\epsilon(n)/m$ to $\epsilon(n)$ and get the following:

$$\begin{aligned} Pr[A'(b_1...b_{i-1}) = b_i] &= Pr[b' = b_i] \cdot Pr[A(X_i) = 1] \\ &\quad + Pr[b' = 1 - b_1] \cdot Pr[A(b_1...b_{i-1}\hat{b_i}r_{i+1}...r_n) = 0] \\ &= \frac{1}{2} \cdot \epsilon(n) + \frac{1}{2} \cdot q \end{aligned}$$

Before solving for q, the first part of the above probability is the probability that the guessed b' is actually b_i times the probability A outputs 1. We say that $P_i = Pr[A(X_i) = 1]$ To solve for q, we need to find out:

$$Pr[A(X_{i-1}) = 1] = Pr[A(b_1..b_{i-1}b_iR) = 1] \cdot \frac{1}{2} + \frac{1}{2} \cdot Pr[A(b_1...b_{i-1}\overline{b_i}R) = 1]$$
$$= \frac{1}{2} \cdot P_i + \frac{1}{2} \cdot (1 - q)$$
$$q = P_i + 1 - 2 \cdot P_{i-1}$$

Plugging q back into the orginal equation we get:

$$= \frac{1}{2} \cdot P_i + \frac{1}{2} \cdot q$$
$$\frac{1}{2}(2P_i + 1 - 2p_{i-1})$$
$$\geq \frac{1}{2} + \epsilon(n)$$

3 Commitment protocol From PRG

The following commitment scheme uses a PRG that takes a seed of length n and outputs a pseudorandom value of length $3n$. Formally we have:

$$G : \{0,1\}^n \to \{0,1\}^3 n$$

The commitment scheme is performed as follows:

1. Receiver R generates a random length $3n$ bitstring r and sends it to commiter C.
2. Commiter generates random seed s of length n and commits $G(s)$ if they want to commit a 0 and $G(s) \oplus r$ if they want to commit a 1.
3. The commiter decommits by sending r.

Hiding
First, to demonstrate the **computational** hiding property of the protocol we consider an adversary A that can determine if the commitment c is either $G(s)$ or $G(s) \oplus r$. Observe that $G(s) \oplus r$ is a random bitstring. Therefore, the receiver must be able to distinguish $G(s)$ from a uniform random string $G(s) \oplus r$ to be able to break the hiding property of the protocol which is not possible since G is a PRG

Binding
To demonstrate the **statistically** binding property of this protocol we have consider what must be possible for the commiter C to cheat. For this to be possible, C must find two inputs s and s' such that either:

- $G(s) = G(s') \oplus r$
- $G(s) \oplus r = G(s')$

In otherwords there must be two inputs s and s' such that

$$G(s) \oplus G(s') = r$$

We demonstrate that the probability that this can be possible is at most negligible. Firstly, we observe that the codomain of this PRG is 2^{2n} times larger than the domain. Because of this, there can be at most $2^n/2^{3n} = 1/2^{2n}$ fraction of the codomain values can be mapped to. Since it is clear that

$$\underset{r \leftarrow \$}{Pr}[\exists x\ G(x) = r] \geq \underset{r \leftarrow \$}{Pr}[\exists x, y, x \neq y \mid G(x) = G(y) = r]$$

It is clear that if R chooses a random $3n$ length value r, then the probability that C can cheat is negligible.