Pseudorandomness (III) & One-Way Functions

601.642/442: Modern Cryptography

Fall 2022

Recap: One-bit stretch PRG $\implies$ Poly-bit stretch PRG

- A PRG with **one-bit stretch** can be used to construct a PRG with arbitrary **polynomial-bit stretch**.

Recap: One-bit stretch PRG $\implies$ Poly-bit stretch PRG

- A PRG with **one-bit stretch** can be used to construct a PRG with arbitrary **polynomial-bit stretch**.

Construction of $G_{poly} : \{0,1\}^n \rightarrow \{0,1\}^{\ell(n)}$

Let $G : \{0,1\}^n \rightarrow \{0,1\}^{n+1}$ be a one-bit stretch PRG.

$$\begin{aligned} s &= x_0 \\ G(x_0) &= x_1 \| b_1 \\ &\vdots \\ G(x_{\ell(n)-1}) &= x_{\ell(n)} \| b_{\ell(n)} \end{aligned}$$

$G_{poly}(s) := b_1 \ldots b_{\ell(n)}$

Recap: Pseudorandomnes of G_{poly}

- In order to show $\left\{G_{poly}(s);\ s \xleftarrow{\$} \{0,1\}^n\right\} \approx_c \left\{r \xleftarrow{\$} \{0,1\}^{\ell(n)}\right\}$ we considered the following hybrid experiments:

Recap: Pseudorandomnes of G_{poly}

- In order to show $\left\{G_{poly}(s);\ s \xleftarrow{\$} \{0,1\}^n\right\} \approx_c \left\{r \xleftarrow{\$} \{0,1\}^{\ell(n)}\right\}$ we considered the following hybrid experiments:

Experiment $\mathcal{H}_1$	Experiment $\mathcal{H}_2$	Experiment $\mathcal{H}_{\ell(n)}$
$s = x_0$	$s = x_0$	$s = X_0$
$G(x_0) = x_1 \| b_1$	$s_1 \| u_1 = x_1 \| u_1$	$s_1 \| u_1 = x_1 \| u_1$
$G(x_1) = x_2 \| b_2$	$G(x_1) = x_2 \| b_2$	$s_2 \| u_2 = x_2 \| u_2$
$\ldots$	$\ldots$	$\ldots$
$\ldots$	$\ldots$	$\ldots$
$G(x_{\ell(n)-1}) = x_{\ell(n)} \| b_{\ell(n)}$	$G(X_{\ell(n)-1}) = x_{\ell(n)} \| b_{\ell(n)}$	$s_{\ell(n)} \| u_{\ell(n)} = x_{\ell(n)} \| u_{\ell(n)}$
$\boxed{\text{Output } G(s) := b_1 b_2 \ldots b_{\ell(n)}}$	$\boxed{\text{Output } G(s) := u_1 b_2 \ldots b_{\ell(n)}}$	$\boxed{\text{Output } G(s) := u_1 u_2 \ldots u_{\ell(n)}}$

Recap: Pseudorandomnes of G_{poly}

- In order to show $\left\{G_{poly}(s);\ s \xleftarrow{\$} \{0,1\}^n\right\} \approx_c \left\{r \xleftarrow{\$} \{0,1\}^{\ell(n)}\right\}$ we considered the following hybrid experiments:

Experiment $\mathcal{H}_1$	Experiment $\mathcal{H}_2$	Experiment $\mathcal{H}_{\ell(n)}$
$s = x_0$	$s = x_0$	$s = X_0$
$G(x_0) = x_1 \| b_1$	$s_1 \| u_1 = x_1 \| u_1$	$s_1 \| u_1 = x_1 \| u_1$
$G(x_1) = x_2 \| b_2$	$G(x_1) = x_2 \| b_2$	$s_2 \| u_2 = x_2 \| u_2$
$\ldots$	$\ldots$	$\ldots$
$\ldots$	$\ldots$	$\ldots$
$G(x_{\ell(n)-1}) = x_{\ell(n)} \| b_{\ell(n)}$	$G(X_{\ell(n)-1}) = x_{\ell(n)} \| b_{\ell(n)}$	$s_{\ell(n)} \| u_{\ell(n)} = x_{\ell(n)} \| u_{\ell(n)}$
Output $G(s) := b_1 b_2 \ldots b_{\ell(n)}$	Output $G(s) := u_1 b_2 \ldots b_{\ell(n)}$	Output $G(s) := u_1 u_2 \ldots u_{\ell(n)}$

- And established that $\forall i \in [\ell(n) - 1]$, $\mathcal{H}_i \approx_c \mathcal{H}_{i+1}$.

Contrapositive Point of View

- So far, we have proved statements in the forward direction.

  "If G is a one-bit stretch PRG, then G_{poly} is a poly-bit stretch PRG."

Contrapositive Point of View

- So far, we have proved statements in the forward direction.

  "If G is a one-bit stretch PRG, then G_{poly} is a poly-bit stretch PRG."

- Let's now think about the **contrapositve** of these statements.

  "If G_{poly} is a ***not*** *poly-bit stretch PRG, then G is* ***not*** *a one-bit stretch PRG."*

Contrapositive Point of View

- So far, we have proved statements in the forward direction.

  "If G is a one-bit stretch PRG, then G_{poly} is a poly-bit stretch PRG."

- Let's now think about the **contrapositve** of these statements.

  *"If G_{poly} is a **not** poly-bit stretch PRG, then G is **not** a one-bit stretch PRG."*

- If G_{poly} is not a PRG, then there exists a n.u. PPT adversary $\mathcal{A}$ who can distinguish between its output on a random input and a uniformly sampled string with some **non-negligible** advantage μ.

Contrapositive Point of View

- So far, we have proved statements in the forward direction.

  "If G is a one-bit stretch PRG, then G_{poly} is a poly-bit stretch PRG."

- Let's now think about the **contrapositve** of these statements.

  *"If G_{poly} is a **not** poly-bit stretch PRG, then G is **not** a one-bit stretch PRG."*

- If G_{poly} is not a PRG, then there exists a n.u. PPT adversary $\mathcal{A}$ who can distinguish between its output on a random input and a uniformly sampled string with some **non-negligible** advantage μ.

Contrapositive Point of View

- So far, we have proved statements in the forward direction.

  "If G is a one-bit stretch PRG, then G_{poly} is a poly-bit stretch PRG."

- Let's now think about the **contrapositve** of these statements.

  *"If G_{poly} is a **not** poly-bit stretch PRG, then G is **not** a one-bit stretch PRG."*

- If G_{poly} is not a PRG, then there exists a n.u. PPT adversary $\mathcal{A}$ who can distinguish between its output on a random input and a uniformly sampled string with some **non-negligible** advantage μ.

Can we somehow use $\mathcal{A}$ to also break security of G?

Contrapositive Point of View

- We just proved security of G_{poly} using a sequence of hybrids.

Contrapositive Point of View

- We just proved security of G_{poly} using a sequence of hybrids.
- If we assume that an adversary $\mathcal{A}$ exists, who has non-negligible advantage in breaking the security of G_{poly}, then at least one of the steps of our previous proof must break down.

Contrapositive Point of View

- We just proved security of G_{poly} using a sequence of hybrids.
- If we assume that an adversary $\mathcal{A}$ exists, who has non-negligible advantage in breaking the security of G_{poly}, then at least one of the steps of our previous proof must break down.
- By hybrid lemma, $\mathcal{A}$ can distinguish between at least 1 pair of consecutive hybrids (say $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$) with at least $\mu/\ell(n)$ advantage.

Contrapositive Point of View

- We just proved security of G_{poly} using a sequence of hybrids.
- If we assume that an adversary $\mathcal{A}$ exists, who has non-negligible advantage in breaking the security of G_{poly}, then at least one of the steps of our previous proof must break down.
- By hybrid lemma, $\mathcal{A}$ can distinguish between at least 1 pair of consecutive hybrids (say $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$) with at least $\mu/\ell(n)$ advantage.

Contrapositive Point of View

- We just proved security of G_{poly} using a sequence of hybrids.
- If we assume that an adversary $\mathcal{A}$ exists, who has non-negligible advantage in breaking the security of G_{poly}, then at least one of the steps of our previous proof must break down.
- By hybrid lemma, $\mathcal{A}$ can distinguish between at least 1 pair of consecutive hybrids (say $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$) with at least $\mu/\ell(n)$ advantage.

Lemma (Alternate way to state Hybrid Lemma)

Let $X^1, \ldots, X^m$ be distribution ensembles for $m = \mathsf{poly}(n)$. Suppose there exists a distinguisher/adversary $\mathcal{A}$ that distinguishes between X^1 and X^m with probability μ. Then $\exists i \in [m-1]$, such that $\mathcal{A}$ distinguishes between X^i and X^{i+1} with advantage at least μ/m.

Contrapositive Point of View

- We just proved security of G_{poly} using a sequence of hybrids.
- If we assume that an adversary $\mathcal{A}$ exists, who has non-negligible advantage in breaking the security of G_{poly}, then at least one of the steps of our previous proof must break down.
- By hybrid lemma, $\mathcal{A}$ can distinguish between at least 1 pair of consecutive hybrids (say $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$) with at least $\mu/\ell(n)$ advantage.
- In our previous proof, we relied on the security of G to argue indistinguishability of each pair of consecutive hybrids.

Contrapositive Point of View

- We just proved security of G_{poly} using a sequence of hybrids.
- If we assume that an adversary $\mathcal{A}$ exists, who has non-negligible advantage in breaking the security of G_{poly}, then at least one of the steps of our previous proof must break down.
- By hybrid lemma, $\mathcal{A}$ can distinguish between at least 1 pair of consecutive hybrids (say $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$) with at least $\mu/\ell(n)$ advantage.
- In our previous proof, we relied on the security of G to argue indistinguishability of each pair of consecutive hybrids.
- We will now use $\mathcal{A}$ that has non-negligible advantage in distinguishing between $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$, to construct another adversary $\mathcal{B}$ to break security of G.

Contrapositive Point of View

- We just proved security of G_{poly} using a sequence of hybrids.
- If we assume that an adversary $\mathcal{A}$ exists, who has non-negligible advantage in breaking the security of G_{poly}, then at least one of the steps of our previous proof must break down.
- By hybrid lemma, $\mathcal{A}$ can distinguish between at least 1 pair of consecutive hybrids (say $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$) with at least $\mu/\ell(n)$ advantage.
- In our previous proof, we relied on the security of G to argue indistinguishability of each pair of consecutive hybrids.
- We will now use $\mathcal{A}$ that has non-negligible advantage in distinguishing between $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$, to construct another adversary $\mathcal{B}$ to break security of G.
- However, since G is a secure PRG, no such n.u. PPT $\mathcal{A}$ should exist. This will give us a contradiction and imply that our assumption was incorrect. G_{poly} is in fact secure.

Proof via Reduction

- How do we construct $\mathcal{B}$?

Proof via Reduction

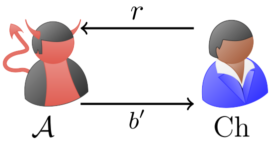
- How do we construct $\mathcal{B}$?
- We consider the game-based definition of PRG.

Proof via Reduction

- How do we construct $\mathcal{B}$?
- We consider the game-based definition of PRG.

Proof via Reduction

- How do we construct $\mathcal{B}$?
- We consider the game-based definition of PRG.



$b \xleftarrow{\$} \{0,1\}$

if $b = 1$: $s \xleftarrow{\$} \{0,1\}^n$

$r := G(s)$

if $b = 0$: $r \xleftarrow{\$} \{0,1\}^{\ell(n)}$

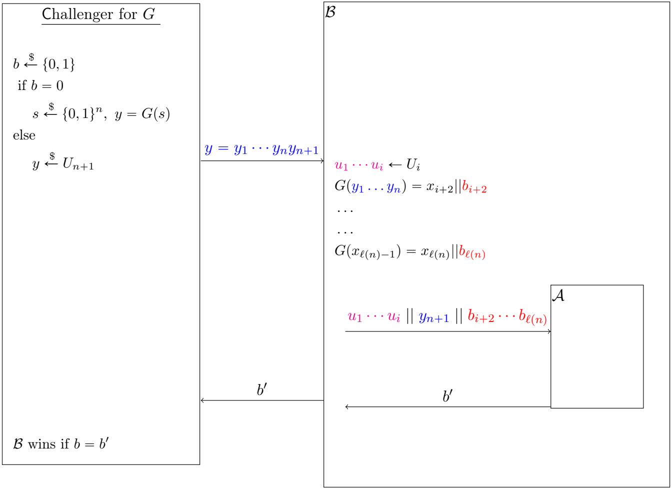
$\boxed{\mathcal{A} \text{ wins if } b' = b}$

$$\left| \Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0] \right| \leqslant \nu(n)$$

Proof by Reduction

- This proof technique is also called **proof by reduction**.

- This proof technique is also called **proof by reduction**.



Proof by Reduction

- If y is pseudorandom, i.e., sampled as $y = G(s)$, then the input to $\mathcal{A}$ is distributed identically to the output of $\mathcal{H}_i$.

Proof by Reduction

- If y is pseudorandom, i.e., sampled as $y = G(s)$, then the input to $\mathcal{A}$ is distributed identically to the output of $\mathcal{H}_i$.
- Otherwise, i.e., y is (truly) random, and therefore the input to $\mathcal{A}$ is distributed identically to the output of $\mathcal{H}_{i+1}$.

Proof by Reduction

- If y is pseudorandom, i.e., sampled as $y = G(s)$, then the input to $\mathcal{A}$ is distributed identically to the output of $\mathcal{H}_i$.
- Otherwise, i.e., y is (truly) random, and therefore the input to $\mathcal{A}$ is distributed identically to the output of $\mathcal{H}_{i+1}$.
- Hence, $\mathcal{B}$ has the **same advantage** in distinguishing between the output of G and a pseudorandom string that $\mathcal{A}$ has in distinguishing between $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$.

Proof by Reduction

- If y is pseudorandom, i.e., sampled as $y = G(s)$, then the input to $\mathcal{A}$ is distributed identically to the output of $\mathcal{H}_i$.
- Otherwise, i.e., y is (truly) random, and therefore the input to $\mathcal{A}$ is distributed identically to the output of $\mathcal{H}_{i+1}$.
- Hence, $\mathcal{B}$ has the **same advantage** in distinguishing between the output of G and a pseudorandom string that $\mathcal{A}$ has in distinguishing between $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$.
- Moreover, since $\mathcal{A}$ is n.u. PPT, so is $\mathcal{B}$. This is a contradiction!

Proof by Reduction

- If y is pseudorandom, i.e., sampled as $y = G(s)$, then the input to $\mathcal{A}$ is distributed identically to the output of $\mathcal{H}_i$.
- Otherwise, i.e., y is (truly) random, and therefore the input to $\mathcal{A}$ is distributed identically to the output of $\mathcal{H}_{i+1}$.
- Hence, $\mathcal{B}$ has the **same advantage** in distinguishing between the output of G and a pseudorandom string that $\mathcal{A}$ has in distinguishing between $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$.
- Moreover, since $\mathcal{A}$ is n.u. PPT, so is $\mathcal{B}$. This is a contradiction!
- Hence, $G_{\mathsf{poly}(n)}$ is a PRG.

Proof by Reduction: Key Points

- These are four important things that you must work through for a valid reduction:

Proof by Reduction: Key Points

- These are four important things that you must work through for a valid reduction:
  1. **Input Mapping:** How to map the input that "outer adversary" $\mathcal{B}$ recieves from the challenger to an input to the "internal adversary" $\mathcal{A}$?

Proof by Reduction: Key Points

- These are four important things that you must work through for a valid reduction:
  1. **Input Mapping:** How to map the input that "outer adversary" $\mathcal{B}$ recieves from the challenger to an input to the "internal adversary" $\mathcal{A}$?
  2. **Input Distribution:** Does the input mapping provide the right distribution of inputs that $\mathcal{A}$ expects?

Proof by Reduction: Key Points

- These are four important things that you must work through for a valid reduction:
  1. **Input Mapping:** How to map the input that "outer adversary" $\mathcal{B}$ recieves from the challenger to an input to the "internal adversary" $\mathcal{A}$?
  2. **Input Distribution:** Does the input mapping provide the right distribution of inputs that $\mathcal{A}$ expects?
  3. **Output Mapping:** How do we map the output that $\mathcal{A}$ provides to an output for $\mathcal{B}$?

Proof by Reduction: Key Points

- These are four important things that you must work through for a valid reduction:
  1. **Input Mapping:** How to map the input that "outer adversary" $\mathcal{B}$ recieves from the challenger to an input to the "internal adversary" $\mathcal{A}$?
  2. **Input Distribution:** Does the input mapping provide the right distribution of inputs that $\mathcal{A}$ expects?
  3. **Output Mapping:** How do we map the output that $\mathcal{A}$ provides to an output for $\mathcal{B}$?
  4. **Probability:** When we assume existence of $\mathcal{A}$, we also assume that $\mathcal{A}$ wins with non-negligible advantage. What is the probability/advantage that $\mathcal{B}$ wins, given the mappings above?

One-Way Functions

One Way Functions

$$\begin{tikzcd} x \arrow[r, bend left, "\text{EASY}"] & f(x) \arrow[l, bend left, "\text{HARD}"] \end{tikzcd}$$

- A function is one-way if it “easy to compute,” but “hard to invert”

One Way Functions

$$\begin{tikzcd} x \arrow[r, bend left, "\text{EASY}"] & f(x) \arrow[l, bend left, "\text{HARD}"] \end{tikzcd}$$

- A function is one-way if it “easy to compute,” but “hard to invert”
- Necessary for the existence of most cryptographic primitives (e.g., multi-message encryption, digital signatures)

One Way Functions

$$\begin{tikzcd} x \arrow[rr, bend left, "\text{EASY}"] & & f(x) \arrow[ll, bend left, "\text{HARD}"] \end{tikzcd}$$

- A function is one-way if it “easy to compute,” but “hard to invert”
- Necessary for the existence of most cryptographic primitives (e.g., multi-message encryption, digital signatures)
- Also sufficient for some cryptographic primitives (e.g., pseudorandom generators, secret-key encryption, digital signatures).

One Way Functions

$$\begin{tikzcd} x \arrow[r, bend left, "\text{EASY}"] & f(x) \arrow[l, bend left, "\text{HARD}"] \end{tikzcd}$$

- A function is one-way if it “easy to compute,” but “hard to invert”
- Necessary for the existence of most cryptographic primitives (e.g., multi-message encryption, digital signatures)
- Also sufficient for some cryptographic primitives (e.g., pseudorandom generators, secret-key encryption, digital signatures).

One Way Functions

$$\begin{tikzcd} x \arrow[r, bend left, "\text{EASY}"] & f(x) \arrow[l, bend left, "\text{HARD}"] \end{tikzcd}$$

- A function is one-way if it “easy to compute,” but “hard to invert”
- Necessary for the existence of most cryptographic primitives (e.g., multi-message encryption, digital signatures)
- Also sufficient for some cryptographic primitives (e.g., pseudorandom generators, secret-key encryption, digital signatures).

How to define one-way functions?

Defining One Way Functions: Attempt 1

Attempt 1: A function $f : \{0,1\}^* \to \{0,1\}^*$ is a one-way function (OWF) if it satisfies the following two conditions:

- **Easy to compute:** there is a polynomial-time algorithm $\mathcal{C}$ s.t. $\forall x \in \{0,1\}^*$,

$$\Pr\left[\mathcal{C}(x) = f(x)\right] = 1.$$

Defining One Way Functions: Attempt 1

Attempt 1: A function $f : \{0,1\}^* \rightarrow \{0,1\}^*$ is a one-way function (OWF) if it satisfies the following two conditions:

- **Easy to compute:** there is a polynomial-time algorithm $\mathcal{C}$ s.t. $\forall x \in \{0,1\}^*$,
$$\Pr\left[\mathcal{C}(x) = f(x)\right] = 1.$$
- **Hard to invert:** for every non-uniform PPT adversary $\mathcal{A}$, for any input length $n \in \mathbb{N}$

Probability of Inversion is Negligible

Defining One Way Functions: Attempt 1

Attempt 1: A function $f : \{0,1\}^* \rightarrow \{0,1\}^*$ is a one-way function (OWF) if it satisfies the following two conditions:

- **Easy to compute:** there is a polynomial-time algorithm $\mathcal{C}$ s.t. $\forall x \in \{0,1\}^*$,
$$\Pr\left[\mathcal{C}(x) = f(x)\right] = 1.$$
- **Hard to invert:** for every non-uniform PPT adversary $\mathcal{A}$, for any input length $n \in \mathbb{N}$
$$\Pr\left[\mathcal{A} \text{ inverts } f(x) \text{ for random } x\right] \leqslant \textit{negligible}.$$

This is called **average-case** hardness.

One Way Functions: Definition

Definition (One Way Function)

A function $f : \{0,1\}^* \to \{0,1\}^*$ is a one-way function (OWF) if it satisfies the following two conditions:

- **Easy to compute:** there is a polynomial-time algorithm $\mathcal{C}$ s.t. $\forall x \in \{0,1\}^*$,
$$\Pr\left[\mathcal{C}(x) = f(x)\right] = 1.$$
- **Hard to invert:** there exists a negligible function $\nu : \mathbb{N} \to \mathbb{R}$ s.t. for every non-uniform PPT adversary $\mathcal{A}$ and $\forall n \in \mathbb{N}$:
$$\Pr\left[x \xleftarrow{\$} \{0,1\}^n, x' \leftarrow \mathcal{A}(1^n, f(x)) : f(x') = f(x)\right] \leqslant \nu(n).$$

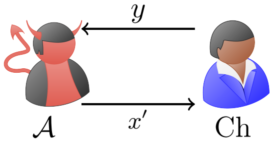
- The above definition is also called **strong** one-way functions.

One Way Functions: Game Based Definition

It is also instructive to think of that definition in this game-based form.

One Way Functions: Game Based Definition

It is also instructive to think of that definition in this game-based form.



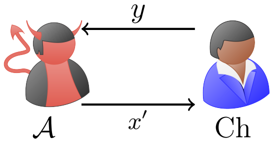
$$x \xleftarrow{\$} \{0,1\}^n$$

$$y = f(x)$$

$$\boxed{\mathcal{A} \text{ wins if } f(x') = y}$$

One Way Functions: Game Based Definition

It is also instructive to think of that definition in this game-based form.



$$x \xleftarrow{\$} \{0,1\}^n$$
$$y = f(x)$$

$$\boxed{\mathcal{A} \text{ wins if } f(x') = y}$$

We say that $f : \{0,1\}^* \to \{0,1\}^*$ is a one-way function if there exists a negligible function $\nu : \mathbb{N} \to \mathbb{R}$ s.t. for every n.u. PPT adversary $\mathcal{A}$ and $\forall n \in \mathbb{N}$:

$$\Pr[\mathcal{A} \text{ wins}] \leqslant \nu(n).$$

Injective OWFs and One Way Permutations (OWP)

- **Injective or 1-1 OWFs**: each image has a unique pre-image:

$$f(x_1) = f(x_2) \implies x_1 = x_2$$

- **One Way Permutations (OWP)**: 1-1 OWF with the additional conditional that "each image has a pre-image"

  (Equivalently: domain and range are of same size.)

Existence of OWFs

Do OWFs exist?

Existence of OWFs

Do OWFs exist?

- NOT Unconditionally — proving that f is one-way requires proving (at least) $\mathbf{P} \neq \mathbf{NP}$.

Existence of OWFs

Do OWFs exist?

- NOT Unconditionally — proving that f is one-way requires proving (at least) $\mathbf{P} \neq \mathbf{NP}$.
- However, we can construct them ASSUMING that certain problems are hard.

Existence of OWFs

Do OWFs exist?

- NOT Unconditionally — proving that f is one-way requires proving (at least) $\mathbf{P} \neq \mathbf{NP}$.
- However, we can construct them ASSUMING that certain problems are hard.
- Such constructions are sometimes called "candidates" because they are based on an assumption or a conjecture.

Factoring Problem

- Consider the **multiplication** function $f_\times : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$:

$$f_\times(x, y) = \begin{cases} \perp & \text{if } x = 1 \vee y = 1 \\ x \cdot y & \text{otherwise} \end{cases}$$

Factoring Problem

- Consider the **multiplication** function $f_\times : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$:

$$f_\times(x, y) = \begin{cases} \perp & \text{if } x = 1 \vee y = 1 \\ x \cdot y & \text{otherwise} \end{cases}$$

- The first condition helps exclude the trivial factor 1.

Factoring Problem

- Consider the **multiplication** function $f_\times : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$:

$$f_\times(x, y) = \begin{cases} \perp & \text{if } x = 1 \vee y = 1 \\ x \cdot y & \text{otherwise} \end{cases}$$

- The first condition helps exclude the trivial factor 1.
- Is $f_\times$ a OWF?

Factoring Problem

- Consider the **multiplication** function $f_\times : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$:

$$f_\times(x, y) = \begin{cases} \perp & \text{if } x = 1 \vee y = 1 \\ x \cdot y & \text{otherwise} \end{cases}$$

- The first condition helps exclude the trivial factor 1.
- Is $f_\times$ a OWF?
- **Clearly not!** With prob. 1/2, a random number (of any fixed size) is <u>even</u>. I.e., xy is even w/ prob. $\frac{3}{4}$ for random (x, y).

Factoring Problem

- Consider the **multiplication** function $f_\times : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$:

$$f_\times(x,y) = \begin{cases} \perp & \text{if } x = 1 \vee y = 1 \\ x \cdot y & \text{otherwise} \end{cases}$$

- The first condition helps exclude the trivial factor 1.
- Is $f_\times$ a OWF?
- **Clearly not!** With prob. 1/2, a random number (of any fixed size) is <u>even</u>. I.e., xy is even w/ prob. $\frac{3}{4}$ for random (x, y).
- Inversion: given number z, output $(2, z/2)$ if z is even and $(0, 0)$ otherwise! (succeeds 75% time)

Factoring Problem (continued)

- Eliminate such trivial small factors.

Factoring Problem (continued)

- Eliminate such trivial small factors.
- Let Π_n be the set of all **prime** numbers $< 2^n$.

Factoring Problem (continued)

- Eliminate such trivial small factors.
- Let Π_n be the set of all **prime** numbers $< 2^n$.
- Choose numbers p and q randomly from Π_n and multiply.

Factoring Problem (continued)

- Eliminate such trivial small factors.
- Let Π_n be the set of all **prime** numbers $< 2^n$.
- Choose numbers p and q randomly from Π_n and multiply.
- This is unlikely to have small trivial factors.
  [Factoring Assumption] For every (non-uniform PPT) adversary $\mathcal{A}$, there exists a negligible function ν such that

$$\Pr\left[p \xleftarrow{\$} \Pi_n; q \xleftarrow{\$} \Pi_n; N = pq : \mathcal{A}(N) \in \{p, q\}\right] \leqslant \nu(n).$$

Factoring Problem (continued)

- Factoring assumption is a well established conjecture.

Factoring Problem (continued)

- Factoring assumption is a well established conjecture.
- Studied for a long time, with no known polynomial-time attack.

Factoring Problem (continued)

- Factoring assumption is a well established conjecture.
- Studied for a long time, with no known polynomial-time attack.
- Best known algorithms for breaking Factoring Assumption:

$$2^{O\left(\sqrt{n \log n}\right)} \qquad \text{(provable)}$$
$$2^{O\left(\sqrt[3]{n \log^2 n}\right)} \qquad \text{(heuristic)}$$

Factoring Problem (continued)

- Factoring assumption is a well established conjecture.
- Studied for a long time, with no known polynomial-time attack.
- Best known algorithms for breaking Factoring Assumption:

$$
\begin{array}{ll}
2^{O\left(\sqrt{n \log n}\right)} & \text{(provable)} \\
2^{O\left(\sqrt[3]{n \log^2 n}\right)} & \text{(heuristic)}
\end{array}
$$

- Can we construct OWFs from the Factoring Assumption?

Multiplication Function

- Going back to the multiplication function $f_\times : \mathbb{N}^2 \to \mathbb{N}$.

$$f_\times(x, y) = \begin{cases} \perp & \text{if } x = 1 \vee y = 1 \\ x \cdot y & \text{otherwise} \end{cases}$$

Multiplication Function

- Going back to the multiplication function $f_\times : \mathbb{N}^2 \rightarrow \mathbb{N}$.

$$f_\times(x,y) = \begin{cases} \perp & \text{if } x = 1 \vee y = 1 \\ x \cdot y & \text{otherwise} \end{cases}$$

- **Observation 1:** If randomly chosen x and y happen to be primes, no PPT $\mathcal{A}$ can invert (except with negligible probability). Call it the **GOOD** case.

Multiplication Function

- Going back to the multiplication function $f_\times : \mathbb{N}^2 \rightarrow \mathbb{N}$.

$$f_\times(x, y) = \begin{cases} \perp & \text{if } x = 1 \vee y = 1 \\ x \cdot y & \text{otherwise} \end{cases}$$

- **Observation 1:** If randomly chosen x and y happen to be primes, no PPT $\mathcal{A}$ can invert (except with negligible probability). Call it the **GOOD** case.

- If GOOD case occurs with probability $> \varepsilon$,

  $\Rightarrow$ every PPT $\mathcal{A}$ must fail to invert $f_\times$ with probability at least ε.

Multiplication Function

- Going back to the multiplication function $f_\times : \mathbb{N}^2 \to \mathbb{N}$.

$$f_\times(x, y) = \begin{cases} \perp & \text{if } x = 1 \vee y = 1 \\ x \cdot y & \text{otherwise} \end{cases}$$

- **Observation 1:** If randomly chosen x and y happen to be primes, no PPT $\mathcal{A}$ can invert (except with negligible probability). Call it the **GOOD** case.

- If GOOD case occurs with probability $> \varepsilon$,

  $\Rightarrow$ every PPT $\mathcal{A}$ must fail to invert $f_\times$ with probability at least ε.

- Now suppose that ε is a **noticeable function** (say e.g. an inverse polynomial, i.e., $\frac{1}{p(\cdot)}$)

  $\Rightarrow$ every $\mathcal{A}$ must fail to invert $f_\times$ with **noticeable** probability.

Multiplication Function

- Going back to the multiplication function $f_\times : \mathbb{N}^2 \to \mathbb{N}$.

$$f_\times(x,y) = \begin{cases} \perp & \text{if } x = 1 \vee y = 1 \\ x \cdot y & \text{otherwise} \end{cases}$$

- **Observation 1:** If randomly chosen x and y happen to be primes, no PPT $\mathcal{A}$ can invert (except with negligible probability). Call it the **GOOD** case.

- If GOOD case occurs with probability $> \varepsilon$,

  $\Rightarrow$ every PPT $\mathcal{A}$ must fail to invert $f_\times$ with probability at least ε.

- Now suppose that ε is a **noticeable function** (say e.g. an inverse polynomial, i.e., $\frac{1}{p(\cdot)}$)

  $\Rightarrow$ every $\mathcal{A}$ must fail to invert $f_\times$ with **noticeable** probability.

- This is already useful!

Multiplication Function

- Going back to the multiplication function $f_\times : \mathbb{N}^2 \rightarrow \mathbb{N}$.

$$f_\times(x,y) = \begin{cases} \perp & \text{if } x = 1 \vee y = 1 \\ x \cdot y & \text{otherwise} \end{cases}$$

- **Observation 1:** If randomly chosen x and y happen to be primes, no PPT $\mathcal{A}$ can invert (except with negligible probability). Call it the **GOOD** case.

- If GOOD case occurs with probability $> \varepsilon$,

  $\Rightarrow$ every PPT $\mathcal{A}$ must fail to invert $f_\times$ with probability at least ε.

- Now suppose that ε is a **noticeable function** (say e.g. an inverse polynomial, i.e., $\frac{1}{p(\cdot)}$)

  $\Rightarrow$ every $\mathcal{A}$ must fail to invert $f_\times$ with **noticeable** probability.

- This is already useful!

- Usually called a **weak** OWF.

Noticeable Functions

Let us start by formally defining noticeable functions. These are functions that are **at most polynomially small**.

Noticeable Functions

Let us start by formally defining noticeable functions. These are functions that are **at most polynomially small**.

Definition (Noticeable Function)

A function $\nu(n)$ is noticeable if $\exists c, n_0$ such that $\forall n > n_0$, $\nu(n) \geqslant \frac{1}{n^c}$.

Noticeable Functions

Let us start by formally defining noticeable functions. These are functions that are **at most polynomially small**.

Definition (Noticeable Function)

A function $\nu(n)$ is noticeable if $\exists c, n_0$ such that $\forall n > n_0$, $\nu(n) \geqslant \frac{1}{n^c}$.

Note that a non-negligible function is not necessarily a noticeable function. Example:

$$f(n) = \begin{cases} 1 & \text{if } n \text{ is even} \\ 2^{-n} & \text{if } n \text{ is odd} \end{cases}.$$

This function is non-negligible, but not noticeable. Why?