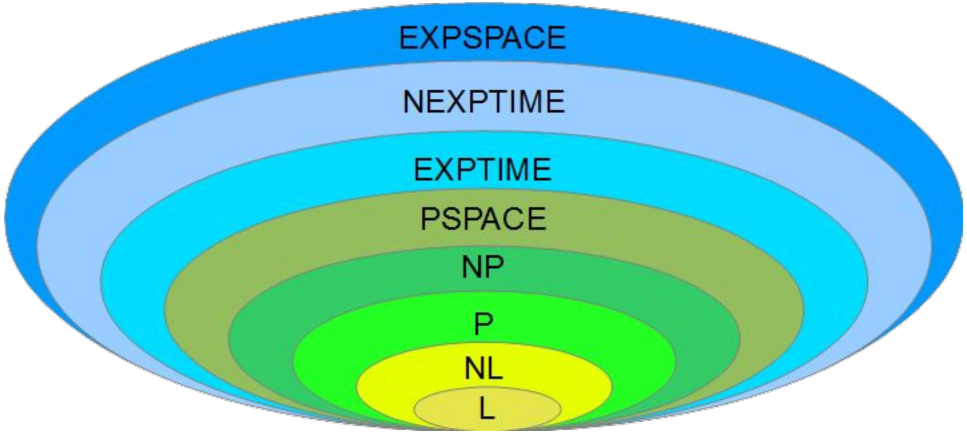
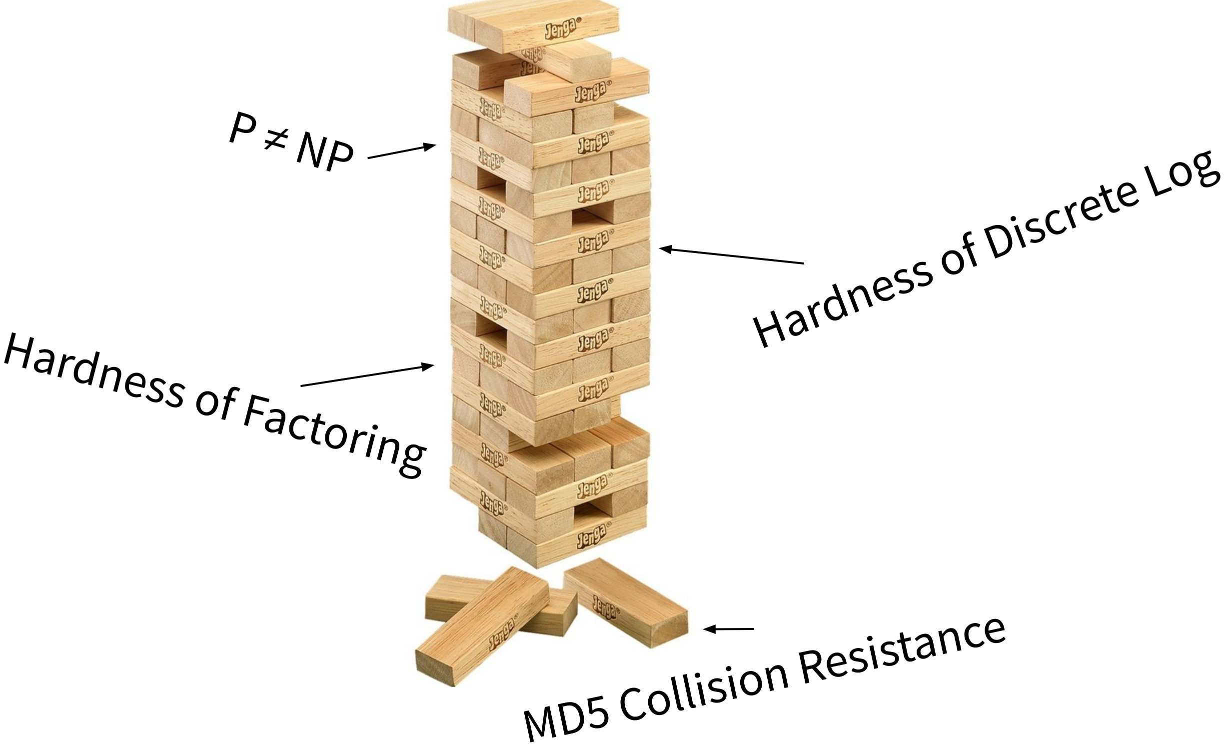
Fine-grained Cryptography

Naganesh Jaladanki

Fine-grained Cryptography



Modern Cryptography



P ≠ NP

Hardness of Discrete Log

Hardness of Factoring

MD5 Collision Resistance

Fine-Grained Cryptography

(also called moderately hard cryptography [Dwork-Naor])

- Honest prover: complexity class $\mathbf{C_{hon}}$
- Adversary: complexity class $\mathbf{C_{adv}} > \mathbf{C_{hon}}$

Examples:

$\mathbf{C_{hon}} = \text{Time}(n)$	$\mathbf{C_{adv}} = \text{Time}(n^2)$
$\mathbf{C_{hon}} = \text{Space}(s)$	$\mathbf{C_{adv}} = \text{Space}(s^2)$
$\mathbf{C_{hon}} = \text{ParTime}(d)$	$\mathbf{C_{adv}} = \text{ParTime}(d^2)$

Time resource bounds

↓

Space resource bounds

↓

Parallel time resource bounds

Time resource bounds

↓

Space resource bounds

↓

Parallel time resource bounds

Merkle Puzzles [Mer78]

- $\mathbf{C}_{\mathbf{hon}}$ = Time(n)
- $\mathbf{C}_{\mathbf{adv}}$ = Time(n^2)
- Key exchange protocol

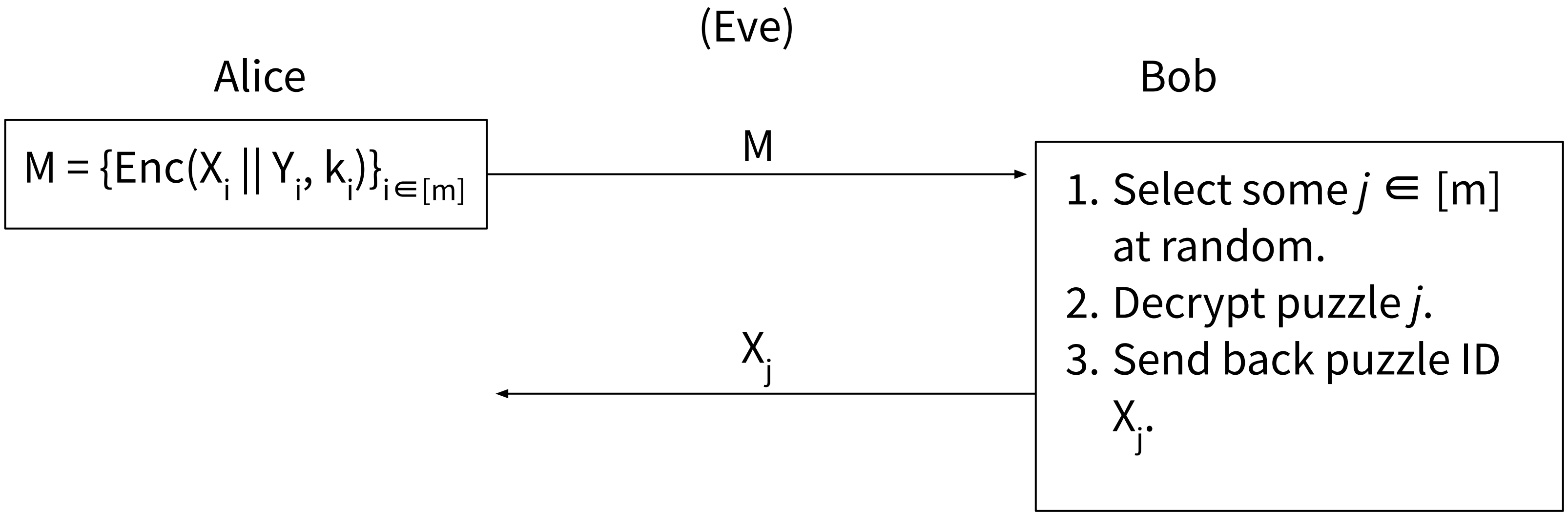
Merkle Puzzles [Mer78]

$$\boxed{\mathrm{Enc}(X \,||\, Y, k)}$$

X = message ID (think UUID)
Y = randomly generated symmetric key
k = randomly generated encryption key

$k \in \boldsymbol{K}$ such that $|\boldsymbol{K}| = n$
Time to break: $O(n)$

Merkle Puzzles [Mer78]



(Eve)

Alice

Bob

$M = \{Enc(X_i \,||\, Y_i, k_i)\}_{i \in [m]}$

M

1. Select some $j \in [m]$ at random.
2. Decrypt puzzle *j*.
3. Send back puzzle ID X_j.

X_j

Key Exchange: both parties know Y_j at the end.
Honest party time: $O(m + n)$
Adversary time: $O(mn)$

Recent advances

- [VLW15] gave a key-exchange protocol extending Merkle’s Puzzles to exchange a lg(**n**) bit key in time $\mathbf{n}^{2\mathbf{k}-\mathbf{g}}$ for the honest prover and $O(\mathbf{n}^{3\mathbf{k}-2\mathbf{g}})$ for an adversary.
    - Constants **k**, **g** depend on the difficulty of particular “puzzle” used for the protocol. Described 3 sufficient properties needed of a computational problem to work with this protocol.
- [BRSV17] used specific reductions in fine-grained complexity to obtain a worst-case to average-case reduction, used to build a Proof Of Work cryptographic primitive.
    - Unfortunately, they showed that building a true one-way function using their approach would violate NSETH, a popular hardness assumption.

Time resource bounds

Space resource bounds

Parallel time resource bounds

Memory-bounded Adversaries [CM97]

- $\mathbf{C}_{\mathbf{hon}}$ = Space(s)
- $\mathbf{C}_{\mathbf{adv}}$ = Space(s^2)
- Key exchange protocol

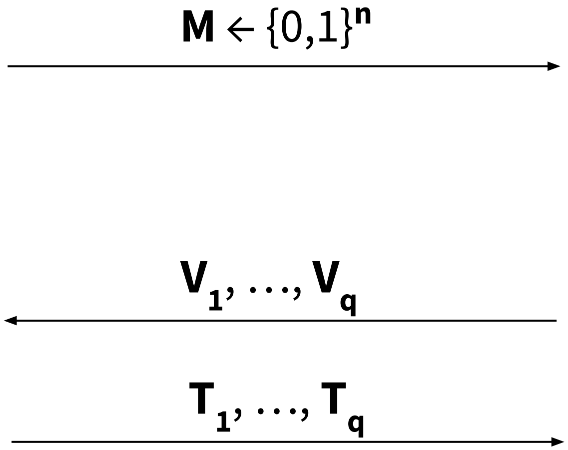
Memory-bounded Adversaries [CM97]

Alice

(Eve)

Bob

1. Select **q** uniform and pairwise independent indices $\mathbf{T_1, \ldots, T_q} \in [\mathbf{n}]$. Record values of stream at those indices.



$\mathbf{M} \leftarrow \{0,1\}^{\mathbf{n}}$

$\mathbf{V_1}, \ldots, \mathbf{V_q}$

$\mathbf{T_1}, \ldots, \mathbf{T_q}$

1. Select **q** uniform and pairwise independent indices $\mathbf{V_1, \ldots, V_q} \in [n]$. Record values of stream at those indices.

2. Compute common indices: $\mathbf{S} = \mathbf{T} \cap \mathbf{V}$.

3. Compute key **K** as values of stream at common indices.

2. Compute common indices: $\mathbf{S} = \mathbf{T} \cap \mathbf{V}$.

3. Compute key **K** as values of stream at common indices.

Memory-bounded Adversaries [CM97]

Lemma 1: The expected number of common indices $\mathbf{l} = \mathbf{q}^2 / \mathbf{n}$.

So, for a constant key size **c**, we would expect to set **q** = O(sqrt(**n**))

Lemma 2: If $\mathbf{T}_1, \ldots, \mathbf{T}_q$ and $\mathbf{V}_1, \ldots, \mathbf{V}_q$ are independent sequences of uniform and pairwise independent random variables, then their intersection $\{\mathbf{S}_1,\ldots,\mathbf{S}_l\} = \mathbf{T} \cap \mathbf{V}$ is pairwise independent.

So, Eve has no hope but to store all information from the stream until the indices are exchanged between Alice and Bob.

Memory-bounded Adversaries [CM97]

Theorem: This protocol uses O(**s**) space for the honest party and O($\mathbf{s}^2$) space for any adversary with a constant probability of guessing the key, where **s** = O(sqrt(**n**)).

For a constant key size **c**, we would expect to set **q** = sqrt(**cn**). Alice and Bob only need to save sqrt(**cn**) information, but Alice needs to store the entire stream of **n** bits.

Recent advances

- Lots of recent advances with memory-bounded adversaries
    - Oblivious Transfer [Ding01] [Ding04]
    - Randomness Extractors [Vad03]
    - Quantum Adversaries [Ding01]

Time resource bounds

↓

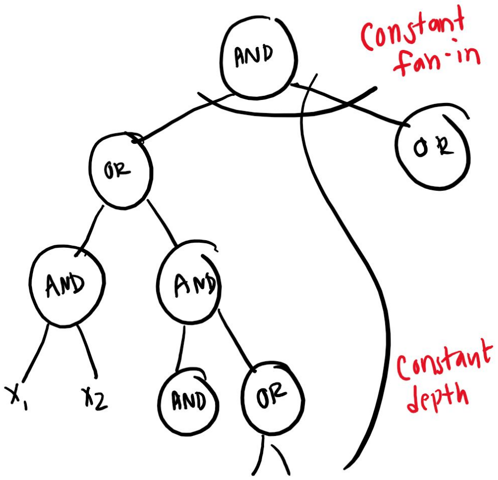
Space resource bounds

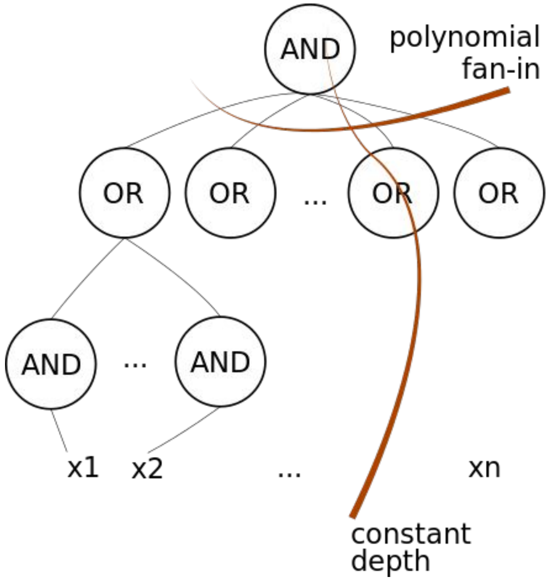
↓

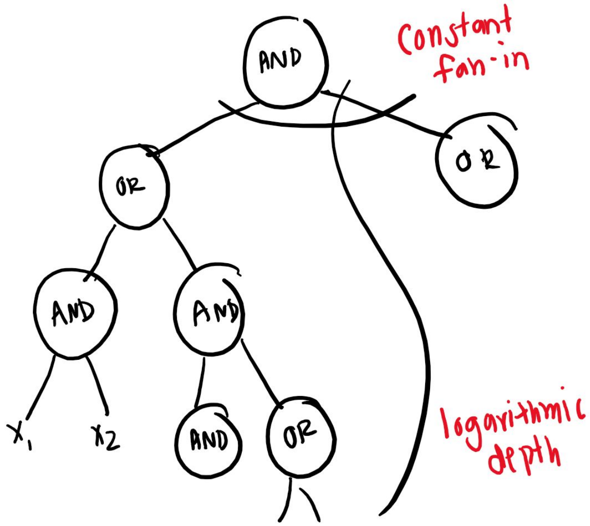
Parallel time resource bounds

Circuit Complexity

$\mathbf{NC^0} \subseteq \mathbf{AC^0} \subseteq \mathbf{NC^1}$







OWFs in NC^0 [Has87]

Definition: (One-Way Function). Let $\mathbf{F} = \{\mathbf{f_n} : \{0, 1\}^{\mathbf{n}} \rightarrow \{0, 1\}^{\mathbf{l(n)}}\}$ be a function family. **F** is a $\mathbf{C_1}$-One-Way Function against $\mathbf{C_2}$ if:

- Computability: for each **n**, $\mathbf{f_n}$ is deterministic and can be computed in $\mathbf{C_1}$.
- One-wayness: for any $\mathbf{G} = \{\mathbf{g_n} : \{0, 1\}^{\mathbf{l(n)}} \rightarrow \{0, 1\}^{\mathbf{n}}\} \in \mathbf{C_2}$, and any **n**, we have:

$$\Pr[\mathbf{f_n}(\mathbf{g_n}(\mathbf{y}) = y \mid y \leftarrow \mathbf{f_n}(\mathbf{x})] < \mathrm{negl}(\mathbf{n})$$

We show a NC^0-One-Way Function against AC^0.

OWFs in NC^0 [Has87]

Theorem: (OWFs against AC^0) Let:

$$\mathbf{f_n}(\mathbf{x}) = (\mathbf{x_1} \oplus \mathbf{x_2}, \mathbf{x_2} \oplus \mathbf{x_3}, \ldots, \mathbf{x_{n-1}} \oplus \mathbf{x_n}, \mathbf{x_n})$$

Then $\mathbf{f_n}(\mathbf{x})$ is an NC^0-One-Way Function against AC^0.

Proof: Computability is satisfied, since $\mathbf{f_n}$ is deterministic.

Note that $\mathbf{f_n}$ is bijective. That is, every **y** has a unique inverse under $\mathbf{f_n}$, which is $(\oplus_{i=1}^{n} y_i, \oplus_{i=2}^{n} y_i, \ldots, y_{n-1} \oplus y_n, y_n)$. In particular, the first bit of the inverse is PARITY(y).

OWFs in NC^0 [Has87]

Proof: (ctd). Consider any AC^0 function family $\mathbf{G} = \{\mathbf{g}_n\}$. Then, we can define another function family $\mathbf{H} = \{\mathbf{h}_n\}$, where $\mathbf{h}_n$ does the following on input **y**:

1. Compute $z \leftarrow \mathbf{g}_n(\mathbf{y})$
2. Check whether $\mathbf{f}_n(\mathbf{z}) = \mathbf{y}$
3. If so, output the first bit of **z**.
4. If not, output a random bit.

Note **H** is also an AC^0 function family, because $\mathbf{f}_n$ and $\mathbf{g}_n$ can be computed in equal depth, as well as checking equality.

OWFs in NC^0 [Has87]

Proof: (ctd). By that observation, we get that for any **n**:

$$\Pr[\mathbf{h_n}(\mathbf{y}) = \text{PARITY}(\mathbf{y})] = \Pr[\mathbf{g_n}(\mathbf{y}) = \mathbf{f_n}^{\mathbf{-1}}(\mathbf{y})] + 0.5 \Pr[\mathbf{g_n}(\mathbf{y}) \neq \mathbf{f_n}^{\mathbf{-1}}(\mathbf{y})]$$

$$= \Pr[\mathbf{g_n}(\mathbf{y}) = \mathbf{f_n}^{\mathbf{-1}}(\mathbf{y})] + 0.5\ (1 - \Pr[\mathbf{g_n}(\mathbf{y}) = \mathbf{f_n}^{\mathbf{-1}}(\mathbf{y})])$$

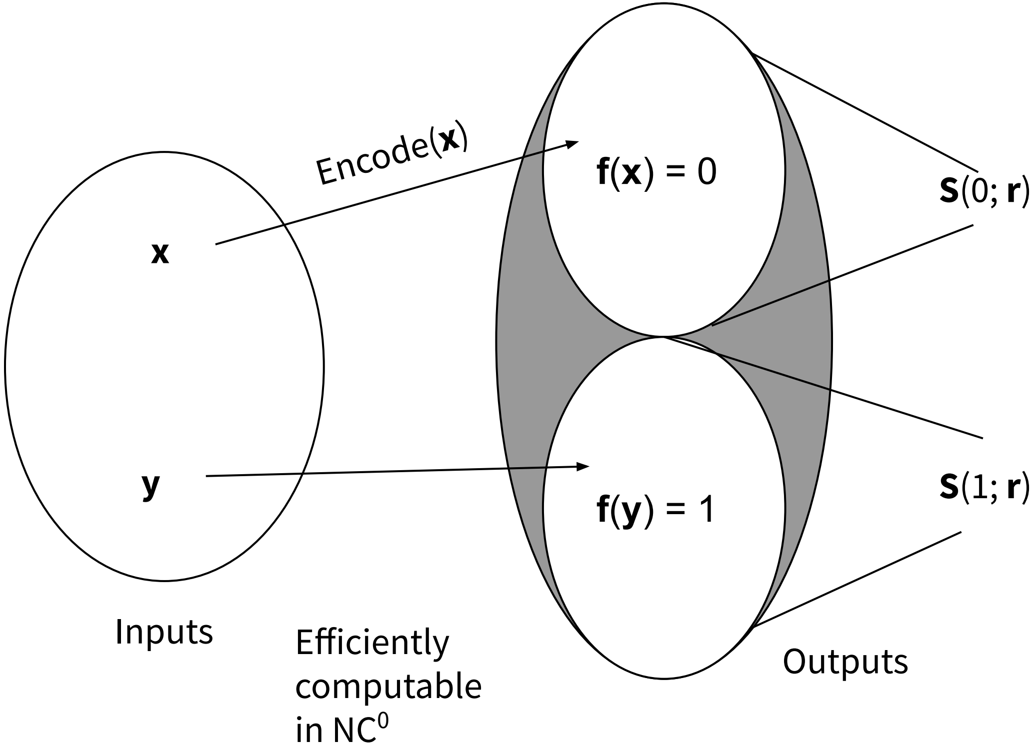
$$= 0.5 + 0.5 \Pr[\mathbf{g_n}(\mathbf{y}) = \mathbf{f_n}^{\mathbf{-1}}(\mathbf{y})]$$

However, a seminal result from Hastad shows that no AC^0 function can compute parity with probability greater than:

$$\Pr[\mathbf{h_n}(\mathbf{y}) = \text{PARITY}(\mathbf{y})] \leq 0.5 + 2^{-O(n\ /\ (\log s(n)))}$$

So, there cannot be an AC^0 family of functions **G** that has a non-negligible advantage in inverted **F**.

Randomized Encodings [IK00, AIK04]



Encode(**x**)

x

y

f(**x**) = 0

f(**y**) = 1

S(0; **r**)

S(1; **r**)

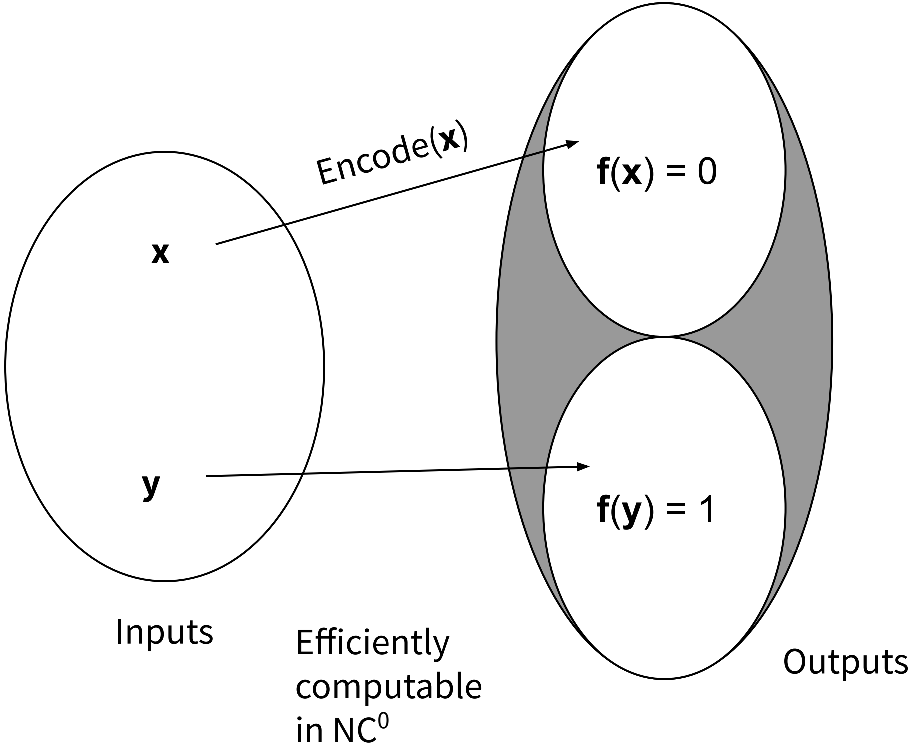
Inputs

Efficiently computable in NC^0

Outputs

Decode(Encode(**x**)) = **f**(**x**)
[expensive operation]

Randomized Encodings [IK00, AIK04]



Surjective Perfect Randomized Encoding:

Given a deterministic function **f** : $\{0, 1\}^n \rightarrow \{0, 1\}^t$, we say that the deterministic function **g** : $\{0, 1\}^n$ x $\{0, 1\}^m \rightarrow \{0, 1\}^s$ is a *perfect randomized encoding* of **f** if the following conditions are satisfied:

1. Input independence
2. Output disjointness
3. Uniformity
4. Balance
5. Stretch preservation
6. Surjectivity

Theorem: [AIK04] Any logspace **f** has NC^0 randomized encodings.

OWFs against NC^1 [BVV15]

Assumption: $\mathbf{L} \neq \mathrm{NC}^1$. Then, there must exist some $\mathbf{f} \in \mathbf{L}$, $\mathbf{f} \notin \mathrm{NC}^1$.

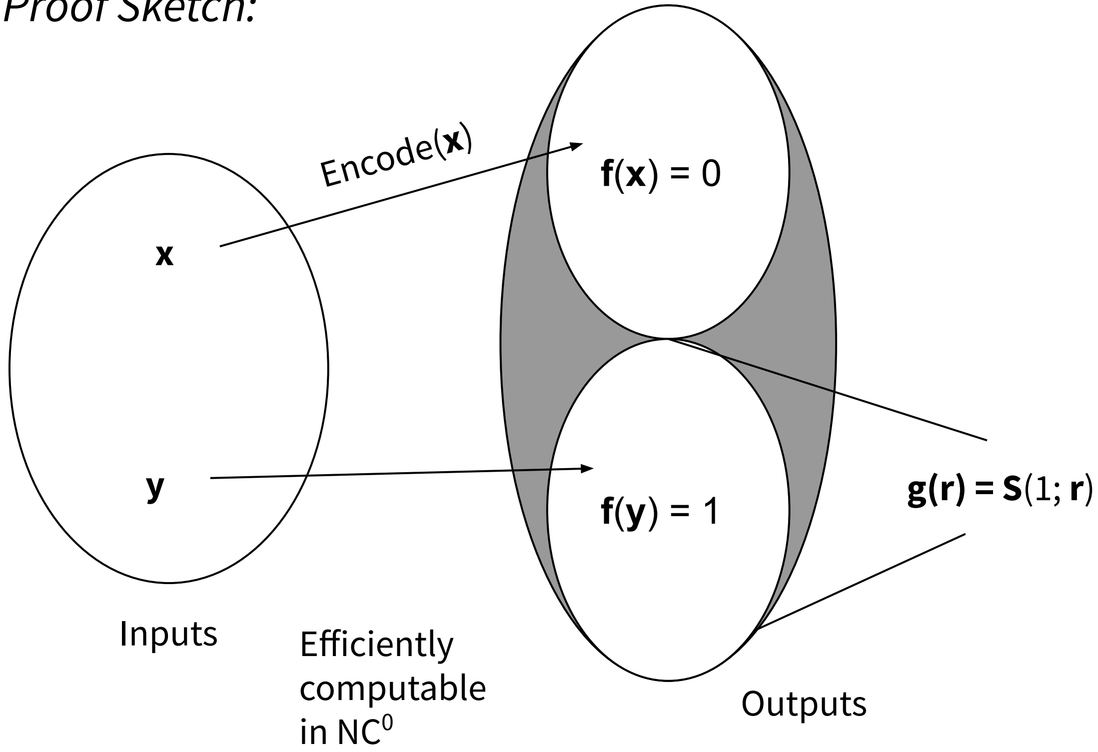
Construction:

$$g(\mathbf{r}) = \mathbf{S}(1; \mathbf{r})$$

is a one-way function secure against NC^1 adversaries.

OWFs against NC¹ [BVV15]

Proof Sketch:



Assume an NC^1 adversary could invert **g**(**r**).

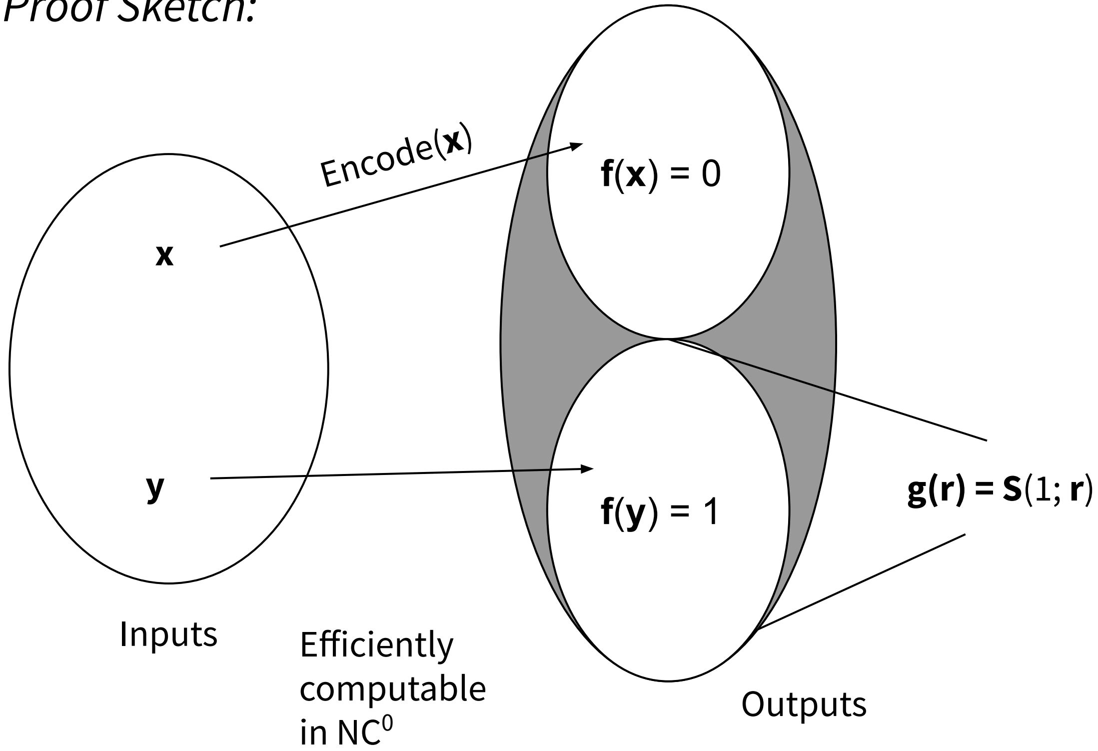
Then, if we fed the adversary Encode(**x**) (where **f**(**x**) = 1), the adversary would tell us what **x** was.

If we fed them Encode(**x**) (where **f**(**x**) = 0), the adversary cannot invert it, since the two distributions are disjoint.

So, we have a decider for the language **f**(**x**) that runs in NC^1. This is a contradiction, since we took **f** to be in **L** but not NC^1.

OWFs against NC¹ [BVV15]

Proof Sketch:



More generally, we have:

$$S(0; U_m) \approx_{NC1} S(1; U_m)$$

Or, the two distributions of the randomized encoding are indistinguishable under NC^1.

$AC^0[2]$-PKE against NC^1 [BVV15]

We open the black box of randomized encodings [IK00]:

$$\mathbf{M}_0^{n} = \begin{pmatrix} 0 & & \cdots & 0 & 0 \\ 1 & 0 & & & 0 \\ 0 & 1 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & 0 & \\ 0 & \cdots & 0 & 1 & 0 \end{pmatrix}, \mathbf{M}_1^{n} = \begin{pmatrix} 0 & & \cdots & 0 & 1 \\ 1 & 0 & & & 0 \\ 0 & 1 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & 0 & \\ 0 & \cdots & 0 & 1 & 0 \end{pmatrix}$$

LSamp(**n**):

1. Output an **n** × **n** upper triangular matrix where all entries in the diagonal are 1 and all other entries in the upper triangular part are chosen at random.

Rsamp(**n**):

1. Sample at random $\mathbf{r} \leftarrow \{0, 1\}^{\mathbf{n}-1}$.
2. Output $\mathbf{M}_0$ with the last column $[\mathbf{r}\ 1]^T$.

$AC^0[2]$-PKE against NC^1 [BVV15]

We open the black box of randomized encodings [IK00]:

$$\mathbf{M}_0^{\mathsf{n}} = \begin{pmatrix} 0 & & \cdots & 0 & 0 \\ 1 & 0 & & & 0 \\ 0 & 1 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & 0 & \\ 0 & \cdots & 0 & 1 & 0 \end{pmatrix}, \mathbf{M}_1^{\mathsf{n}} = \begin{pmatrix} 0 & & \cdots & 0 & 1 \\ 1 & 0 & & & 0 \\ 0 & 1 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & 0 & \\ 0 & \cdots & 0 & 1 & 0 \end{pmatrix}$$

LSamp(**n**):

1. Output an **n** × **n** upper triangular matrix where all entries in the diagonal are 1 and all other entries in the upper triangular part are chosen at random.

Rsamp(**n**):

1. Sample at random $\mathbf{r} \leftarrow \{0, 1\}^{\mathbf{n}-1}$.
2. Output $\mathbf{M}_0$ with the last column $[\mathbf{r}\ 1]^T$.

$AC^0[2]$-PKE against NC^1 [BVV15]

Randomized Encoding scheme:

- Sample $\mathbf{R}_1 \leftarrow \mathrm{LSamp}(\mathbf{n})$ and $\mathbf{R}_2 \leftarrow \mathrm{RSamp}(\mathbf{n})$.
- When $\mathbf{f}(\mathbf{x}) = 0$, sample and return matrix $\mathbf{M} \leftarrow \mathbf{R}_1 \mathbf{M}_0^{\mathbf{n}} \mathbf{R}_2$. This matrix has rank ($\mathbf{n}$-1).
- When $\mathbf{f}(\mathbf{x}) = 1$, sample and return matrix $\mathbf{M} \leftarrow \mathbf{R}_1 \mathbf{M}_1^{\mathbf{n}} \mathbf{R}_2$. This matrix has rank $\mathbf{n}$.

We know from earlier that

$$\mathbf{M}_{\mathbf{f(x) = 1}} \approx_{\mathbf{NC1}} \mathbf{M}_{\mathbf{f(x) = 0}}$$

In other words, the two distributions are indistinguishable to an NC^1 adversary.

$AC^0[2]$-PKE against NC^1 [BVV15]

Theorem: Assume $\oplus L/poly \not\subseteq NC^1$. Then, the following construction is an $AC^0[2]$-Public Key Encryption Scheme against NC^1.

KeyGen$_n$:

1. Sample $\mathbf{R}_1 \leftarrow \mathrm{LSamp}(\mathbf{n})$ and $\mathbf{R}_2 \leftarrow \mathrm{RSamp}(\mathbf{n})$.
2. Let $\mathbf{k} = (\mathbf{r}\ 1)^T$ be the last column of $\mathbf{R}_2$.
3. Compute $\mathbf{M} = \mathbf{R}_1\mathbf{M}_0{}^n\mathbf{R}_2$.
4. Output (pk = $\mathbf{M}$, sk = $\mathbf{k}$).

Enc$_n$*(pk =* $\mathbf{M}$*, b):*

1. Sample $\mathbf{r} \in \{0, 1\}^n$.
2. Let $\mathbf{t}^T = (0 \ldots 0\ 1)$ of length $\mathbf{n}$.
3. Output $\mathbf{c}^T = \mathbf{r}^T\mathbf{M} + b\mathbf{t}^T$.

Dec$_n$(sk = $\mathbf{k}$, $\mathbf{c}$)**:**

1. Output $\langle \mathbf{c}, \mathbf{k} \rangle$.

$AC^0[2]$-PKE against NC^1 [BVV15]

*ZeroSamp(**n**):*

1. Sample $\mathbf{R}_1 \leftarrow$ LSamp(**n**) and $\mathbf{R}_2 \leftarrow$ RSamp(**n**).
2. Output $\mathbf{R}_1\mathbf{M}_0\mathbf{R}_2$.

*OneSamp(**n**):*

1. Sample $\mathbf{R}_1 \leftarrow$ LSamp(**n**) and $\mathbf{R}_2 \leftarrow$ RSamp(**n**).
2. Output $\mathbf{R}_1\mathbf{M}_1\mathbf{R}_2$.

Theorem: [IK00, AIK04] For any boolean function family $\mathbf{F} = \{\mathbf{f}_\mathbf{n}\}$ in $\oplus$L/poly, there exists a polynomial **p** and a perfect randomized encoding $\mathbf{g}_\mathbf{n}$ for $\mathbf{f}_\mathbf{n}$ such that the distribution of $\mathbf{g}_\mathbf{n}$ is identical to ZeroSamp(**p(n)**) when $\mathbf{f}_n(\mathbf{x}) = 0$ and identical to OneSamp(**p(n)**) when $\mathbf{f}_n(\mathbf{x}) = 1$.

Essentially, this theorem implies that if there is some function in $\oplus$L/poly that is hard to compute in the worst-case, then it is hard to distinguish between samples from **S**(0; **r**) and **S**(1; **r**).

$AC^0[2]$-PKE against NC^1 [BVV15]

So, this means that:

$$(\mathrm{pk}, \mathrm{Enc}_n(\mathrm{pk}, 0)) = (\mathbf{M}, \mathbf{r}^T\mathbf{M} \mid \mathbf{M} \leftarrow \mathrm{ZeroSamp}(\mathbf{p(n)}), \mathbf{r}) \approx_{NC1} (\mathbf{M}, \mathbf{r}^T\mathbf{M} \mid \mathbf{M} \leftarrow \mathrm{OneSamp}(\mathbf{p(n)}), r)$$

However, the output of OneSamp is always full rank. So, the distribution of $\mathbf{r}^T\mathbf{M}$ is just uniform over $\{0, 1\}^n$. s a result, we get:

$$(\mathbf{M}, \mathbf{r}^T\mathbf{M} \mid \mathbf{M} \leftarrow \mathrm{OneSamp}(\mathbf{p(n)}), r) = (\mathbf{M}, \mathbf{r}^T\mathbf{M} + \mathbf{t}^\mathbf{T} \mid \mathbf{M} \leftarrow \mathrm{OneSamp}(\mathbf{p(n)}), r)$$

since flipping the last bit does not change the distribution. Using the same idea as above, we get:

$$(\mathbf{M}, \mathbf{r}^T\mathbf{M} \mid \mathbf{M} \leftarrow \mathrm{OneSamp}(\mathbf{p(n)}), r) \approx_{NC1} (\mathbf{M}, \mathbf{r}^T\mathbf{M} + \mathbf{t}^\mathbf{T} \mid \mathbf{M} \leftarrow \mathrm{ZeroSamp}(\mathbf{p(n)}), r) = (\mathrm{pk}, \mathrm{Enc}_n(\mathrm{pk}, 1))$$

Since the distributions are the same regardless of which bit we've encrypted, we have shown semantic security.

Conclusion

Thank you!