Pseudorandomness

A Very Short Introduction

Xinyu Mao

2021/06/06

Motivation:
Get the Power of Randomness
at Lower Cost or even for Free

The Power of Randomness

- Randomized algorithms
  - Quick sort
  - Polynomial identity testing: Given *black-box* access to two n-variable polynomial $f, g \in \mathbb{F}[X_1, \ldots, X_n]$, determine whether $f \equiv g$.
    - Solution: test on a small number of *random* points
- Proving the existence of combinatorial objects with desired property
  - For some property $P: \mathcal{D} \to \{yes, no\}$, if $\Pr_{o \leftarrow \mathcal{D}}[P(o) = yes] > 0$, then there exists an object in domain $\mathcal{D}$ with property P.
  - This is known as *probability method*.
- Cryptography
  - There is no much thing we can do in cryptography without randomness

Main Question: *Can we do all the things with less or even no randomness?*

> Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin.
>
> John von Neumann

- Can all randomized algorithms be derandomized?
  - **BPP = P?**
- Randomness in the physical world: what can we do with a source of biased and correlated bits?
- Seek for explicit construction of combinatorial objects.
- Make cryptographic constructions more efficient.

Pseudorandomness:
A Conception and Paradigm

Idea: To generate objects that 'looks random' efficiently with less or no truly randomness.

Indistinguishable things are identical.
G.W. Leibniz

Pseudorandom Generator (PRG)

$$\text{Truly random} \quad \boxed{n \text{ bits}} \rightarrow \boxed{\text{PRG}} \rightarrow \boxed{\ell(n) \text{ bits}}$$

$\ell(n) > n$

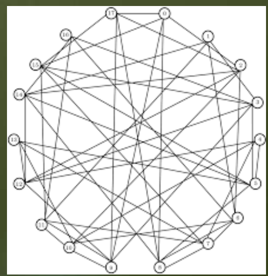
Pseudorandom, or 'look random'

Randomness Extractor

$$\boxed{\text{Weak randomness}} \rightarrow \boxed{\text{Randomness Extractor}} \rightarrow \boxed{\text{Close to perfect randomness}}$$

Expander Graph

Graphs that are both sparse and well-connected.



Pseudorandom Generator (PRG)

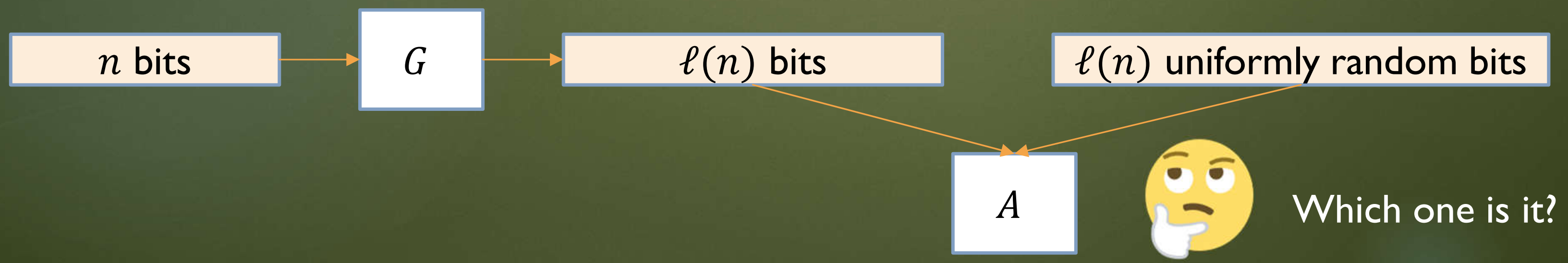
Definition of Pseudorandom Generator (PRG)

Definition. A function $G: \{0,1\}^* \to \{0,1\}^*$ **has stretch $\ell: \mathbb{N} \to \mathbb{N}$** if $|G(x)| = \ell(|x|)$.

Definition. Let $G: \{0 , 1\}^* \to \{0,1\}^*$ be a function with stretch ℓ. We say **G fools algorithm A with error ϵ** if

$$\left|\Pr\left[A\left(G(U_n)\right) = 1\right] - \Pr\left[A\left(U_{\ell(n)}\right) = 1\right]\right| \leq \epsilon(n),$$

where U_n is the uniform distribution on $\{0,1\}^n$.



Definition of PRG

Definition. Let $\mathcal{C}$ be a class of algorithms. $G: \{0,1\}^* \to \{0,1\}^*$ is a $(\mathcal{C}, \epsilon)$-PRG with stretch ℓ if

- G has stretch ℓ;
- G fool every algorithm $A \in \mathcal{C}$ with error ϵ.

- A common setting:
  - $\mathcal{C}$: all probabilistic polynomial time (PPT) algorithms;
  - ϵ is a negligible function, i.e., $\epsilon(n) = o(n^{-c})$ for all $n \in \mathbb{N}$. E.g. $\epsilon(n) = 2^{-n}$.
- Remark: this definition is somewhat *subjective*!

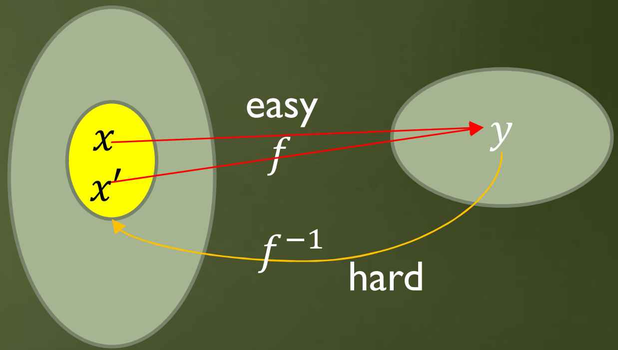
From One-way Function to PRG

Does PRG exist?
The answer is yes under *minimal* cryptographic assumption – the existence of *one-way function*.

Definition. A function $f: \{0,1\}^* \to \{0,1\}^*$ is a $(\mathcal{C}, \epsilon)$-one-way function if it is

- Easy to compute: f is polynomial-time computable and
- Hard to invert: for every algorithm $A \in \mathcal{C}$,

$$\Pr_{\substack{x \leftarrow \{0,1\}^n \\ y := f(x)}} [A(y) \in f^{-1}(y)] \le \epsilon(n).$$



Theorem([HILL99]). PRG exists if and only if OWF exists.

- Remark: Modern cryptography builds upon the assumption that OWF exists.

Lower Bounds for OWF→PRG

- Let $f = \{f_n : \{0,1\}^n \to \{0,1\}^{m(n)}\}$ be an OWF and $G^f : \{0,1\}^{k(n)} \to \{0,1\}^{\ell(n)}$ be a PRG constructed from f.
  - *Black-box construction*
- We care about:
  - $k(n)$: seed length
  - $q(n)$: query complexity, i.e., number of calls to f made by G^f

State of Art ([VZ12]). $k(n) = O(n^4),\ q(n) = O(n^3)$.

Theorem.(Lower bound for regular OWF → PRG [HS12]).
If f is regular, $q(n) = \Omega\left(\frac{n}{\log n}\right)$.

Open problem: Is $k(n) = \Omega(n^4),\ q(n) = \Omega(n^3)$ optimal for arbitrary OWF?

This matches the state of art construction.

Lower bound is always so hard… 😅

Derandomization with PRG

Derandomization by enumeration

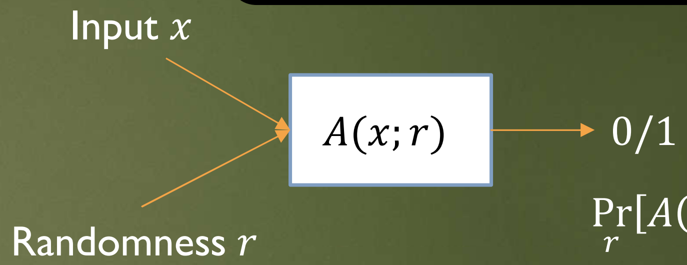
PPT algorithm A that uses $\ell(n)$ bits of randomness

Enumeration

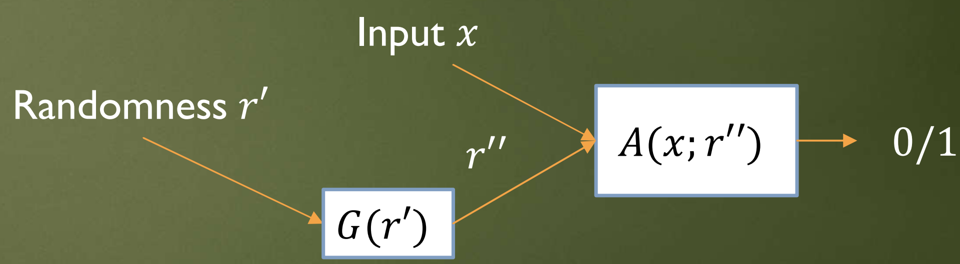
Deterministic A' runs in $2^{\ell(n)} \cdot poly(n)$ time.

- To derandomize BPP, we want
  - G has logarithmic seed length: $k = O(\log \ell)$;
  - G is efficient: computable in $O(2^k)$ time
  - G fools all PPT algorithm with error $\epsilon = 0.1$

Use PRG $G: \{0,1\}^{k(n)} \to \{0,1\}^{\ell(n)}$ to reduce randomness



$\Pr_r[A(x; r) \text{ is correct}] \geq 0.7$



If G fools A with error $\epsilon := 0.1$, then $\Pr_r[A(x; r'') \text{ is correct}] \geq 0.6$.

Does such a PRG exist?

Hardness vs. Randomness: Evidence for $\mathbf{BPP} = \mathbf{P}$

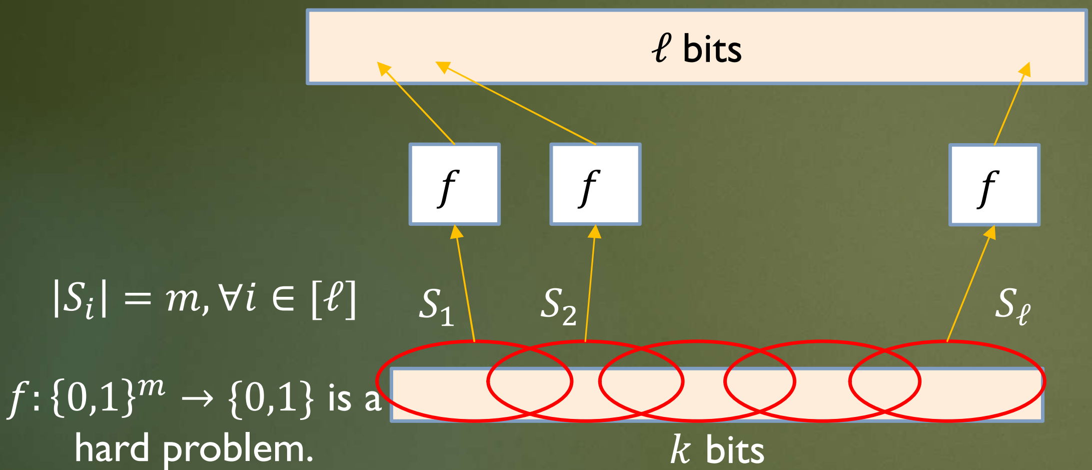


Fig. Main idea of Nisan-Wigderson generator.

Theorem ([NW88]). If there exists a function $f: \{0,1\}^* \to \{0,1\}$ such that:

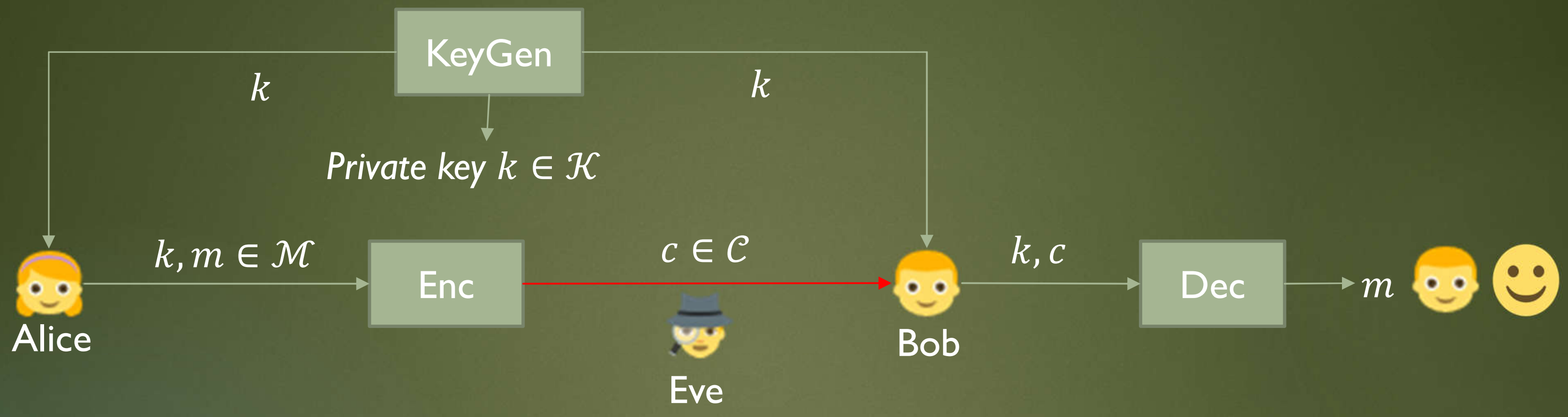
- computable in $2^{O(n)}$ time by Turing Machines, and
- not computable by $2^{0.001n}$ size circuits,

then $\mathbf{BPP} = \mathbf{P}$.

Pseudorandomness in Cryptography: An Example

OWF → PRG → Pseudorandom Function → CPA-Secure Symmetric Encryption

Symmetric Encryption and One-time Pad



One-time pad

- $KeyGen(1^\lambda)$: return $k \leftarrow \{0,1\}^\lambda$.
- $Enc(k, m)$: return $c \coloneqq k \oplus m$.
- $Dec(k, c)$: return $m \coloneqq c \oplus k$.
- $\mathcal{K} = \mathcal{C} = \mathcal{M} = \{0, 1\}^\lambda$

- Drawback of One-time pad: No randomization in encryption → insecure under chosen ciphertext attack (CPA).
  - It leaks whether two ciphertexts encode the same plaintext.

Pseudorandom Function (PRF) → CPA-Secure Symmetric Encryption

- $KeyGen(1^\lambda)$: return $F \leftarrow \mathcal{F}_\lambda$.
- $Enc(F, m)$:
  - choose $r \leftarrow \{0,1\}^\lambda$;
  - return $c := (r, F(r) \oplus m)$.
- $Dec(F, c = (r, \tilde{c}))$:
  - return $m := F(r) \oplus \tilde{c}$.
- $\mathcal{K} = \mathcal{C} = \mathcal{M} = \{0,1\}^\lambda$

$\mathcal{F}_\lambda$:= all functions from $\{0,1\}^\lambda$ to $\{0,1\}^\lambda$.

- Problem: key size is too large!
- Solution: use a pseudorandom function that is:
  - indistinguishable from real random function
  - has short description

Pseudorandom Function (PRF)

Definition. $F: \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ is a PRF if the following holds:

- For every $k \in \{0,1\}^\lambda$, define $F_k(x) := F(k,x)$, then $F_k(x) \in \mathcal{F}_\lambda$.
- For all PPT oracle-aided algorithm A:

$$\left| \Pr_{k \leftarrow \{0,1\}^\lambda}\left[A^{F_k(\cdot)}(1^\lambda) = 1\right] - \Pr_{F \leftarrow \mathcal{F}_\lambda}\left[A^{F(\cdot)}(1^\lambda) = 1\right] \right| \text{ is negligible in } \lambda.$$

- $KeyGen(1^\lambda)$: return $k \leftarrow \mathcal{F}_\lambda$.
- $Enc(k,m)$:
  - choose $r \leftarrow \{0,1\}^\lambda$;
  - return $c := (r, F_k(r) \oplus m)$.
- $Dec(F, c = (r, \tilde{c}))$:
  - return $m := F_k(r) \oplus \tilde{c}$.

Efficiency: The description of F_k is of length $|k| = \lambda$, if we want to encrypt λ bits.

Theorem. PRF exists if and only if PRG exists.

Constrained PRF (CPRF)

PRF: F

Master key: k

Circuit $C: \{0\,,1\}^n \to \{0,1\}$

k, C → **Constrain** → Constrained key k_C

k_C, x → **Constrain Eval** → If $C(x) = 1$: $F(k, x)$; If $C(x) = 0$: whatever

- k_C reveals nothing about $F(k, x)$ when $C(x) = 0$.
- *Constrained hiding*: k_C reveals nothing about C.
- Proposed Boneh and Waters [BW13].

Open problem: Can we construct a adaptively secure Constrained hiding CPRF for polynomial-size circuits (upon standard assumptions)?

Epilogue

Three Perspectives on Randomness

- Information theoretic view: Randomness is lack of information.
  - Consider the probability distribution of the missing data.
  - By definition, one cannot generate more random bits.
- Kolmogorov complexity: Randomness in terms of effective description.
- Computational view: Pseudorandomness -- randomness is something in the eye of the observer.
  - Subjectivity: The ability of the observer matters.

Thanks for Listening ☺

Reference

- [HILL99] Håstad J, Impagliazzo R, Levin LA, Luby M. A pseudorandom generator from any one-way function. SIAM Journal on Computing. 1999;28(4):1364-96.
- [NW88] N. Nisan and A. Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994. Prelim version FOCS '88
- [VZ12] Vadhan S, Zheng CJ. Characterizing pseudoentropy and simplifying pseudorandom generator constructions. InProceedings of the forty-fourth annual ACM symposium on Theory of computing 2012 May 19 (pp. 817-836).
- [HS12] Holenstein T, Sinha M. Constructing a pseudorandom generator requires an almost linear number of calls. In2012 IEEE 53rd Annual Symposium on Foundations of Computer Science 2012 Oct 20 (pp. 698-707). IEEE.
- [BW13] Boneh D, Waters B. Constrained pseudorandom functions and their applications. InInternational conference on the theory and application of cryptology and information security 2013 Dec 1 (pp. 280-300). Springer, Berlin, Heidelberg.