Hardness vs. Randomness

Shubham Sharma*, Rahul Gupta*

*Indian Institute of Technology Kanpur, India

18 April 2019

1

[1] https://www.math.ias.edu/avi/node/780 Noam Nisan and Avi Wigderson

Pseudorandom Generator (PRG)

Pseudorandom Generator (PRG)

- $G : \{0,1\}^r \to \{0,1\}^n$ is a pseudorandom generator such that it appears random (fools) to a class of test functions

Pseudorandom Generator (PRG)

- $G : \{0,1\}^r \to \{0,1\}^n$ is a pseudorandom generator such that it appears random (fools) to a class of test functions
- G ϵ-fools F if $\forall f \in F$ such that $f : \{0,1\}^n \to \{0,1\}$

$$\left| \Pr\left[f(y) = 1 \right] - \Pr\left[f(G(x)) = 1 \right] \right| \le \epsilon$$

where x is choosen uniformly from $\{0,1\}^r$ and y from $\{0,1\}^n$

Pseudorandom Generator (PRG)

- $G : \{0,1\}^r \to \{0,1\}^n$ is a pseudorandom generator such that it appears random (fools) to a class of test functions
- G ϵ-fools F if $\forall f \in F$ such that $f : \{0,1\}^n \to \{0,1\}$

$$\left|\Pr\left[f(y) = 1\right] - \Pr\left[f(G(x)) = 1\right]\right| \leq \epsilon$$

  where x is choosen uniformly from $\{0,1\}^r$ and y from $\{0,1\}^n$
  - Parity-function
  - Polynomial-funtions
  - Boolean-circuits

Pseudorandom Generator (PRG)

- $G : \{0,1\}^r \to \{0,1\}^n$ is a pseudorandom generator such that it appears random (fools) to a class of test functions
- G ϵ-fools F if $\forall f \in F$ such that $f : \{0,1\}^n \to \{0,1\}$

$$\Big|\Pr\big[f(y) = 1\big] - \Pr\big[f(G(x)) = 1\big]\Big| \leq \epsilon$$

  where x is choosen uniformly from $\{0,1\}^r$ and y from $\{0,1\}^n$
  - Parity-function
  - Polynomial-funtions
  - Boolean-circuits
- G is a quick pseudorandom generator if it runs in deterministic time exponential in its input size, $G \in \mathrm{DTIME}(2^{O(r)})$

Pseudorandom Generator (PRG)

- $G : \{0,1\}^r \to \{0,1\}^n$ is a pseudorandom generator such that it appears random (fools) to a class of test functions
- G ϵ-fools F if $\forall f \in F$ such that $f : \{0,1\}^n \to \{0,1\}$

$$\Big| \Pr\big[f(y) = 1\big] - \Pr\big[f(G(x)) = 1\big] \Big| \leq \epsilon$$

  where x is choosen uniformly from $\{0,1\}^r$ and y from $\{0,1\}^n$
  - Parity-function
  - Polynomial-funtions
  - Boolean-circuits
- G is a quick pseudorandom generator if it runs in deterministic time exponential in its input size, $G \in \mathrm{DTIME}(2^{O(r)})$
  - quality of the output – n, ϵ
  - price – r

Motivation

Motivation

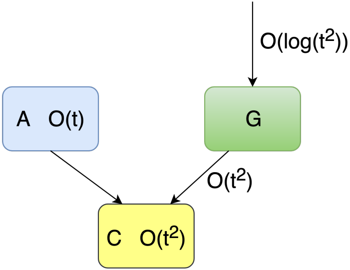
- If there exists a quick pseudorandom generator $G : \log(n) \to n$ then for any time constructible bound $t = t(n)$:

$$\mathrm{RTIME}(t) \subset \mathrm{DTIME}(2^{O(\log(t^2))})$$

Motivation

- If there exists a quick pseudorandom generator $G : \log(n) \rightarrow n$ then for any time constructible bound $t = t(n)$:

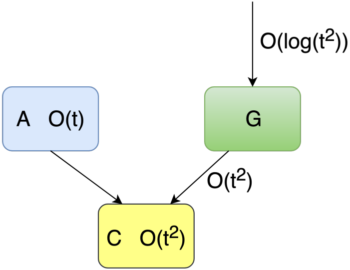
$$\mathrm{RTIME}(t) \subset \mathrm{DTIME}(2^{O(\log(t^2))})$$



Motivation

- If there exists a quick pseudorandom generator $G : \log(n) \to n$ then for any time constructible bound $t = t(n)$:

$$\mathrm{RTIME}(t) \subset \mathrm{DTIME}(2^{O(\log(t^2))})$$



- Simulate A deterministically, by trying all the possible random seeds and taking a majority vote

Applications

Applications

- $\mathsf{BPP} \subset \cap_{\epsilon>0}\mathsf{DTIME}(2^{n^\epsilon})$
- $\mathsf{BPP} \subset \mathsf{DTIME}(2^{(\log n)^c})$
- $\mathsf{BPP} = P$
- $\mathsf{RNC} \subset \cap_{\epsilon>0}\mathsf{DSPACE}(n^\epsilon)$
- $\mathsf{RNC} \subset \mathsf{DSPACE}(polylog)$

$\vdots$

Prior Work

- Shamir: RSA function
- Blum and Micali: Intractability of Discrete Logarithm function
- Yao: One-way permutation

Prior Work

- Shamir: RSA function
- Blum and Micali: Intractability of Discrete Logarithm function
- Yao: One-way permutation
  - Require a strong unproven assumption. (the existence of a one-way function, an assumption which is even stronger than $P \neq NP$)
  - sequential

Contribution of Paper

- Construction of pseudorandom generator based on much weaker assumptions (EXPTIME cannot be approximated by circuits of small size)

Contribution of Paper

- Construction of pseudorandom generator based on much weaker assumptions (EXPTIME cannot be approximated by circuits of small size)
- Equivalence between the problem of proving lower bounds for the size of circuits approximating functions in EXPTIME, and the problem of constructing pseudorandom generators which run "sufficiently fast"

Contribution of Paper

- Construction of pseudorandom generator based on much weaker assumptions (EXPTIME cannot be approximated by circuits of small size)
- Equivalence between the problem of proving lower bounds for the size of circuits approximating functions in EXPTIME, and the problem of constructing pseudorandom generators which run "sufficiently fast"
- Randomized Parallel Computation

Contribution of Paper

- Construction of pseudorandom generator based on much weaker assumptions (EXPTIME cannot be approximated by circuits of small size)
- Equivalence between the problem of proving lower bounds for the size of circuits approximating functions in EXPTIME, and the problem of constructing pseudorandom generators which run "sufficiently fast"
- Randomized Parallel Computation

$\vdots$

Hardness

- Generator is constructed on the assumption of existence of hard functions

Hardness

- Generator is constructed on the assumption of existence of hard functions
- Hard: function can not be computed/approximated by small-circuits

Hardness

- Generator is constructed on the assumption of existence of hard functions
- Hard: function can not be computed/approximated by small-circuits
- Let $f : \{0,1\}^n \to \{0,1\}$ be a boolean function. We say that f is (ϵ, S)-hard if for any circuit C of size S,

$$\left|\Pr\left[C(x) = f(x)\right] - \frac{1}{2}\right| < \epsilon$$

Hardness

- Generator is constructed on the assumption of existence of hard functions
- Hard: function can not be computed/approximated by small-circuits
- Let $f : \{0,1\}^n \to \{0,1\}$ be a boolean function. We say that f is (ϵ, S)-hard if for any circuit C of size S,

$$\left|\Pr\left[C(x) = f(x)\right] - \frac{1}{2}\right| < \epsilon$$

- Yao proved that even harder function can be constructed by xor-ing multiple copies of f

Hardness

- Generator is constructed on the assumption of existence of hard functions
- Hard: function can not be computed/approximated by small-circuits
- Let $f : \{0,1\}^n \rightarrow \{0,1\}$ be a boolean function. We say that f is (ϵ, S)-hard if for any circuit C of size S,

$$\left| \Pr\left[C(x) = f(x)\right] - \frac{1}{2} \right| < \epsilon$$

- Yao proved that even harder function can be constructed by xor-ing multiple copies of f
- Let $f_1, \cdots, f_k$ all be (ϵ, S)-hard. Then for any $\delta > 0$, the function $f(x_1, \cdot, x_k)$ defined by

$$f(x_1, \cdot, x_k) = \sum_{i=i}^{k} f_i(x_i)(mod\ 2)$$

is $(\epsilon^k + \delta, \delta^2(1-\epsilon)^2 S)$-hard

Hardness

- Let $f = \{0,1\}^* \to \{0,1\}$ be a boolean function. We say that f cannot be approximated by circuits of size $s(n)$ if for some constant k, all large enough n, and all circuits C_n of size $s(n)$:

$$\Pr\left[C_n(x) \neq f(x)\right] > n^{-k}$$

  - small circuits attempting to compute f have a non-negligible fraction of error

- Let $f : \{0,1\}^* \to \{0,1\}$ be a boolean function, and let f_m be the restriction of f to strings of length m. The Hardness of f at m, $H_{f(m)}$ is defined to be the maximum integer h_m such that fm is $(1/h_m, h_m)$-hard

Hardness

- Let $s(m)$ be any function such that $m \leq s(m) \leq 2^m$; if there exists a function f in EXPTIME that cannot be approximated by circuits of size $s(m)$, then for some $c > 0$ there exists a function f' in EXPTIME that has hardness $H_{f'(m)} \geq s(m^c)$.

Construction

Construction

- A collection of sets $\{S_1, \cdots, S_n\}$, where $S_i \subset \{1, \cdots, l\}$ is called a (k, m)-design if:
  - $\forall i : \ |S_i| = m$
  - $\forall i \neq j : \ |S_i \cap S_j| \leq k$

Construction

- A collection of sets $\{S_1, \cdots, S_n\}$, where $S_i \subset \{1, \cdots, l\}$ is called a (k, m)-design if:
  - $\forall i: \ |S_i| = m$
  - $\forall i \neq j: \ |S_i \cap S_j| \leq k$
- A $n \times l$ $(0-1)$ matrix is called a (k, m)-design if the collection of its n rows, interpreted as subsets of $\{1 \cdots l\}$, is a (k, m)-design.

Construction

- A collection of sets $\{S_1, \cdots, S_n\}$, where $S_i \subset \{1, \cdots, l\}$ is called a (k, m)-design if:
  - $\forall i : \ |S_i| = m$
  - $\forall i \neq j : \ |S_i \cap S_j| \leq k$
- A $n \times l$ $(0-1)$ matrix is called a (k, m)-design if the collection of its n rows, interpreted as subsets of $\{1 \cdots l\}$, is a (k, m)-design.
- Construction:

Construction

- A collection of sets $\{S_1, \cdots, S_n\}$, where $S_i \subset \{1, \cdots, l\}$ is called a (k, m)-design if:
  - $\forall i : \ |S_i| = m$
  - $\forall i \neq j : \ |S_i \cap S_j| \leq k$
- A $n \times l$ $(0-1)$ matrix is called a (k, m)-design if the collection of its n rows, interpreted as subsets of $\{1 \cdots l\}$, is a (k, m)-design.
- Construction:
  - f a boolean function, $f : \{0,1\}^m \to \{0,1\}$, such that $H_{f(m)} \geq n^2$

Construction

- A collection of sets $\{S_1, \cdots, S_n\}$, where $S_i \subset \{1, \cdots, l\}$ is called a (k, m)-design if:
  - $\forall i : \; |S_i| = m$
  - $\forall i \neq j : \; |S_i \cap S_j| \leq k$
- A $n \times l$ $(0-1)$ matrix is called a (k, m)-design if the collection of its n rows, interpreted as subsets of $\{1 \cdots l\}$, is a (k, m)-design.
- Construction:
  - f a boolean function, $f : \{0,1\}^m \to \{0,1\}$, such that $H_{f(m)} \geq n^2$
  - A a boolean $n \times l$ matrix which is a $(logn, m)$ design

Construction

- A collection of sets $\{S_1, \cdots, S_n\}$, where $S_i \subset \{1, \cdots, l\}$ is called a (k, m)-design if:
  - $\forall i: \ |S_i| = m$
  - $\forall i \neq j: \ |S_i \cap S_j| \leq k$
- A $n \times l$ $(0-1)$ matrix is called a (k, m)-design if the collection of its n rows, interpreted as subsets of $\{1 \cdots l\}$, is a (k, m)-design.
- Construction:
  - f a boolean function, $f: \{0, 1\}^m \rightarrow \{0, 1\}$, such that $H_{f(m)} \geq n^2$
  - A a boolean $n \times l$ matrix which is a $(logn, m)$ design
  - $G: l \rightarrow n$ given by $G(x) = f_{A(x)}$ is a pseudorandom generator

Construction

- A collection of sets $\{S_1, \cdots, S_n\}$, where $S_i \subset \{1, \cdots, l\}$ is called a (k, m)-design if:
  - $\forall i : \ |S_i| = m$
  - $\forall i \neq j : \ |S_i \cap S_j| \leq k$
- A $n \times l$ $(0-1)$ matrix is called a (k, m)-design if the collection of its n rows, interpreted as subsets of $\{1 \cdots l\}$, is a (k, m)-design.
- Construction:
  - f a boolean function, $f : \{0, 1\}^m \rightarrow \{0, 1\}$, such that $H_{f(m)} \geq n^2$
  - A a boolean $n \times l$ matrix which is a $(logn, m)$ design
  - $G : l \rightarrow n$ given by $G(x) = f_{A(x)}$ is a pseudorandom generator
  - $f_{A(x)}$ the n bit vector of bits computed by applying the function f to the subsets of the $x's$ denoted by the n different rows of A

Proof (Main Idea)

- Assume that G is not pseudorandom generator

$$\left|\Pr\left[C(y) = 1\right] - \Pr\left[C(G(x)) = 1\right]\right| > \frac{1}{n}$$

Proof (Main Idea)

- Assume that G is not pseudorandom generator

$$\Big|\Pr\big[C(y) = 1\big] - \Pr\big[C(G(x)) = 1\big]\Big| > \frac{1}{n}$$

- This implies that one of the bits of $f_{A(x)}$ can be predicted from the previous ones

Proof (Main Idea)

- Assume that G is not pseudorandom generator

$$\left|\Pr\left[C(y)=1\right]-\Pr\left[C(G(x))=1\right]\right|>\frac{1}{n}$$

- This implies that one of the bits of $f_{A(x)}$ can be predicted from the previous ones
- Construct the circuit that predicts the next bit of $f_{A(x)}$

Proof (Main Idea)

- Assume that G is not pseudorandom generator

$$\Big|\Pr\big[C(y)=1\big]-\Pr\big[C(G(x))=1\big]\Big| > \frac{1}{n}$$

- This implies that one of the bits of $f_{A(x)}$ can be predicted from the previous ones
- Construct the circuit that predicts the next bit of $f_{A(x)}$
- This contradicts that $H_{f(m)} \geq n^2$

What did we achieved?

- For every function $s, l \leq s(l) \leq 2^l$ the following are equivalent:

What did we achieved?

- For every function $s, l \leq s(l) \leq 2^l$ the following are equivalent:
  - For some $c > 0$ some function in EXPTIME cannot be approximated by circuits of size $s(l^c)$

What did we achieved?

- For every function $s, l \leq s(l) \leq 2^l$ the following are equivalent:
  - For some $c > 0$ some function in EXPTIME cannot be approximated by circuits of size $s(l^c)$
  - For some $c > 0$ there exists a function in EXPTIME with hardness $s(l^c)$

What did we achieved?

- For every function $s, l \leq s(l) \leq 2^l$ the following are equivalent:
  - For some $c > 0$ some function in EXPTIME cannot be approximated by circuits of size $s(l^c)$
  - For some $c > 0$ there exists a function in EXPTIME with hardness $s(l^c)$
  - For some $c > 0$ there exists a quick pseudorandom generator $G : l \rightarrow s(l^c)$

Questions?