PRGs For Two-party Communication Protocols

TSS

19/04/23

Outline of The Talk.

- Majority of the time: motivating the theory of unconditional pseudo-randomness
- Something about communication protocols
- Something about expander graphs
- Main Result (sketch)

* All the credit to recent survey by Hatami-Hoza (ECCC '23)

Perspective: Randomness as a Resource

Perspective: Randomness as a Resource

→ What to do when limited supply of random bits?

Perspective: Randomness as a Resource

→ What to do when limited supply of random bits?

→ A PRG uses small amt. of 'true' randomness (aka "seed") - generates long seq. appearing random.

Perspective: Randomness as a Resource

- What to do when limited supply of random bits?
- A PRG uses small amt. of 'true' randomness (aka 'seed') - generates long seq. appearing random.
- 'Observer' - to be modelled as a function.

Tooling.

Yosling. (Def n.) $f : \{0,1\}^n \longrightarrow \mathbb{R}$, X: distribution

Fooling (Defn.) $f : \{0,1\}^n \longrightarrow \mathbb{R}$, X: distribution

X fools f with error ε if

$$\left| \mathbb{E}[f(X)] - \mathbb{E}[f(U_n)] \right| \leq \varepsilon$$

Fooling (def n.) $f : \{0,1\}^n \longrightarrow \mathbb{R}$, X: distribution

X fools f with error ε if

$$\left| \mathbb{E}[f(X)] - \mathbb{E}[f(U_n)] \right| \leq \varepsilon$$

U_n ↳ unif. dist.

Fooling (Defn.) $f: \{0,1\}^n \longrightarrow \mathbb{R}$, X: distribution

X fools f with error ε if

$$\left| \mathbb{E}[f(X)] - \mathbb{E}[f(U_n)] \right| \leq \varepsilon$$

(U_n ↳ unif. dist.)

> I.e., though X may not be unif., X & U_n are still indistinguishable from f's perspective.

Fooling (Defn.) $f: \{0,1\}^n \longrightarrow \mathbb{R}$, X: distribution

X fools f with error ε if

$$\left| \mathbb{E}[f(X)] - \mathbb{E}[f(U_n)] \right| \leq \varepsilon$$

(U_n ↳ unif. dist.)

> I.e., though X may not be unif., X & U_n are still indistinguishable from f's perspective.

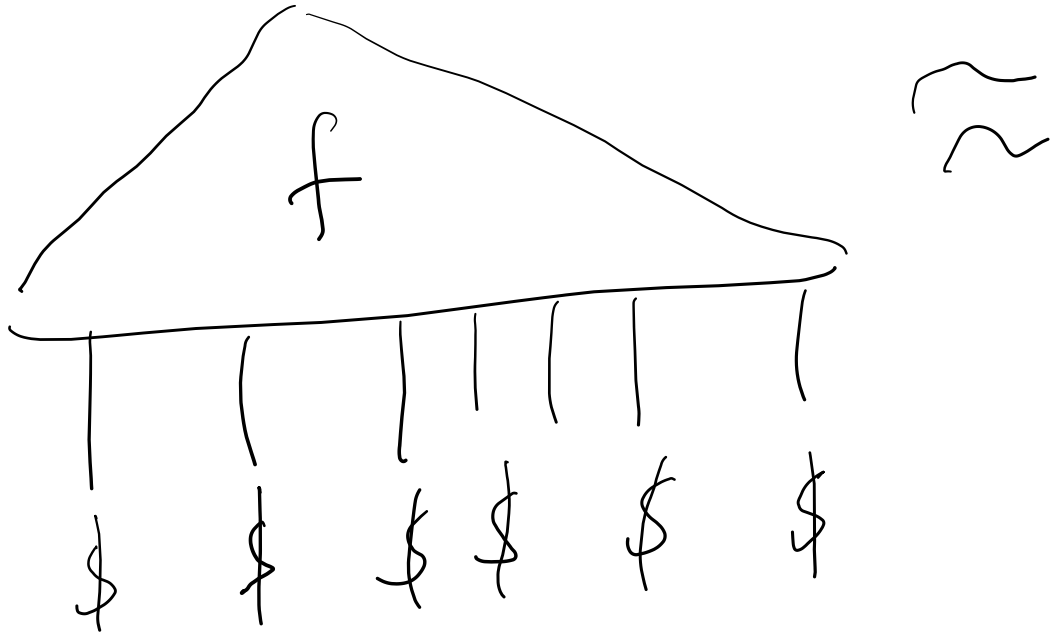
> PRG's job: generate such an X with few truly random bits.

PRG Def^n: Let $f: \{0,1\}^n \to \mathbb{R}$, $G: \{0,1\}^s \to \{0,1\}^n$ and $\varepsilon > 0$.

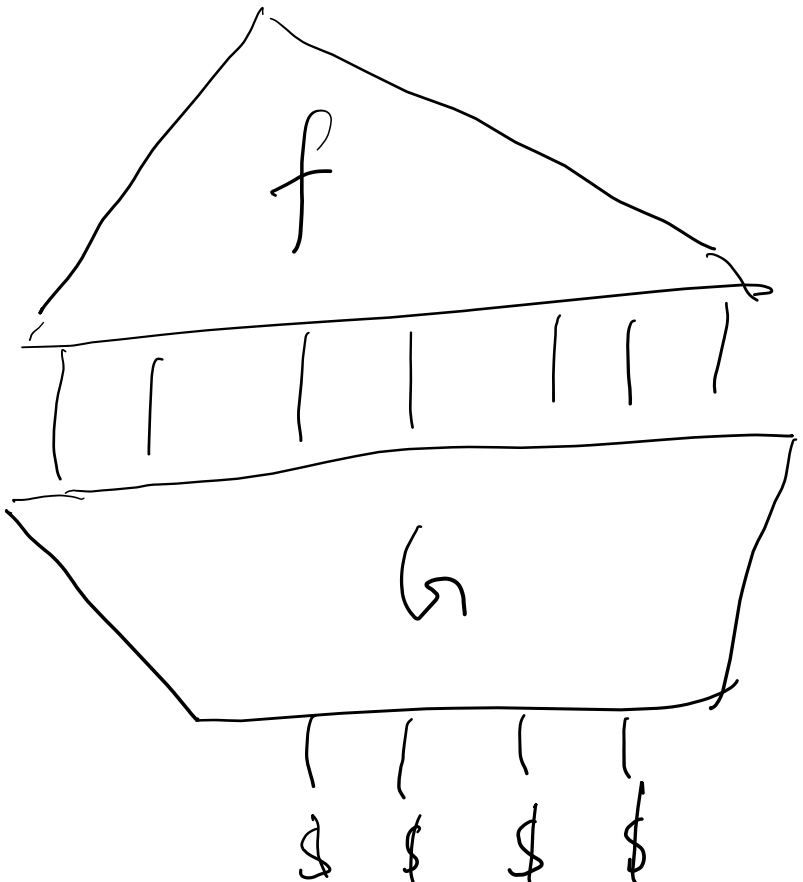
PRG Def^n: Let $f: \{0,1\}^n \to \mathbb{R}$, $G: \{0,1\}^s \to \{0,1\}^n$ and $\varepsilon > 0$. Say G is an ε-PRG for f if $G(U_s)$ fools f with error ε.

PRG Def^n: Let $f: \{0,1\}^n \to \mathbb{R}$, $G: \{0,1\}^s \to \{0,1\}^n$ and $\varepsilon > 0$. Say G is an ε-PRG for f if $G(U_s)$ fools f with error ε. Also say G fools f with error ε.

PRG Def^n: Let $f: \{0,1\}^n \to R$, $G: \{0,1\}^s \to \{0,1\}^n$ and $\varepsilon > 0$. Say G is an ε-PRG for f if $G(U_s)$ fools f with error ε. Also say G fools f with error ε.



$\approx$



Whom shall we fold?

Whom shall we fool?

Fact: for any non-trivial PKG, ∃ fn. which isn't fooled.

Whom shall we fool?

Fact: for any non-trivial PRG, $\exists$ fn. which isn't fooled.

Clm: Let $G: \{0,1\}^s \to \{0,1\}^n$, $s < n$. $\exists f: \{0,1\}^n \to \{0,1\}$ s.t. G does not 0.49-fool f.

Whom shall we fool?

Fact: for any non-trivial PRG, $\exists$ fn. which isn't fooled.

Clm: Let $G: \{0,1\}^s \to \{0,1\}^n$, $s < n$. $\exists f: \{0,1\}^n \to \{0,1\}$ s.t. G does not 0.49-fool f.

Pf: f: ind. fn. of img. of G. Then $\mathbb{E}[f(G(U_s))] = 1$, but

$\mathbb{E}[f] \leq 1/2$ as $s < n$.

$\left(|\{0,1\}^s| \text{ as a fraction of } |\{0,1\}^n| \right)$

(i.e. $f = \chi_{G(U_s)}$)

Best hope: Want to generate bits that fool large classes of observers but not all of them.

Best hope: Want to generate bits that fool large classes of observers but not all of them.

Def^n: $\mathcal{F}$: class of fns. $f : \{0,1\}^n \to \mathbb{R}$. G is an ε-PRG for $\mathcal{F}$ if G ε-fools every $f \in \mathcal{F}$.

Best hope: Want to generate bits that fool large class of observers but not all of them.

Def^n: $\mathcal{Y}$: class of fns. $f: \{0,1\}^n \to \mathbb{R}$. G is an ε-PRG for $\mathcal{Y}$ if G ε-fools every $f \in \mathcal{Y}$.

So, back to: which observers shall we fool?

Best hope: Want to generate bits that fool large classes of observers but not all of them.

Def^n: $\mathcal{F}$: class of fns. $f: \{0,1\}^n \to \mathbb{R}$. G is an ε-PRG for $\mathcal{F}$ if G ε-fools every $f \in \mathcal{F}$.

So, back to: which observers shall we fool? 3 paradigms:

1. Everyday non-adversarial applications
2. All efficient observers
3. 'Restricted' models of computation

1. PRGs for everyday non-adversarial apps.

1. PRGs for everyday non-adversarial apps.

→ Programmers: just use random()!

1. PRGs for everyday non-adversarial apps.

- Programmers: just use `random()`!
  - ↪ very sophisticated under the hood!

1. PRGs for everyday non-adversarial apps.

> Programmers: just use random()!

> When referring to 'PRNGs', they typically refer to the entire randomness system as a whole, including what goes into producing an initial seed.

1. PRGs for everyday non-adversarial apps.

> Programmers: just use random()!
> When referring to 'PRNGs', they typically refer to the entire randomness system as a whole, including what goes into producing an initial seed.
> In theory CS, we sidestep this issue completely and focus on stretching out to a long pseudorandom string

1. PRGs for everyday non-adversarial apps.

> Programmers: just use random()!

> When referring to 'PRNGs', they typically refer to the entire randomness system as a whole, including what goes into producing an initial seed.

> In theory CS, we sidestep this issue completely and focus on stretching out to a long pseudorandom string

↳ So in practice, this is only one of the multiple components of a practical random system.

1. PRGs for everyday non-adversarial apps.

> Programmers: just use random()!

> When referring to 'PRNGs', they typically refer to the entire randomness system as a whole, including what goes into producing an initial seed.

> In theory CS, we sidestep this issue completely and focus on stretching out to a long pseudorandom string

1. PRGs for everyday non-adversarial apps. (contd)

Example: Java's math.random() (Linear congruential gen.)

1. PRGs for everyday non-adversarial apps. (contd)

Example: Java's math.random() (Linear congruential gen.)

$X_0 \in \{0, 1, \ldots, M-1\}$ u.a.r. Output $(X_1, X_2, X_3, \ldots)$

where $X_{i+1} = aX_i + b \bmod M$. for some M, a, b.

1. PRGs for everyday non-adversarial apps. (contd)

Example: Java's math.random() (Linear congruential gen.)

$X_0 \in \{0, 1, \ldots, M-1\}$ u.a.r. Output $(X_1, X_2, X_3, \ldots)$

where $X_{i+1} = aX_i + b \bmod M$. for some M, a, b.

> Python's random.random() uses 'Mersenne twister'

1. PRGs for everyday non-adversarial apps. (contd)

Example: Java's math.random() (Linear congruential gen.)

$X_0 \in \{0, 1, \ldots, M-1\}$ u.a.r. Output $(X_1, X_2, X_3, \ldots)$

where $X_{i+1} = aX_i + b \bmod M$. for some M, a, b.

> Python's random.random() uses 'Mersenne twister'

> Web browsers: 'xorshift+ family' in Javascript

1. PRGs for everyday non-adversarial apps. (contd)

Example: Java's math.random() (Linear congruential gen.)

$X_0 \in \{0, 1, \ldots, M-1\}$ u.a.r. Output $(X_1, X_2, X_3, \ldots)$

where $X_{i+1} = aX_i + b \bmod M$. for some M, a, b.

> Python's random.random() uses 'Mersenne twister'

> Web browsers: 'xorshift+ family' in Javascript

1. PRGs for everyday non-adversarial apps (contd)

Example: Java's math.random() (Linear congruential gen.)

$X_0 \in \{0, 1, \ldots, M-1\}$ u.a.r. Output $(X_1, X_2, X_3, \ldots)$

where $X_{i+1} = aX_i + b \bmod M$. for some M, a, b.

> Python's random.random() uses 'Mersenne twister'

> Web browsers: 'xorshift+ family' in Javascript

> But these are all heuristics with no firm mathematical guarantee. Unsatisfactory for us theoreticians!

(Contd.).

> Credit to designers for making them work.

> They wisely do not claim that they work in adversarial scenarios — unsuitable for crypto.

(Contd.)

> Credit to designers for making them work.

> They wisely do not claim that they work in adversarial scenarios — unsuitable for crypto.

> In fact, sometimes programs do come up which 'accidentally' distinguish random numbers from PRNG. numbers.

(Contd.)

> Credit to designers for making them work.

> They wisely do not claim that they work in adversarial scenarios — unsuitable for crypto.

> In fact, sometimes programs do come up which 'accidentally' distinguish random numbers from PRNG. numbers.

> Us in theory: success of practical PRGs largely a mystery.

2. PRGs for all efficient observers.

2. PRGs for all efficient observers.

> Great ideal Given such a PRG and seed, can execute any rand. algo worth executing!

> Such a PRG could also work for crypto - we can assume adversaries only have so much computational pow.

2. PRGs for all efficient observers.

> Great ideal Given such a PRG and seed, can execute any rand. algo worth executing!

> Such a PRG could also work for crypto - we can assume adversaries only have so much computational pow.

> Example (BBSB6). Pick $X_0 \in \{0, 1, \ldots, M-1\}$ rar.

Output: $(X_1 \bmod 2, X_2 \bmod 2, \ldots)$ where

$$X_{i+1} = X_i^2 \bmod M.$$

2. PRGs for all efficient observers.

> Great ideal Given such a PRG and seed, can execute any rand. algo worth executing!

> Such a PRG could also work for crypto - we can assume adversaries only have so much computational pow.

> Example (BBS86). Pick $X_0 \in \{0, 1, \ldots, M-1\}$ rand.

Output: $(X_1 \bmod 2, X_2 \bmod 2, \ldots)$ where

$$X_{i+1} = X_i^2 \bmod M.$$

> Belief: Fools all poly-time algorithms!

Situation (Currently) Unsatisfactory as well

Situation (Currently) Unsatisfactory as well

> while this is a well-defd. goal, nobody knows how to prove that an efficiently computable PRG has this ppty.

Situation (Currently) Unsatisfactory as well

> while this is a well-defd. goal, nobody knows how to prove that an efficiently computable PRG has this ppty.

> Conditional results exist: for e.g., BBS has it assuming 'quadratic residuosity' is hard. Other examples - - -

Situation (Currently) Unsatisfactory as well

> while this is a well-defd. goal, nobody knows how to prove that an efficiently computable PRG has this ppty.
> Conditional results exist: for e.g., BBS has it assuming 'quadratic residuosity' is hard. Other examples - - -
> Genuine room for doubt whether known PRGs work, and even if they do, don't know why they work.
> Conditional proofs are partial explanations at best.

3. PRGs for restricted models

3. PRGs for restricted models

> Here, we identify an interesting and well-defd. rest. model. Then, design PRGs unconditionally and optimize the seed length.

3. PRGs for restricted models

> Here, we identify an interesting and well-defd. rest. model. Then, design PRGs unconditionally and optimize the seed length.

> Toy example: Design $G: \{0,1\}^2 \to \{0,1\}^3$ that fools any observer that can only look at two of the three i/p bits.

3. PRGs for restricted models

> Here, we identify an interesting and well-defd. rest. model. Then, design PRGs unconditionally and optimize the seed length.

> Toy example: Design $G: \{0,1\}^2 \to \{0,1\}^3$ that fools any observer that can only look at two of the three i/p bits. Not trivial - don't know which two bits!

3. PRGs for restricted models

> Here, we identify an interesting and well-defd. rest. model. Then, design PRGs unconditionally and optimize the seed length.

> Toy example: Design $G: \{0,1\}^2 \to \{0,1\}^3$ that fools any observer that can only look at two of the three i/p bits. Not trivial - don't know which two bits!

$G(u_1, u_2) = (u_1, u_2, u_1 \oplus u_2)$ works

3. PRGs for restricted models

> Here, we identify an interesting and well-defd. rest. model. Then, design PRGs unconditionally and optimize the seed length.

> Toy example: Design $G: \{0,1\}^2 \to \{0,1\}^3$ that fools any observer that can only look at two of the three i/p bits. Not trivial - don't know which two bits!

$G(u_1, u_2) = (u_1, u_2, u_1 \oplus u_2)$ works

- when u_1, u_2 are uar, the 3 o/p bits are correlated but any 2 of the bits are random, indep.

> Unconditional PRGs are know for much richer models.

>

> Unconditional PRGs are know for much richer models.

> Want: o/p of PRG to appear random to a "sufficiently" efficient observer.

> Unconditional PRGs are know for much richer models.

> Want: o/p of PRG to appear random to a "sufficiently" efficient observer.

> Two landmark results: for AC^0 & read-once branching programs (ROBPs).

> Unconditional PRGs are know for much richer models.
> Want: o/p of PRG to appear random to a "sufficiently" efficient observer.
> Two landmark results: for AC^0 & read-once branching programs (ROBPs).
> Unconditional PRGs are invaluable within theory of computing (in particular in attacking: L vs RL

Generic Probabilistic Existence Proof

Claim (Non-explicit PRGs)

Generic Probabilistic Existence Proof

Claim (Non-explicit PRGs) $\mathcal{Y}$: class of fns. $f: \{0,1\}^n \to \{0,1\}$.
For every $\varepsilon > 0$, $\exists$ ε-PRG for $\mathcal{Y}$ with seed length

$$\log\log|\mathcal{Y}| + 2\log(1/\varepsilon) + O(1).$$

Generic Probabilistic Existence Proof

Claim (Non-explicit PRGs) $\mathcal{Y}$: class of fns. $f: \{0,1\}^n \to \{0,1\}$.
For any $\varepsilon > 0$, $\exists$ ε-PRG for $\mathcal{Y}$ with seed length

$$\log\log|\mathcal{Y}| + 2\log(1/\varepsilon) + O(1).$$

Pf: Pick $G: \{0,1\}^\Delta \to \{0,1\}^n$ uar (Δ: TBD). Notice
for any $f \in \mathcal{Y}$, for any seed y, $\mathop{\mathbb{E}}_{G}[f(G(y))] = \mathop{\mathbb{E}}_{U_n}[f(U_n)]$.

Generic Probabilistic Existence Proof

Claim (Non-explicit PRGs) $\mathcal{Y}$: class of fns $f: \{0,1\}^n \to \{0,1\}$.
For any $\varepsilon > 0$, $\exists$ ε-PRG for $\mathcal{Y}$ with seed length

$$\log\log|\mathcal{Y}| + 2\log(1/\varepsilon) + O(1).$$

Pf: Pick $G: \{0,1\}^\Delta \to \{0,1\}^n$ uar (Δ: TBD). Notice
for any $f \in \mathcal{Y}$, for any seed y, $\mathop{\mathbb{E}}_{G}[f(G(y))] = \mathop{\mathbb{E}}_{U_n}[f(U_n)]$.

$\Rightarrow f(G(y))$ are indep, so by Chernoff, $\Pr_G\left[\left|\mathbb{E}[f] - 2^{-\Delta}\sum_{y\in\{0,1\}^\Delta} f(G(y))\right| > \varepsilon\right]$
$\leq 2e^{-2\varepsilon^2 \cdot 2^\Delta}$

> Done by Union bd.

> Done by Union bd.

> Typical case: $\mathcal{H} = \{\text{ckts of size} \le n\}$ - then each $f \in \mathcal{H}$ can be described at most poly(n) bits i.e. $|\mathcal{H}| \le 2^{\text{poly}(n)}$.

So, claim $\Rightarrow$ seed length $O(\log(n/\varepsilon))$.

> Done by Union bd.

> Typical case: $\mathcal{Y} = \{\text{ckts of size} \le n\}$ - then each $f \in \mathcal{Y}$ can be described at most poly(n) bits i.e. $|\mathcal{Y}| \le 2^{\text{poly}(n)}$.

So, claim $\Rightarrow$ seed length $O(\log(n/\varepsilon))$.

↳ Major weakness: doesn't guarantee PRG is efficiently computable!

> Done by Union bd.

> Typical case: $\mathcal{F} = \{\text{ckts of size} \le n\}$ - then each $f \in \mathcal{F}$ can be described at most poly(n) bits i.e. $|\mathcal{F}| \le 2^{\text{poly}(n)}$.

So, claim $\Rightarrow$ seed length $O(\log(n/\varepsilon))$.

↳ Major weakness: doesn't guarantee PRG is efficiently computable!

Defⁿ (explicitness) A PRG $G: \{0,1\}^s \to \{0,1\}^n$ is explicit if it can be computed in poly(n) time.

FYI: Default conjecture is that explicit PRGs exist.

FYI: Default conjecture is that explicit PRGs exist.

> with same seed length as the generic bd.

FYI: Default conjecture is that explicit PRGs exist.

> with same seed length as the generic bd.

> Often, conj. can be supported by conditional constructions.

FYI: Default conjecture is that explicit PRGs exist.

> with same seed length as the generic bd.

> Often, conj. can be supported by conditional constructions.

> Example: [IW97] There is an explict PRG for all size n Boolean ckts with seed length $O(\log(n/\varepsilon))$

FYI: Default conjecture is that explicit PRGs exist.

> with same seed length as the generic bd.

> Often, conj. can be supported by conditional constructions.

> Example: [IW97] There is an explict PRG for all size n Boolean ckts with seed length $O(\log(n/\varepsilon))$

— assuming some complexity theoretic conjectures.

Practice: Triangle Inequality for PRGs.

Practice: Triangle Inequality for PRGs.

Lemma: Let $f_1, \dots, f_k : \{0,1\}^n \to \mathbb{R}$ be fns, let $\lambda_0, \dots, \lambda_k \in \mathbb{R}$ and let $f(x) = \lambda_0 + \sum_{i=1}^{k} \lambda_i f_i(x)$. Let X be a dist. over $\{0,1\}^n$ and assume X fools f_i with error ε_i. Then X fools f with error $\varepsilon = \sum_{1}^{k} |\lambda_i| \cdot \varepsilon_i$.

Practice: Triangle Inequality for PRGs.

Lemma: Let $f_1, \dots, f_k : \{0,1\}^n \to \mathbb{R}$ be fns, let $\lambda_0, \dots, \lambda_k \in \mathbb{R}$ and let $f(x) = \lambda_0 + \sum_{i=1}^{k} \lambda_i f_i(x)$. Let X be a dist. over $\{0,1\}^n$ and assume X fools f_i with error ε_i. Then X fools f with error $\varepsilon = \sum_{1}^{k} |\lambda_i| \cdot \varepsilon_i$.

Pf:

$$\left| \mathbb{E}[f(x)] - \mathbb{E}[f] \right| = \left| \sum_{1}^{k} \lambda_i \mathbb{E}[f_i(x)] - \sum_{1}^{k} \lambda_i \mathbb{E}[f_i] \right|$$

$$\leq \sum_{1}^{k} |\lambda_i| \left| \mathbb{E}[f_i(x)] - \mathbb{E}[f_i] \right|$$

$$\leq \sum |\lambda_i| \cdot \varepsilon_i$$

Let's talk about Communication

Let's talk about Communication

> want to compute $F : \{0,1\}^{n/2} \times \{0,1\}^{n/2} \longrightarrow \{0,1\}$.

Let's talk about Communication

> want to compute $F : \{0,1\}^{n/2} \times \{0,1\}^{n/2} \to \{0,1\}$.

given to Alice

given to Bob.

Let's talk about Communication

> want to compute $F : \{0,1\}^{n/2} \times \{0,1\}^{n/2} \to \{0,1\}$.

given to Alice ← F → given to Bob.

Use as few bits of communication as possible to compute F.

Let's talk about Communication

> want to compute $F : \{0,1\}^{n/2} \times \{0,1\}^{n/2} \longrightarrow \{0,1\}$.

given to Alice — given to Bob.

Use as few bits of communication as possible to compute F.

> no. of bits exchanged = cost of protocol.

Let's talk about Communication

> want to compute $F : \{0,1\}^{n/2} \times \{0,1\}^{n/2} \to \{0,1\}$.

given to Alice — given to Bob.

Use as few bits of communication as possible to compute F.

> no. of bits exchanged = cost of protocol.

> In a protocol of cost m, after m rounds of communication, the protocol terminates and both parties output the same bit $F(x, y)$.

Designing PRGs for Communication Protocols?

Designing PRGs for Communication Protocols?

> Note we want to fool F, a fn. of x & y.
(x: Alice's i/p, y: Bob's i/p)

Designing PRGs for Communication Protocols?

> Note we want to fool F, a fn. of x & y.
(x: Alice's i/p, y: Bob's i/p.)

> So we should think of $x \Delta y$ as random c/ps; and we wanting to replace them with pseudorandom bits, in order to fool F.

Designing PRGs for Communication Protocols?

> Note we want to fool F, a fn. of x & y.
Alice's i/p Bob's i/p

> So we should think of $x \circ y$ as random i/p's; and we wanting to replace them with pseudorandom bits in order to fool F.

Thm (INW94) For every $n, m \in \mathbb{N}$, $\forall \varepsilon > 0$, $\exists$ ε-PRG for 2-party protocols on n bits with communication cost m with seed length $\frac{n}{2} + O(m + \log(1/\varepsilon))$.

Designing PRGs for Communication Protocols?

> Note we want to fool F, a fn. of x & y.
Alice's i/p Bob's i/p

> So we should think of $x \circ y$ as random i/p's; and we wanting to replace them with pseudorandom bits in order to fool F.

Thm (INW94) For every $n, m \in \mathbb{N}$, $\forall \varepsilon > 0$, $\exists$ ε-PRG for 2-party protocols on n bits with communication cost m with seed length $\frac{n}{2} + O(m + \log(1/\varepsilon))$.

→ optimal!

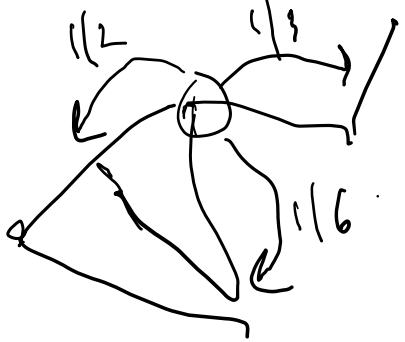
Expander Graphs!

Expander Graphs!

Def^n G: regular undirected graph with transitional probability matrix $M \in [0,1]^{N \times N}$. Let the e.v.s of M be $1 = \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N$. Say G is an ε-spectral expander if $|\lambda_i| \leq \varepsilon \ \forall i = 2, \cdots, N$.

Expander Graphs!

Def^n G: regular undirected graph with transitional probability matrix $M \in [0,1]^{N \times N}$. Let the e.v.s of M be $1 = \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N$. Say G is an ε-spectral expander if $|\lambda_i| \leq \varepsilon \ \forall i = 2, \cdots, N$.

G: 

+ so on.

PRG Characterization of expander graphs.

PRG Characterization of expander graphs.

Lemma n : even. $f : \{0,1\}^n \to \mathbb{R}$ of the form $f(x,y) = g(x) \cdot h(y)$ where $g, h : \{0,1\}^{n/2} \to \mathbb{R}$ satisfy.

$\mathrm{Var}(g) \leq 1$, $\mathrm{Var}(h) \leq 1$.

PRG Characterization of expander graphs.

Lemma $n \in \mathbb{N}$. $\forall f: \{0,1\}^n \to \mathbb{R}$ of the form $f(x,y) = g(x) \cdot h(y)$ where $g, h: \{0,1\}^{n/2} \to \mathbb{R}$ satisfy. $\mathrm{Var}(g) \leq 1$, $\mathrm{Var}(h) \leq 1$. G: regular undirected graph on $\{0,1\}^{n/2}$, sample a uar vertex X in G, sample a uar neighbour Y of X.

PRG Characterization of expander graphs.

Lemma $n \in \mathbb{N}$. $\mathcal{F}: f: \{0,1\}^n \to \mathbb{R}$ of the form $f(x,y) = g(x) \cdot h(y)$ where $g, h: \{0,1\}^{n/2} \to \mathbb{R}$ satisfy. $\mathrm{Var}(g) \le 1$, $\mathrm{Var}(h) \le 1$. G: regular undirected graph on $\{0,1\}^n$, sample a uar vertex X in G, sample a uar neighbour Y of X. Then $\forall \varepsilon > 0$,

G is an ε-spectral expander $\iff$

distribution (X, Y) fools $\mathcal{F}$ with error ε.

PRG Characterization of expander graphs.

Lemma ("combinatorial rectangles") $n \in \mathbb{N}$. Any $f: \{0,1\}^n \to \mathbb{R}$ of the form $f(x,y) = g(x) \cdot h(y)$ where $g, h: \{0,1\}^{n/2} \to \mathbb{R}$ satisfy $\mathrm{Var}(g) \le 1$, $\mathrm{Var}(h) \le 1$. G: regular undirected graph on $\{0,1\}^{n/2}$, sample a uar vertex X in G, sample a uar neighbour Y of X. Then $\forall \varepsilon > 0$,

G is an ε-spectral expander $\iff$

distribution (X, Y) fools f with error ε.

Proof: Think of fns. from $\{0,1\}^{n/2} \to \mathbb{R}$ as long vectors with the inner product $\langle g, h \rangle = \mathbb{E}_{v \sim U_{n/2}} [g(v) \cdot h(v)]$ & the norm $\|g\| = \sqrt{\langle g, g \rangle}$.

Proof: Think of fns. from $\{0,1\}^{n/2} \to \mathbb{R}$ as long vectors with the inner product $\langle g, h \rangle = \mathbb{E}_{v \sim U_{n/2}}[g(v) \cdot h(v)]$ & the norm $\|g\| = \sqrt{\langle g, g \rangle}$.

First, suppose $\mathbb{E}[f] = \mathbb{E}[g] = 0$. then,

$$|\mathbb{E}[f(x,y)]| = |\langle g, Mh \rangle| \leq \|g\| \cdot \|Mh\|$$

by Cauchy - Schwartz.

Proof: Think of fns. from $\{0,1\}^{n/2} \to \mathbb{R}$ as long vectors with the inner product $\langle g, h \rangle = \mathbb{E}_{U \sim U_{n/2}}[g(U) \cdot h(U)]$ & the norm $\|g\| = \sqrt{\langle g, g \rangle}$.

First, suppose $\mathbb{E}[f] = \mathbb{E}[g] = 0$. then,

$$|\mathbb{E}[f(x,y)]| = |\langle g, Mh \rangle| \leq \|g\| \cdot \|Mh\|$$

by Cauchy - Schwartz.

Since G is regular, $\overline{1}$ is an eigenvector of M with ev $\lambda_1 = 1$.

Proof: Think of fns. from $\{0,1\}^{n/2} \to \mathbb{R}$ as long vectors with the inner product $\langle g, h \rangle = \mathbb{E}_{U \sim U_{n/2}}[g(U) \cdot h(U)]$ & the norm $\|g\| = \sqrt{\langle g, g \rangle}$.

First, suppose $\mathbb{E}[f] = \mathbb{E}[g] = 0$. then,

$$|\mathbb{E}[f(x,y)]| = |\langle g, Mh \rangle| \leq \|g\| \cdot \|Mh\|$$

by Cauchy - Schwartz.

Since G is regular, $\overline{1}$ is an eigenvector of M with ev $\lambda_1 = 1$. Since $\mathbb{E}[g] = 0$, $g \perp \overline{1}$.

Proof: Think of fns. from $\{0,1\}^{n/2} \to \mathbb{R}$ as long vectors with the inner product $\langle g, h \rangle = \mathbb{E}_{U \sim U_{n/2}}[g(U) \cdot h(U)]$ & the norm $\|g\| = \sqrt{\langle g, g \rangle}$.

First, suppose $\mathbb{E}[f] = \mathbb{E}[g] = 0$. then,

$$|\mathbb{E}[f(x,y)]| = |\langle g, Mh \rangle| \leq \|g\| \cdot \|Mh\|$$

by Cauchy - Schwartz.

Since G is regular, $\overline{1}$ is an eigenvector of M with ev $\lambda_1 = 1$. Since $\mathbb{E}[g] = 0$, $g \perp \overline{1}$. $\Rightarrow g$ is a l.c of e.v.s other than $\overline{1}$.

therefore, $\|Mg\| \leq \varepsilon - \|g\|$.

therefore, $\|Mg\| \le \varepsilon \cdot \|g\|$. So,

$$|\mathbb{E}[f(X,Y)]| \le \varepsilon \cdot \|g\| \cdot \|h\| = \varepsilon \cdot \sqrt{\mathrm{Var}[g]} \cdot \sqrt{\mathrm{Var}[h]} \le \varepsilon.$$

therefore, $\|Mg\| \le \varepsilon \cdot \|g\|$. So,

$$|\mathbb{E}[f(X,Y)]| \le \varepsilon \cdot \|g\| \cdot \|h\| = \varepsilon \cdot \sqrt{\mathrm{Var}[g]} \cdot \sqrt{\mathrm{Var}[h]} \le \varepsilon.$$

For the gen. case, write $g = \mathbb{E}[g] + \bar{g}$, $h = \mathbb{E}[h] + \bar{h}$,

so, $f(x,y) = \mathbb{E}[g] \cdot \mathbb{E}[h] + \bar{g}(x) \cdot \bar{h}(y) + \mathbb{E}[g] \cdot \bar{h}(y) + \mathbb{E}[h] \bar{g}(x)$

therefore, $\|Mg\| \le \varepsilon \cdot \|g\|$. So,

$$|\mathbb{E}[f(x,y)]| \le \varepsilon \cdot \|g\| \cdot \|h\| = \varepsilon \cdot \sqrt{\mathrm{Var}[g]} \cdot \sqrt{\mathrm{Var}[h]} \le \varepsilon.$$

For the gen. case, write $g = \mathbb{E}[g] + \bar{g}$, $h = \mathbb{E}[h] + \bar{h}$,

so, $f(x,y) = \mathbb{E}[g] \cdot \mathbb{E}[h] + \bar{g}(x) \cdot \bar{h}(y) + \mathbb{E}[g] \cdot \bar{h}(y) + \mathbb{E}[h] \bar{g}(x)$

Since G is regular, each marginal dist. X & Y is uniform over $\{0,1\}^{n/2}$. Therefore, $\mathbb{E}[\bar{g}(x)] = \mathbb{E}[\bar{h}(y)] = 0$.

therefore, $\|Mg\| \le \varepsilon \cdot \|g\|$. So,

$$|\mathbb{E}[f(X,Y)]| \le \varepsilon \cdot \|g\| \cdot \|h\| = \varepsilon \cdot \sqrt{\mathrm{Var}[g]} \cdot \sqrt{\mathrm{Var}[h]} \le \varepsilon.$$

For the gen. case, write $g = \mathbb{E}[g] + \bar{g}$, $h = \mathbb{E}[h] + \bar{h}$,

so, $f(x,y) = \mathbb{E}[g] \cdot \mathbb{E}[h] + \bar{g}(x) \cdot \bar{h}(y) + \mathbb{E}[g] \cdot \bar{h}(y) + \mathbb{E}[h] \bar{g}(x)$

Since G is regular, each marginal dist. X & Y is uniform over $\{0,1\}^{n/2}$. Therefore, $\mathbb{E}[\bar{g}(X)] = \mathbb{E}[\bar{h}(Y)] = 0$.

Moreover, $\mathrm{Var}[\bar{g}] = \mathrm{Var}[g] \le 1$ & $\mathrm{Var}[\bar{h}] = \mathrm{Var}[h] \le 1$, so,

$$|\mathbb{E}[f(X,Y) - \mathbb{E}[f]]| = |\mathbb{E}[\bar{g}(X) \cdot \bar{h}(Y)]| \le \varepsilon.$$

Explicit Expanders Exist!

Explicit Expanders Exist!

Thm $\forall n \in \mathbb{N}, \forall \varepsilon > 0$, $\exists$ ε-spectral expander with vertex set $\{0,1\}^n$ & degree $D = \text{poly}(1/\varepsilon)$.

Explicit Expanders Exist!

Thm $\forall n \in \mathbb{N}, \forall \varepsilon > 0$, $\exists$ ε-spectral expander with vertex set $\{0,1\}^n$ & degree $D = \text{poly}(1/\varepsilon)$. The expander is "strongly explicit" i.e., given a vertex x and a value $i \in [D]$, the i^{th} neighbour of x can be computed in time $\text{poly}(n, \log(1/\varepsilon))$.

Explicit Expanders Exist!

Thm $\forall n \in \mathbb{N}, \forall \varepsilon > 0$, $\exists$ ε-spectral expander with vertex set $\{0,1\}^n$ & degree $D = \text{poly}(1/\varepsilon)$. The expander is "strongly explicit" i.e., given a vertex x and a value $i \in [D]$, the i^{th} neighbour of x can be computed in time $\text{poly}(n, \log(1/\varepsilon))$.

Cor: $\exists$ explicit ε-PRG for combinatorial rectangle on n bits, with seed length $\frac{n}{2} + O(\log(1/\varepsilon))$

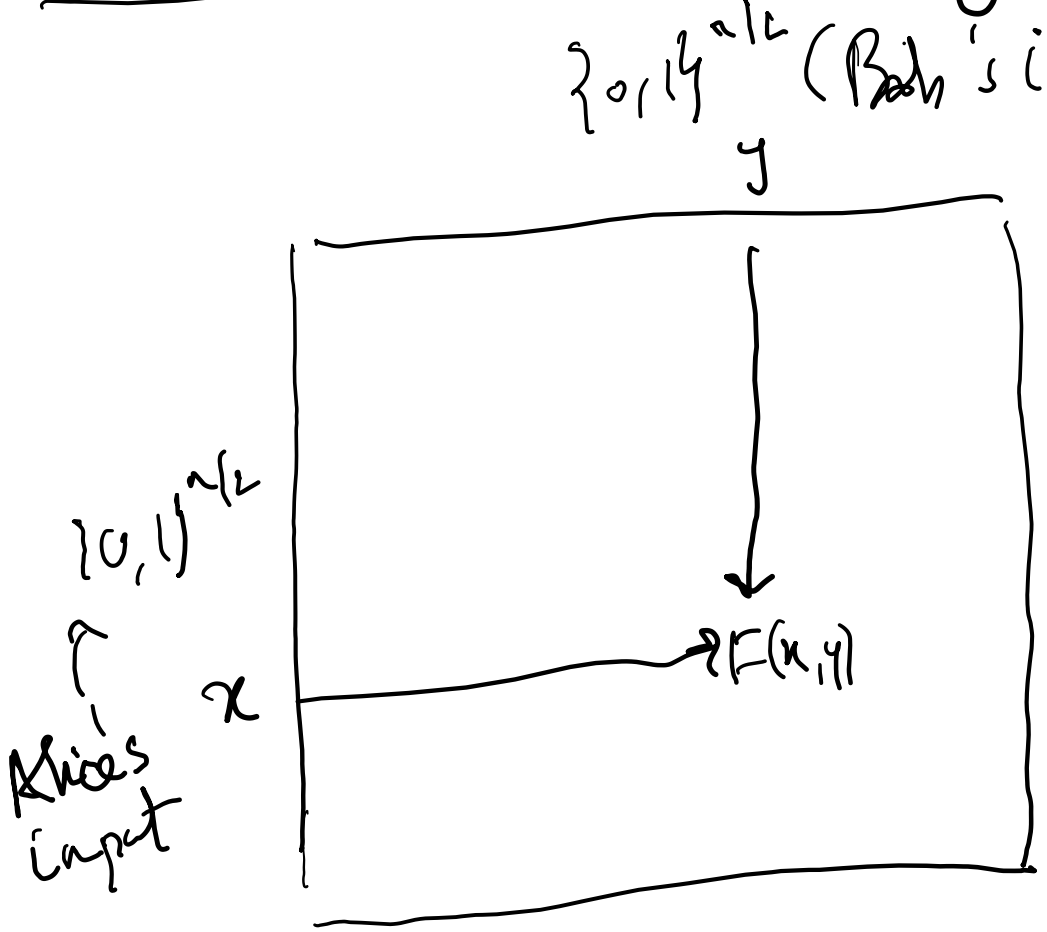
Proof of PRGs for 2-party Communication Protocols

Proof of PRGs for 2-party Communication Protocols

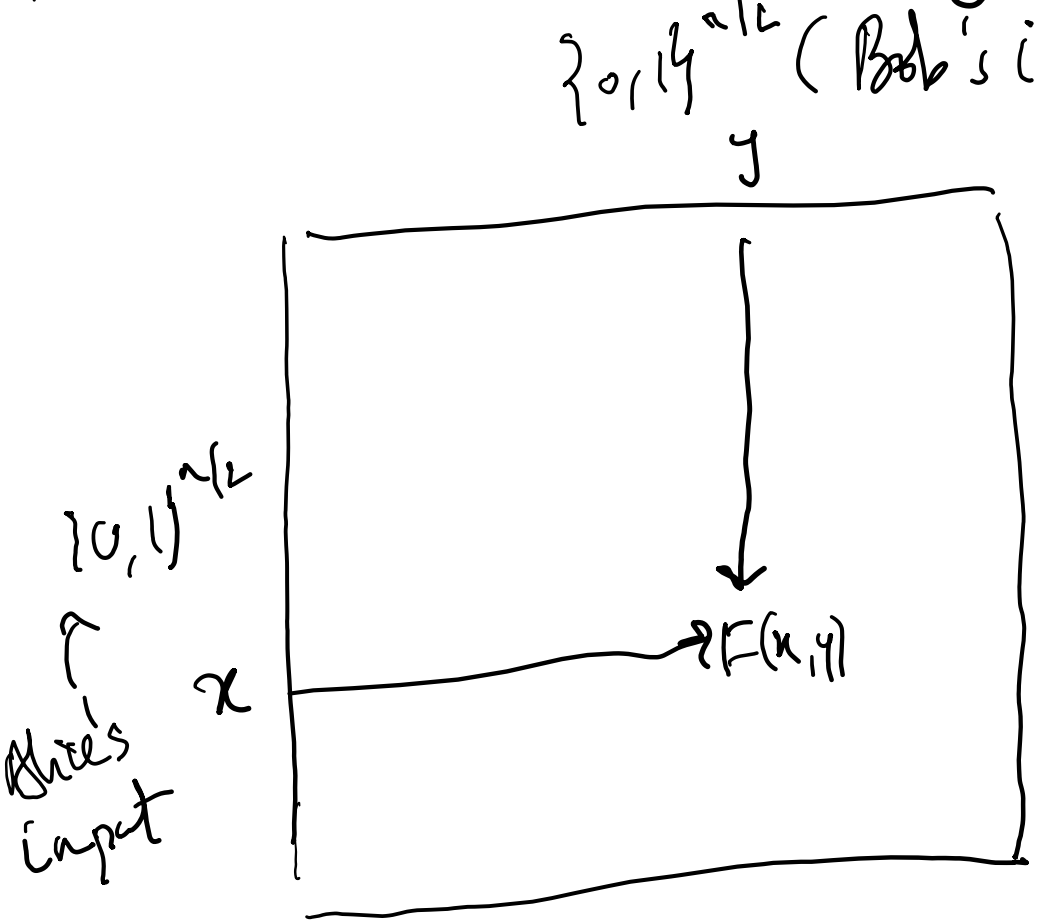
Idea: Any $f(x,y)$ computed by a cost m protocol, can be "broken up" into 2^m - many combinatorial rectangles. Then just use prev. result and the Δ-inequality for PRGs.

Combinatorial Rectangles in Communication

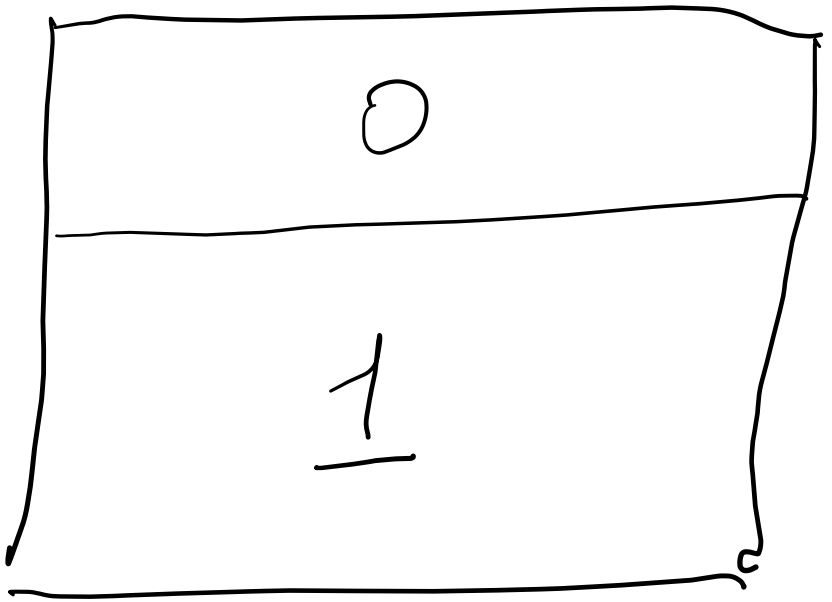
Combinatorial Rectangles in Communication



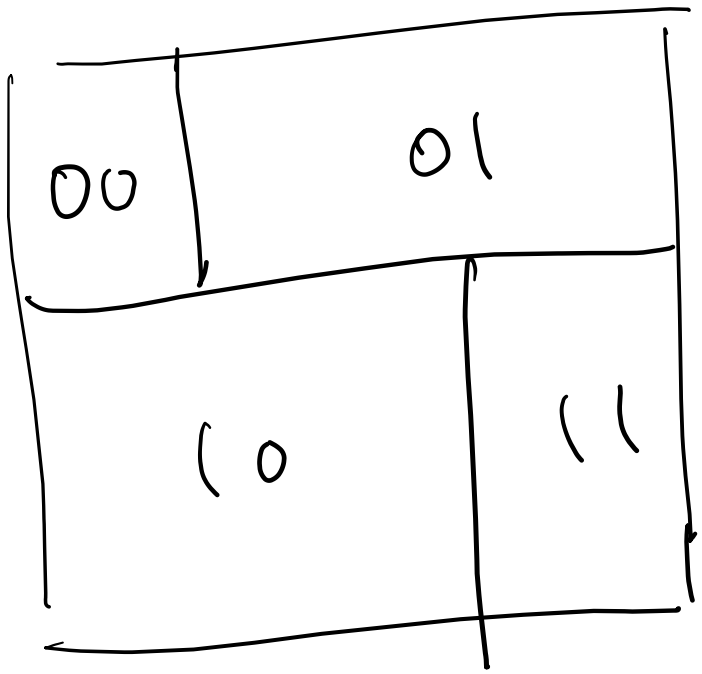
Combinatorial Rectangles in Communication



Now, if Π halts after 1 rd, then necessarily:



If TT stops after two rds, then.



and so on...

Thank You!