An Introduction to the Hardness vs. Randomness Paradigm

Tutorial for the Simons HDX and Pseudorandomness Workshop

Ronen Shaltiel

University of Haifa

August 2026

Outline

About these talks

- Who is this talk for?

About these talks

- Who is this talk for? I am not sure...

About these talks

- Who is this talk for? I am not sure...
- Who isn't this talk for?

About these talks

- Who is this talk for? I am not sure...
- Who isn't this talk for? Experts!

About these talks

- Who is this talk for? I am not sure...
- Who isn't this talk for? Experts!
- If you consider yourself an expert, I ask you to get up and leave!

About these talks

- Who is this talk for? I am not sure...
- Who isn't this talk for? Experts!
- If you consider yourself an expert, I ask you to get up and leave!

Plan:

About these talks

- Who is this talk for? I am not sure...
- Who isn't this talk for? Experts!
- If you consider yourself an expert, I ask you to get up and leave!

Plan:

- Cover classical works in hardness vs. randomness: [NW88,IW97,IW98,STV99,Tre99...]

About these talks

- Who is this talk for? I am not sure...
- Who isn't this talk for? Experts!
- If you consider yourself an expert, I ask you to get up and leave!

Plan:

- Cover classical works in hardness vs. randomness: [NW88,IW97,IW98,STV99,Tre99...]
- I won't specifically *cover recent work!* Left to future talks.

About these talks

- Who is this talk for? I am not sure...
- Who isn't this talk for? Experts!
- If you consider yourself an expert, I ask you to get up and leave!

Plan:

- Cover classical works in hardness vs. randomness: [NW88,IW97,IW98,STV99,Tre99...]
- I won't specifically *cover recent work!* Left to future talks.
- Mention recent research directions when surveying "the classics".

About these talks

- Who is this talk for? I am not sure...
- Who isn't this talk for? Experts!
- If you consider yourself an expert, I ask you to get up and leave!

Plan:

- Cover classical works in hardness vs. randomness: [NW88,IW97,IW98,STV99,Tre99...]
- I won't specifically *cover recent work!* Left to future talks.
- Mention recent research directions when surveying "the classics".
- Many of these classic results and ideas are components in recent work.

About these talks

- Who is this talk for? I am not sure...
- Who isn't this talk for? Experts!
- If you consider yourself an expert, I ask you to get up and leave!

Plan:

- Cover classical works in hardness vs. randomness: [NW88,IW97,IW98,STV99,Tre99...]
- I won't specifically *cover recent work!* Left to future talks.
- Mention recent research directions when surveying "the classics".
- Many of these classic results and ideas are components in recent work.
- Relatively precise about statements: definitions, theorem statements.

About these talks

- Who is this talk for? I am not sure...
- Who isn't this talk for? Experts!
- If you consider yourself an expert, I ask you to get up and leave!

Plan:

- Cover classical works in hardness vs. randomness: [NW88,IW97,IW98,STV99,Tre99...]
- I won't specifically *cover recent work!* Left to future talks.
- Mention recent research directions when surveying "the classics".
- Many of these classic results and ideas are components in recent work.
- Relatively precise about statements: definitions, theorem statements.
- Nontechnical about proofs: proof sketches focusing on ideas and suppressing details.

About these talks

- Who is this talk for? I am not sure...
- Who isn't this talk for? Experts!
- If you consider yourself an expert, I ask you to get up and leave!

Plan:

- Cover classical works in hardness vs. randomness: [NW88,IW97,IW98,STV99,Tre99...]
- I won't specifically *cover recent work!* Left to future talks.
- Mention recent research directions when surveying "the classics".
- Many of these classic results and ideas are components in recent work.
- Relatively precise about statements: definitions, theorem statements.
- Nontechnical about proofs: proof sketches focusing on ideas and suppressing details.
- I will sometimes blatantly cheat!

About these talks

- Who is this talk for? I am not sure...
- Who isn't this talk for? Experts!
- If you consider yourself an expert, I ask you to get up and leave!

Plan:

- Cover classical works in hardness vs. randomness: [NW88,IW97,IW98,STV99,Tre99...]
- I won't specifically *cover recent work!* Left to future talks.
- Mention recent research directions when surveying "the classics".
- Many of these classic results and ideas are components in recent work.
- Relatively precise about statements: definitions, theorem statements.
- Nontechnical about proofs: proof sketches focusing on ideas and suppressing details.
- I will sometimes blatantly cheat!

Let's see how this goes...

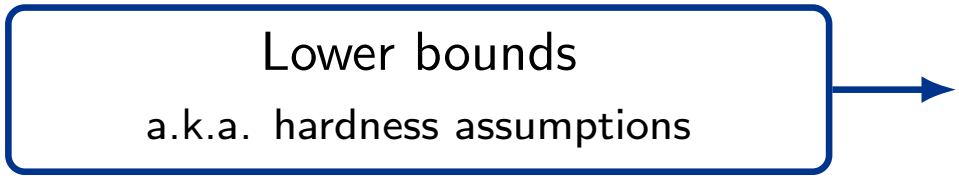
The hardness vs. randomness paradigm at a glance

The hardness vs. randomness paradigm at a glance

Lower bounds

a.k.a. hardness assumptions

The hardness vs. randomness paradigm at a glance



The hardness vs. randomness paradigm at a glance

The hardness vs. randomness paradigm at a glance

Thm: [IW97]

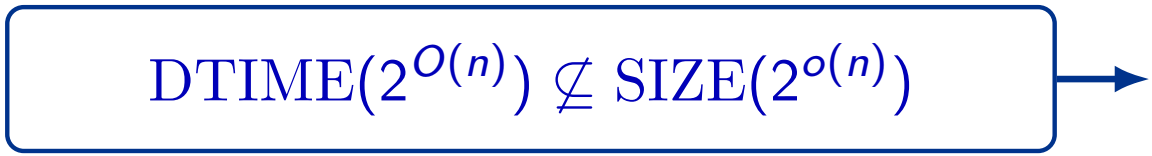
The hardness vs. randomness paradigm at a glance

Thm: [IW97]

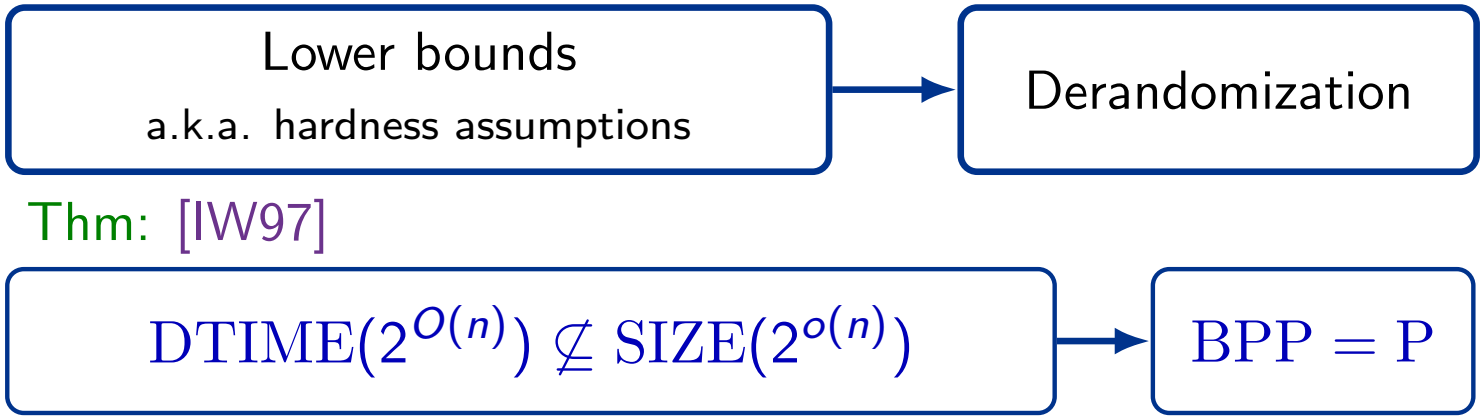
$$\mathrm{DTIME}(2^{O(n)}) \not\subseteq \mathrm{SIZE}(2^{o(n)})$$

The hardness vs. randomness paradigm at a glance

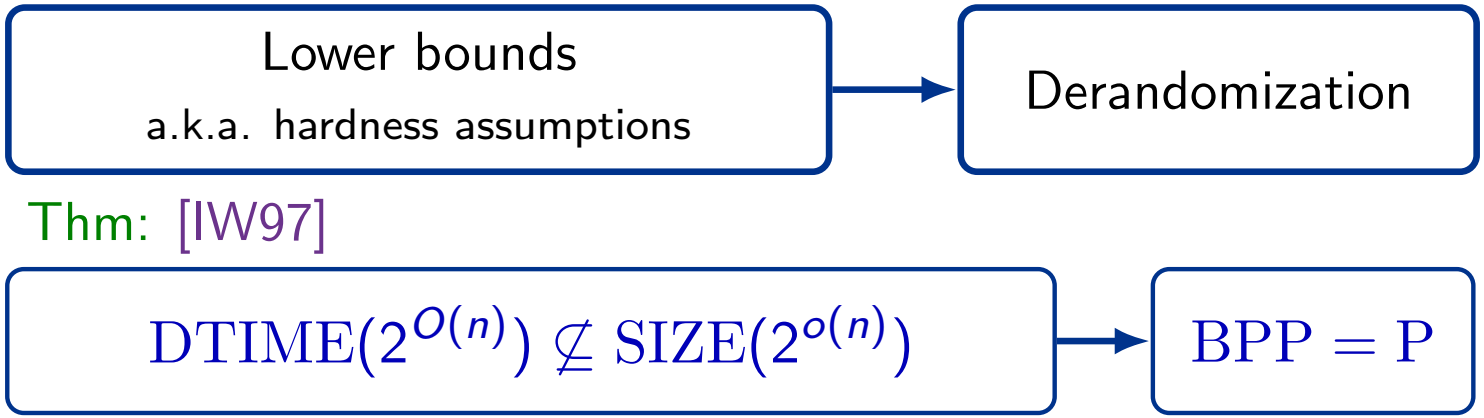
Thm: [IW97]



The hardness vs. randomness paradigm at a glance



The hardness vs. randomness paradigm at a glance



Conditional result...

The hardness vs. randomness paradigm at a glance

Thm: [IW97]

Conditional result... But many complexity results are conditional.

The hardness vs. randomness paradigm at a glance

Thm: [IW97]

Conditional result... But many complexity results are conditional.

Example: $3 - SAT \leq_P CLIQUE$

The hardness vs. randomness paradigm at a glance

Thm: [IW97]

Conditional result... But many complexity results are conditional.

Example: $3-SAT \leq_P CLIQUE$

Interpretation: *CLIQUE* easy $\Rightarrow$ $3-SAT$ easy

The hardness vs. randomness paradigm at a glance

Thm: [IW97]

Conditional result... But many complexity results are conditional.

Example: $3-SAT \leq_P CLIQUE$

Interpretation: $CLIQUE$ easy $\Rightarrow$ $3-SAT$ easy

Equivalently, $3-SAT$ hard $\Rightarrow$ $CLIQUE$ hard

The hardness vs. randomness paradigm at a glance

Thm: [IW97]

Conditional result... But many complexity results are conditional.

Example: $3-SAT \leq_P CLIQUE$

Interpretation: $CLIQUE$ easy $\Rightarrow$ $3-SAT$ easy

Equivalently, $3-SAT$ hard $\Rightarrow$ $CLIQUE$ hard

Hardness vs. randomness has different structure:

The hardness vs. randomness paradigm at a glance

Thm: [IW97]

Conditional result... But many complexity results are conditional.

Example: $3-SAT \leq_P CLIQUE$

Interpretation: $CLIQUE$ easy $\Rightarrow$ $3-SAT$ easy

Equivalently, $3-SAT$ hard $\Rightarrow$ $CLIQUE$ hard

Hardness vs. randomness has different structure:

hard $\Rightarrow$ easy

The hardness vs. randomness paradigm at a glance

Thm: [IW97]

Conditional result... But many complexity results are conditional.

Example: $3-SAT \leq_P CLIQUE$

Interpretation: $CLIQUE$ easy $\Rightarrow$ $3-SAT$ easy

Equivalently, $3-SAT$ hard $\Rightarrow$ $CLIQUE$ hard

Hardness vs. randomness has different structure:

$$\text{hard} \Rightarrow \text{easy}$$

This is achieved using pseudorandom generators (PRGs)

The hardness vs. randomness paradigm at a glance

Thm: [IW97]

Conditional result... But many complexity results are conditional.

Example: $3-SAT \leq_P CLIQUE$

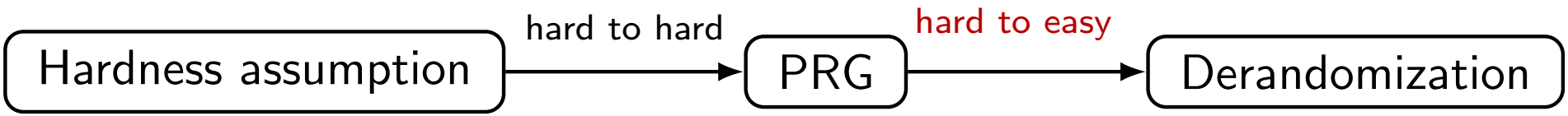
Interpretation: $CLIQUE$ easy $\Rightarrow$ $3-SAT$ easy

Equivalently, $3-SAT$ hard $\Rightarrow$ $CLIQUE$ hard

Hardness vs. randomness has different structure:

$$\text{hard} \Rightarrow \text{easy}$$

This is achieved using pseudorandom generators (PRGs)



Plan for talks

Plan for talks

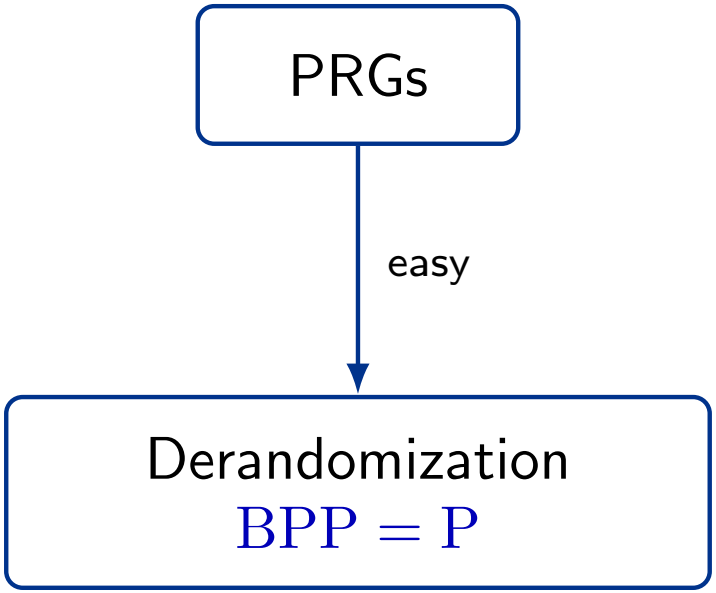
PRGs

Plan for talks

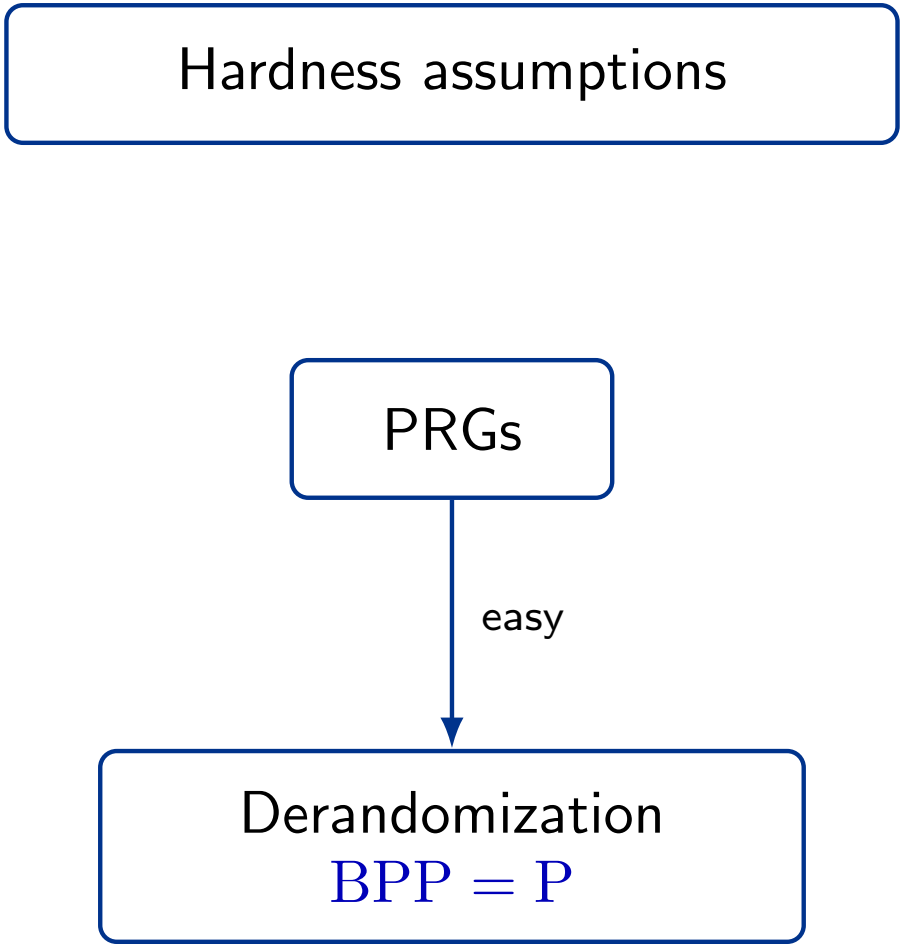
PRGs

Derandomization
$\mathrm{BPP} = \mathrm{P}$

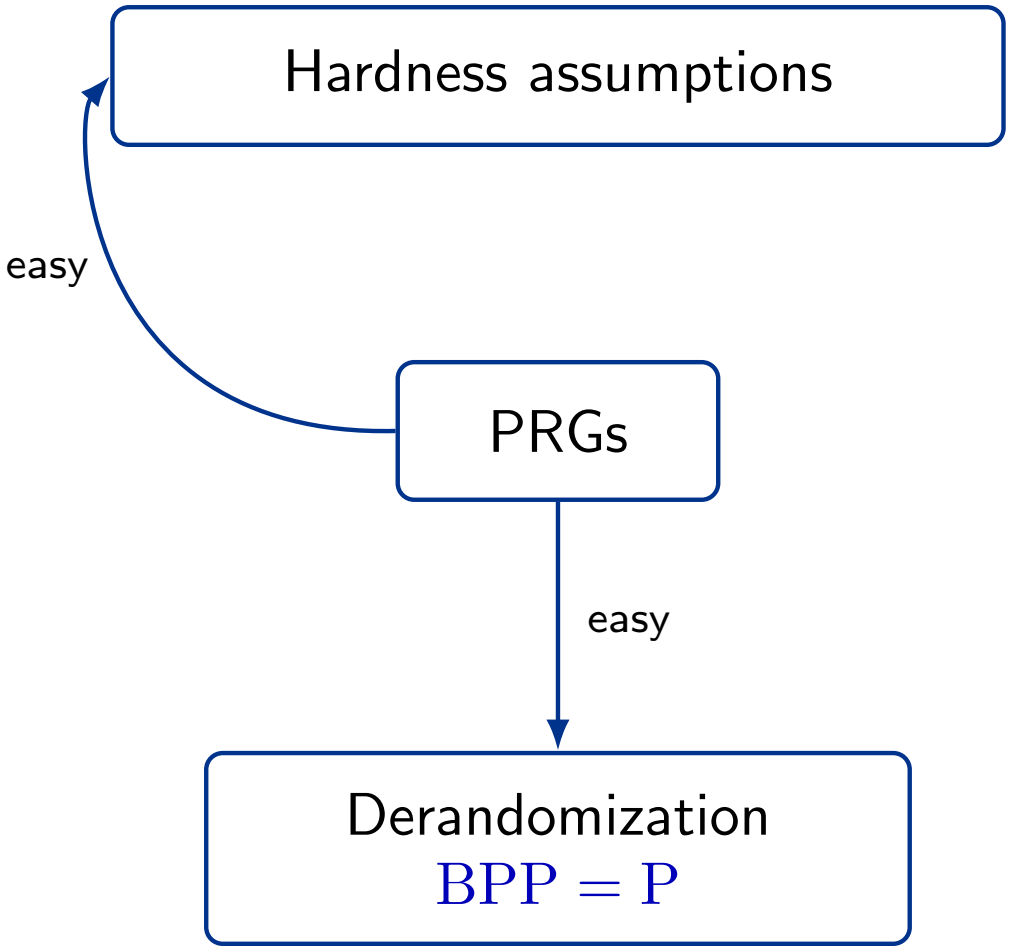
Plan for talks



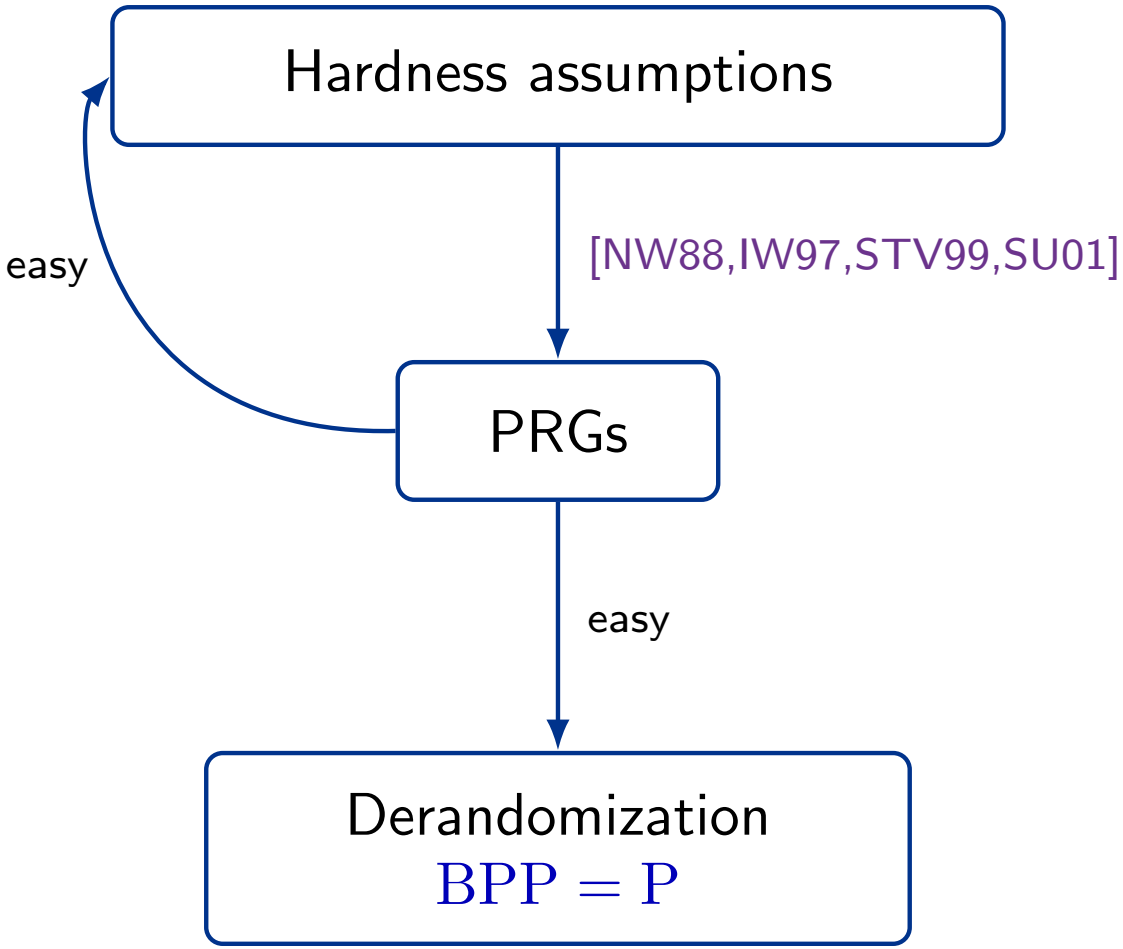
Plan for talks



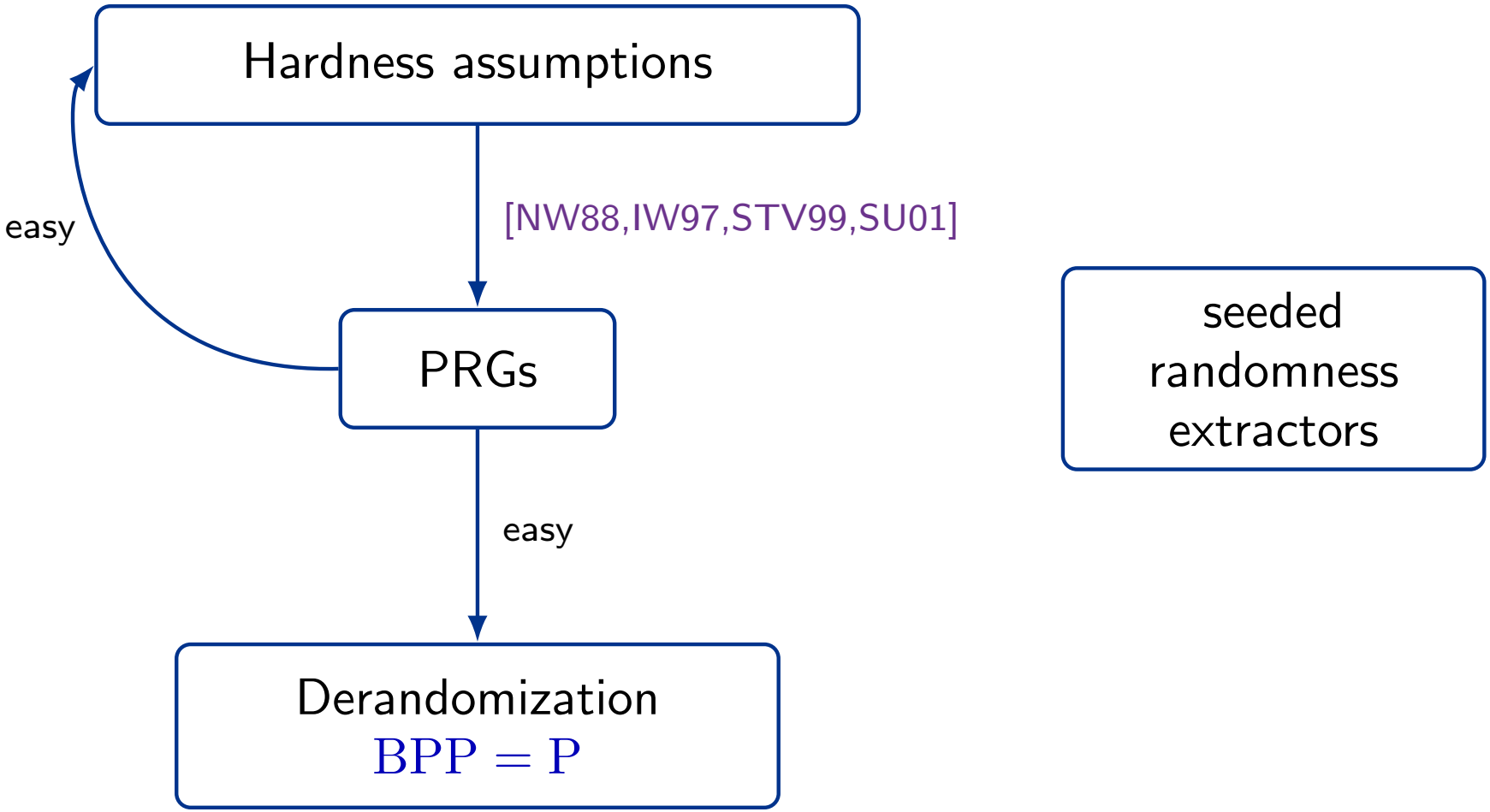
Plan for talks



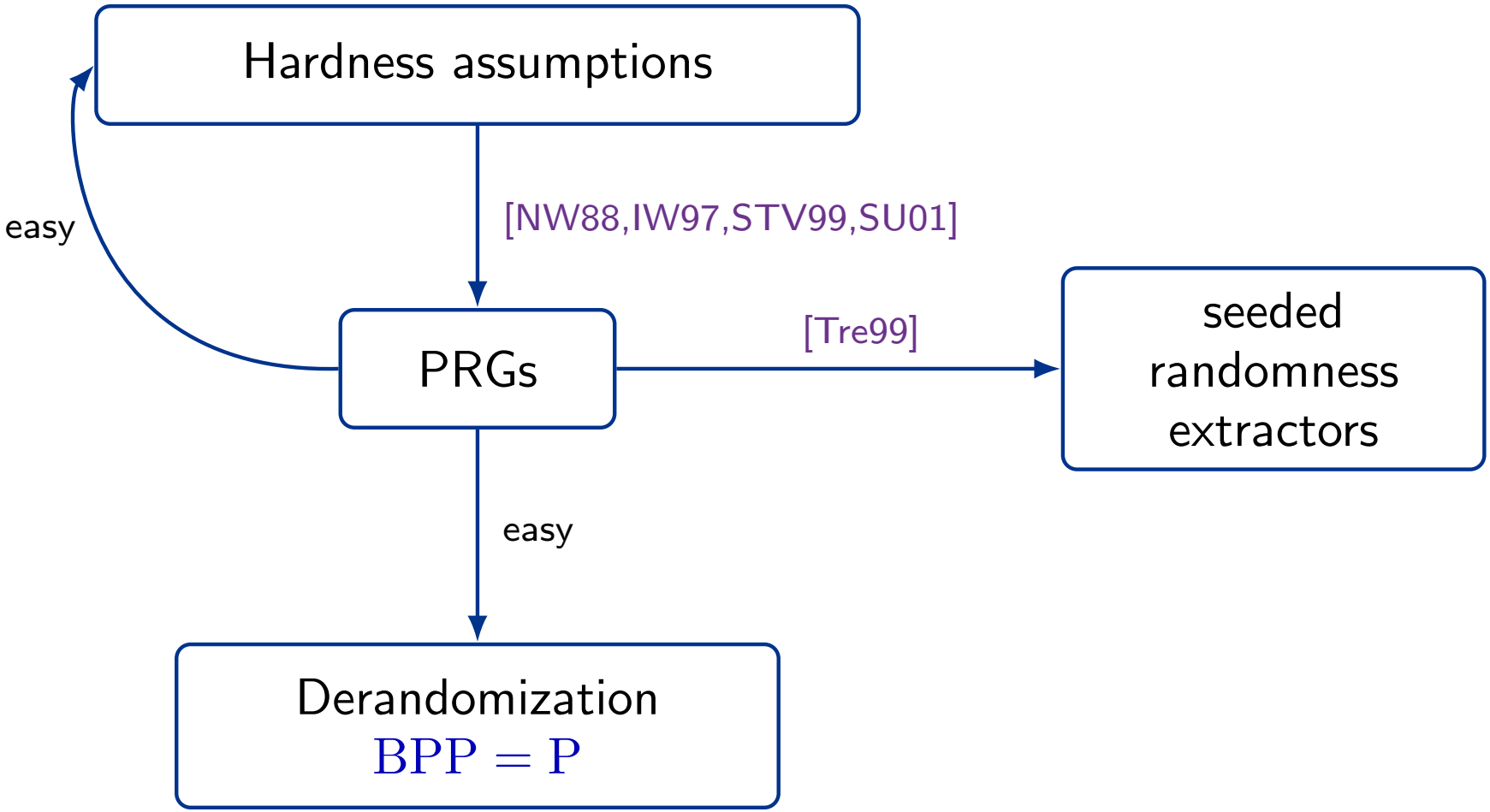
Plan for talks



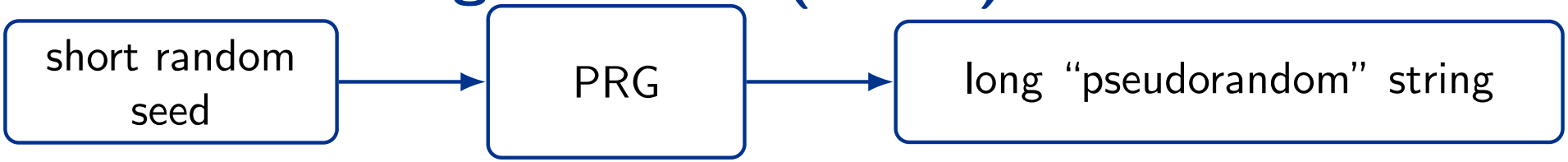
Plan for talks



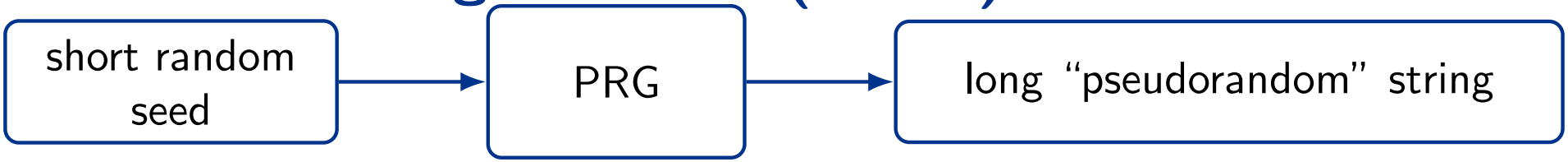
Plan for talks



Pseudorandom generators (PRGs)



Pseudorandom generators (PRGs)



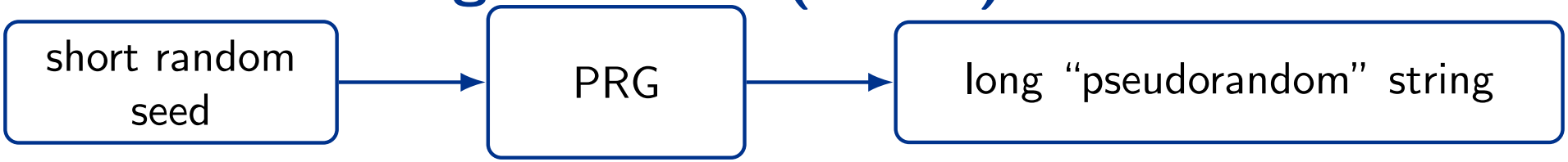
Definition (Pseudorandom generator (PRG))

$G : \{0,1\}^r \to \{0,1\}^m$ is an ϵ-PRG for $D : \{0,1\}^m \to \{0,1\}$ if

$$|\Pr[D(G(U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon.$$

Throughout this talk, $\epsilon = 1/10$.

Pseudorandom generators (PRGs)



Definition (Pseudorandom generator (PRG))

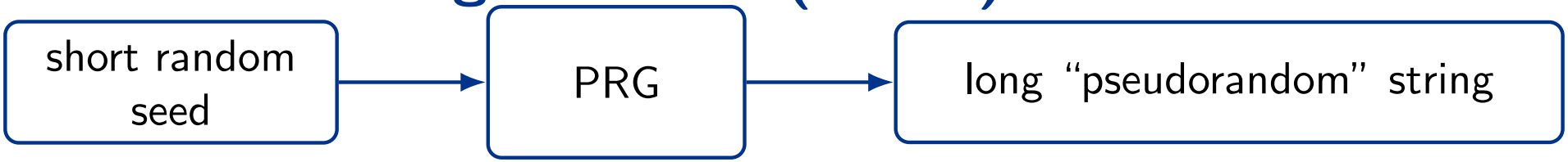
$G : \{0,1\}^r \to \{0,1\}^m$ is an ϵ-PRG for $D : \{0,1\}^m \to \{0,1\}$ if

$$|\Pr[D(G(U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon.$$

Throughout this talk, $\epsilon = 1/10$.

Big idea: "Randomness is in the eye of the beholder".

Pseudorandom generators (PRGs)



Definition (Pseudorandom generator (PRG))

$G : \{0,1\}^r \to \{0,1\}^m$ is an ϵ-PRG for $D : \{0,1\}^m \to \{0,1\}$ if

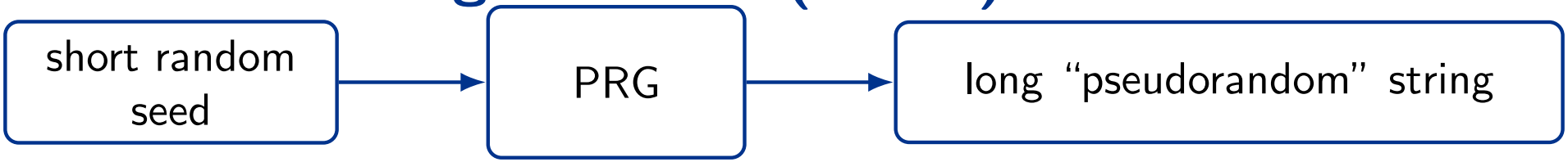
$$|\Pr[D(G(U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon.$$

Throughout this talk, $\epsilon = 1/10$.

Big idea: "Randomness is in the eye of the beholder".

- Typically interested in a class of distinguishers.

Pseudorandom generators (PRGs)



Definition (Pseudorandom generator (PRG))

$G : \{0,1\}^r \to \{0,1\}^m$ is an ϵ-PRG for $D : \{0,1\}^m \to \{0,1\}$ if

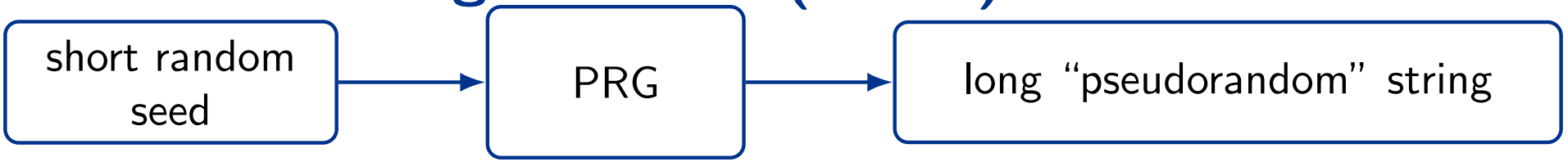
$$|\Pr[D(G(U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon.$$

Throughout this talk, $\epsilon = 1/10$.

Big idea: "Randomness is in the eye of the beholder".

- Typically interested in a class of distinguishers. This talk: circuits of size m.

Pseudorandom generators (PRGs)



Definition (Pseudorandom generator (PRG))

$G : \{0,1\}^r \to \{0,1\}^m$ is an ϵ-PRG for $D : \{0,1\}^m \to \{0,1\}$ if

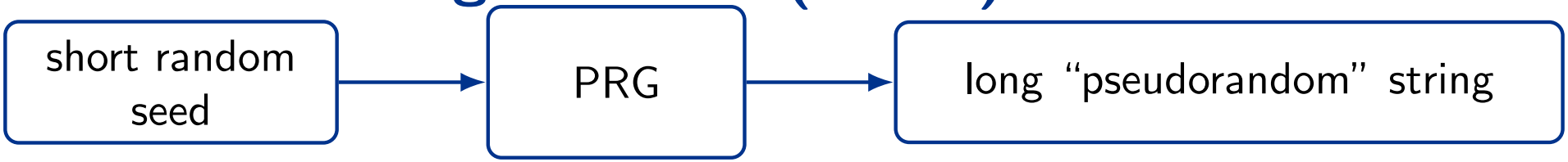
$$|\Pr[D(G(U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon.$$

Throughout this talk, $\epsilon = 1/10$.

Big idea: "Randomness is in the eye of the beholder".

- Typically interested in a class of distinguishers. This talk: circuits of size m.
- Explicit PRGs: G is efficiently computable.

Pseudorandom generators (PRGs)



Definition (Pseudorandom generator (PRG))

$G : \{0,1\}^r \to \{0,1\}^m$ is an ϵ-PRG for $D : \{0,1\}^m \to \{0,1\}$ if

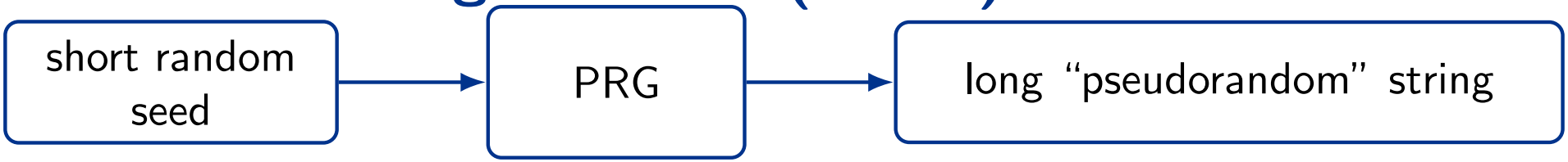
$$|\Pr[D(G(U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon.$$

Throughout this talk, $\epsilon = 1/10$.

Big idea: "Randomness is in the eye of the beholder".

- Typically interested in a class of distinguishers. This talk: circuits of size m.
- Explicit PRGs: G is efficiently computable.
  - Strongly explicit: G computable in time polynomial in output length m.

Pseudorandom generators (PRGs)



Definition (Pseudorandom generator (PRG))

$G : \{0,1\}^r \to \{0,1\}^m$ is an ϵ-PRG for $D : \{0,1\}^m \to \{0,1\}$ if

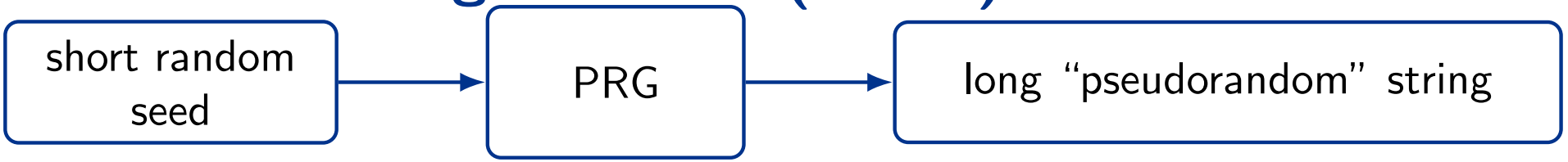
$$|\Pr[D(G(U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon.$$

Throughout this talk, $\epsilon = 1/10$.

Big idea: "Randomness is in the eye of the beholder".

- Typically interested in a class of distinguishers. This talk: circuits of size m.
- Explicit PRGs: G is efficiently computable.
  - Strongly explicit: G computable in time polynomial in output length m.
  - Weakly explicit: G computable in time exponential in seed length r.

Pseudorandom generators (PRGs)



Definition (Pseudorandom generator (PRG))

$G : \{0,1\}^r \to \{0,1\}^m$ is an ϵ-PRG for $D : \{0,1\}^m \to \{0,1\}$ if

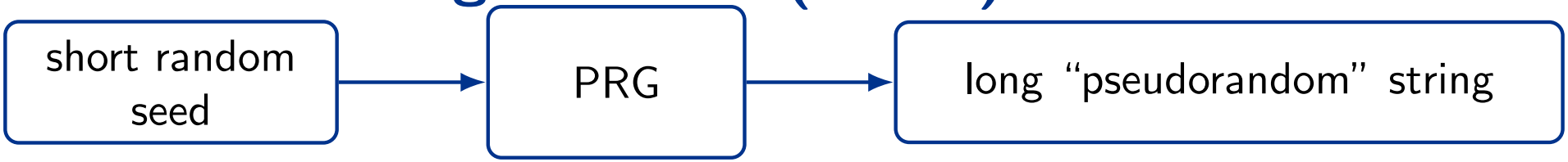
$$|\Pr[D(G(U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon.$$

Throughout this talk, $\epsilon = 1/10$.

Big idea: "Randomness is in the eye of the beholder".

- Typically interested in a class of distinguishers. This talk: circuits of size m.
- Explicit PRGs: G is efficiently computable.
  - Strongly explicit: G computable in time polynomial in output length m.
  - Weakly explicit: G computable in time exponential in seed length r.
- In this talk: $m = 2^{\Theta(r)} \Rightarrow r = \Theta(\log m)$, two notions coincide.

Pseudorandom generators (PRGs)



Definition (Pseudorandom generator (PRG))

$G : \{0,1\}^r \to \{0,1\}^m$ is an ϵ-PRG for $D : \{0,1\}^m \to \{0,1\}$ if

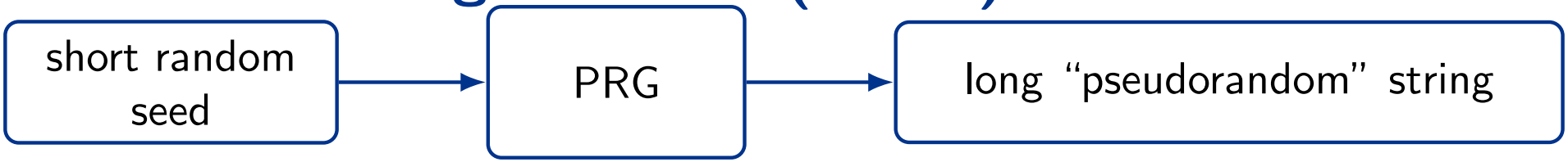
$$|\Pr[D(G(U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon.$$

Throughout this talk, $\epsilon = 1/10$.

Big idea: "Randomness is in the eye of the beholder".

- Typically interested in a class of distinguishers. This talk: circuits of size m.
- Explicit PRGs: G is efficiently computable.
  - Strongly explicit: G computable in time polynomial in output length m.
  - Weakly explicit: G computable in time exponential in seed length r.
- In this talk: $m = 2^{\Theta(r)} \Rightarrow r = \Theta(\log m)$, two notions coincide.
- "Nisan-Wigderson setup": "running time of G" $= m^{O(1)}$

Pseudorandom generators (PRGs)



Definition (Pseudorandom generator (PRG))

$G : \{0,1\}^r \to \{0,1\}^m$ is an ϵ-PRG for $D : \{0,1\}^m \to \{0,1\}$ if

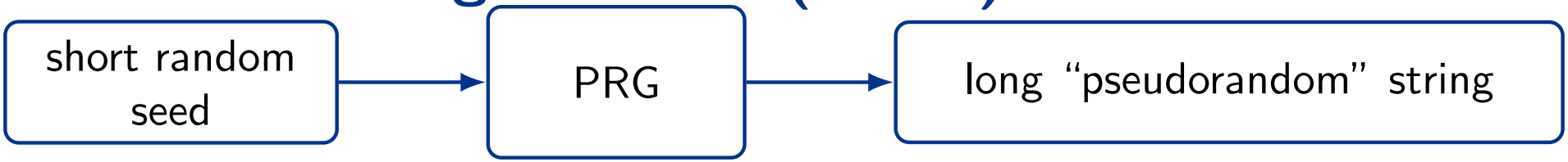
$$|\Pr[D(G(U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon.$$

Throughout this talk, $\epsilon = 1/10$.

Big idea: "Randomness is in the eye of the beholder".

- Typically interested in a class of distinguishers. This talk: circuits of size m.
- Explicit PRGs: G is efficiently computable.
  - Strongly explicit: G computable in time polynomial in output length m.
  - Weakly explicit: G computable in time exponential in seed length r.
- In this talk: $m = 2^{\Theta(r)} \Rightarrow r = \Theta(\log m)$, two notions coincide.
- "Nisan-Wigderson setup": "running time of G"$= m^{O(1)} >$

Pseudorandom generators (PRGs)



Definition (Pseudorandom generator (PRG))

$G : \{0,1\}^r \to \{0,1\}^m$ is an ϵ-PRG for $D : \{0,1\}^m \to \{0,1\}$ if

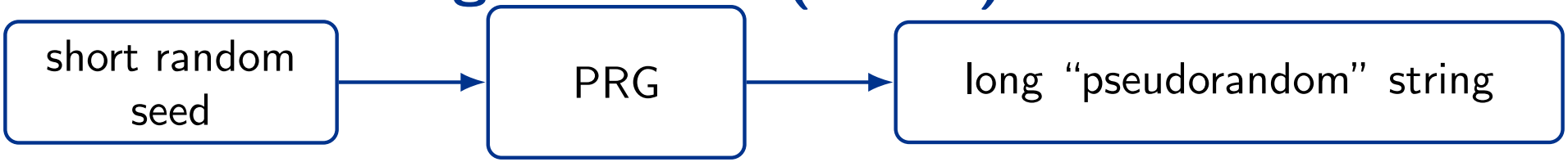
$$|\Pr[D(G(U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon.$$

Throughout this talk, $\epsilon = 1/10$.

Big idea: "Randomness is in the eye of the beholder".

- Typically interested in a class of distinguishers. This talk: circuits of size m.
- Explicit PRGs: G is efficiently computable.
  - Strongly explicit: G computable in time polynomial in output length m.
  - Weakly explicit: G computable in time exponential in seed length r.
- In this talk: $m = 2^{\Theta(r)} \Rightarrow r = \Theta(\log m)$, two notions coincide.
- "Nisan-Wigderson setup": "running time of G"$= m^{O(1)} > m =$ "size of D".

Pseudorandom generators (PRGs)



Definition (Pseudorandom generator (PRG))

$G : \{0,1\}^r \to \{0,1\}^m$ is an ϵ-PRG for $D : \{0,1\}^m \to \{0,1\}$ if

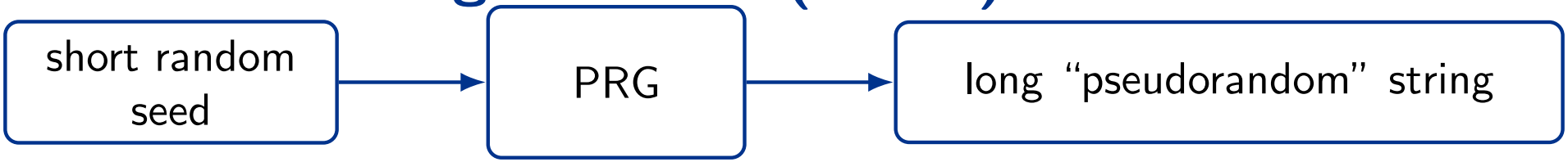
$$|\Pr[D(G(U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon.$$

Throughout this talk, $\epsilon = 1/10$.

Big idea: "Randomness is in the eye of the beholder".

- Typically interested in a class of distinguishers. This talk: circuits of size m.
- Explicit PRGs: G is efficiently computable.
  - Strongly explicit: G computable in time polynomial in output length m.
  - Weakly explicit: G computable in time exponential in seed length r.
- In this talk: $m = 2^{\Theta(r)} \Rightarrow r = \Theta(\log m)$, two notions coincide.
- "Nisan-Wigderson setup": "running time of G"$= m^{O(1)} > m =$ "size of D".
- "Crypto setup": D is sufficiently strong to run G.

Pseudorandom generators (PRGs)



Definition (Pseudorandom generator (PRG))

$G : \{0,1\}^r \to \{0,1\}^m$ is an ϵ-PRG for $D : \{0,1\}^m \to \{0,1\}$ if

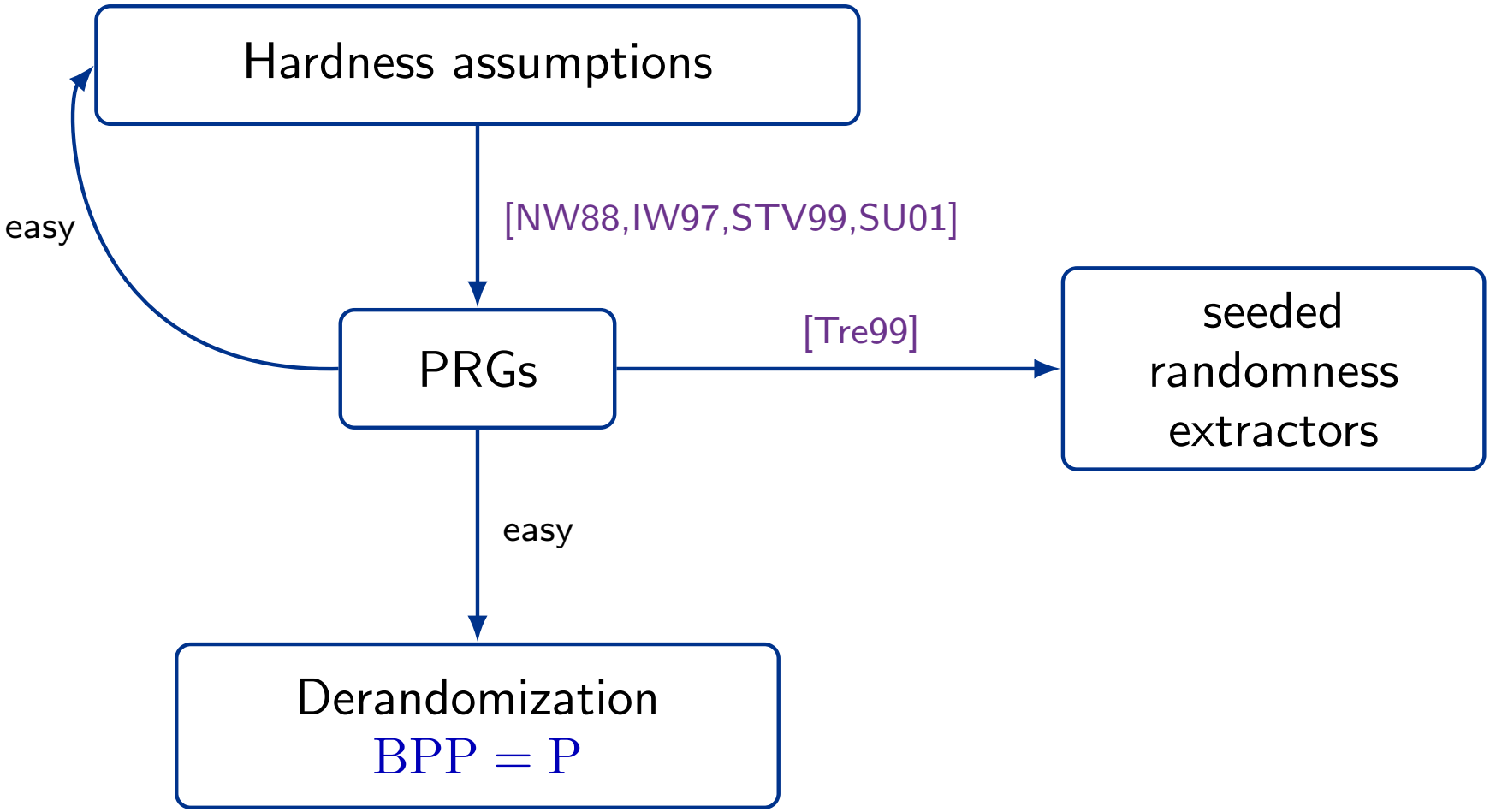
$$|\Pr[D(G(U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon.$$

Throughout this talk, $\epsilon = 1/10$.

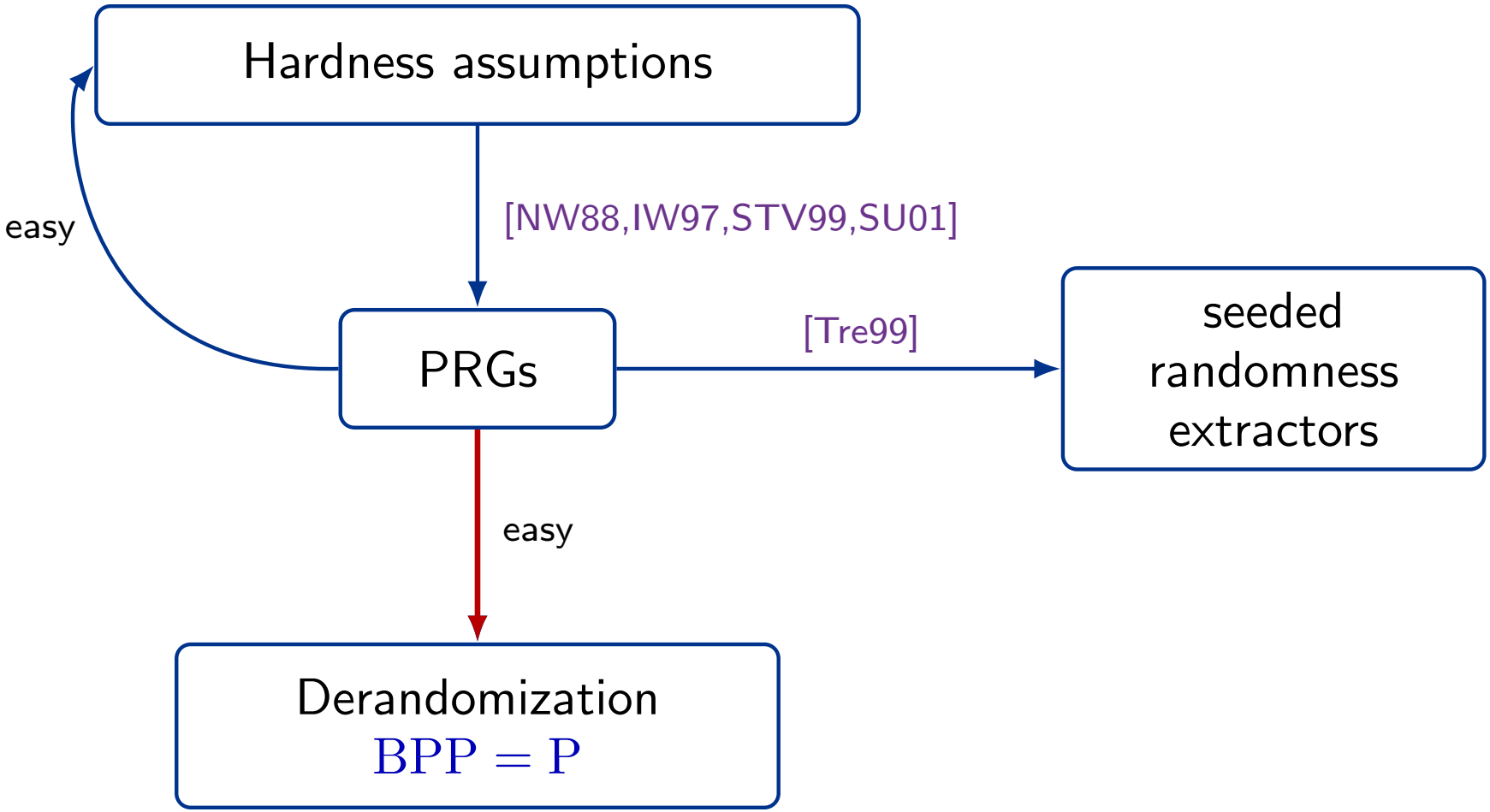
Big idea: "Randomness is in the eye of the beholder".

- Typically interested in a class of distinguishers. This talk: circuits of size m.
- Explicit PRGs: G is efficiently computable.
  - Strongly explicit: G computable in time polynomial in output length m.
  - Weakly explicit: G computable in time exponential in seed length r.
- In this talk: $m = 2^{\Theta(r)} \Rightarrow r = \Theta(\log m)$, two notions coincide.
- "Nisan-Wigderson setup": "running time of G" $= m^{O(1)} > m =$ "size of D".
- "Crypto setup": D is sufficiently strong to run G.
- Comment: Recent work replaces PRGs with "targeted PRGs".

Plan for talks



Plan for talks



PRGs $\Rightarrow$ BPP $=$ P

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

PRGs $\Rightarrow$ BPP $=$ P

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: Why do nonuniform circuits show up?

PRGs $\Rightarrow$ BPP $=$ P

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: Why do nonuniform circuits show up?

Proof: Let A be a randomized algorithm that runs in poly-time $m(n) = n^{O(1)}$.

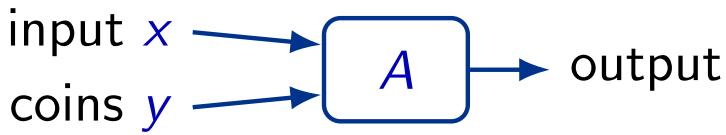
PRGs $\Rightarrow$ BPP = P

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP = P.

Question: Why do nonuniform circuits show up?

Proof: Let A be a randomized algorithm that runs in poly-time $m(n) = n^{O(1)}$.



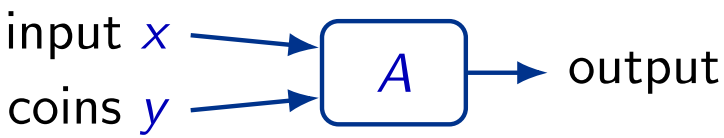
PRGs $\Rightarrow$ BPP = P

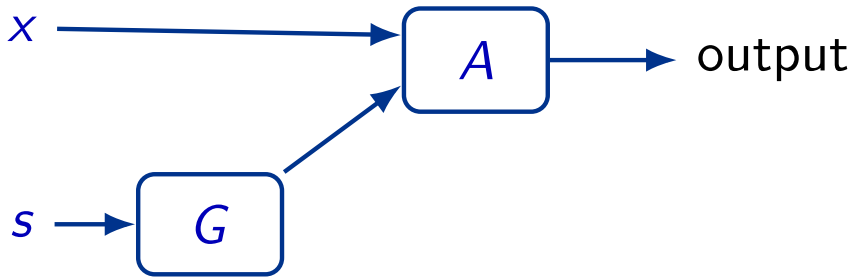
Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP = P.

Question: Why do nonuniform circuits show up?

Proof: Let A be a randomized algorithm that runs in poly-time $m(n) = n^{O(1)}$.





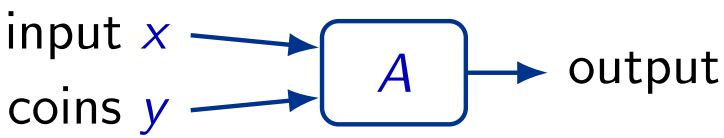
PRGs $\Rightarrow$ BPP = P

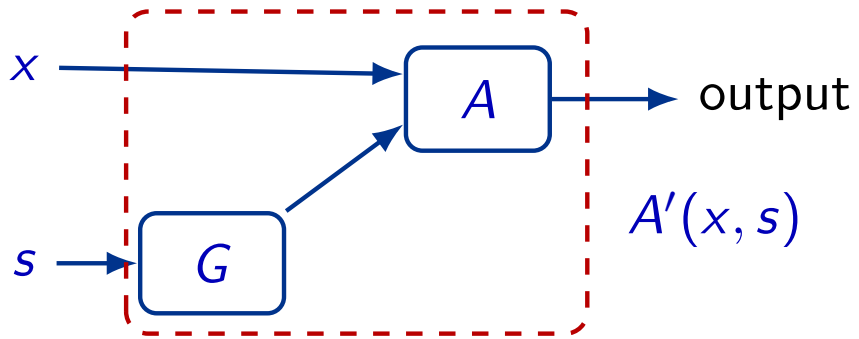
Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP = P.

Question: Why do nonuniform circuits show up?

Proof: Let A be a randomized algorithm that runs in poly-time $m(n) = n^{O(1)}$.





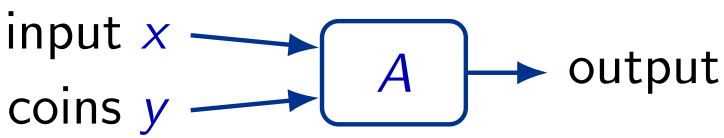
PRGs $\Rightarrow$ BPP = P

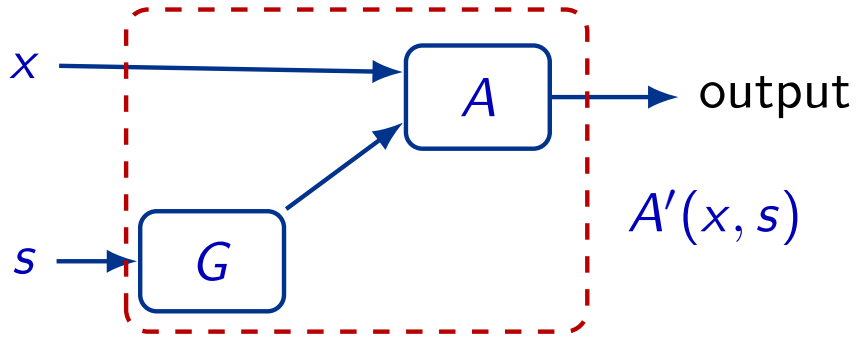
Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP = P.

Question: Why do nonuniform circuits show up?

Proof: Let A be a randomized algorithm that runs in poly-time $m(n) = n^{O(1)}$.





A' uses only $r = O(\log m)$ random bits, and runs in time $m^{O(1)}$ (explicitness).

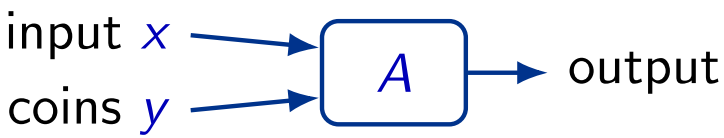
PRGs $\Rightarrow$ BPP = P

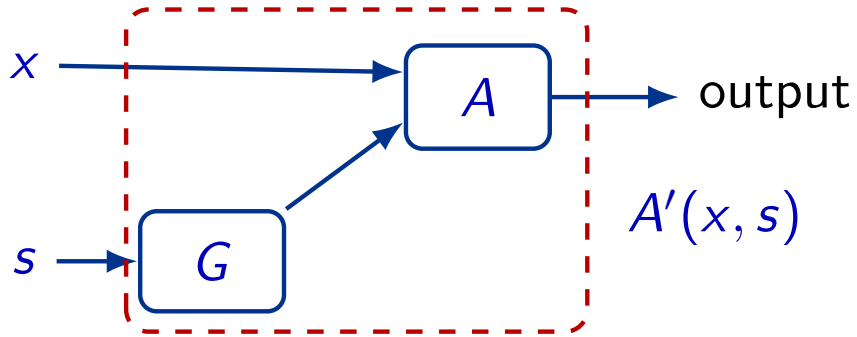
Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP = P.

Question: Why do nonuniform circuits show up?

Proof: Let A be a randomized algorithm that runs in poly-time $m(n) = n^{O(1)}$.





A' uses only $r = O(\log m)$ random bits, and runs in time $m^{O(1)}$ (explicitness).

Claim: A' is a randomized algorithm for the same language

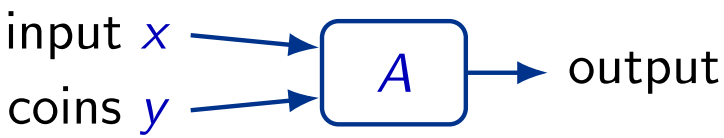
PRGs $\Rightarrow$ BPP = P

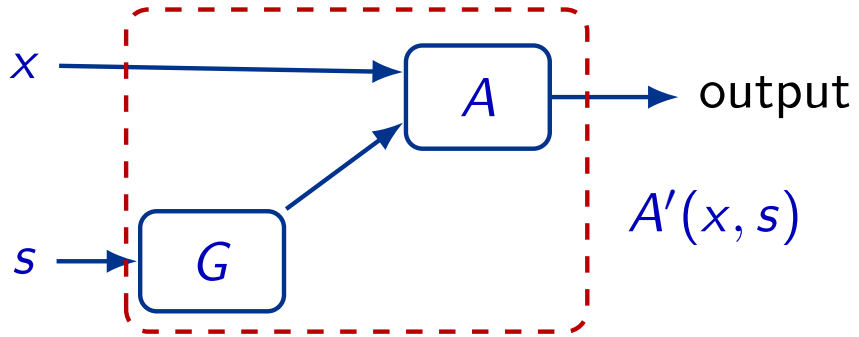
Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP = P.

Question: Why do nonuniform circuits show up?

Proof: Let A be a randomized algorithm that runs in poly-time $m(n) = n^{O(1)}$.





A' uses only $r = O(\log m)$ random bits, and runs in time $m^{O(1)}$ (explicitness).

Claim: A' is a randomized algorithm for the same language

Corollary: The deterministic algorithm $B(x) = \mathrm{maj}_{s\in\{0,1\}^r} A'(x,s)$ simulates $A'(x)$ in time $2^r \cdot m^{O(1)} = m^{O(1)} = n^{O(1)}$.

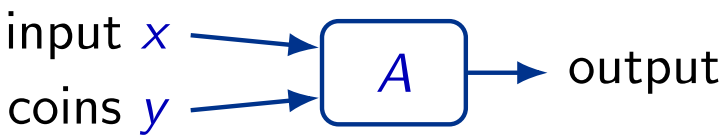
PRGs $\Rightarrow$ BPP = P

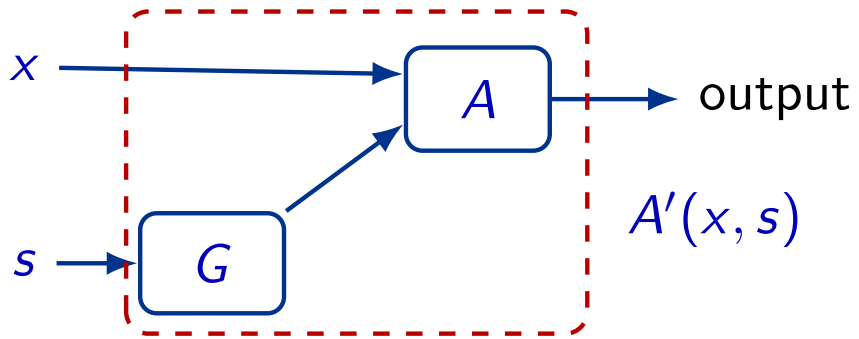
Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP = P.

Question: Why do nonuniform circuits show up?

Proof: Let A be a randomized algorithm that runs in poly-time $m(n) = n^{O(1)}$.





A' uses only $r = O(\log m)$ random bits, and runs in time $m^{O(1)}$ (explicitness).

Claim: A' is a randomized algorithm for the same language

Corollary: The deterministic algorithm $B(x) = \mathrm{maj}_{s\in\{0,1\}^r} A'(x,s)$ simulates $A'(x)$ in time $2^r \cdot m^{O(1)} = m^{O(1)} = n^{O(1)}$.

Proof of Claim: $\forall x \in \{0,1\}^n$, define the circuit $D_x(y) = A(x,y)$.

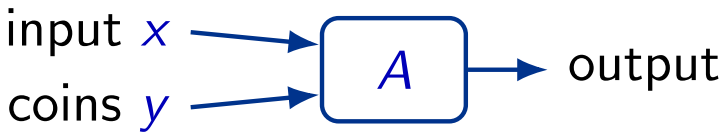
PRGs $\Rightarrow$ BPP = P

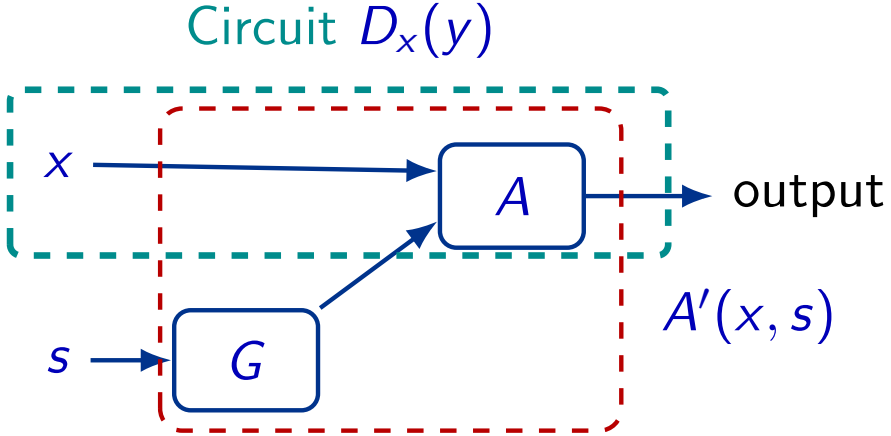
Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP = P.

Question: Why do nonuniform circuits show up?

Proof: Let A be a randomized algorithm that runs in poly-time $m(n) = n^{O(1)}$.





A' uses only $r = O(\log m)$ random bits, and runs in time $m^{O(1)}$ (explicitness).

Claim: A' is a randomized algorithm for the same language

Corollary: The deterministic algorithm $B(x) = \text{maj}_{s\in\{0,1\}^r} A'(x,s)$ simulates $A'(x)$ in time $2^r \cdot m^{O(1)} = m^{O(1)} = n^{O(1)}$.

Proof of Claim: $\forall x \in \{0,1\}^n$, define the circuit $D_x(y) = A(x,y)$.

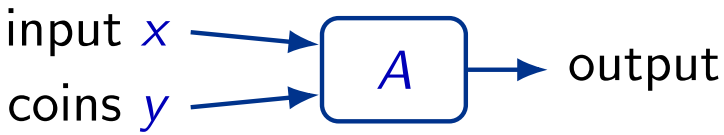
PRGs $\Rightarrow$ BPP = P

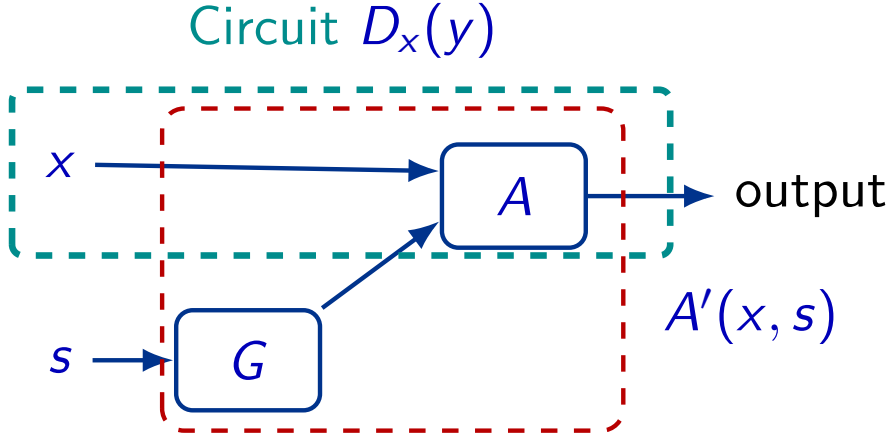
Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP = P.

Question: Why do nonuniform circuits show up?

Proof: Let A be a randomized algorithm that runs in poly-time $m(n) = n^{O(1)}$.





A' uses only $r = O(\log m)$ random bits, and runs in time $m^{O(1)}$ (explicitness).

Claim: A' is a randomized algorithm for the same language

Corollary: The deterministic algorithm $B(x) = \text{maj}_{s\in\{0,1\}^r} A'(x,s)$ simulates $A'(x)$ in time $2^r \cdot m^{O(1)} = m^{O(1)} = n^{O(1)}$.

Proof of Claim: $\forall x \in \{0,1\}^n$, define the circuit $D_x(y) = A(x,y)$.
size$(D_x) = m \Rightarrow G$ is a PRG for D_x (D_x doesn't distinguish y from $G(s)$).

PRGs $\Rightarrow$ BPP $=$ P (reflection: uniform derand)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: Why do nonuniform circuits show up?

PRGs $\Rightarrow$ BPP $=$ P (reflection: uniform derand)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: Why do nonuniform circuits show up?

Answer: We want the deterministic simulation $B(x)$ to work for every input x.

PRGs $\Rightarrow$ BPP $=$ P (reflection: uniform derand)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: Why do nonuniform circuits show up?

Answer: We want the deterministic simulation $B(x)$ to work for every input x. G needed to be a PRG for $D_x(y) = A(x,y)$ for every $x \in \{0,1\}^n$.

PRGs $\Rightarrow$ BPP $=$ P (reflection: uniform derand)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: Why do nonuniform circuits show up?

Answer: We want the deterministic simulation $B(x)$ to work for every input x.
G needed to be a PRG for $D_x(y) = A(x,y)$ for every $x \in \{0,1\}^n$.
No guarantee that x is feasibly generated, so treat x as nonuniformity.

PRGs $\Rightarrow$ BPP $=$ P (reflection: uniform derand)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: Why do nonuniform circuits show up?

Answer: We want the deterministic simulation $B(x)$ to work for every input x.
G needed to be a PRG for $D_x(y) = A(x,y)$ for every $x \in \{0,1\}^n$.
No guarantee that x is feasibly generated, so treat x as nonuniformity.

PRG against uniform adversaries suffices inputs are "feasibly generated".

PRGs $\Rightarrow$ BPP $=$ P (reflection: uniform derand)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: Why do nonuniform circuits show up?

Answer: We want the deterministic simulation $B(x)$ to work for every input x.
G needed to be a PRG for $D_x(y) = A(x,y)$ for every $x \in \{0,1\}^n$.
No guarantee that x is feasibly generated, so treat x as nonuniformity.

PRG against uniform adversaries suffices inputs are "feasibly generated".

Theorem (Informal)

Explicit ϵ*-PRG* $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for uniform algorithms running in time* m*, implies that* $\forall L \in$ BPP, $\exists$*deterministic poly-time algorithm* B *s.t. it is infeasible to generate an input* x *on which* B *errs.*

In some cases, one can obtain:*

$$\Pr_{x \leftarrow U_n}[B(x) \neq 1_L(x)] \leq O(\epsilon).$$

In fact, U_n *can sometimes be replaced by any efficiently samplable distribution.*

PRGs $\Rightarrow$ BPP $=$ P (reflection: uniform derand)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: Why do nonuniform circuits show up?

Answer: We want the deterministic simulation $B(x)$ to work for every input x.
G needed to be a PRG for $D_x(y) = A(x,y)$ for every $x \in \{0,1\}^n$.
No guarantee that x is feasibly generated, so treat x as nonuniformity.

PRG against uniform adversaries suffices inputs are "feasibly generated".

Theorem (Informal)

Explicit ϵ*-PRG* $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for uniform algorithms running in time* m*, implies that* $\forall L \in$ BPP, $\exists$*deterministic poly-time algorithm* B *s.t. it is infeasible to generate an input* x *on which* B *errs.*

In some cases, one can obtain:*

$$\Pr_{x \leftarrow U_n}[B(x) \neq 1_L(x)] \leq O(\epsilon).$$

In fact, U_n *can sometimes be replaced by any efficiently samplable distribution.*

Uniform derandomization is extensively studied [IW98,Kab00,Lu00,TV02,GW02, GST03,SU07,Sha09,KvMS09,CIS18,CRTY20,CT21,CRT22,BCT25...].

PRGs $\Rightarrow$ BPP $=$ P (reflection: uniform derand)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: Why do nonuniform circuits show up?

Answer: We want the deterministic simulation $B(x)$ to work for every input x. G needed to be a PRG for $D_x(y) = A(x,y)$ for every $x \in \{0,1\}^n$. No guarantee that x is feasibly generated, so treat x as nonuniformity.

PRG against uniform adversaries suffices inputs are "feasibly generated".

PRGs $\Rightarrow$ BPP $=$ P (reflection: uniform derand)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: Why do nonuniform circuits show up?

Answer: We want the deterministic simulation $B(x)$ to work for every input x. G needed to be a PRG for $D_x(y) = A(x,y)$ for every $x \in \{0,1\}^n$. No guarantee that x is feasibly generated, so treat x as nonuniformity.

PRG against uniform adversaries suffices inputs are "feasibly generated".
Example: nonuniform vs. uniform tradeoff.

PRGs $\Rightarrow$ BPP $=$ P (reflection: uniform derand)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: Why do nonuniform circuits show up?

Answer: We want the deterministic simulation $B(x)$ to work for every input x. G needed to be a PRG for $D_x(y) = A(x,y)$ for every $x \in \{0,1\}^n$. No guarantee that x is feasibly generated, so treat x as nonuniformity.

PRG against uniform adversaries suffices inputs are "feasibly generated".
Example: nonuniform vs. uniform tradeoff.

Theorem (BFNW92 nonuniform hardness vs. randomness tradeoff)

EXP $\not\subseteq$ P/poly $\Rightarrow$ BPP $\subseteq i.o. -$ SUBEXP.

PRGs $\Rightarrow$ BPP $=$ P (reflection: uniform derand)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: Why do nonuniform circuits show up?

Answer: We want the deterministic simulation $B(x)$ to work for every input x. G needed to be a PRG for $D_x(y) = A(x,y)$ for every $x \in \{0,1\}^n$. No guarantee that x is feasibly generated, so treat x as nonuniformity.

PRG against uniform adversaries suffices inputs are "feasibly generated".
Example: nonuniform vs. uniform tradeoff.

Theorem (BFNW92 nonuniform hardness vs. randomness tradeoff)

EXP $\not\subseteq$ P/poly $\Rightarrow$ BPP $\subseteq$ *i.o.* $-$ SUBEXP.

Theorem (IW98 uniform hardness vs. randomness tradeoff)

EXP $\not\subseteq$ BPP $\Rightarrow$ BPP $\subseteq$ *i.o.* $-$ *Heuristic* $-$ SUBEXP.

PRGs $\Rightarrow$ BPP $=$ P (reflection: uniform derand)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: Why do nonuniform circuits show up?

Answer: We want the deterministic simulation $B(x)$ to work for every input x. G needed to be a PRG for $D_x(y) = A(x,y)$ for every $x \in \{0,1\}^n$. No guarantee that x is feasibly generated, so treat x as nonuniformity.

PRG against uniform adversaries suffices inputs are "feasibly generated".
Example: nonuniform vs. uniform tradeoff.

Theorem (BFNW92 nonuniform hardness vs. randomness tradeoff)

EXP $\not\subseteq$ P/poly $\Rightarrow$ BPP $\subseteq$ *i.o.* $-$ SUBEXP.

Theorem (IW98 uniform hardness vs. randomness tradeoff)

EXP $\not\subseteq$ BPP $\Rightarrow$ BPP $\subseteq$ *i.o.* $-$ *Heuristic* $-$ SUBEXP.

Main point: weaker assumption and weaker conclusion

PRGs $\Rightarrow$ BPP $=$ P (reflection: uniform derand)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: Why do nonuniform circuits show up?

Answer: We want the deterministic simulation $B(x)$ to work for every input x. G needed to be a PRG for $D_x(y) = A(x,y)$ for every $x \in \{0,1\}^n$. No guarantee that x is feasibly generated, so treat x as nonuniformity.

PRG against uniform adversaries suffices inputs are "feasibly generated".
Example: nonuniform vs. uniform tradeoff.

Theorem (BFNW92 nonuniform hardness vs. randomness tradeoff)

EXP $\not\subseteq$ P/poly $\Rightarrow$ BPP $\subseteq$ *i.o.* $-$ SUBEXP.

Theorem (IW98 uniform hardness vs. randomness tradeoff)

EXP $\not\subseteq$ BPP $\Rightarrow$ BPP $\subseteq$ *i.o.* $-$ *Heuristic* $-$ SUBEXP.

Main point: weaker assumption and weaker conclusion

Recent work: Keep assumption uniform, shoot for derandomization on every input.

PRGs $\Rightarrow$ BPP $=$ P (reflection: targeted PRGs)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: PRGs suffice, but aren't they an overkill?

PRGs $\Rightarrow$ BPP $=$ P (reflection: targeted PRGs)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: PRGs suffice, but aren't they an overkill?

Intuition: When deterministically simulating randomized algorithm $A(x, r)$ on a given input x, we know both "targets" A and x.

PRGs $\Rightarrow$ BPP $=$ P (reflection: targeted PRGs)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: PRGs suffice, but aren't they an overkill?

Intuition: When deterministically simulating randomized algorithm $A(x, r)$ on a given input x, we know both "targets" A and x.

Observation: We can weaken the definition of a PRG, to a "targeted PRG" that is supplied with the "targets" A and x that it needs to fool.

PRGs $\Rightarrow$ BPP $=$ P (reflection: targeted PRGs)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: PRGs suffice, but aren't they an overkill?

Intuition: When deterministically simulating randomized algorithm $A(x, r)$ on a given input x, we know both "targets" A and x.

Observation: We can weaken the definition of a PRG, to a "targeted PRG" that is supplied with the "targets" A and x that it needs to fool.

This was observed (and sometimes used) even very early, but we didn't know how to strongly take advantage of this fact.

PRGs $\Rightarrow$ BPP $=$ P (reflection: targeted PRGs)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: PRGs suffice, but aren't they an overkill?

Intuition: When deterministically simulating randomized algorithm $A(x, r)$ on a given input x, we know both "targets" A and x.

Observation: We can weaken the definition of a PRG, to a "targeted PRG" that is supplied with the "targets" A and x that it needs to fool.

This was observed (and sometimes used) even very early, but we didn't know how to strongly take advantage of this fact.

Recently,there is an exciting line of work: [Gol11,CT21,LP22,LP23,vMS23,CTW23,DT23,DPT24,BCT25] that construct targeted PRGs from seemingly weaker assumptions to give "instance-wise" derandomization.

PRGs $\Rightarrow$ BPP $=$ P (reflection: targeted PRGs)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: PRGs suffice, but aren't they an overkill?

Intuition: When deterministically simulating randomized algorithm $A(x, r)$ on a given input x, we know both "targets" A and x.

Observation: We can weaken the definition of a PRG, to a "targeted PRG" that is supplied with the "targets" A and x that it needs to fool.

This was observed (and sometimes used) even very early, but we didn't know how to strongly take advantage of this fact.

Recently,there is an exciting line of work: [Gol11,CT21,LP22,LP23,vMS23,CTW23,DT23,DPT24,BCT25] that construct targeted PRGs from seemingly weaker assumptions to give "instance-wise" derandomization.

I will not cover this. Many of these works rely on the machinery that I will cover.

PRGs $\Rightarrow$ BPP $=$ P (reflection: targeted PRGs)

Theorem

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for circuits of size* $m \Rightarrow$ BPP $=$ P.

Question: PRGs suffice, but aren't they an overkill?

Intuition: When deterministically simulating randomized algorithm $A(x,r)$ on a given input x, we know both "targets" A and x.

Observation: We can weaken the definition of a PRG, to a "targeted PRG" that is supplied with the "targets" A and x that it needs to fool.

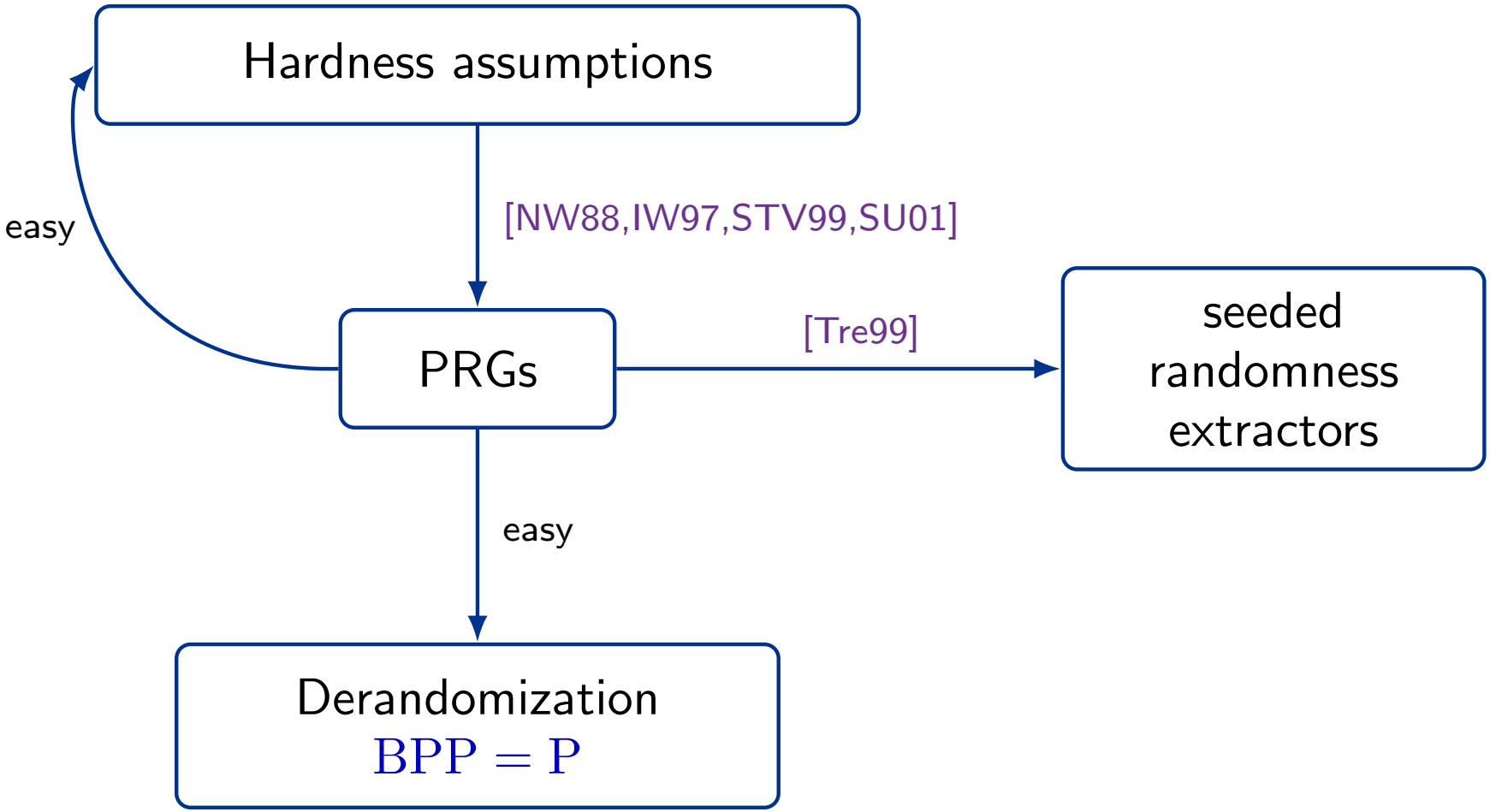
This was observed (and sometimes used) even very early, but we didn't know how to strongly take advantage of this fact.

Recently,there is an exciting line of work: [Gol11,CT21,LP22,LP23,vMS23,CTW23,DT23,DPT24,BCT25] that construct targeted PRGs from seemingly weaker assumptions to give "instance-wise" derandomization.

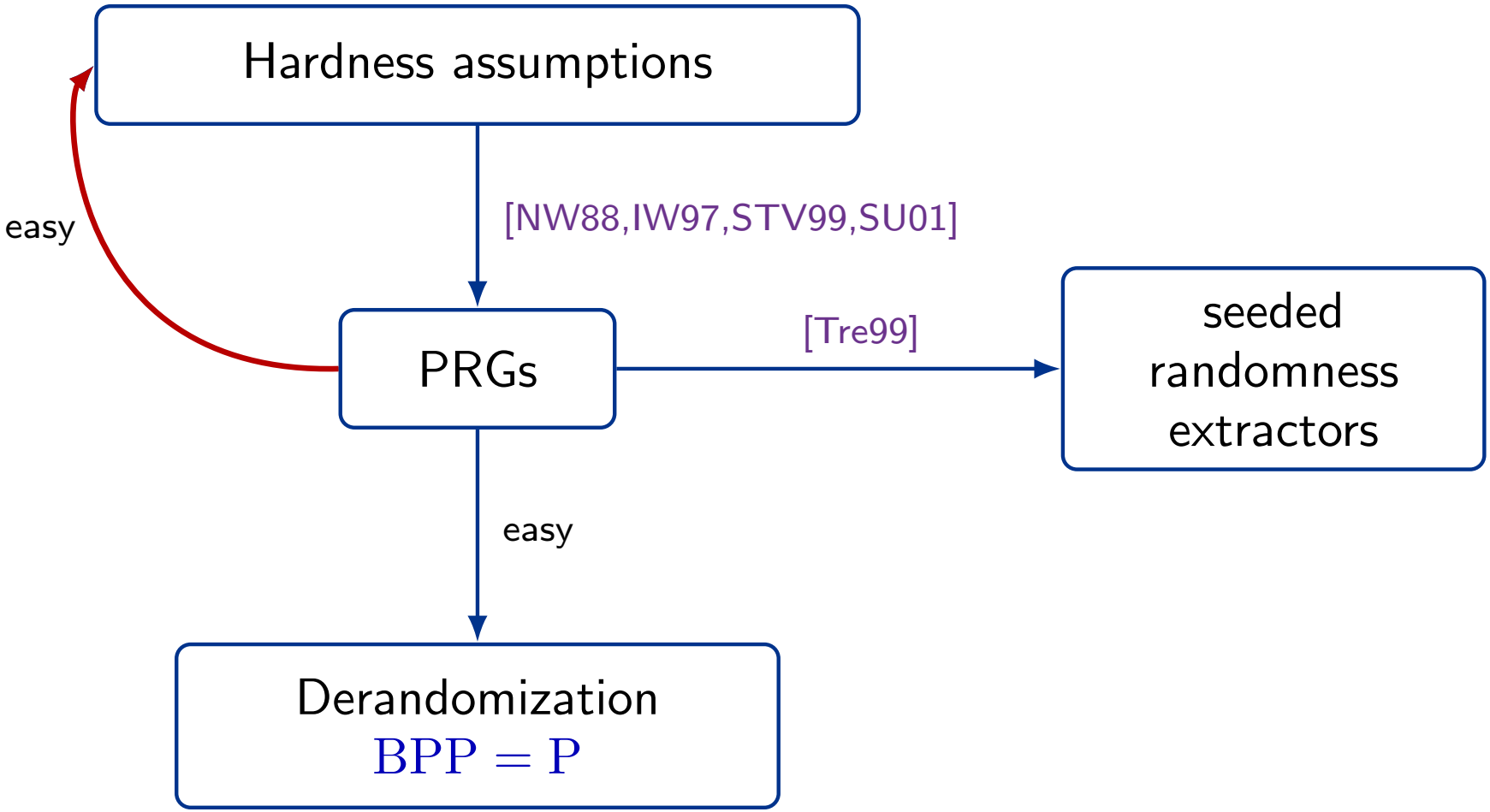
I will not cover this. Many of these works rely on the machinery that I will cover.

Comment: Some applications of PRGs (other than derandomization) require the full power of the definition, and targetd PRGs do not suffice.

Plan for talks



Plan for talks



Hardness assumptions

Definition (Hard functions)

$f : \{0,1\}^n \to \{0,1\}$ is

- s-hard if $\forall$size s circuit C, $\exists x \in \{0,1\}^n$ s.t. $C(x) \neq f(x)$.
- s-average-case-hard if $\forall$size s circuit C, $\Pr_{x \leftarrow U_n}[C(x) = f(x)] \leq \frac{1}{2} + \frac{1}{s}$.

Hardness assumptions

Definition (Hard functions)

$f : \{0,1\}^n \to \{0,1\}$ is

- s-hard if $\forall$size s circuit C, $\exists x \in \{0,1\}^n$ s.t. $C(x) \neq f(x)$.
- s-average-case-hard if $\forall$size s circuit C, $\Pr_{x \leftarrow U_n}[C(x) = f(x)] \leq \frac{1}{2} + \frac{1}{s}$.

Definition (Hardness assumptions)

- E is hard for exponential size circuits if $\exists \beta > 0$, and $\exists L \in \mathrm{E} = \mathrm{DTIME}(2^{O(n)})$ s.t. $\forall n$, 1_L is $2^{\beta n}$-hard (on inputs of length n).
- E is average-case hard for exponential size circuits if $\exists \beta > 0$, and $\exists L \in \mathrm{E} = \mathrm{DTIME}(2^{O(n)})$ s.t. $\forall n$, 1_L is $2^{\beta n}$-average-case-hard.

Hardness assumptions

Definition (Hard functions)

$f : \{0,1\}^n \to \{0,1\}$ is

- s-hard if $\forall$size s circuit C, $\exists x \in \{0,1\}^n$ s.t. $C(x) \neq f(x)$.
- s-average-case-hard if $\forall$size s circuit C, $\Pr_{x \leftarrow U_n}[C(x) = f(x)] \leq \frac{1}{2} + \frac{1}{s}$.

Definition (Hardness assumptions)

- E is hard for exponential size circuits if $\exists \beta > 0$, and $\exists L \in \mathrm{E} = \mathrm{DTIME}(2^{O(n)})$ s.t. $\forall n$, 1_L is $2^{\beta n}$-hard (on inputs of length n).
- E is average-case hard for exponential size circuits if $\exists \beta > 0$, and $\exists L \in \mathrm{E} = \mathrm{DTIME}(2^{O(n)})$ s.t. $\forall n$, 1_L is $2^{\beta n}$-average-case-hard.

Theorem (PRGs $\Rightarrow$ worst-case hardness (appears as an exercise))

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for size* m *circuits* $\Rightarrow$
E *is hard for exponential size circuits.*

Hardness assumptions

Definition (Hard functions)

$f : \{0,1\}^n \to \{0,1\}$ is

- s-hard if $\forall$size s circuit C, $\exists x \in \{0,1\}^n$ s.t. $C(x) \neq f(x)$.
- s-average-case-hard if $\forall$size s circuit C, $\Pr_{x \leftarrow U_n}[C(x) = f(x)] \leq \frac{1}{2} + \frac{1}{s}$.

Definition (Hardness assumptions)

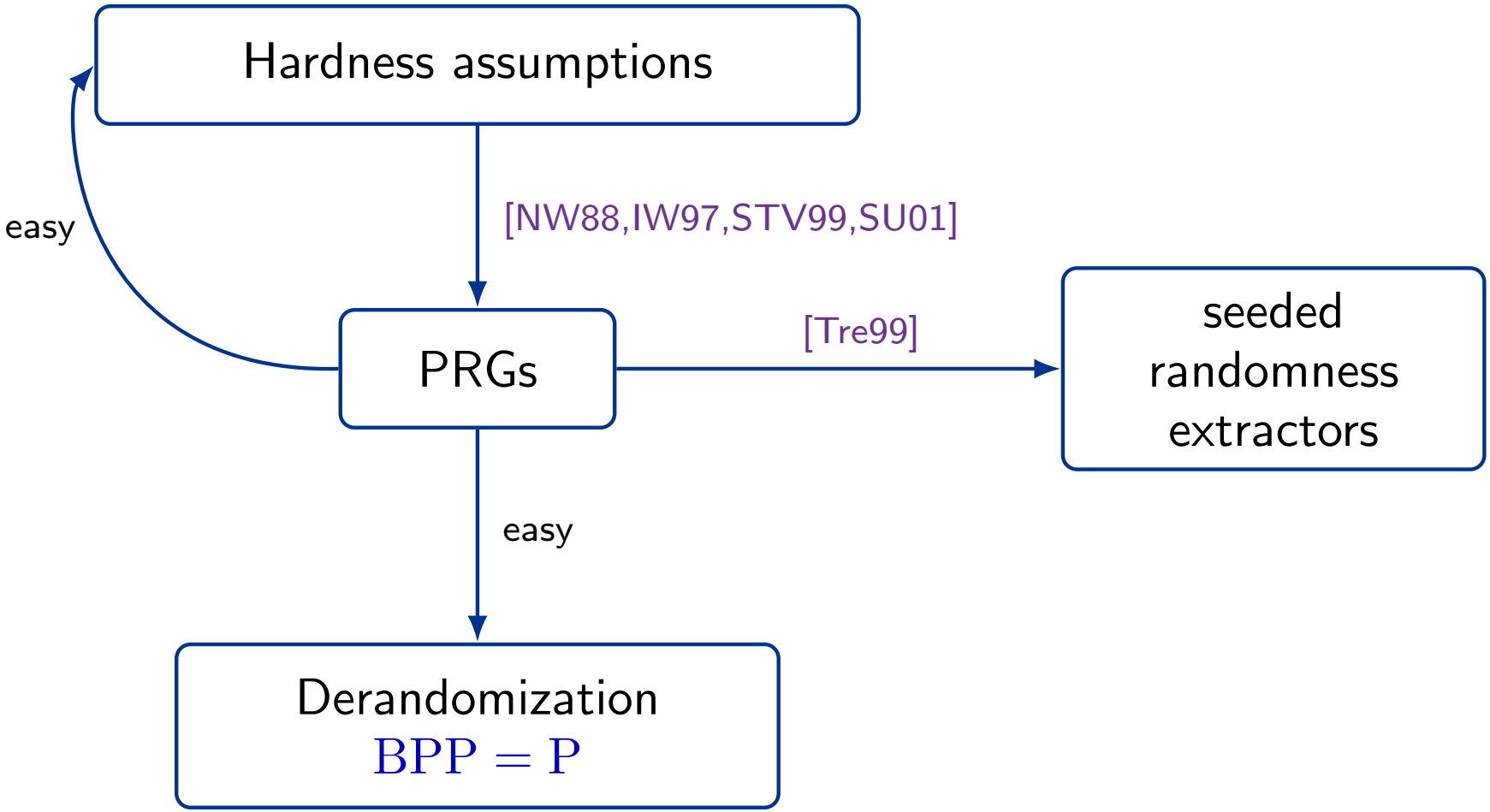
- E is hard for exponential size circuits if $\exists \beta > 0$, and $\exists L \in \mathsf{E} = \mathsf{DTIME}(2^{O(n)})$ s.t. $\forall n$, 1_L is $2^{\beta n}$-hard (on inputs of length n).
- E is average-case hard for exponential size circuits if $\exists \beta > 0$, and $\exists L \in \mathsf{E} = \mathsf{DTIME}(2^{O(n)})$ s.t. $\forall n$, 1_L is $2^{\beta n}$-average-case-hard.

Theorem (PRGs $\Rightarrow$ worst-case hardness (appears as an exercise))

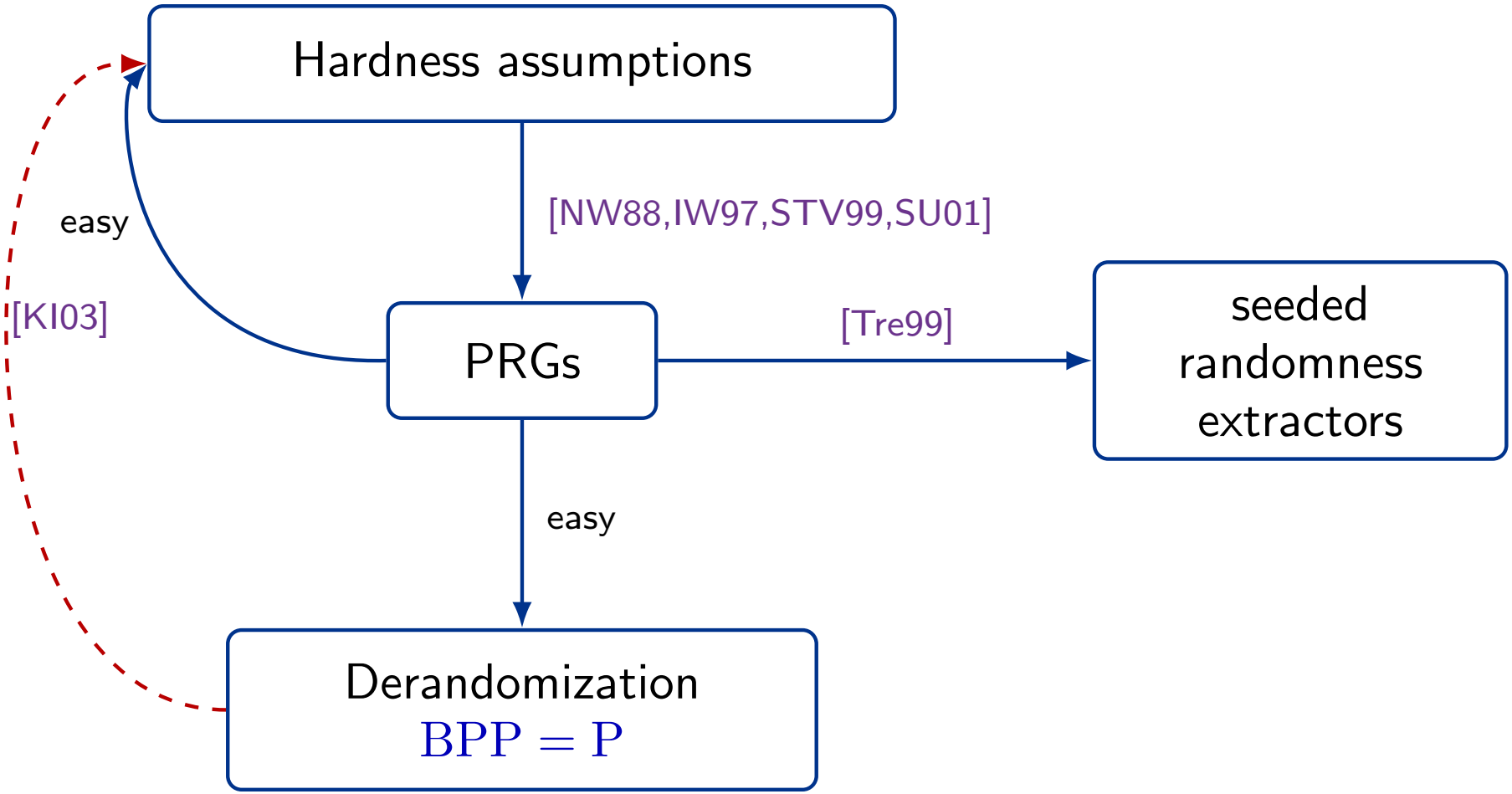
Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *for size* m *circuits* $\Rightarrow$ E *is hard for exponential size circuits.*

We don't have such lower bounds $\Rightarrow$ We cannot expect to construct explicit PRGs, unless we assume that E is hard for exponential size circuits.

Does derandomization imply hardness assumptions?

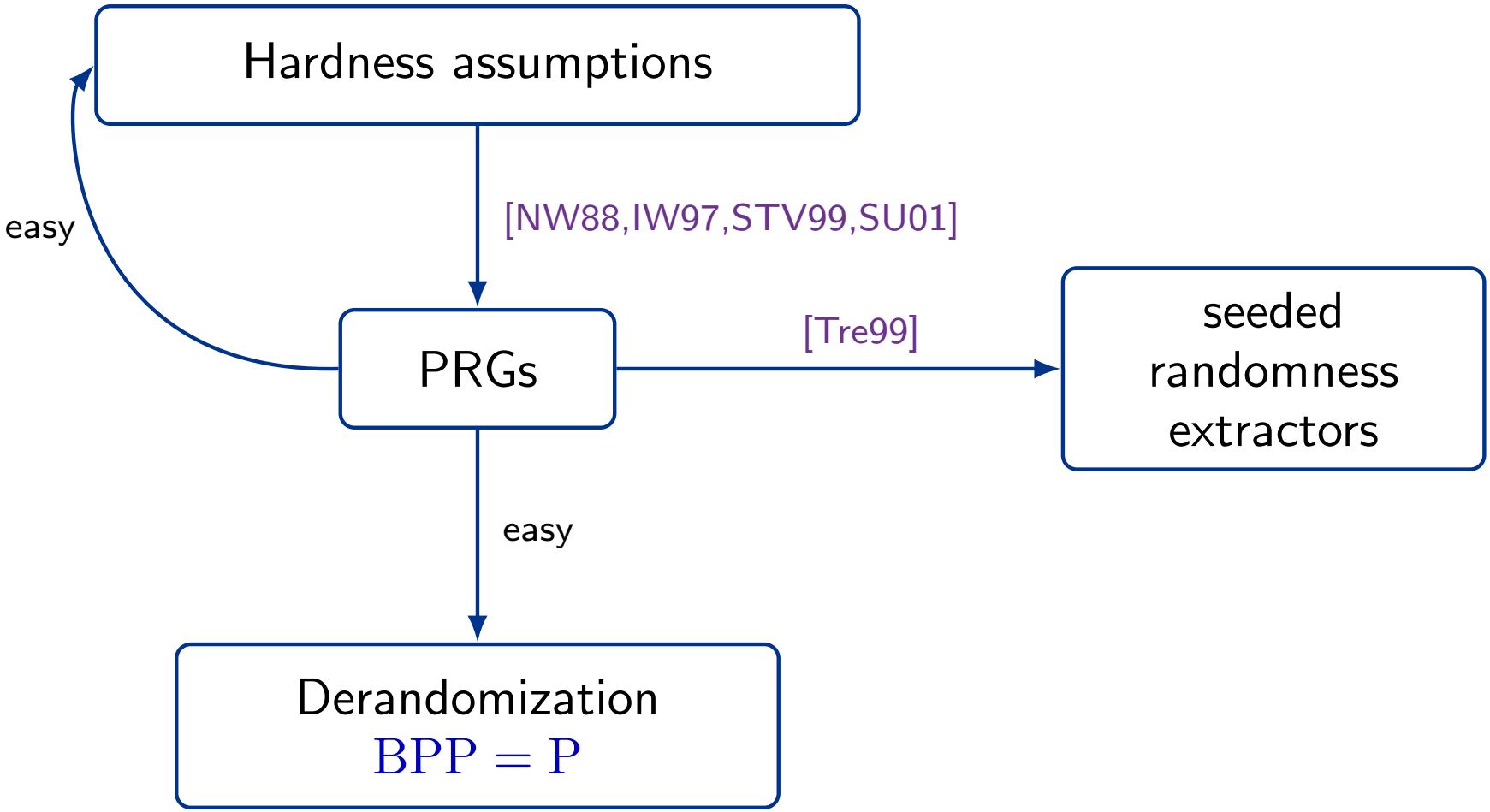


Does derandomization imply hardness assumptions?

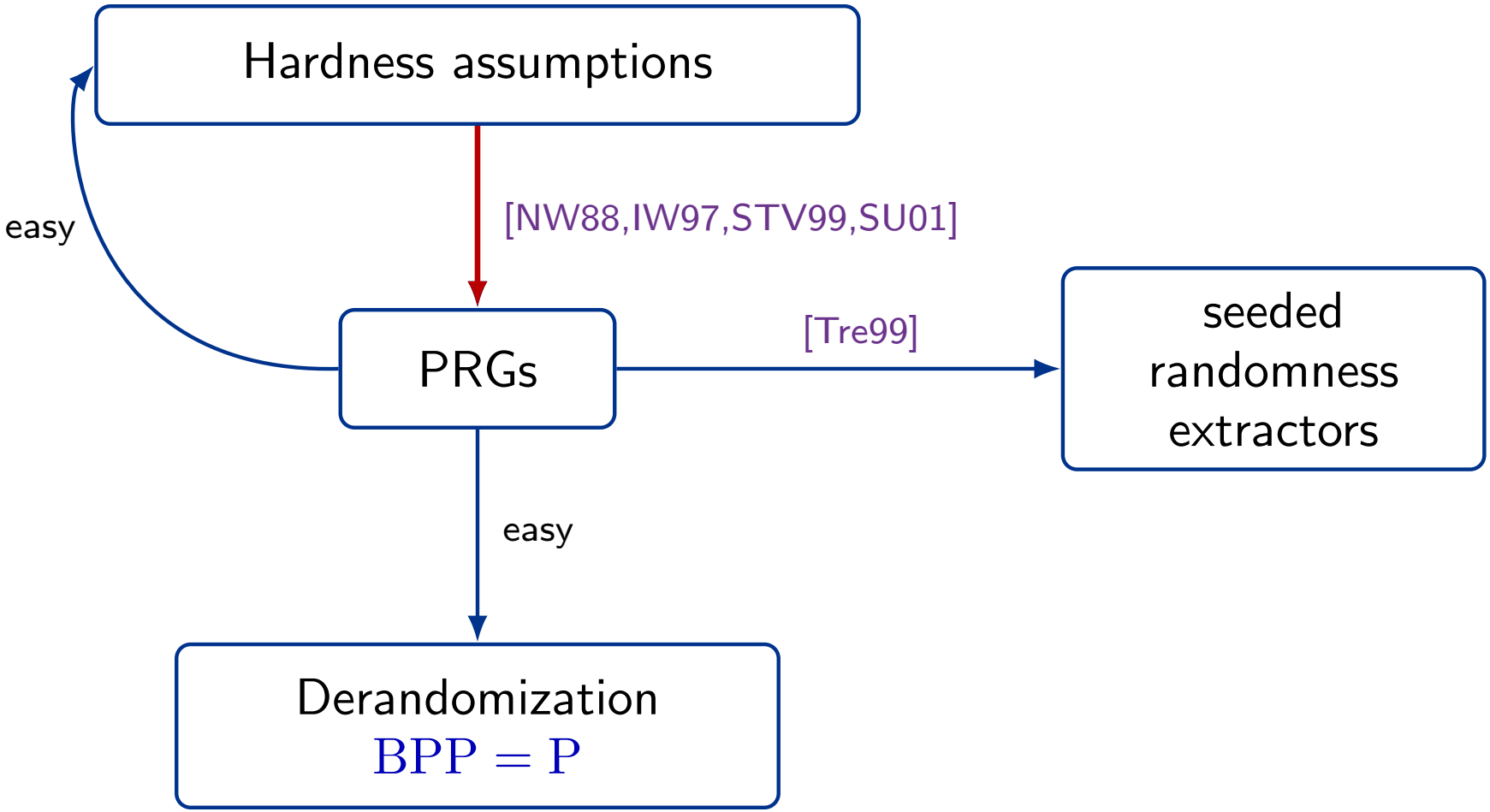


* The hardness assumption implied by $\mathrm{BPP} = \mathrm{P}$ in [KI03] is different and less clean than the one we considered.

Plan for talks



Plan for talks



Hardness assumptions $\Rightarrow$ PRGs

Theorem (IW97)

E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.

Hardness assumptions $\Rightarrow$ PRGs

Theorem (IW97)

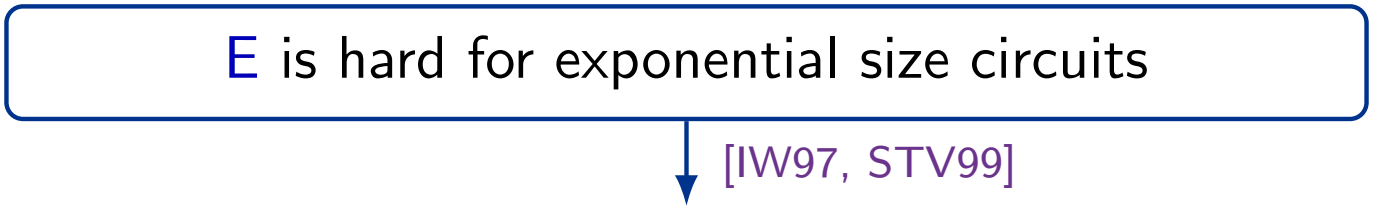
E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.

E is hard for exponential size circuits

Hardness assumptions $\Rightarrow$ PRGs

Theorem (IW97)

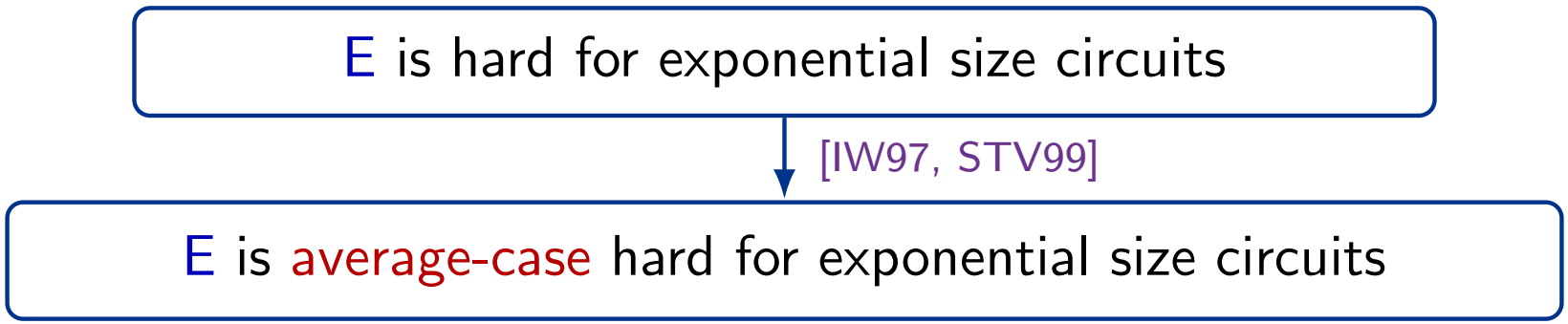
E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.



Hardness assumptions $\Rightarrow$ PRGs

Theorem (IW97)

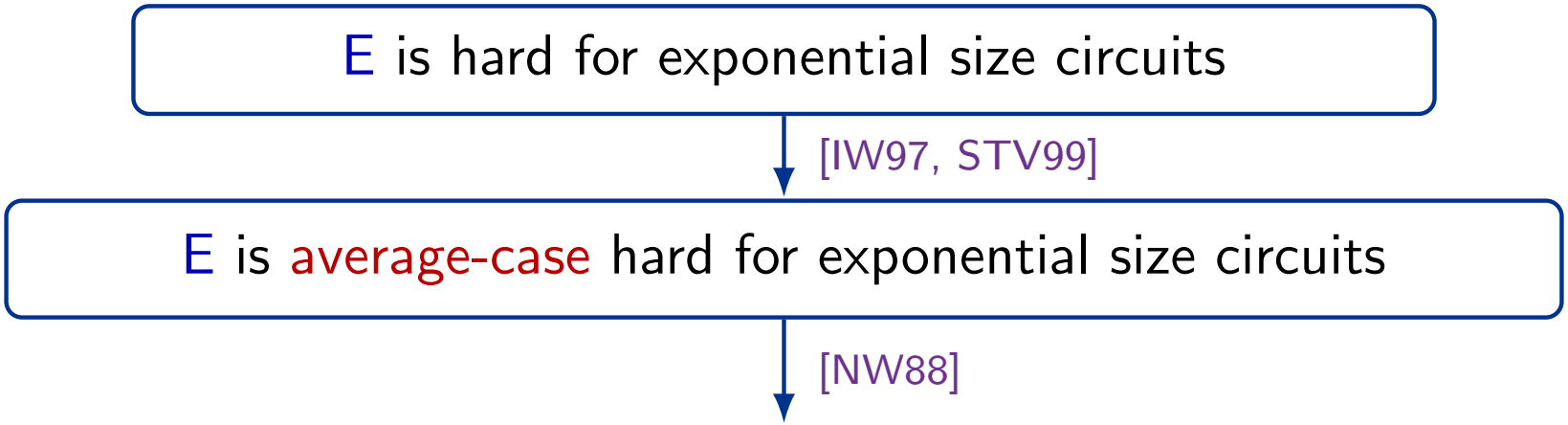
E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.



Hardness assumptions $\Rightarrow$ PRGs

Theorem (IW97)

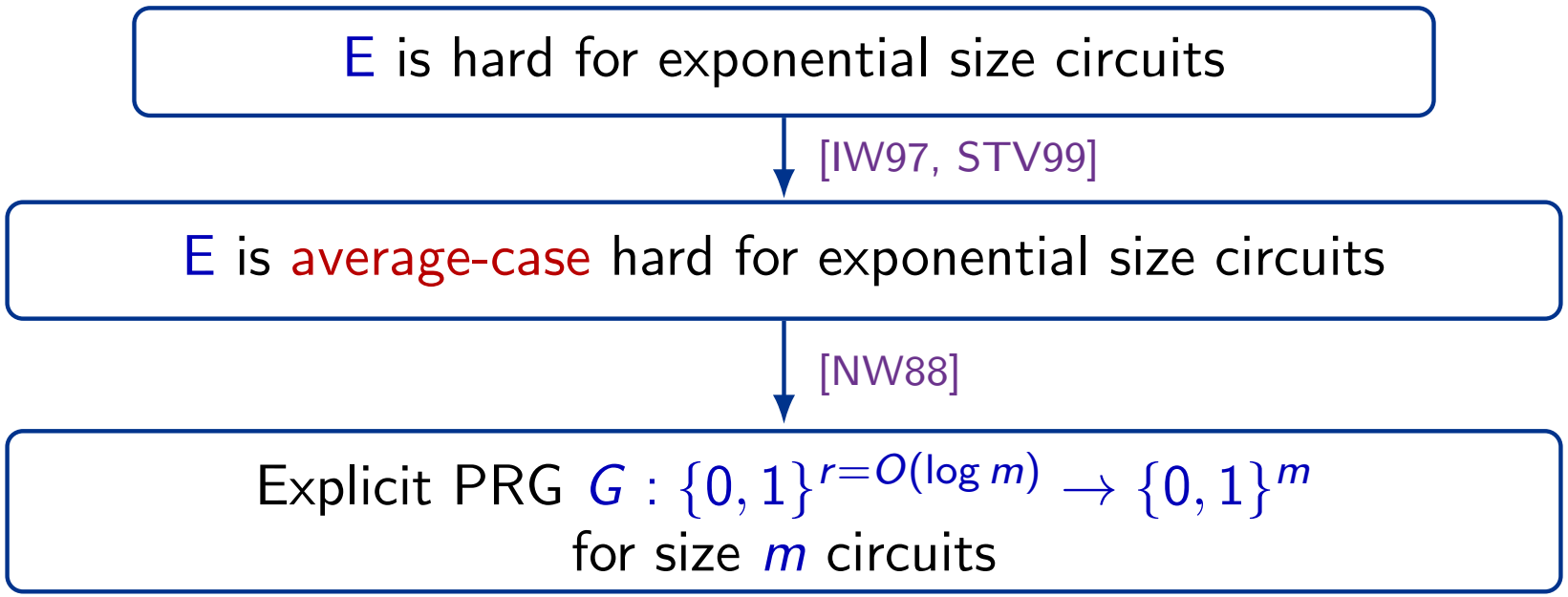
E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.



Hardness assumptions $\Rightarrow$ PRGs

Theorem (IW97)

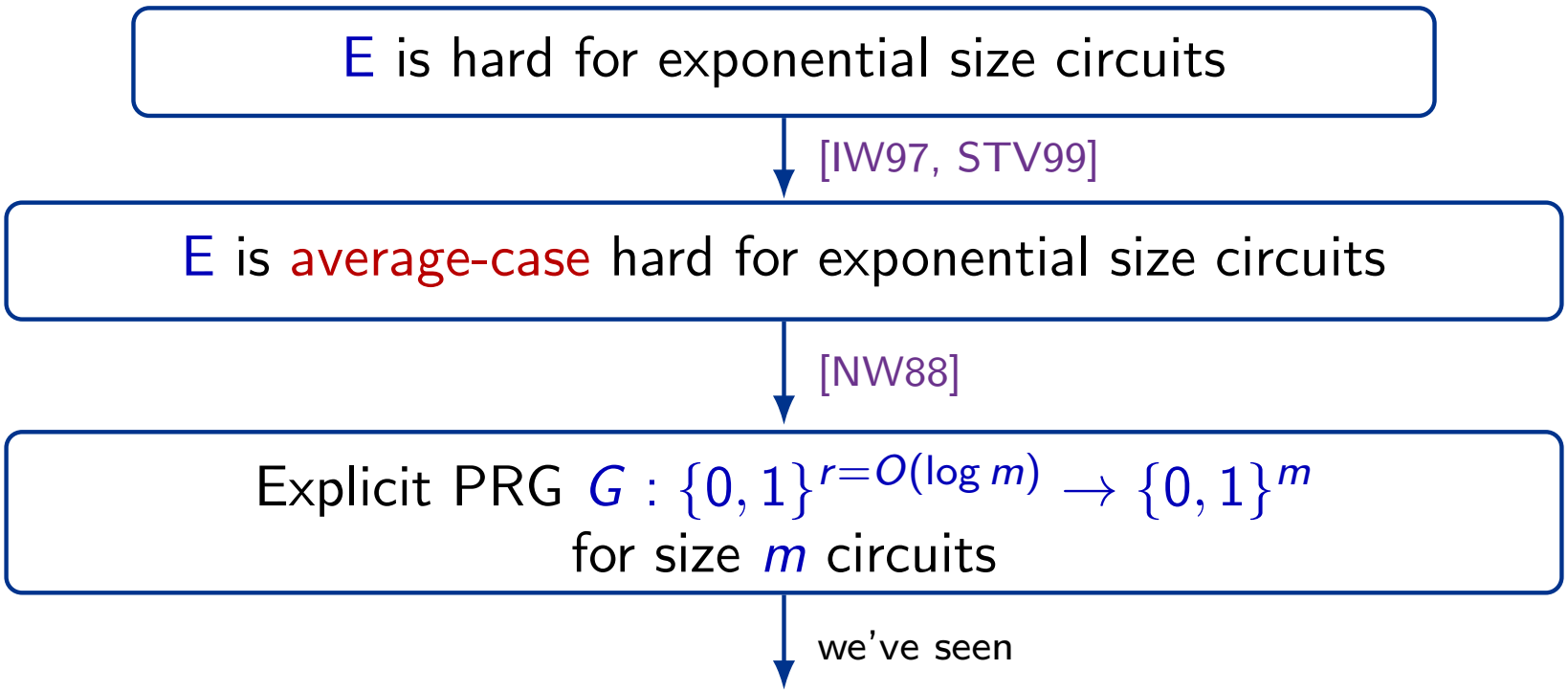
E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.



Hardness assumptions $\Rightarrow$ PRGs

Theorem (IW97)

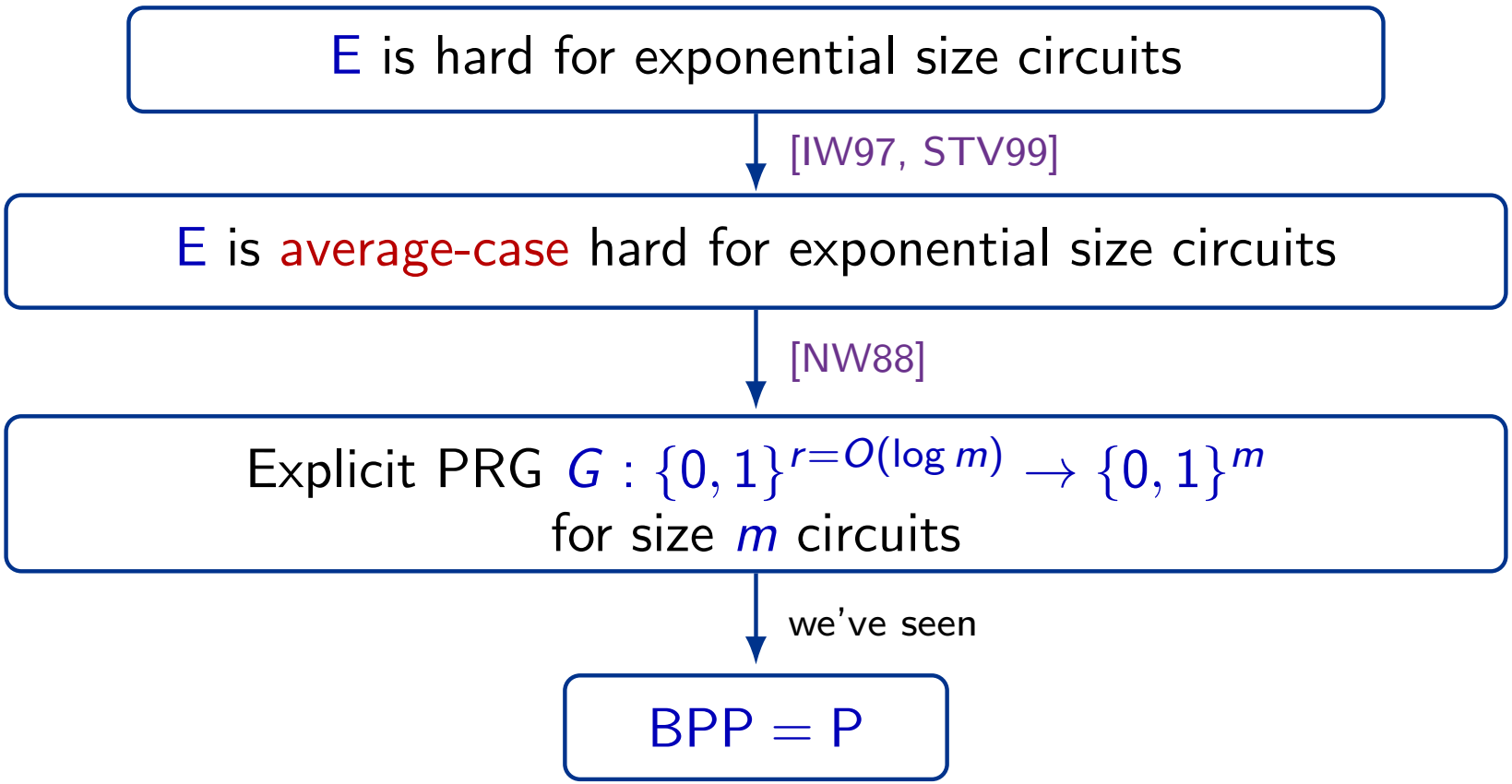
E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.



Hardness assumptions $\Rightarrow$ PRGs

Theorem (IW97)

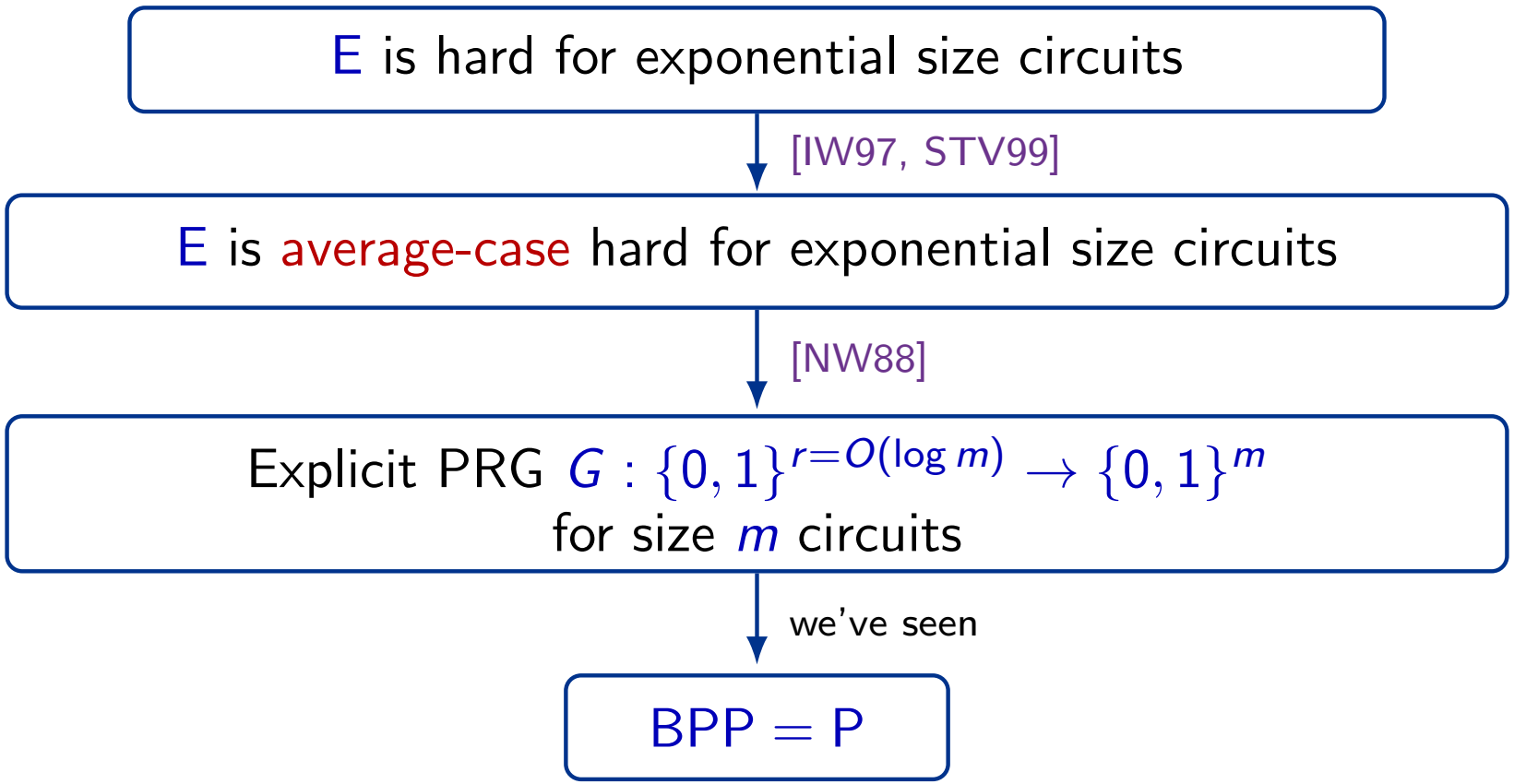
E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.



Hardness $\Rightarrow$ PRGs (NW PRG)

Theorem (IW97)

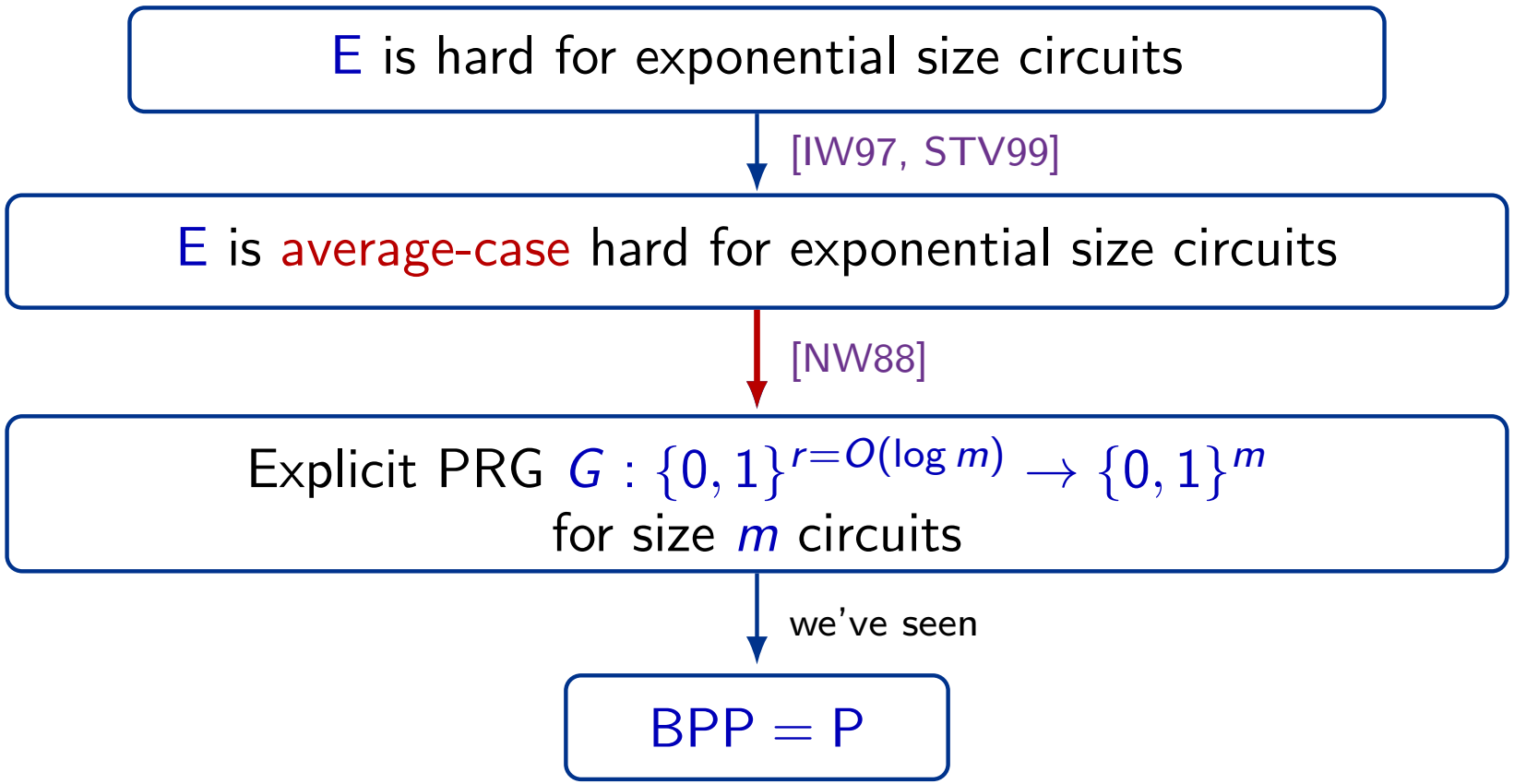
E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.



Hardness $\Rightarrow$ PRGs (NW PRG)

Theorem (IW97)

E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.



PRGs from average-case hardness

E is average-case hard for exponential size circuits

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ for size m circuits

PRGs from average-case hardness

E is average-case hard for exponential size circuits

↓ goal

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ for size m circuits

Idea: Use assumption on input length $\ell = O(\log m)$

PRGs from average-case hardness

E is average-case hard for exponential size circuits

↓ goal

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ for size m circuits

Idea: Use assumption on input length $\ell = O(\log m) \Rightarrow$

PRGs from average-case hardness

E is average-case hard for exponential size circuits

↓ goal

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ for size m circuits

Idea: Use assumption on input length $\ell = O(\log m) \Rightarrow f : \{0,1\}^\ell \to \{0,1\}$ s.t.

PRGs from average-case hardness

E is average-case hard for exponential size circuits

↓ goal

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ for size m circuits

Idea: Use assumption on input length $\ell = O(\log m) \Rightarrow f : \{0,1\}^\ell \to \{0,1\}$ s.t.

- f is computable in time $2^{O(\ell)} = m^{O(1)}$.

PRGs from average-case hardness

E is average-case hard for exponential size circuits

↓ goal

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ for size m circuits

Idea: Use assumption on input length $\ell = O(\log m) \Rightarrow f : \{0,1\}^\ell \to \{0,1\}$ s.t.

- f is computable in time $2^{O(\ell)} = m^{O(1)}$.
- f is $2^{\beta\ell} = m$-average-case hard.

PRGs from average-case hardness

E is average-case hard for exponential size circuits

↓ goal

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ for size m circuits

Idea: Use assumption on input length $\ell = O(\log m) \Rightarrow f : \{0,1\}^\ell \to \{0,1\}$ s.t.

- f is computable in time $2^{O(\ell)} = m^{O(1)}$.
- f is $2^{\beta\ell} = m$-average-case hard.

This explains why "exponential" comes up in the assumptions.

PRGs from average-case hardness

E is average-case hard for exponential size circuits

↓ goal

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ for size m circuits

Idea: Use assumption on input length $\ell = O(\log m) \Rightarrow f : \{0,1\}^\ell \to \{0,1\}$ s.t.

- f is computable in time $2^{O(\ell)} = m^{O(1)}$.
- f is $2^{\beta\ell} = m$-average-case hard.

This explains why "exponential" comes up in the assumptions.

Idea: Let's set $r = \ell = O(\log m)$, and shoot for a PRG $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$.

PRGs from average-case hardness

E is average-case hard for exponential size circuits

↓ goal

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ for size m circuits

Idea: Use assumption on input length $\ell = O(\log m) \Rightarrow f : \{0,1\}^\ell \to \{0,1\}$ s.t.

- f is computable in time $2^{O(\ell)} = m^{O(1)}$.
- f is $2^{\beta\ell} = m$-average-case hard.

This explains why "exponential" comes up in the assumptions.

Idea: Let's set $r = \ell = O(\log m)$, and shoot for a PRG $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$.

Construction: $G(x) = x, f(x)$.

PRGs from average-case hardness

E is average-case hard for exponential size circuits

↓ goal

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ for size m circuits

Idea: Use assumption on input length $\ell = O(\log m) \Rightarrow f : \{0,1\}^\ell \to \{0,1\}$ s.t.

- f is computable in time $2^{O(\ell)} = m^{O(1)}$.
- f is $2^{\beta\ell} = m$-average-case hard.

This explains why "exponential" comes up in the assumptions.

Idea: Let's set $r = \ell = O(\log m)$, and shoot for a PRG $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$.

Construction: $G(x) = x, f(x)$. (explicit, because f is computable in time $m^{O(1)}$).

PRGs from average-case hardness

E is average-case hard for exponential size circuits

↓ goal

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ for size m circuits

Idea: Use assumption on input length $\ell = O(\log m) \Rightarrow f : \{0,1\}^\ell \to \{0,1\}$ s.t.

- f is computable in time $2^{O(\ell)} = m^{O(1)}$.
- f is $2^{\beta\ell} = m$-average-case hard.

This explains why "exponential" comes up in the assumptions.

Idea: Let's set $r = \ell = O(\log m)$, and shoot for a PRG $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$.

Construction: $G(x) = x, f(x)$. (explicit, because f is computable in time $m^{O(1)}$).

Intuition: f average-case hard for size m circuits is necessary.

PRGs from average-case hardness

E is average-case hard for exponential size circuits

↓ goal

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ for size m circuits

Idea: Use assumption on input length $\ell = O(\log m) \Rightarrow f : \{0,1\}^\ell \to \{0,1\}$ s.t.

- f is computable in time $2^{O(\ell)} = m^{O(1)}$.
- f is $2^{\beta\ell} = m$-average-case hard.

This explains why "exponential" comes up in the assumptions.

Idea: Let's set $r = \ell = O(\log m)$, and shoot for a PRG $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$.

Construction: $G(x) = x, f(x)$. (explicit, because f is computable in time $m^{O(1)}$).

Intuition: f average-case hard for size m circuits is necessary. but is it sufficient?

PRGs from average-case hardness

E is average-case hard for exponential size circuits

↓ goal

Explicit PRG $G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ for size m circuits

Idea: Use assumption on input length $\ell = O(\log m) \Rightarrow f : \{0,1\}^\ell \to \{0,1\}$ s.t.

- f is computable in time $2^{O(\ell)} = m^{O(1)}$.
- f is $2^{\beta\ell} = m$-average-case hard.

This explains why "exponential" comes up in the assumptions.

Idea: Let's set $r = \ell = O(\log m)$, and shoot for a PRG $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$.

Construction: $G(x) = x, f(x)$. (explicit, because f is computable in time $m^{O(1)}$).

Intuition: f average-case hard for size m circuits is necessary. but is it sufficient?

Theorem (Distinguisher to Predictor (toy version appears as an exercise))

$G : \{0,1\}^r \to \{0,1\}^m$ is a PRG for size m circuits,
if for every $i \in [m]$, and every size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$,

$$\Pr_{x \leftarrow U_r}[P(G(x)_{1,\ldots,i-1}) = G(x)_i] \leq \frac{1}{2} + \frac{1}{10m}.$$

PRGs from average-case hardness (1 bit stretch)

Theorem (Distinguisher to Predictor)

$G : \{0,1\}^r \to \{0,1\}^m$ is a PRG for size m circuits,
if for every $i \in [m]$, and every size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$,

$$\Pr_{x \leftarrow U_r}[P(G(x)_{1,\ldots,i-1}) = G(x)_i] \leq \frac{1}{2} + \frac{1}{10m}.$$

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard
$\Rightarrow$ $G(x) = x, f(x)$ is a PRG for circuits of size m.

Proof.

PRGs from average-case hardness (1 bit stretch)

Theorem (Distinguisher to Predictor)

$G : \{0,1\}^r \to \{0,1\}^m$ is a PRG for size m circuits,
if for every $i \in [m]$, and every size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$,

$$\Pr_{x \leftarrow U_r}[P(G(x)_{1,\ldots,i-1}) = G(x)_i] \leq \frac{1}{2} + \frac{1}{10m}.$$

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard
$\Rightarrow$ $G(x) = x, f(x)$ is a PRG for circuits of size m.

Proof.

By Theorem, if G is not a PRG, then $\exists i \in [\ell + 1]$, $\exists$size m circuit
$P : \{0,1\}^{i-1} \to \{0,1\}$, s.t. $\Pr_{x \leftarrow U_\ell}[P(G(x)_{1,..i-1}) = G(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

PRGs from average-case hardness (1 bit stretch)

Theorem (Distinguisher to Predictor)

$G : \{0,1\}^r \to \{0,1\}^m$ is a PRG for size m circuits,
if for every $i \in [m]$, and every size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$,

$$\Pr_{x \leftarrow U_r}[P(G(x)_{1,\ldots,i-1}) = G(x)_i] \leq \frac{1}{2} + \frac{1}{10m}.$$

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard
$\Rightarrow$ $G(x) = x, f(x)$ is a PRG for circuits of size m.

Proof.

By Theorem, if G is not a PRG, then $\exists i \in [\ell+1]$, $\exists$size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$, s.t. $\Pr_{x \leftarrow U_\ell}[P(G(x)_{1,..i-1}) = G(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

This is obviously impossible if $i \neq \ell + 1$.

□

PRGs from average-case hardness (1 bit stretch)

Theorem (Distinguisher to Predictor)

$G : \{0,1\}^r \to \{0,1\}^m$ is a PRG for size m circuits,
if for every $i \in [m]$, and every size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$,

$$\Pr_{x \leftarrow U_r}[P(G(x)_{1,\ldots,i-1}) = G(x)_i] \leq \frac{1}{2} + \frac{1}{10m}.$$

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard
$\Rightarrow$ $G(x) = x, f(x)$ is a PRG for circuits of size m.

Proof.

By Theorem, if G is not a PRG, then $\exists i \in [\ell+1]$, $\exists$size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$, s.t. $\Pr_{x \leftarrow U_\ell}[P(G(x)_{1,..i-1}) = G(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

This is obviously impossible if $i \neq \ell+1$. For $i = \ell+1$ this gives,

$$\Pr_{x \leftarrow U_\ell}[P(x) = f(x)] > \frac{1}{2} + \frac{1}{10m},$$

□

PRGs from average-case hardness (1 bit stretch)

Theorem (Distinguisher to Predictor)

$G : \{0,1\}^r \to \{0,1\}^m$ is a PRG for size m circuits,
if for every $i \in [m]$, and every size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$,

$$\Pr_{x \leftarrow U_r}[P(G(x)_{1,\ldots,i-1}) = G(x)_i] \leq \frac{1}{2} + \frac{1}{10m}.$$

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard
$\Rightarrow$ $G(x) = x, f(x)$ is a PRG for circuits of size m.

Proof.

By Theorem, if G is not a PRG, then $\exists i \in [\ell+1]$, $\exists$size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$, s.t. $\Pr_{x \leftarrow U_\ell}[P(G(x)_{1,..i-1}) = G(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

This is obviously impossible if $i \neq \ell+1$. For $i = \ell+1$ this gives,

$$\Pr_{x \leftarrow U_\ell}[P(x) = f(x)] > \frac{1}{2} + \frac{1}{10m},$$

which contradicts the average-case hardness of f. □

PRGs from average-case hardness (2 bit stretch)

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $G : \{0,1\}^{2\ell} \to \{0,1\}^{2\ell+2}$ defined by $G(x_1, x_2) = x_1, x_2, f(x_1), f(x_2)$ is a PRG for circuits of size m.

Proof.

PRGs from average-case hardness (2 bit stretch)

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $G : \{0,1\}^{2\ell} \to \{0,1\}^{2\ell+2}$ defined by $G(x_1, x_2) = x_1, x_2, f(x_1), f(x_2)$ is a PRG for circuits of size m.

Proof.

By Theorem, if G is not a PRG, then $\exists i \in [2\ell + 2]$, $\exists$size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$, s.t.
$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(G(x_1, x_2)_{1,..i-1}) = G(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

PRGs from average-case hardness (2 bit stretch)

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $G : \{0,1\}^{2\ell} \to \{0,1\}^{2\ell+2}$ defined by $G(x_1, x_2) = x_1, x_2, f(x_1), f(x_2)$ is a PRG for circuits of size m.

Proof.

By Theorem, if G is not a PRG, then $\exists i \in [2\ell + 2]$, $\exists$size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$, s.t.
$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(G(x_1, x_2)_{1,..i-1}) = G(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

This is obviously impossible if $i \leq 2\ell$.

□

PRGs from average-case hardness (2 bit stretch)

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $G : \{0,1\}^{2\ell} \to \{0,1\}^{2\ell+2}$ defined by $G(x_1, x_2) = x_1, x_2, f(x_1), f(x_2)$ is a PRG for circuits of size m.

Proof.

By Theorem, if G is not a PRG, then $\exists i \in [2\ell + 2]$, $\exists$size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$, s.t.
$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(G(x_1, x_2)_{1,..i-1}) = G(x)_i] > \frac{1}{2} + \frac{1}{10m}$.
This is obviously impossible if $i \leq 2\ell$. If $i = 2\ell + 1$, this gives:

$$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(x_1, x_2) = f(x_1)] > \frac{1}{2} + \frac{1}{10m}.$$

□

PRGs from average-case hardness (2 bit stretch)

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $G : \{0,1\}^{2\ell} \to \{0,1\}^{2\ell+2}$ defined by $G(x_1, x_2) = x_1, x_2, f(x_1), f(x_2)$ is a PRG for circuits of size m.

Proof.

By Theorem, if G is not a PRG, then $\exists i \in [2\ell+2]$, $\exists$size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$, s.t.
$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(G(x_1,x_2)_{1,..i-1}) = G(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

This is obviously impossible if $i \leq 2\ell$. If $i = 2\ell + 1$, this gives:

$$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(x_1, x_2) = f(x_1)] > \frac{1}{2} + \frac{1}{10m}.$$

$\Rightarrow \exists x_2'$, s.t. $\Pr_{x_1 \leftarrow U_\ell}[P(x_1, x_2') = f(x_1)] > \frac{1}{2} + \frac{1}{10m}$.

□

PRGs from average-case hardness (2 bit stretch)

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $G : \{0,1\}^{2\ell} \to \{0,1\}^{2\ell+2}$ defined by $G(x_1, x_2) = x_1, x_2, f(x_1), f(x_2)$ is a PRG for circuits of size m.

Proof.

By Theorem, if G is not a PRG, then $\exists i \in [2\ell+2]$, $\exists$size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$, s.t.
$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(G(x_1,x_2)_{1,..i-1}) = G(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

This is obviously impossible if $i \leq 2\ell$. If $i = 2\ell + 1$, this gives:

$$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(x_1, x_2) = f(x_1)] > \frac{1}{2} + \frac{1}{10m}.$$

$\Rightarrow \exists x_2'$, s.t. $\Pr_{x_1 \leftarrow U_\ell}[P(x_1, x_2') = f(x_1)] > \frac{1}{2} + \frac{1}{10m}$.

The fixing x_2' can be hardwired.

PRGs from average-case hardness (2 bit stretch)

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $G : \{0,1\}^{2\ell} \to \{0,1\}^{2\ell+2}$ defined by $G(x_1, x_2) = x_1, x_2, f(x_1), f(x_2)$ is a PRG for circuits of size m.

Proof.

By Theorem, if G is not a PRG, then $\exists i \in [2\ell+2]$, $\exists$size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$, s.t.
$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(G(x_1,x_2)_{1,..i-1}) = G(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

This is obviously impossible if $i \leq 2\ell$. If $i = 2\ell + 1$, this gives:

$$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(x_1, x_2) = f(x_1)] > \frac{1}{2} + \frac{1}{10m}.$$

$\Rightarrow \exists x_2'$, s.t. $\Pr_{x_1 \leftarrow U_\ell}[P(x_1, x_2') = f(x_1)] > \frac{1}{2} + \frac{1}{10m}$.

The fixing x_2' can be hardwired. That is define $C(x_1) = P(x_1, x_2')$, and then:

$$\Pr_{x_1 \leftarrow U_\ell}[C(x_1) = f(x_1)] > \frac{1}{2} + \frac{1}{10m},$$

□

PRGs from average-case hardness (2 bit stretch)

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $G : \{0,1\}^{2\ell} \to \{0,1\}^{2\ell+2}$ defined by $G(x_1, x_2) = x_1, x_2, f(x_1), f(x_2)$ is a PRG for circuits of size m.

Proof.

By Theorem, if G is not a PRG, then $\exists i \in [2\ell+2]$, $\exists$size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$, s.t.
$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(G(x_1,x_2)_{1,..i-1}) = G(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

This is obviously impossible if $i \leq 2\ell$. If $i = 2\ell + 1$, this gives:

$$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(x_1, x_2) = f(x_1)] > \frac{1}{2} + \frac{1}{10m}.$$

$\Rightarrow \exists x_2'$, s.t. $\Pr_{x_1 \leftarrow U_\ell}[P(x_1, x_2') = f(x_1)] > \frac{1}{2} + \frac{1}{10m}$.

The fixing x_2' can be hardwired. That is define $C(x_1) = P(x_1, x_2')$, and then:

$$\Pr_{x_1 \leftarrow U_\ell}[C(x_1) = f(x_1)] > \frac{1}{2} + \frac{1}{10m},$$

which is a contradiction as C is of size $\approx m$. □

PRGs from average-case hardness (2 bit stretch)

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $G : \{0,1\}^{2\ell} \to \{0,1\}^{2\ell+2}$ defined by $G(x_1, x_2) = x_1, x_2, f(x_1), f(x_2)$ is a PRG for circuits of size m.

Proof.

By Theorem, if G is not a PRG, then $\exists i \in [2\ell + 2]$, $\exists$size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$, s.t.
$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(G(x_1, x_2)_{1,..i-1}) = G(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

This is obviously impossible if $i \leq 2\ell$.

□

PRGs from average-case hardness (2 bit stretch)

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $G : \{0,1\}^{2\ell} \to \{0,1\}^{2\ell+2}$ defined by $G(x_1, x_2) = x_1, x_2, f(x_1), f(x_2)$ is a PRG for circuits of size m.

Proof.

By Theorem, if G is not a PRG, then $\exists i \in [2\ell+2]$, $\exists$size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$, s.t.
$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(G(x_1,x_2)_{1,..i-1}) = G(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

This is obviously impossible if $i \leq 2\ell$. If $i = 2\ell + 2$, this gives:

$$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(x_1, x_2, f(x_1)) = f(x_2)] > \frac{1}{2} + \frac{1}{10m}.$$

PRGs from average-case hardness (2 bit stretch)

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $G : \{0,1\}^{2\ell} \to \{0,1\}^{2\ell+2}$ defined by $G(x_1, x_2) = x_1, x_2, f(x_1), f(x_2)$ is a PRG for circuits of size m.

Proof.

By Theorem, if G is not a PRG, then $\exists i \in [2\ell+2]$, $\exists$size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$, s.t.
$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(G(x_1,x_2)_{1,..i-1}) = G(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

This is obviously impossible if $i \leq 2\ell$. If $i = 2\ell+2$, this gives:

$$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(x_1, x_2, f(x_1)) = f(x_2)] > \frac{1}{2} + \frac{1}{10m}.$$

$\Rightarrow \exists x_1'$, s.t. $\Pr_{x_2 \leftarrow U_\ell}[P(x_1', x_2, f(x_1')) = f(x_2)] > \frac{1}{2} + \frac{1}{10m}$.

PRGs from average-case hardness (2 bit stretch)

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $G : \{0,1\}^{2\ell} \to \{0,1\}^{2\ell+2}$ defined by $G(x_1, x_2) = x_1, x_2, f(x_1), f(x_2)$ is a PRG for circuits of size m.

Proof.

By Theorem, if G is not a PRG, then $\exists i \in [2\ell+2]$, $\exists$size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$, s.t.
$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(G(x_1,x_2)_{1,..i-1}) = G(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

This is obviously impossible if $i \leq 2\ell$. If $i = 2\ell+2$, this gives:

$$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(x_1, x_2, f(x_1)) = f(x_2)] > \frac{1}{2} + \frac{1}{10m}.$$

$\Rightarrow \exists x_1'$, s.t. $\Pr_{x_2 \leftarrow U_\ell}[P(x_1', x_2, f(x_1')) = f(x_2)] > \frac{1}{2} + \frac{1}{10m}$.

The fixings x_1' and $f(x_1')$ can be hardwired.

□

PRGs from average-case hardness (2 bit stretch)

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $G : \{0,1\}^{2\ell} \to \{0,1\}^{2\ell+2}$ defined by $G(x_1, x_2) = x_1, x_2, f(x_1), f(x_2)$ is a PRG for circuits of size m.

Proof.

By Theorem, if G is not a PRG, then $\exists i \in [2\ell+2]$, $\exists$size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$, s.t.
$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(G(x_1,x_2)_{1,..i-1}) = G(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

This is obviously impossible if $i \leq 2\ell$. If $i = 2\ell+2$, this gives:

$$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(x_1, x_2, f(x_1)) = f(x_2)] > \frac{1}{2} + \frac{1}{10m}.$$

$\Rightarrow \exists x_1'$, s.t. $\Pr_{x_2 \leftarrow U_\ell}[P(x_1', x_2, f(x_1')) = f(x_2)] > \frac{1}{2} + \frac{1}{10m}$.

The fixings x_1' and $f(x_1')$ can be hardwired. That is define $C(x_2) = P(x_1', x_2, f(x_1'))$, and then:

$$\Pr_{x_2 \leftarrow U_\ell}[C(x_2) = f(x_2)] > \frac{1}{2} + \frac{1}{10m},$$

□

PRGs from average-case hardness (2 bit stretch)

Claim

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $G : \{0,1\}^{2\ell} \to \{0,1\}^{2\ell+2}$ defined by $G(x_1, x_2) = x_1, x_2, f(x_1), f(x_2)$ is a PRG for circuits of size m.

Proof.

By Theorem, if G is not a PRG, then $\exists i \in [2\ell+2]$, $\exists$size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$, s.t.
$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(G(x_1,x_2)_{1,..i-1}) = G(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

This is obviously impossible if $i \leq 2\ell$. If $i = 2\ell+2$, this gives:

$$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_\ell}[P(x_1, x_2, f(x_1)) = f(x_2)] > \frac{1}{2} + \frac{1}{10m}.$$

$\Rightarrow \exists x_1'$, s.t. $\Pr_{x_2 \leftarrow U_\ell}[P(x_1', x_2, f(x_1')) = f(x_2)] > \frac{1}{2} + \frac{1}{10m}$.

The fixings x_1' and $f(x_1')$ can be hardwired. That is define $C(x_2) = P(x_1', x_2, f(x_1'))$, and then:

$$\Pr_{x_2 \leftarrow U_\ell}[C(x_2) = f(x_2)] > \frac{1}{2} + \frac{1}{10m},$$

which is a contradiction as C is of size $\approx m$. □

PRGs from average-case hardness (NW PRG)

We got 2 bit stretch, but we'll never get large stretch this way...

PRGs from average-case hardness (NW PRG)

We got 2 bit stretch, but we'll never get large stretch this way...

Definition (designs)

A collection of sets $S_1, \ldots, S_m \subseteq [r]$ is an (ℓ, u)-design if:

- For every $1 \leq i \leq m$, $|S_i| = \ell$.
- For every $1 \leq i < j \leq m$, $|S_i \cap S_j| \leq u$.

PRGs from average-case hardness (NW PRG)

We got 2 bit stretch, but we'll never get large stretch this way...

Definition (designs)

A collection of sets $S_1, \ldots, S_m \subseteq [r]$ is an (ℓ, u)-design if:

- For every $1 \leq i \leq m$, $|S_i| = \ell$.
- For every $1 \leq i < j \leq m$, $|S_i \cap S_j| \leq u$.

Definition (NW PRG)

For $f : \{0,1\}^\ell \to \{0,1\}$ and (ℓ, u)-design, define: $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$:

$$NW^f(x) = x, f(x|_{S_1}), \ldots, f(x|_{S_m}).$$

PRGs from average-case hardness (NW PRG)

We got 2 bit stretch, but we'll never get large stretch this way...

Definition (designs)

A collection of sets $S_1, \ldots, S_m \subseteq [r]$ is an (ℓ, u)-design if:

- For every $1 \leq i \leq m$, $|S_i| = \ell$.
- For every $1 \leq i < j \leq m$, $|S_i \cap S_j| \leq u$.

Definition (NW PRG)

For $f : \{0,1\}^\ell \to \{0,1\}$ and (ℓ, u)-design, define: $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$:

$$NW^f(x) = x, f(x|_{S_1}), \ldots, f(x|_{S_m}).$$

$G(x_1, x_2) = x_1, x_2, f(x_1), f(x_2)$ is a special case for $r = 2\ell$, $m = 2$, $S_1 = \{1, \ldots, \ell\}$, $S_2 = \{\ell + 1, \ldots, 2\ell\}$.

PRGs from average-case hardness (NW PRG)

We got 2 bit stretch, but we'll never get large stretch this way...

Definition (designs)

A collection of sets $S_1, \ldots, S_m \subseteq [r]$ is an (ℓ, u)-design if:

- For every $1 \leq i \leq m$, $|S_i| = \ell$.
- For every $1 \leq i < j \leq m$, $|S_i \cap S_j| \leq u$.

Definition (NW PRG)

For $f : \{0,1\}^\ell \to \{0,1\}$ and (ℓ, u)-design, define: $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$:

$$NW^f(x) = x, f(x|_{S_1}), \ldots, f(x|_{S_m}).$$

$G(x_1, x_2) = x_1, x_2, f(x_1), f(x_2)$ is a special case for $r = 2\ell$, $m = 2$, $S_1 = \{1, \ldots, \ell\}$, $S_2 = \{\ell + 1, \ldots, 2\ell\}$.

Theorem (Left as an exercise)

$\forall \gamma > 0, \exists a > 1$ s.t. for every sufficiently large ℓ, there is an explicit $(\ell, \gamma \cdot \ell)$-design with $m = 2^{\Omega(\ell)}$ sets, in a universe of size $r = a \cdot \ell$.
$\Rightarrow NW^f : \{0,1\}^{r=O(\ell)=O(\log m)} \to \{0,1\}^m$.

NW PRG analysis

Claim (NW analysis)

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$ defined by $NW^f(x) = x, f(x|_{S_1}), \ldots, f(x|_{S_m})$ is a PRG for circuits of size m.

Proof.

NW PRG analysis

Claim (NW analysis)

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$ defined by $NW^f(x) = x, f(x|_{S_1}), \ldots, f(x|_{S_m})$ is a PRG for circuits of size m.

Proof.

By Theorem, if NW^f is not a PRG, then $\exists i \in [r+m]$, $\exists$size m circuit P, s.t. $\Pr_{x \leftarrow U_r}[P(NW^f(x)_{1,..i-1}) = NW^f(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

NW PRG analysis

Claim (NW analysis)

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$ defined by $NW^f(x) = x, f(x|_{S_1}), \ldots, f(x|_{S_m})$ is a PRG for circuits of size m.

Proof.

By Theorem, if NW^f is not a PRG, then $\exists i \in [r+m]$, $\exists$size m circuit P, s.t. $\Pr_{x \leftarrow U_r}[P(NW^f(x)_{1,..i-1}) = NW^f(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

It's impossible that P predicts a bit of the seed.

□

NW PRG analysis

Claim (NW analysis)

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$ defined by $NW^f(x) = x, f(x|_{S_1}), \ldots, f(x|_{S_m})$ is a PRG for circuits of size m.

Proof.

By Theorem, if NW^f is not a PRG, then $\exists i \in [r+m]$, $\exists$size m circuit P, s.t. $\Pr_{x \leftarrow U_r}[P(NW^f(x)_{1,..i-1}) = NW^f(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

It's impossible that P predicts a bit of the seed. This gives: $\exists i \in [m]$ s.t.

$$\Pr_{x \leftarrow U_r}[P(x, f(x|_{S_1}), \ldots, f(x|_{S_{i-1}})) = f(x|_{S_i})] > \frac{1}{2} + \frac{1}{10m}.$$

□

NW PRG analysis

Claim (NW analysis)

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$ defined by $NW^f(x) = x, f(x|_{S_1}), \ldots, f(x|_{S_m})$ is a PRG for circuits of size m.

Proof.

By Theorem, if NW^f is not a PRG, then $\exists i \in [r+m]$, $\exists$size m circuit P, s.t. $\Pr_{x \leftarrow U_r}[P(NW^f(x)_{1,..i-1}) = NW^f(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

It's impossible that P predicts a bit of the seed. This gives: $\exists i \in [m]$ s.t.

$$\Pr_{x \leftarrow U_r}[P(x, f(x|_{S_1}), \ldots, f(x|_{S_{i-1}})) = f(x|_{S_i})] > \frac{1}{2} + \frac{1}{10m}.$$

Split $x \in \{0,1\}^r$ to $x = (x_1, x_2)$ where $x_1 = x|_{S_i}$ and $x_2 = x|_{[r] \setminus S_i}$.

NW PRG analysis

Claim (NW analysis)

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$ defined by $NW^f(x) = x, f(x|_{S_1}), \ldots, f(x|_{S_m})$ is a PRG for circuits of size m.

Proof.

By Theorem, if NW^f is not a PRG, then $\exists i \in [r+m]$, $\exists$size m circuit P, s.t. $\Pr_{x \leftarrow U_r}[P(NW^f(x)_{1,..i-1}) = NW^f(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

It's impossible that P predicts a bit of the seed. This gives: $\exists i \in [m]$ s.t.

$$\Pr_{x \leftarrow U_r}[P(x, f(x|_{S_1}), \ldots, f(x|_{S_{i-1}})) = f(x|_{S_i})] > \frac{1}{2} + \frac{1}{10m}.$$

Split $x \in \{0,1\}^r$ to $x = (x_1, x_2)$ where $x_1 = x|_{S_i}$ and $x_2 = x|_{[r]\setminus S_i}$.

$$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_{r-\ell}}[P(x, f(x|_{S_1}), \ldots, f(x|_{S_{i-1}})) = f(x_1)] > \frac{1}{2} + \frac{1}{10m}.$$

NW PRG analysis

Claim (NW analysis)

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$ defined by $NW^f(x) = x, f(x|_{s_1}), \ldots, f(x|_{s_m})$ is a PRG for circuits of size m.

Proof.

By Theorem, if NW^f is not a PRG, then $\exists i \in [r+m]$, $\exists$size m circuit P, s.t. $\Pr_{x \leftarrow U_r}[P(NW^f(x)_{1,..i-1}) = NW^f(x)_i] > \frac{1}{2} + \frac{1}{10m}$.

It's impossible that P predicts a bit of the seed. This gives: $\exists i \in [m]$ s.t.

$$\Pr_{x \leftarrow U_r}[P(x, f(x|_{s_1}), \ldots, f(x|_{s_{i-1}})) = f(x|_{s_i})] > \frac{1}{2} + \frac{1}{10m}.$$

Split $x \in \{0,1\}^r$ to $x = (x_1, x_2)$ where $x_1 = x|_{s_i}$ and $x_2 = x|_{[r]\setminus s_i}$.

$$\Pr_{x_1 \leftarrow U_\ell, x_2 \leftarrow U_{r-\ell}}[P(x, f(x|_{s_1}), \ldots, f(x|_{s_{i-1}})) = f(x_1)] > \frac{1}{2} + \frac{1}{10m}.$$

$$\Rightarrow \exists x_2' : \Pr_{x_1 \leftarrow U_\ell, x=(x_1,x_2')}[P(x, f(x|_{s_1}), \ldots, f(x|_{s_{i-1}})) = f(x_1)] > \frac{1}{2} + \frac{1}{10m}.$$

□

NW PRG analysis (continued)

Claim (NW analysis)

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$ defined by $NW^f(x) = x, f(x|_{S_1}), \ldots, f(x|_{S_m})$ is a PRG for circuits of size m.

Proof.

NW PRG analysis (continued)

Claim (NW analysis)

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$ defined by $NW^f(x) = x, f(x|_{S_1}), \ldots, f(x|_{S_m})$ is a PRG for circuits of size m.

Proof.

By Theorem, if NW^f is not a PRG, then $\exists i \in [m]$, $\exists$size m circuit P, s.t.

$$\exists x_2' : \Pr_{x_1 \leftarrow U_\ell, x=(x_1,x_2')}[P(x, f(x|_{S_1}), \ldots, f(x|_{S_{i-1}})) = f(x_1)] > \frac{1}{2} + \frac{1}{10m},$$

where we split $x \in \{0,1\}^r$ to $x = (x_1, x_2)$ where $x_1 = x|_{S_i}$ and $x_2 = x|_{[r]\setminus S_i}$.

NW PRG analysis (continued)

Claim (NW analysis)

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$ defined by $NW^f(x) = x, f(x|_{S_1}), \ldots, f(x|_{S_m})$ is a PRG for circuits of size m.

Proof.

By Theorem, if NW^f is not a PRG, then $\exists i \in [m]$, $\exists$size m circuit P, s.t.

$$\exists x_2' : \Pr_{x_1 \leftarrow U_\ell, x=(x_1,x_2')}[P(x, f(x|_{S_1}), \ldots, f(x|_{S_{i-1}})) = f(x_1)] > \frac{1}{2} + \frac{1}{10m},$$

where we split $x \in \{0,1\}^r$ to $x = (x_1, x_2)$ where $x_1 = x|_{S_i}$ and $x_2 = x|_{[r]\setminus S_i}$.

For $j < i$, how hard is it to compute $f(x|_{S_j}) = f((x_1, x_2')|_{S_j})$ when given x_1?

NW PRG analysis (continued)

Claim (NW analysis)

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$ defined by $NW^f(x) = x, f(x|_{S_1}), \ldots, f(x|_{S_m})$ is a PRG for circuits of size m.

Proof.

By Theorem, if NW^f is not a PRG, then $\exists i \in [m]$, $\exists$size m circuit P, s.t.

$$\exists x_2' : \Pr_{x_1 \leftarrow U_\ell, x=(x_1,x_2')}[P(x, f(x|_{S_1}), \ldots, f(x|_{S_{i-1}})) = f(x_1)] > \frac{1}{2} + \frac{1}{10m},$$

where we split $x \in \{0,1\}^r$ to $x = (x_1, x_2)$ where $x_1 = x|_{S_i}$ and $x_2 = x|_{[r]\setminus S_i}$.

For $j < i$, how hard is it to compute $f(x|_{S_j}) = f((x_1, x_2')|_{S_j})$ when given x_1?

$|S_i \cap S_j| \leq u \Rightarrow$ hardwiring x_2', this is a function of x_1 that depends only u bits

NW PRG analysis (continued)

Claim (NW analysis)

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$ defined by $NW^f(x) = x, f(x|_{S_1}), \ldots, f(x|_{S_m})$ is a PRG for circuits of size m.

Proof.

By Theorem, if NW^f is not a PRG, then $\exists i \in [m]$, $\exists$size m circuit P, s.t.

$$\exists x_2' : \Pr_{x_1 \leftarrow U_\ell, x=(x_1,x_2')}[P(x, f(x|_{S_1}), \ldots, f(x|_{S_{i-1}})) = f(x_1)] > \frac{1}{2} + \frac{1}{10m},$$

where we split $x \in \{0,1\}^r$ to $x = (x_1, x_2)$ where $x_1 = x|_{S_i}$ and $x_2 = x|_{[r]\setminus S_i}$.

For $j < i$, how hard is it to compute $f(x|_{S_j}) = f((x_1, x_2')|_{S_j})$ when given x_1?

$|S_i \cap S_j| \leq u \Rightarrow$ hardwiring x_2', this is a function of x_1 that depends only u bits

$\Rightarrow$ Every such function has a circuit of size 2^u.

NW PRG analysis (continued)

Claim (NW analysis)

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$ defined by $NW^f(x) = x, f(x|_{S_1}), \ldots, f(x|_{S_m})$ is a PRG for circuits of size m.

Proof.

By Theorem, if NW^f is not a PRG, then $\exists i \in [m]$, $\exists$size m circuit P, s.t.

$$\exists x_2' : \Pr_{x_1 \leftarrow U_\ell, x=(x_1,x_2')}[P(x, f(x|_{S_1}), \ldots, f(x|_{S_{i-1}})) = f(x_1)] > \frac{1}{2} + \frac{1}{10m},$$

where we split $x \in \{0,1\}^r$ to $x = (x_1, x_2)$ where $x_1 = x|_{S_i}$ and $x_2 = x|_{[r]\setminus S_i}$.

For $j < i$, how hard is it to compute $f(x|_{S_j}) = f((x_1, x_2')|_{S_j})$ when given x_1?

$|S_i \cap S_j| \leq u \Rightarrow$ hardwiring x_2', this is a function of x_1 that depends only u bits

$\Rightarrow$ Every such function has a circuit of size 2^u.

overall, $\exists$circuit $C(x_1)$ of size $O(m \cdot 2^u)$ s.t. $C(x_1) = P(x, f(x|_{S_1}), \ldots, f(x|_{S_{i-1}}))$.

NW PRG analysis (continued)

Claim (NW analysis)

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$ defined by $NW^f(x) = x, f(x|_{S_1}), \ldots, f(x|_{S_m})$ is a PRG for circuits of size m.

Proof.

By Theorem, if NW^f is not a PRG, then $\exists i \in [m]$, $\exists$size m circuit P, s.t.

$$\exists x_2' : \Pr_{x_1 \leftarrow U_\ell, x=(x_1,x_2')}[P(x, f(x|_{S_1}), \ldots, f(x|_{S_{i-1}})) = f(x_1)] > \frac{1}{2} + \frac{1}{10m},$$

where we split $x \in \{0,1\}^r$ to $x = (x_1, x_2)$ where $x_1 = x|_{S_i}$ and $x_2 = x|_{[r]\setminus S_i}$.

For $j < i$, how hard is it to compute $f(x|_{S_j}) = f((x_1, x_2')|_{S_j})$ when given x_1?

$|S_i \cap S_j| \leq u \Rightarrow$ hardwiring x_2', this is a function of x_1 that depends only u bits

$\Rightarrow$ Every such function has a circuit of size 2^u.

overall, $\exists$circuit $C(x_1)$ of size $O(m \cdot 2^u)$ s.t. $C(x_1) = P(x, f(x|_{S_1}), \ldots, f(x|_{S_{i-1}}))$.

Recall that we chose $\ell = O(\log m)$ and $u = \gamma \cdot \ell$, where we can choose a sufficiently small constant $\gamma > 0$.

NW PRG analysis (continued)

Claim (NW analysis)

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$ defined by $NW^f(x) = x, f(x|_{S_1}), \ldots, f(x|_{S_m})$ is a PRG for circuits of size m.

Proof.

By Theorem, if NW^f is not a PRG, then $\exists i \in [m]$, $\exists$size m circuit P, s.t.

$$\exists x_2' : \Pr_{x_1 \leftarrow U_\ell, x=(x_1,x_2')}[P(x, f(x|_{S_1}), \ldots, f(x|_{S_{i-1}})) = f(x_1)] > \frac{1}{2} + \frac{1}{10m},$$

where we split $x \in \{0,1\}^r$ to $x = (x_1, x_2)$ where $x_1 = x|_{S_i}$ and $x_2 = x|_{[r]\setminus S_i}$.

For $j < i$, how hard is it to compute $f(x|_{S_j}) = f((x_1, x_2')|_{S_j})$ when given x_1?

$|S_i \cap S_j| \leq u \Rightarrow$ hardwiring x_2', this is a function of x_1 that depends only u bits

$\Rightarrow$ Every such function has a circuit of size 2^u.

overall, $\exists$circuit $C(x_1)$ of size $O(m \cdot 2^u)$ s.t. $C(x_1) = P(x, f(x|_{S_1}), \ldots, f(x|_{S_{i-1}}))$.

Recall that we chose $\ell = O(\log m)$ and $u = \gamma \cdot \ell$, where we can choose a sufficiently small constant $\gamma > 0$. We can get that size(C) $\approx$ m,

NW PRG analysis (continued)

Claim (NW analysis)

$f : \{0,1\}^\ell \to \{0,1\}$ is m-average-case hard $\Rightarrow$ $NW^f : \{0,1\}^r \to \{0,1\}^{r+m}$ defined by $NW^f(x) = x, f(x|_{S_1}), \ldots, f(x|_{S_m})$ is a PRG for circuits of size m.

Proof.

By Theorem, if NW^f is not a PRG, then $\exists i \in [m]$, $\exists$size m circuit P, s.t.

$$\exists x_2' : \Pr_{x_1 \leftarrow U_\ell, x=(x_1,x_2')}[P(x, f(x|_{S_1}), \ldots, f(x|_{S_{i-1}})) = f(x_1)] > \frac{1}{2} + \frac{1}{10m},$$

where we split $x \in \{0,1\}^r$ to $x = (x_1, x_2)$ where $x_1 = x|_{S_i}$ and $x_2 = x|_{[r]\setminus S_i}$.

For $j < i$, how hard is it to compute $f(x|_{S_j}) = f((x_1, x_2')|_{S_j})$ when given x_1?

$|S_i \cap S_j| \leq u \Rightarrow$ hardwiring x_2', this is a function of x_1 that depends only u bits

$\Rightarrow$ Every such function has a circuit of size 2^u.

overall, $\exists$circuit $C(x_1)$ of size $O(m \cdot 2^u)$ s.t. $C(x_1) = P(x, f(x|_{S_1}), \ldots, f(x|_{S_{i-1}}))$.

Recall that we chose $\ell = O(\log m)$ and $u = \gamma \cdot \ell$, where we can choose a sufficiently small constant $\gamma > 0$. We can get that size(C) $\approx$ m, and $\Pr_{x_1 \leftarrow U_\ell}[C(x_1) = f(x_1)] > \frac{1}{2} + \frac{1}{10m}$, which is a contradiction.

□

NW PRG (reflection: the power and cost of designs)

One can view the NW-PRG as a "small sample space".

NW PRG (reflection: the power and cost of designs)

One can view the NW-PRG as a "small sample space".

Definition (designs)

A collection of sets $S_1, \ldots, S_m \subseteq [r]$ is an (ℓ, u)-design if:

- For every $1 \leq i \leq m$, $|S_i| = \ell$.
- For every $1 \leq i < j \leq m$, $|S_i \cap S_j| \leq u$.

NW PRG (reflection: the power and cost of designs)

One can view the NW-PRG as a "small sample space".

Definition (designs)

A collection of sets $S_1, \ldots, S_m \subseteq [r]$ is an (ℓ, u)-design if:

- For every $1 \leq i \leq m$, $|S_i| = \ell$.
- For every $1 \leq i < j \leq m$, $|S_i \cap S_j| \leq u$.

Specifically, a map that maps a short $x \in \{0,1\}^r$ to many strings $x|_{S_1}, \ldots, x|_{S_m}$, which have "small intersections" between pairs.

NW PRG (reflection: the power and cost of designs)

One can view the NW-PRG as a "small sample space".

Definition (designs)

A collection of sets $S_1, \ldots, S_m \subseteq [r]$ is an (ℓ, u)-design if:

- For every $1 \leq i \leq m$, $|S_i| = \ell$.
- For every $1 \leq i < j \leq m$, $|S_i \cap S_j| \leq u$.

Specifically, a map that maps a short $x \in \{0,1\}^r$ to many strings $x|_{S_1}, \ldots, x|_{S_m}$, which have "small intersections" between pairs.

Countless applications: see Santhanam's talk, Complexity as a Kaleidoscope 26.

NW PRG (reflection: the power and cost of designs)

One can view the NW-PRG as a "small sample space".

Definition (designs)

A collection of sets $S_1, \ldots, S_m \subseteq [r]$ is an (ℓ, u)-design if:

- For every $1 \leq i \leq m$, $|S_i| = \ell$.
- For every $1 \leq i < j \leq m$, $|S_i \cap S_j| \leq u$.

Specifically, a map that maps a short $x \in \{0,1\}^r$ to many strings $x|_{S_1}, \ldots, x|_{S_m}$, which have "small intersections" between pairs.

Countless applications: see Santhanam's talk, Complexity as a Kaleidoscope 26.

Curiously, [IW97] uses this view to "derandomize Yao's XOR lemma".

NW PRG (reflection: the power and cost of designs)

One can view the NW-PRG as a "small sample space".

Definition (designs)

A collection of sets $S_1, \ldots, S_m \subseteq [r]$ is an (ℓ, u)-design if:

- For every $1 \leq i \leq m$, $|S_i| = \ell$.
- For every $1 \leq i < j \leq m$, $|S_i \cap S_j| \leq u$.

Specifically, a map that maps a short $x \in \{0,1\}^r$ to many strings $x|_{S_1}, \ldots, x|_{S_m}$, which have "small intersections" between pairs.

Countless applications: see Santhanam's talk, Complexity as a Kaleidoscope 26.

Curiously, [IW97] uses this view to "derandomize Yao's XOR lemma".

Designs have $r = O(\ell)$ if $u = \Omega(\ell)$. In general $r = O(\ell^2/u)$, which gives $r \approx \ell^2$ if u is small (this is tight [ISW99], see exercise).

NW PRG (reflection: the power and cost of designs)

One can view the NW-PRG as a "small sample space".

Definition (designs)

A collection of sets $S_1, \ldots, S_m \subseteq [r]$ is an (ℓ, u)-design if:

- For every $1 \leq i \leq m$, $|S_i| = \ell$.
- For every $1 \leq i < j \leq m$, $|S_i \cap S_j| \leq u$.

Specifically, a map that maps a short $x \in \{0,1\}^r$ to many strings $x|_{S_1}, \ldots, x|_{S_m}$, which have "small intersections" between pairs.

Countless applications: see Santhanam's talk, Complexity as a Kaleidoscope 26.

Curiously, [IW97] uses this view to "derandomize Yao's XOR lemma".

Designs have $r = O(\ell)$ if $u = \Omega(\ell)$. In general $r = O(\ell^2/u)$,
which gives $r \approx \ell^2$ if u is small (this is tight [ISW99], see exercise).
$\Rightarrow$ Designs have costs in "low-end derandomization".

NW PRG (reflection: the power and cost of designs)

One can view the NW-PRG as a "small sample space".

Definition (designs)

A collection of sets $S_1, \ldots, S_m \subseteq [r]$ is an (ℓ, u)-design if:

- For every $1 \leq i \leq m$, $|S_i| = \ell$.
- For every $1 \leq i < j \leq m$, $|S_i \cap S_j| \leq u$.

Specifically, a map that maps a short $x \in \{0,1\}^r$ to many strings $x|_{S_1}, \ldots, x|_{S_m}$, which have "small intersections" between pairs.

Countless applications: see Santhanam's talk, Complexity as a Kaleidoscope 26.

Curiously, [IW97] uses this view to "derandomize Yao's XOR lemma".

Designs have $r = O(\ell)$ if $u = \Omega(\ell)$. In general $r = O(\ell^2/u)$,
which gives $r \approx \ell^2$ if u is small (this is tight [ISW99], see exercise).
$\Rightarrow$ Designs have costs in "low-end derandomization".
This led to a different PRG construction [SU01,Umans02] which I'll cover later.

NW PRG (reflection: the power and cost of designs)

One can view the NW-PRG as a "small sample space".

Definition (designs)

A collection of sets $S_1, \ldots, S_m \subseteq [r]$ is an (ℓ, u)-design if:

- For every $1 \leq i \leq m$, $|S_i| = \ell$.
- For every $1 \leq i < j \leq m$, $|S_i \cap S_j| \leq u$.

Specifically, a map that maps a short $x \in \{0,1\}^r$ to many strings $x|_{S_1}, \ldots, x|_{S_m}$, which have "small intersections" between pairs.

Countless applications: see Santhanam's talk, Complexity as a Kaleidoscope 26.

Curiously, [IW97] uses this view to "derandomize Yao's XOR lemma".

Designs have $r = O(\ell)$ if $u = \Omega(\ell)$. In general $r = O(\ell^2/u)$,
which gives $r \approx \ell^2$ if u is small (this is tight [ISW99], see exercise).
$\Rightarrow$ Designs have costs in "low-end derandomization".
This led to a different PRG construction [SU01,Umans02] which I'll cover later.
[NW88,SU01] are building blocks in recent work on hardness vs. randomness.

NW PRG (reflection: the power and cost of designs)

One can view the NW-PRG as a "small sample space".

Definition (designs)

A collection of sets $S_1, \ldots, S_m \subseteq [r]$ is an (ℓ, u)-design if:

- For every $1 \leq i \leq m$, $|S_i| = \ell$.
- For every $1 \leq i < j \leq m$, $|S_i \cap S_j| \leq u$.

Specifically, a map that maps a short $x \in \{0,1\}^r$ to many strings $x|_{S_1}, \ldots, x|_{S_m}$, which have "small intersections" between pairs.

Countless applications: see Santhanam's talk, Complexity as a Kaleidoscope 26.

Curiously, [IW97] uses this view to "derandomize Yao's XOR lemma".

Designs have $r = O(\ell)$ if $u = \Omega(\ell)$. In general $r = O(\ell^2/u)$,
which gives $r \approx \ell^2$ if u is small (this is tight [ISW99], see exercise).
$\Rightarrow$ Designs have costs in "low-end derandomization".
This led to a different PRG construction [SU01,Umans02] which I'll cover later.
[NW88,SU01] are building blocks in recent work on hardness vs. randomness.

Comment: Even if you're happy with $r = \ell^2$, designs have other costs, that can be reduced with "weak designs" [RRV99].

NW PRG (reflection: cost of hybrid argument)

We relied on the following theorem:

Theorem (Distinguisher to Predictor)

$G : \{0,1\}^r \to \{0,1\}^m$ is a PRG for size m circuits,
if for every $i \in [m]$, and every size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$,

$$\Pr_{x \leftarrow U_r}[P(G(x)_{1,\ldots,i-1}) = G(x)_i] \leq \frac{1}{2} + \frac{1}{10m}.$$

NW PRG (reflection: cost of hybrid argument)

We relied on the following theorem:

Theorem (Distinguisher to Predictor)

$G : \{0,1\}^r \to \{0,1\}^m$ is a PRG for size m circuits,
if for every $i \in [m]$, and every size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$,

$$\Pr_{x \leftarrow U_r}[P(G(x)_{1,\ldots,i-1}) = G(x)_i] \leq \frac{1}{2} + \frac{1}{10m}.$$

This relies on the hybrid argument [Yao82,GM84].

.

NW PRG (reflection: cost of hybrid argument)

We relied on the following theorem:

Theorem (Distinguisher to Predictor)

$G : \{0,1\}^r \to \{0,1\}^m$ is a PRG for size m circuits,
if for every $i \in [m]$, and every size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$,

$$\Pr_{x \leftarrow U_r}[P(G(x)_{1,\ldots,i-1}) = G(x)_i] \leq \frac{1}{2} + \frac{1}{10m}.$$

This relies on the hybrid argument [Yao82,GM84].

Even when shooting to get constant error $\epsilon = 1/10$ in the PRG, we have to "amplify" the average-case hardness to $\frac{1}{2} + \frac{1}{10m}$.

.

NW PRG (reflection: cost of hybrid argument)

We relied on the following theorem:

Theorem (Distinguisher to Predictor)

$G : \{0,1\}^r \to \{0,1\}^m$ is a PRG for size m circuits, if for every $i \in [m]$, and every size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$,

$$\Pr_{x \leftarrow U_r}[P(G(x)_{1,\ldots,i-1}) = G(x)_i] \leq \frac{1}{2} + \frac{1}{10m}.$$

This relies on the hybrid argument [Yao82,GM84].

Even when shooting to get constant error $\epsilon = 1/10$ in the PRG, we have to "amplify" the average-case hardness to $\frac{1}{2} + \frac{1}{10m}$.

Such amplification has costs (in seed length, and size loss). See e.g., [SV08,FSUV12,AASY16,GSV18,SV22].

.

NW PRG (reflection: cost of hybrid argument)

We relied on the following theorem:

Theorem (Distinguisher to Predictor)

$G : \{0,1\}^r \to \{0,1\}^m$ is a PRG for size m circuits,
if for every $i \in [m]$, and every size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$,

$$\Pr_{x \leftarrow U_r}[P(G(x)_{1,\ldots,i-1}) = G(x)_i] \leq \frac{1}{2} + \frac{1}{10m}.$$

This relies on the hybrid argument [Yao82,GM84].

Even when shooting to get constant error $\epsilon = 1/10$ in the PRG, we have to "amplify" the average-case hardness to $\frac{1}{2} + \frac{1}{10m}$.

Such amplification has costs (in seed length, and size loss). See e.g., [SV08,FSUV12,AASY16,GSV18,SV22].

This is a barrier for "fast derandomization" where the goal is $\mathrm{BPTIME}(T) \subseteq \mathrm{DTIME}(T^{1+\text{something small}})$.

.

NW PRG (reflection: cost of hybrid argument)

We relied on the following theorem:

Theorem (Distinguisher to Predictor)

$G : \{0,1\}^r \to \{0,1\}^m$ is a PRG for size m circuits,
if for every $i \in [m]$, and every size m circuit $P : \{0,1\}^{i-1} \to \{0,1\}$,

$$\Pr_{x \leftarrow U_r}[P(G(x)_{1,\ldots,i-1}) = G(x)_i] \leq \frac{1}{2} + \frac{1}{10m}.$$

This relies on the hybrid argument [Yao82,GM84].

Even when shooting to get constant error $\epsilon = 1/10$ in the PRG, we have to "amplify" the average-case hardness to $\frac{1}{2} + \frac{1}{10m}$.

Such amplification has costs (in seed length, and size loss). See e.g., [SV08,FSUV12,AASY16,GSV18,SV22].

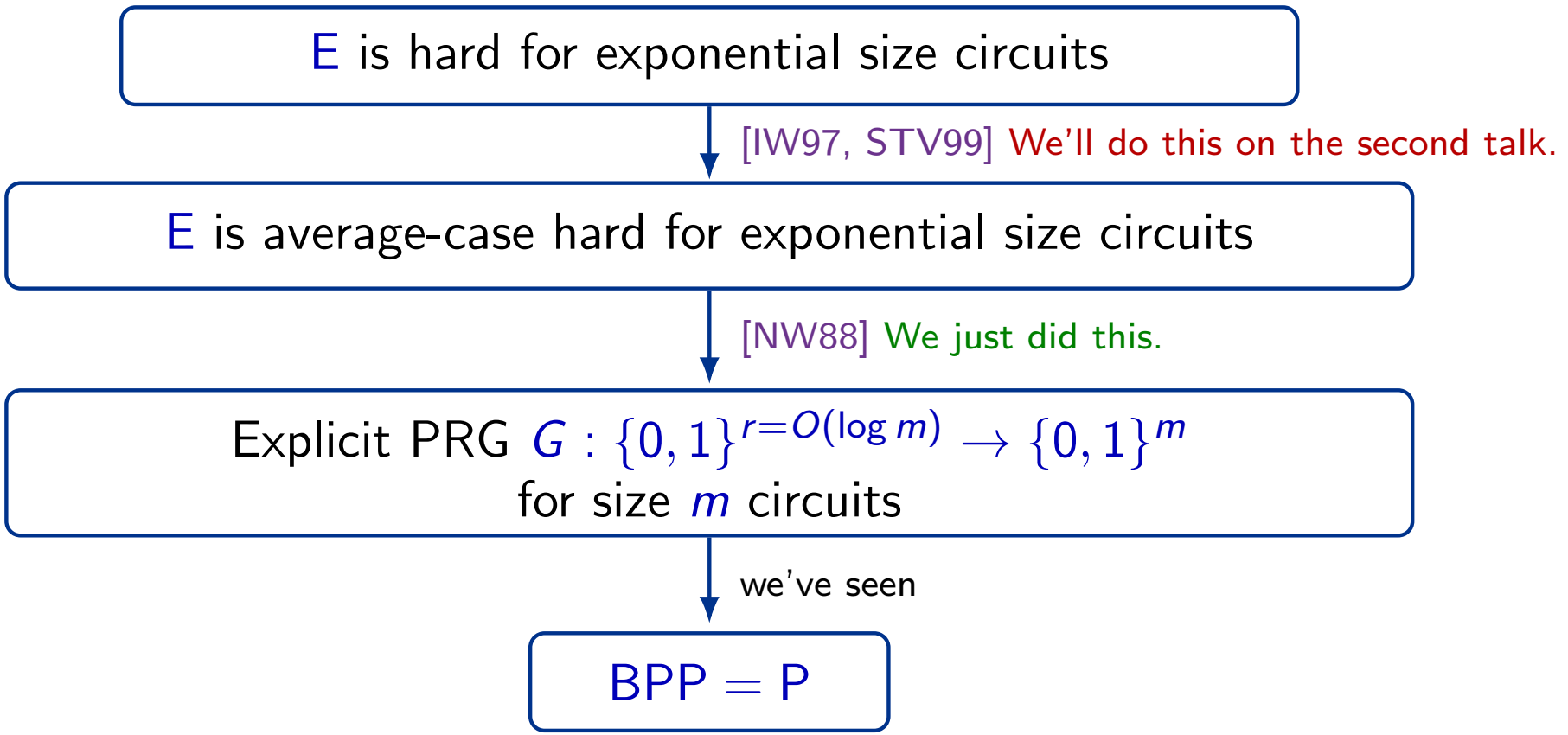
This is a barrier for "fast derandomization" where the goal is $\mathrm{BPTIME}(T) \subseteq \mathrm{DTIME}(T^{1+\text{something small}})$.

Exciting recent work on "fast derandomization" that "bypasses the hybrid argument" (typically using specialized and more elaborate hardness assumptions) [DMOZ20,CT21a,CT21b,CT23,BCT25,DMOZ26...].

Hardness $\Rightarrow$ PRGs (Where we are)

Theorem (IW97)

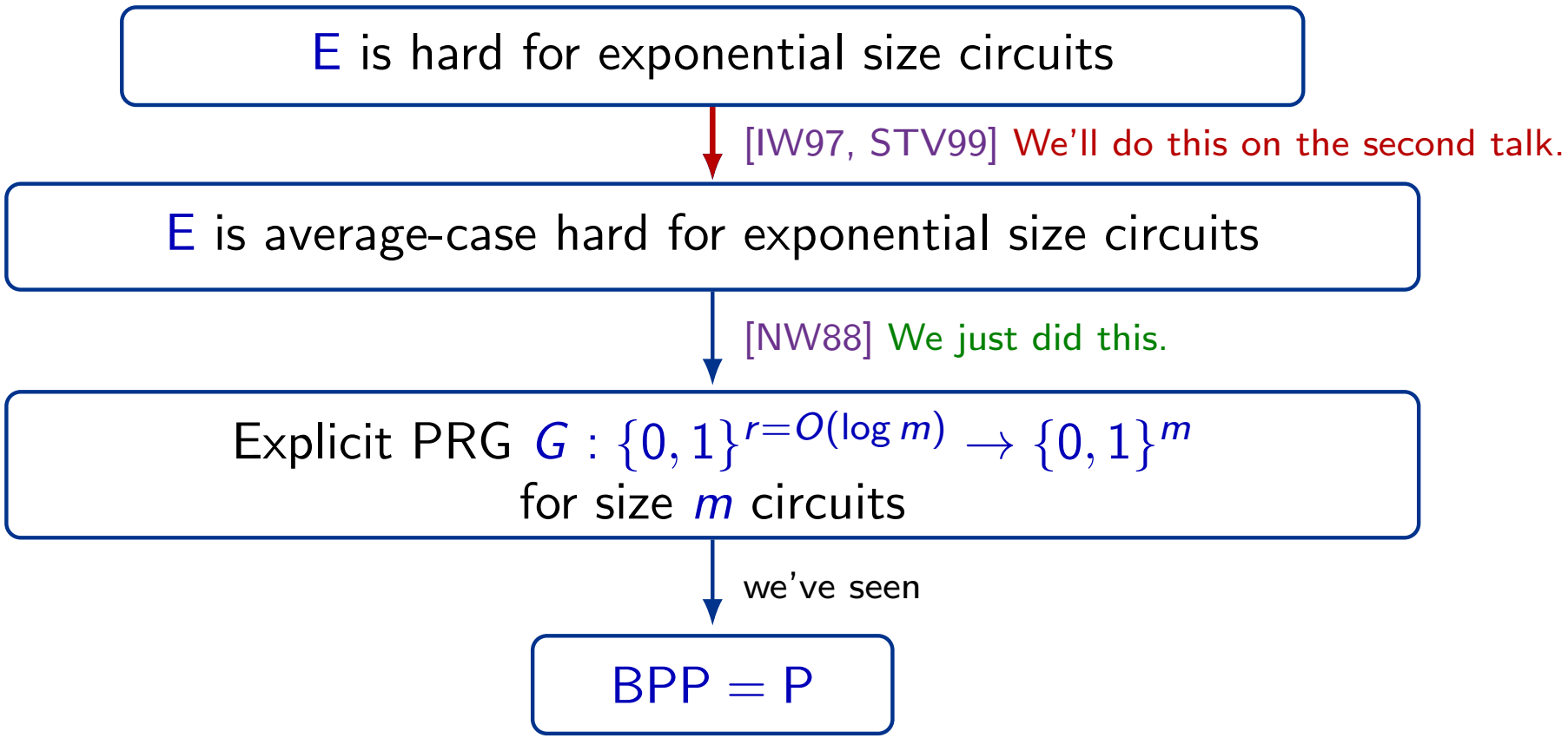
E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.



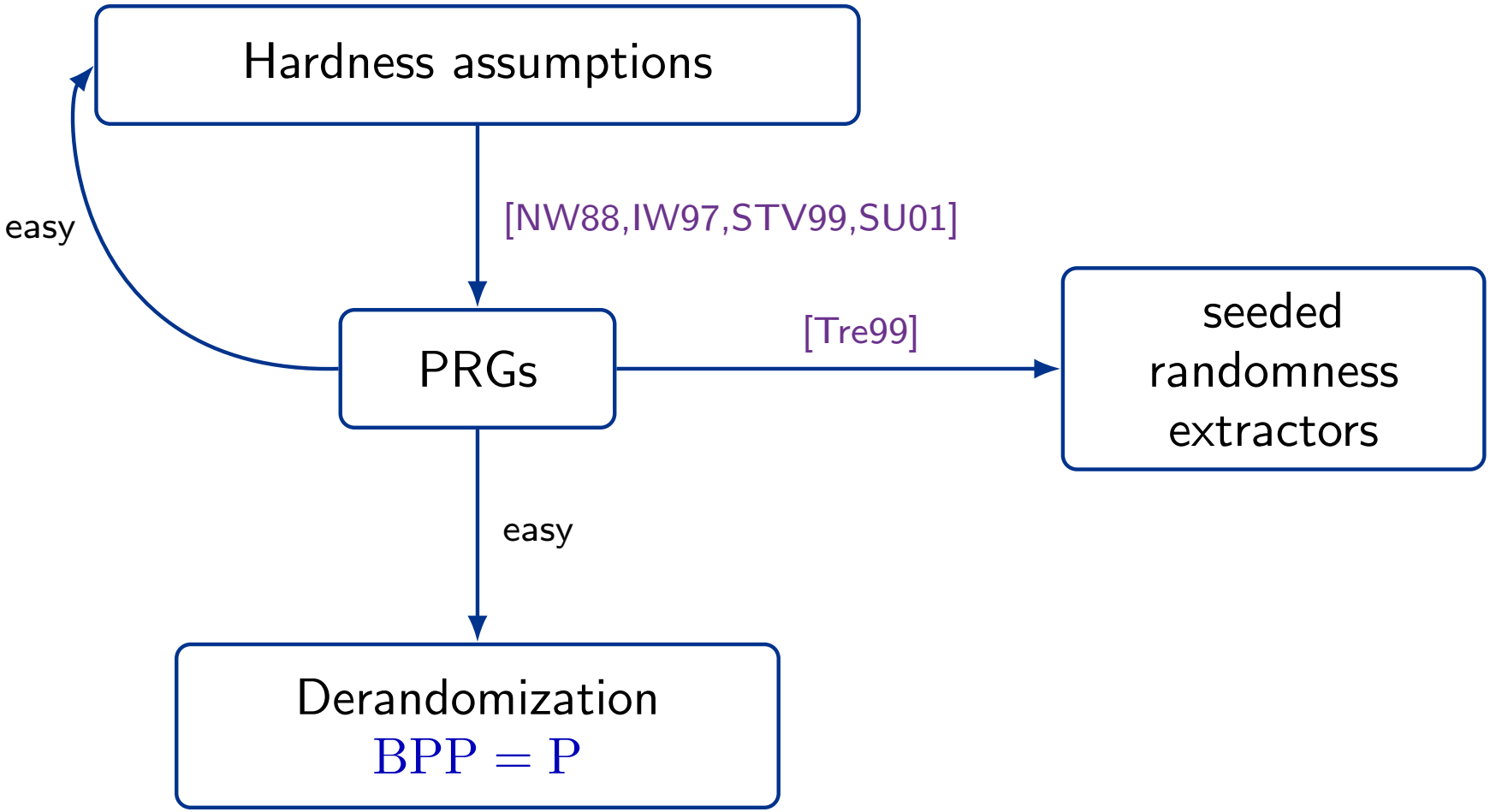
Hardness $\Rightarrow$ PRGs (Where we are)

Theorem (IW97)

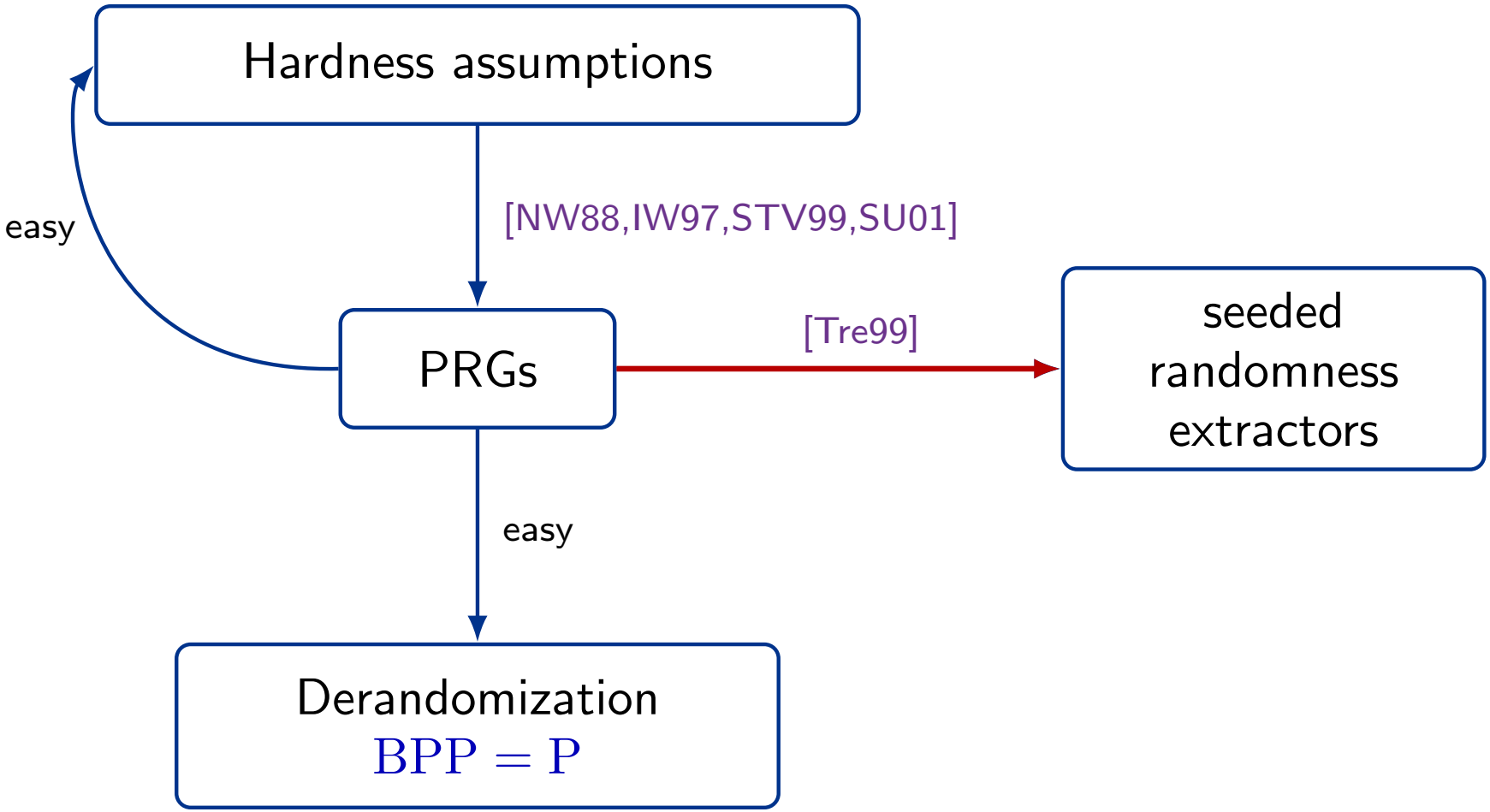
E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.



Plan for talks



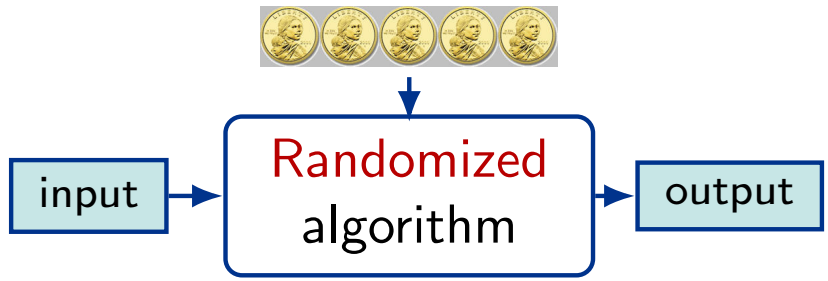
Plan for talks



Randomness extractors (Eshan will give two talks)

How do computers obtain random bits?

Eshan will give detailed talks on randomness extractors.

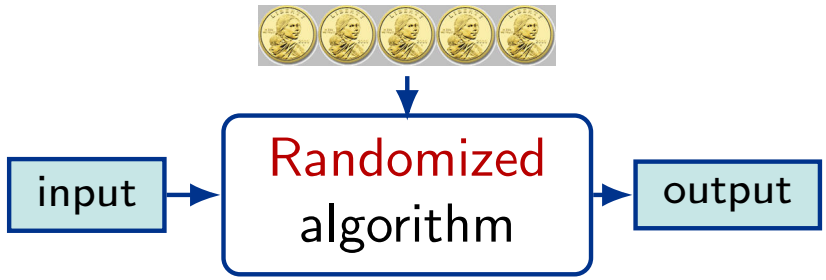


How do computers obtain random bits?

Eshan will give detailed talks on randomness extractors.

Computers can sample from:

- Electro-magnetic noise (Intel)
- Key strokes of user (Unix)
- Timing of past events (Unix)

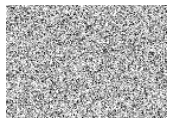


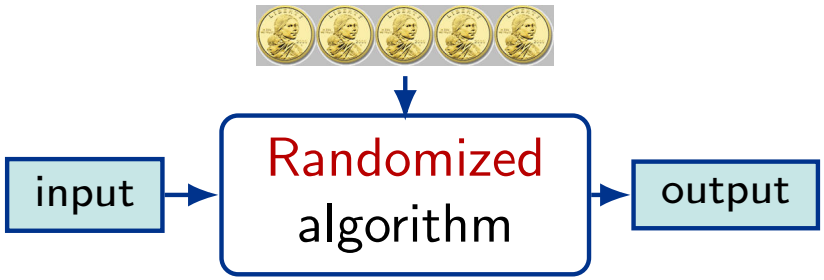
How do computers obtain random bits?

Eshan will give detailed talks on randomness extractors.

Computers can sample from:

- Electro-magnetic noise (Intel)
- Key strokes of user (Unix)
- Timing of past events (Unix)



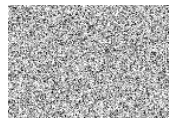


How do computers obtain random bits?

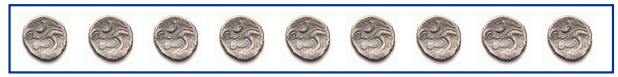
Eshan will give detailed talks on randomness extractors.

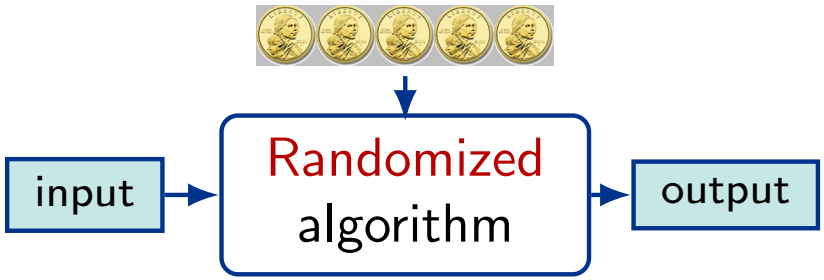
Computers can sample from:

- Electro-magnetic noise (Intel)
- Key strokes of user (Unix)
- Timing of past events (Unix)



"weak source of randomness"



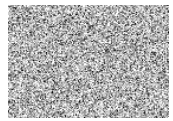


How do computers obtain random bits?

Eshan will give detailed talks on randomness extractors.

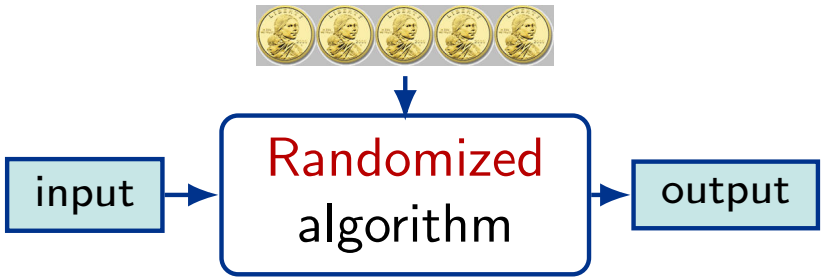
Computers can sample from:

- Electro-magnetic noise (Intel)
- Key strokes of user (Unix)
- Timing of past events (Unix)



"weak source of randomness"

These distributions are
somewhat random but not truly random.

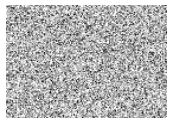


How do computers obtain random bits?

Eshan will give detailed talks on randomness extractors.

Computers can sample from:

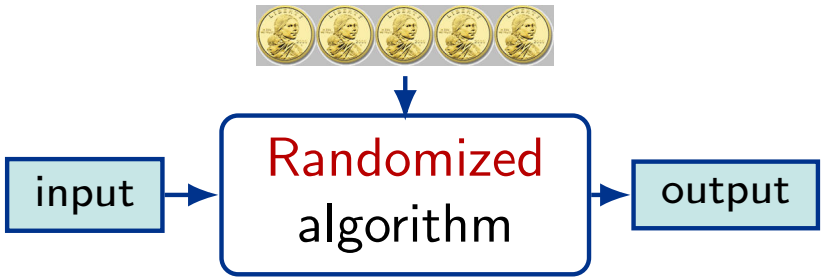
- Electro-magnetic noise (Intel)
- Key strokes of user (Unix)
- Timing of past events (Unix)



"weak source of randomness"

These distributions are somewhat random but not truly random.

Paradigm: randomness extractors

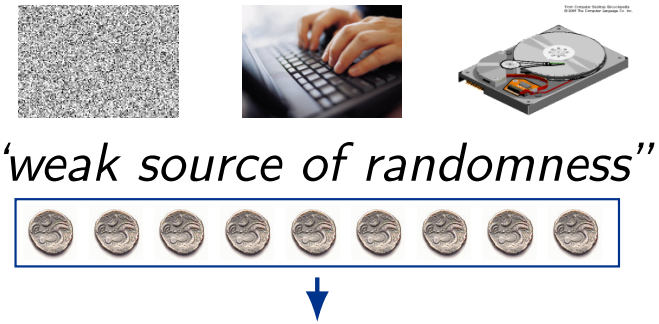


How do computers obtain random bits?

Eshan will give detailed talks on randomness extractors.

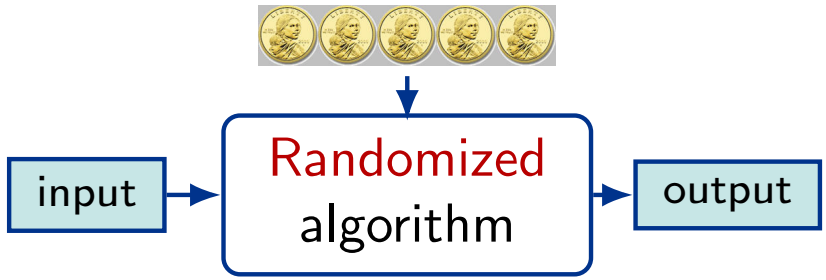
Computers can sample from:

- Electro-magnetic noise (Intel)
- Key strokes of user (Unix)
- Timing of past events (Unix)



These distributions are
somewhat random but not truly random.

Paradigm: randomness extractors



How do computers obtain random bits?

Eshan will give detailed talks on randomness extractors.

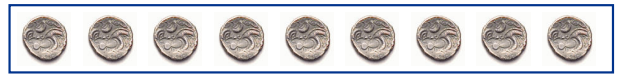
Computers can sample from:

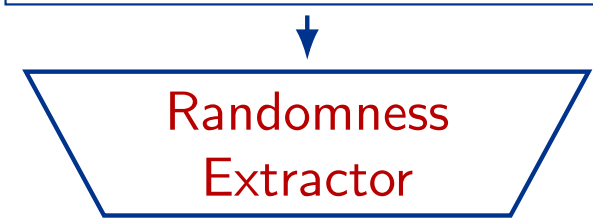
- Electro-magnetic noise (Intel)
- Key strokes of user (Unix)
- Timing of past events (Unix)

These distributions are somewhat random but not truly random.

Paradigm: randomness extractors

"weak source of randomness"





How do computers obtain random bits?

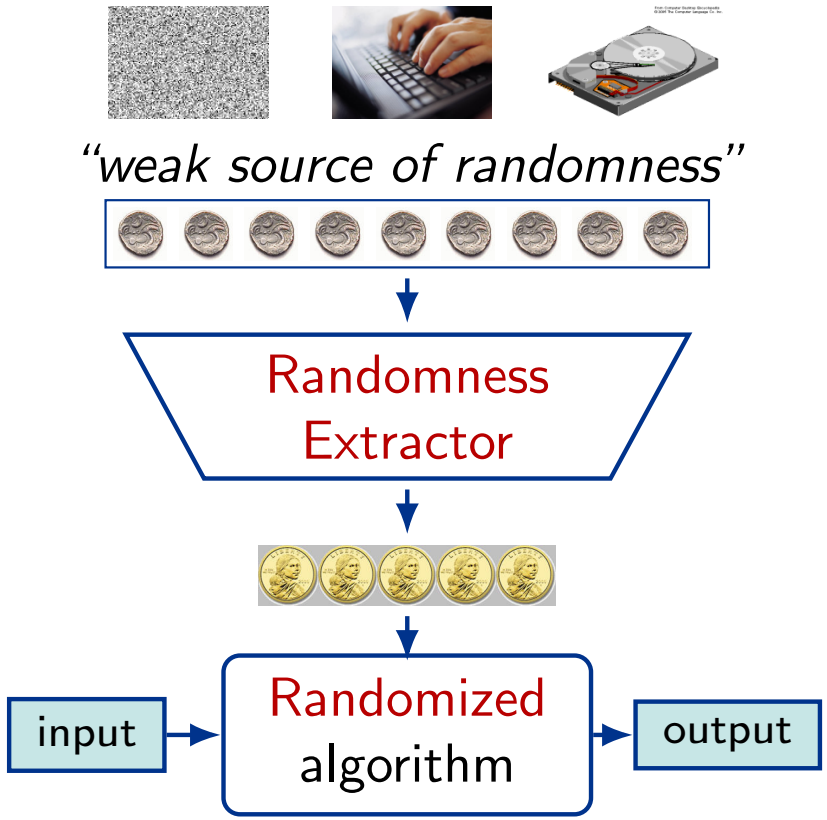
Eshan will give detailed talks on randomness extractors.

Computers can sample from:

- Electro-magnetic noise (Intel)
- Key strokes of user (Unix)
- Timing of past events (Unix)

These distributions are somewhat random but not truly random.

Paradigm: randomness extractors



How do computers obtain random bits?

Eshan will give detailed talks on randomness extractors.

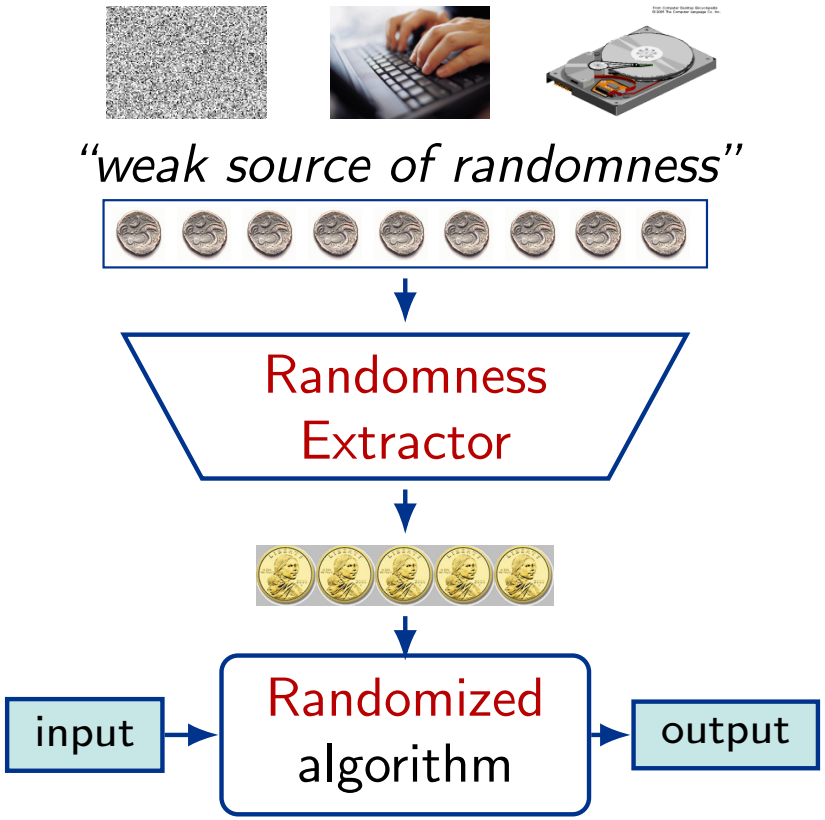
Computers can sample from:

- Electro-magnetic noise (Intel)
- Key strokes of user (Unix)
- Timing of past events (Unix)

These distributions are somewhat random but not truly random.

Paradigm: randomness extractors

Input: one sample from arbitrary "weak source of randomness".



How do computers obtain random bits?

Eshan will give detailed talks on randomness extractors.

Computers can sample from:

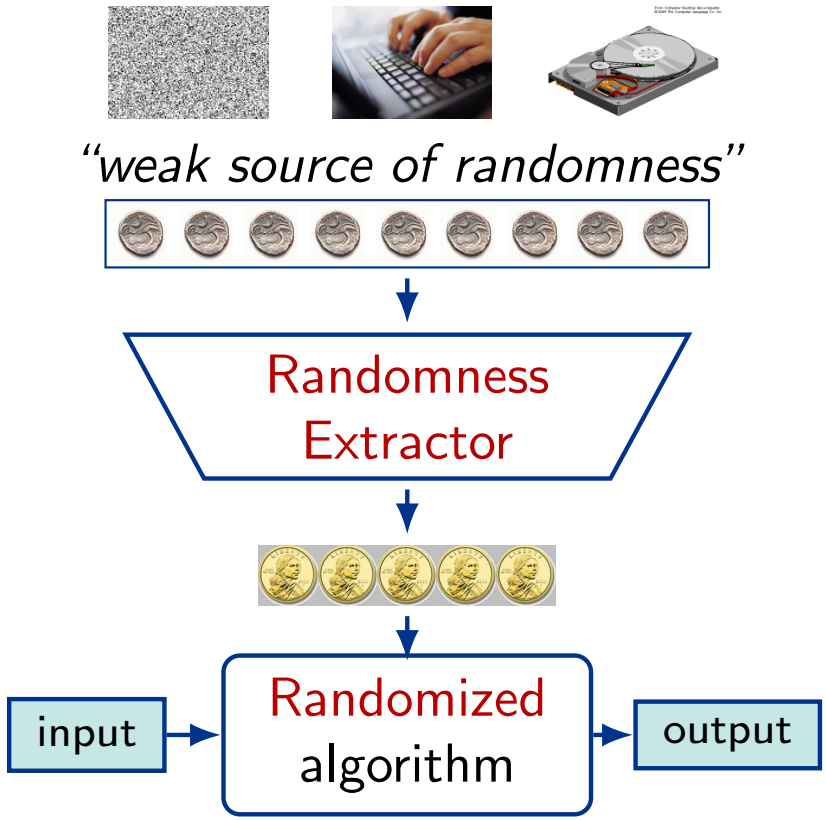
- Electro-magnetic noise (Intel)
- Key strokes of user (Unix)
- Timing of past events (Unix)

These distributions are somewhat random but not truly random.

Paradigm: randomness extractors

Input: one sample from arbitrary "weak source of randomness".

Output: independent coin tosses.



How do computers obtain random bits?

Eshan will give detailed talks on randomness extractors.

Computers can sample from:

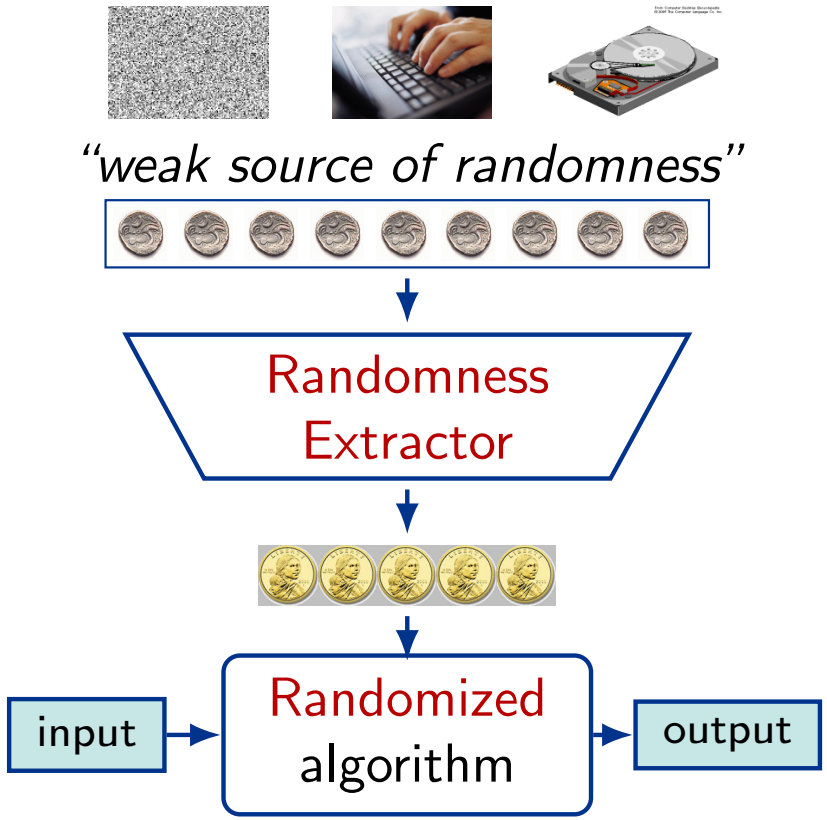
- Electro-magnetic noise (Intel)
- Key strokes of user (Unix)
- Timing of past events (Unix)

These distributions are somewhat random but not truly random.

Paradigm: randomness extractors

Input: one sample from arbitrary "weak source of randomness".

Output: independent coin tosses.



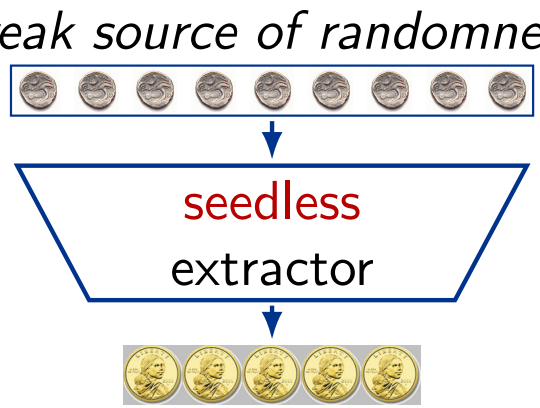
Extensively studied area, dates back to von-Neumann in 1951.

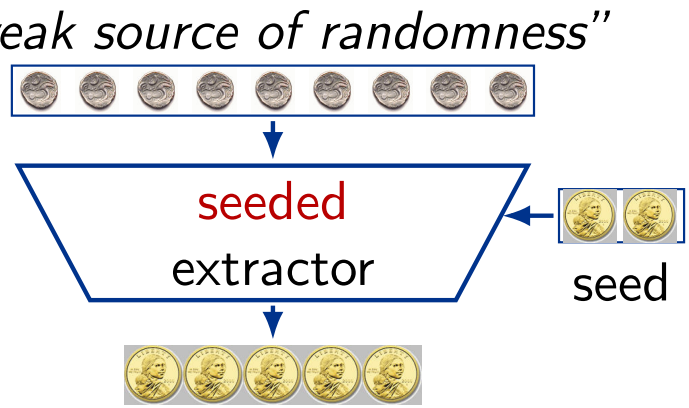
Seeded extractors

Extractors come in two flavors: seeded and deterministic (seedless).

Seeded extractors

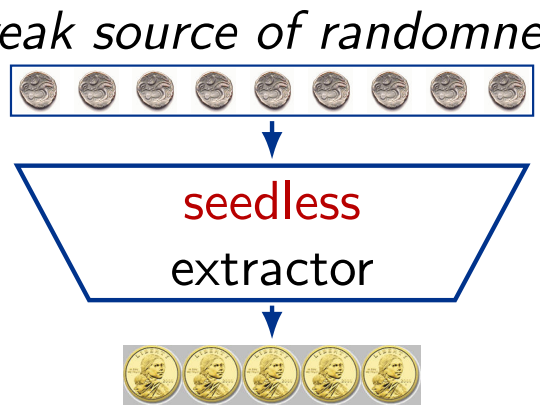
Extractors come in two flavors: seeded and deterministic (seedless).

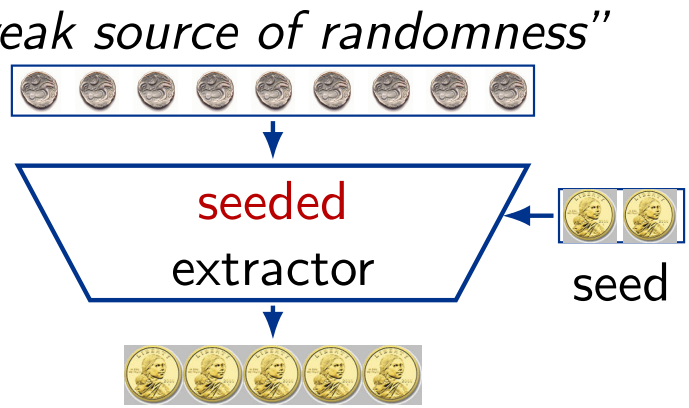




Seeded extractors

Extractors come in two flavors: seeded and deterministic (seedless).



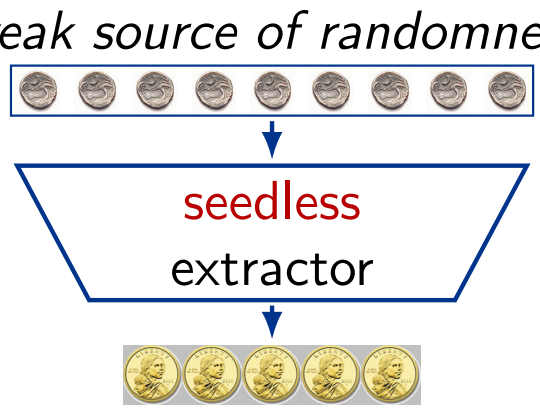


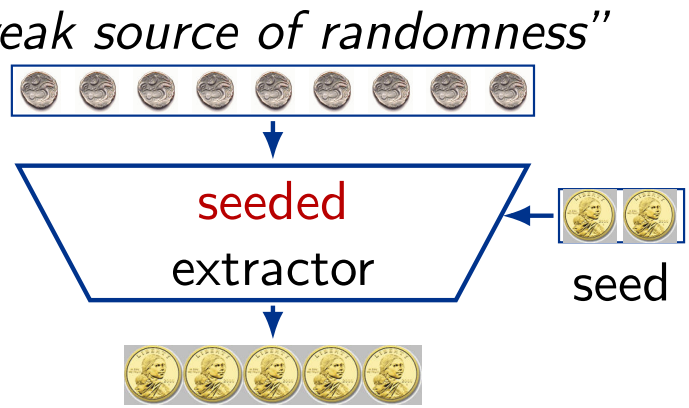
Definition (min-entropy)

For a distribution X, $H_\infty(X) = \min_{x \in \text{Support}(X)} \log \frac{1}{\Pr[X=x]}$.

Seeded extractors

Extractors come in two flavors: seeded and deterministic (seedless).





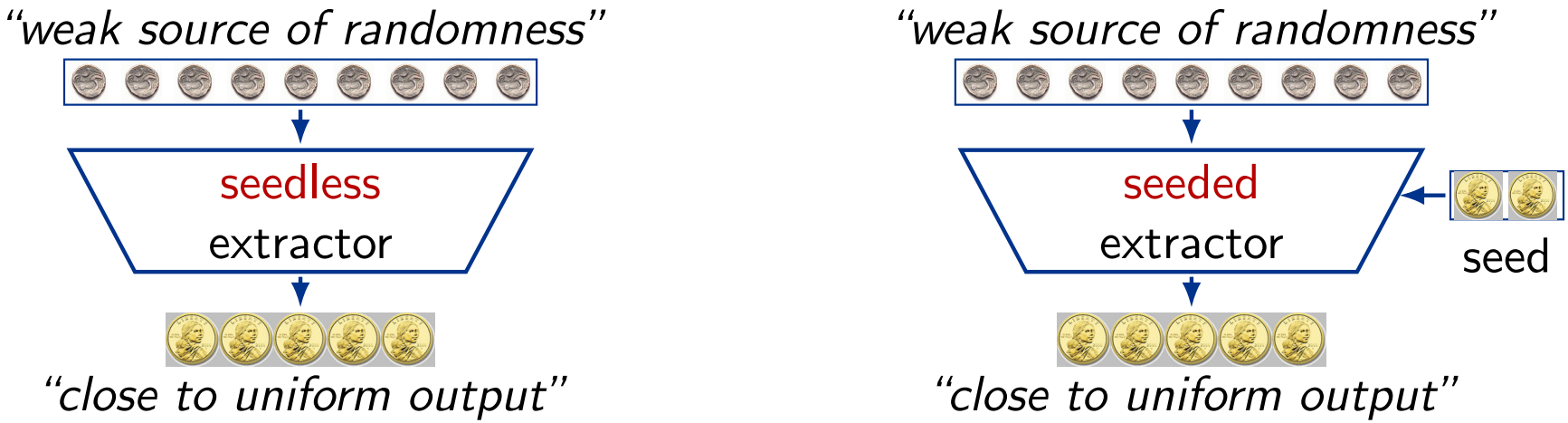
Definition (min-entropy)

For a distribution X, $H_\infty(X) = \min_{x \in \text{Support}(X)} \log \frac{1}{\Pr[X=x]}$.

Example: X that is uniform over a subset $S \subseteq \{0,1\}^n$ of size 2^k has $H_\infty(X) = k$.

Seeded extractors

Extractors come in two flavors: seeded and deterministic (seedless).



Definition (min-entropy)

For a distribution X, $H_\infty(X) = \min_{x \in \text{Support(X)}} \log \frac{1}{\Pr[X=x]}$.

Example: X that is uniform over a subset $S \subseteq \{0,1\}^n$ of size 2^k has $H_\infty(X) = k$.

Definition (Seeded extractors [NZ92])

$E : \{0,1\}^n \times \{0,1\}^r \to \{0,1\}^m$ is a (k, ϵ)-extractor if $\forall X$ with $H_\infty(X) \geq k$, $E(X, U_r)$ is ϵ-close to U_m.

Meaning that $\forall D : \{0,1\}^m \to \{0,1\}$, (not necessarily efficient)
$|\Pr[D(E(X, U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon$.

Brief history of seeded extractors

Definition (Seeded extractors [NZ92])

$E : \{0,1\}^n \times \{0,1\}^r \to \{0,1\}^m$ is a (k, ϵ)-extractor if $\forall X$ with $H_\infty(X) \geq k$, $E(X, U_r)$ is ϵ-close to U_m.

Meaning that $\forall D : \{0,1\}^m \to \{0,1\}$, (not necessarily efficient)
$|\Pr[D(E(X, U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon$.

Brief history of seeded extractors

Definition (Seeded extractors [NZ92])

$E : \{0,1\}^n \times \{0,1\}^r \to \{0,1\}^m$ is a (k, ϵ)-extractor if $\forall X$ with $H_\infty(X) \geq k$, $E(X, U_r)$ is ϵ-close to U_m.

Meaning that $\forall D : \{0,1\}^m \to \{0,1\}$, (not necessarily efficient)
$|\Pr[D(E(X, U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon$.

Theorem (Simulating BPP with a weak random source, Informal)

Seed length $r = O(\log n)$ and output length $m = n^{\Omega(1)}$ $\Rightarrow$ seeded extractors can be used to simulate any poly-time randomized algorithm with access to an arbitrary weak random source.

Brief history of seeded extractors

Definition (Seeded extractors [NZ92])

$E : \{0,1\}^n \times \{0,1\}^r \to \{0,1\}^m$ is a (k, ϵ)-extractor if $\forall X$ with $H_\infty(X) \geq k$, $E(X, U_r)$ is ϵ-close to U_m.

Meaning that $\forall D : \{0,1\}^m \to \{0,1\}$, (not necessarily efficient)
$|\Pr[D(E(X, U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon$.

Theorem (Simulating BPP with a weak random source, Informal)

Seed length $r = O(\log n)$ and output length $m = n^{\Omega(1)}$ $\Rightarrow$ seeded extractors can be used to simulate any poly-time randomized algorithm with access to an arbitrary weak random source.

Extractors are "information theoretic" objects, and were constructed using information theoretic methods.

Brief history of seeded extractors

Definition (Seeded extractors [NZ92])

$E : \{0,1\}^n \times \{0,1\}^r \to \{0,1\}^m$ is a (k, ϵ)-extractor if $\forall X$ with $H_\infty(X) \geq k$, $E(X, U_r)$ is ϵ-close to U_m.

Meaning that $\forall D : \{0,1\}^m \to \{0,1\}$, (not necessarily efficient)
$|\Pr[D(E(X, U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon$.

Theorem (Simulating BPP with a weak random source, Informal)

Seed length $r = O(\log n)$ and output length $m = n^{\Omega(1)}$ $\Rightarrow$ seeded extractors can be used to simulate any poly-time randomized algorithm with access to an arbitrary weak random source.

Extractors are "information theoretic" objects, and were constructed using information theoretic methods.

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ seeded extractor.

Brief history of seeded extractors

Definition (Seeded extractors [NZ92])

$E : \{0,1\}^n \times \{0,1\}^r \to \{0,1\}^m$ is a (k, ϵ)-extractor if $\forall X$ with $H_\infty(X) \geq k$, $E(X, U_r)$ is ϵ-close to U_m.

Meaning that $\forall D : \{0,1\}^m \to \{0,1\}$, (not necessarily efficient)
$|\Pr[D(E(X, U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon$.

Theorem (Simulating BPP with a weak random source, Informal)

Seed length $r = O(\log n)$ and output length $m = n^{\Omega(1)}$ $\Rightarrow$ seeded extractors can be used to simulate any poly-time randomized algorithm with access to an arbitrary weak random source.

Extractors are "information theoretic" objects, and were constructed using information theoretic methods.

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ seeded extractor.

Consequence: The construction of PRGs that we covered yields seeded extractors.

Brief history of seeded extractors

Definition (Seeded extractors [NZ92])

$E : \{0,1\}^n \times \{0,1\}^r \to \{0,1\}^m$ is a (k, ϵ)-extractor if $\forall X$ with $H_\infty(X) \geq k$, $E(X, U_r)$ is ϵ-close to U_m.

Meaning that $\forall D : \{0,1\}^m \to \{0,1\}$, (not necessarily efficient)
$|\Pr[D(E(X, U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon$.

Theorem (Simulating BPP with a weak random source, Informal)

Seed length $r = O(\log n)$ and output length $m = n^{\Omega(1)}$ $\Rightarrow$ seeded extractors can be used to simulate any poly-time randomized algorithm with access to an arbitrary weak random source.

Extractors are "information theoretic" objects, and were constructed using information theoretic methods.

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ seeded extractor.

Consequence: The construction of PRGs that we covered yields seeded extractors.

▸ Conclude talk

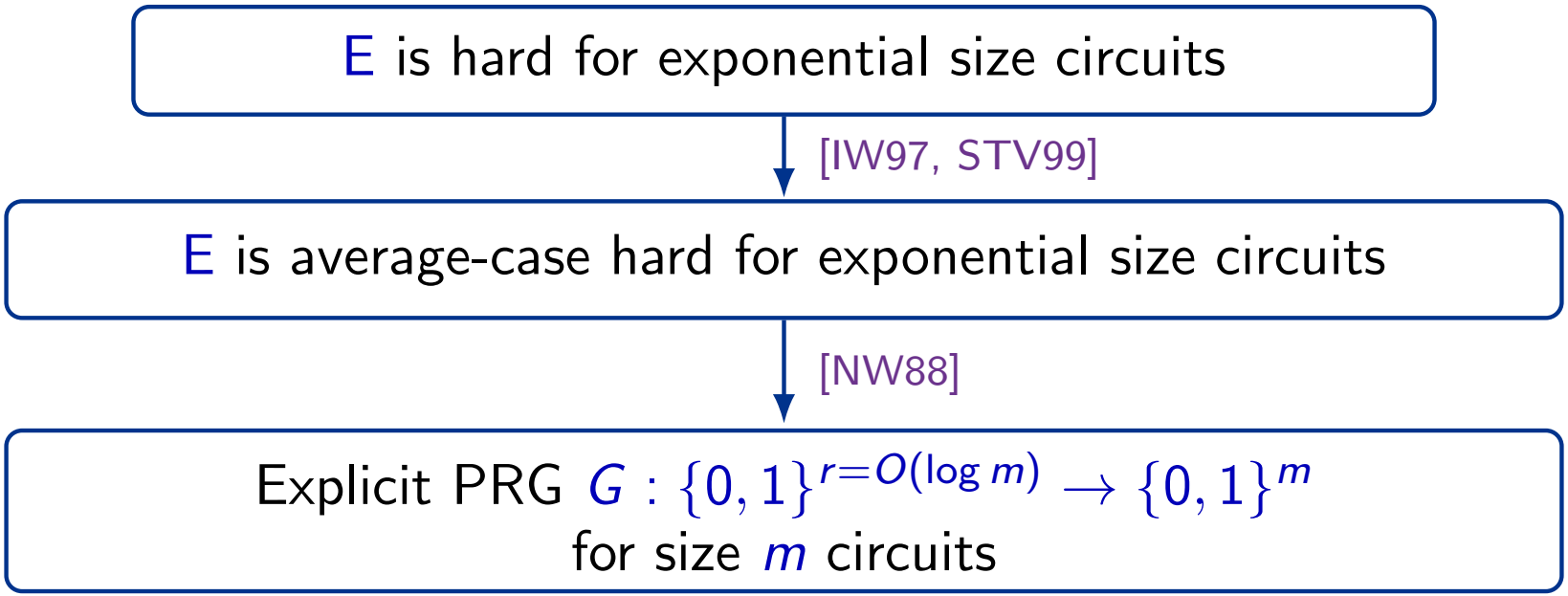
Hardness $\Rightarrow$ PRGs (review of argument):

Theorem (IW97)

E *is hard for exponential size circuits* $\Rightarrow$ *there is an explicit PRG*

$$G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$$

for size m *circuits.*



Hardness $\Rightarrow$ PRGs (fully black-box view)

Theorem (hardness $\Rightarrow$ PRGs - fully-black-box view)

Hardness $\Rightarrow$ PRGs (fully black-box view)

Theorem (hardness $\Rightarrow$ PRGs - fully-black-box view)

- $\exists$ *oracle TM* $G^{(\cdot)}$ *s.t.* $\forall f : \{0,1\}^{\ell} \to \{0,1\}$, *setting* $m = 2^{\Omega(\ell)}$: $G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *is computable in* E^f.

Hardness $\Rightarrow$ PRGs (fully black-box view)

Theorem (hardness $\Rightarrow$ PRGs - fully-black-box view)

- $\exists$ *oracle TM* $G^{(\cdot)}$ *s.t.* $\forall f : \{0,1\}^{\ell} \to \{0,1\}$, *setting* $m = 2^{\Omega(\ell)}$: $G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *is computable in* E^f.

Hard function

f

Hardness $\Rightarrow$ PRGs (fully black-box view)

Theorem (hardness $\Rightarrow$ PRGs - fully-black-box view)

- $\exists$ *oracle TM* $G^{(\cdot)}$ *s.t.* $\forall f : \{0,1\}^{\ell} \to \{0,1\}$, *setting* $m = 2^{\Omega(\ell)}$: $G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *is computable in* E^f.

Hard function

f ——construction——▶

Hardness $\Rightarrow$ PRGs (fully black-box view)

Theorem (hardness $\Rightarrow$ PRGs - fully-black-box view)

- $\exists$ *oracle TM* $G^{(\cdot)}$ *s.t.* $\forall f : \{0,1\}^{\ell} \to \{0,1\}$, *setting* $m = 2^{\Omega(\ell)}$: $G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *is computable in* E^f.

Hard function **PRG**

$$f \xrightarrow{\text{construction}} G^f$$

Hardness $\Rightarrow$ PRGs (fully black-box view)

Theorem (hardness $\Rightarrow$ PRGs - fully-black-box view)

- $\exists$ *oracle TM* $G^{(\cdot)}$ *s.t.* $\forall f : \{0,1\}^{\ell} \to \{0,1\}$, *setting* $m = 2^{\Omega(\ell)}$: $G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *is computable in* E^f.
  Consequently, $f \in \mathsf{E} \Rightarrow G^f$ *is explicit.*

Hard function **PRG**

$$f \xrightarrow{\text{construction}} G^f$$

Hardness $\Rightarrow$ PRGs (fully black-box view)

Theorem (hardness $\Rightarrow$ PRGs - fully-black-box view)

- $\exists$ *oracle TM* $G^{(\cdot)}$ *s.t.* $\forall f : \{0,1\}^{\ell} \to \{0,1\}$, *setting* $m = 2^{\Omega(\ell)}$: $G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *is computable in* E^f.
  Consequently, $f \in \mathsf{E} \Rightarrow G^f$ *is explicit.*
- $\forall D : \{0,1\}^m \to \{0,1\}$ *(not necessarily a small circuit), if*

$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon = 1/10,$$

then $\exists$ *size* $2^{o(\ell)}$ *oracle-circuit* $C^{(\cdot)}$ *s.t.* $\forall x \in \{0,1\}^{\ell}$, $C^D(x) = f(x)$.

Hard function **PRG**

$$f \xrightarrow{\text{construction}} G^f$$

Hardness $\Rightarrow$ PRGs (fully black-box view)

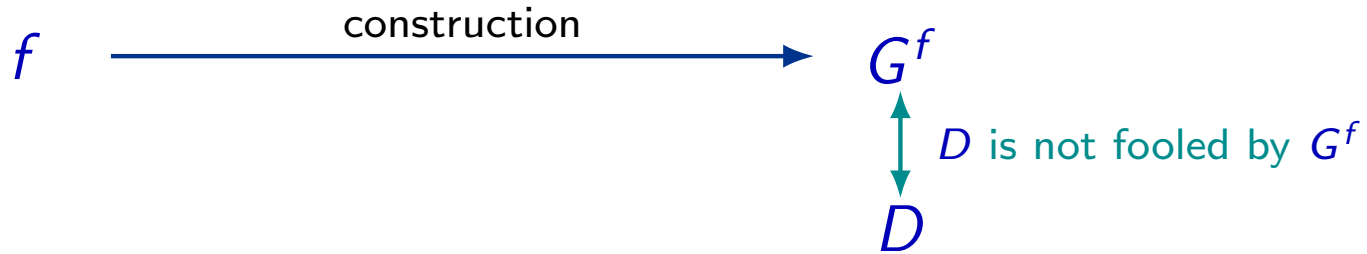
Theorem (hardness $\Rightarrow$ PRGs - fully-black-box view)

- $\exists$ *oracle TM* $G^{(\cdot)}$ *s.t.* $\forall f : \{0,1\}^\ell \to \{0,1\}$, *setting* $m = 2^{\Omega(\ell)}$: $G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *is computable in* E^f.
  Consequently, $f \in \mathsf{E} \Rightarrow G^f$ *is explicit.*
- $\forall D : \{0,1\}^m \to \{0,1\}$ *(not necessarily a small circuit), if*

$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon = 1/10,$$

then $\exists$ *size* $2^{o(\ell)}$ *oracle-circuit* $C^{(\cdot)}$ *s.t.* $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.

Hard function **PRG**



Hardness $\Rightarrow$ PRGs (fully black-box view)

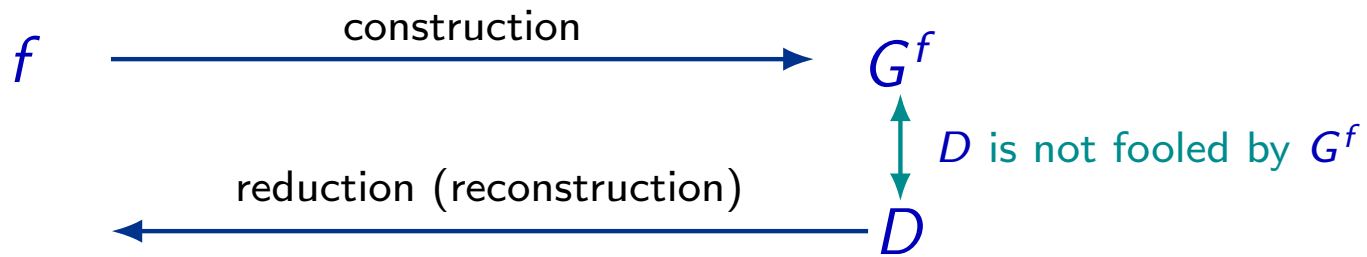
Theorem (hardness $\Rightarrow$ PRGs - fully-black-box view)

- $\exists$ *oracle TM* $G^{(\cdot)}$ *s.t.* $\forall f : \{0,1\}^\ell \to \{0,1\}$, *setting* $m = 2^{\Omega(\ell)}$: $G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *is computable in* E^f.
  Consequently, $f \in \mathrm{E} \Rightarrow G^f$ *is explicit.*
- $\forall D : \{0,1\}^m \to \{0,1\}$ *(not necessarily a small circuit), if*

$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon = 1/10,$$

then $\exists$ *size* $2^{o(\ell)}$ *oracle-circuit* $C^{(\cdot)}$ *s.t.* $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.

Hard function **PRG**



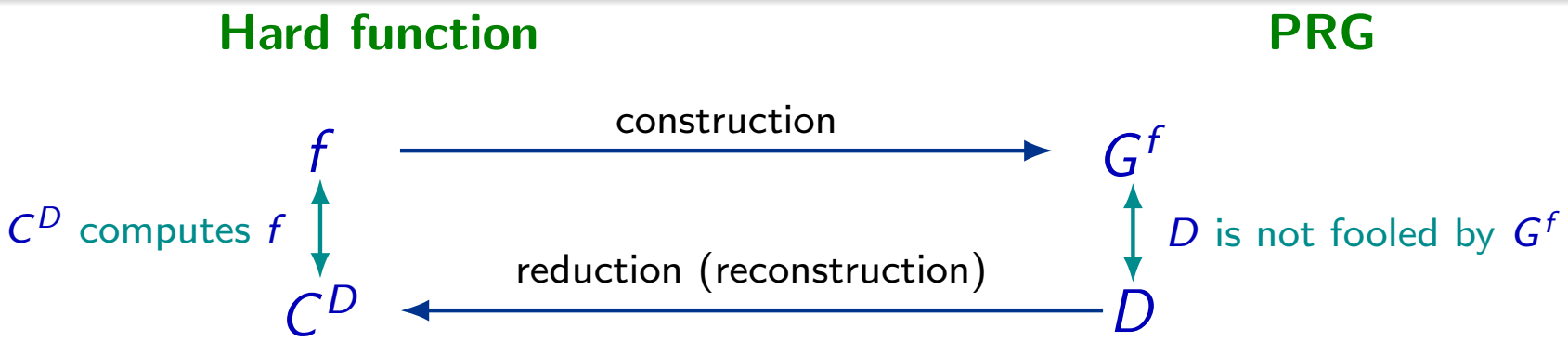
Hardness $\Rightarrow$ PRGs (fully black-box view)

Theorem (hardness $\Rightarrow$ PRGs - fully-black-box view)

- $\exists$ *oracle TM* $G^{(\cdot)}$ *s.t.* $\forall f : \{0,1\}^\ell \to \{0,1\}$, *setting* $m = 2^{\Omega(\ell)}$: $G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *is computable in* E^f.
  Consequently, $f \in \mathrm{E} \Rightarrow G^f$ *is explicit.*
- $\forall D : \{0,1\}^m \to \{0,1\}$ *(not necessarily a small circuit), if*

$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon = 1/10,$$

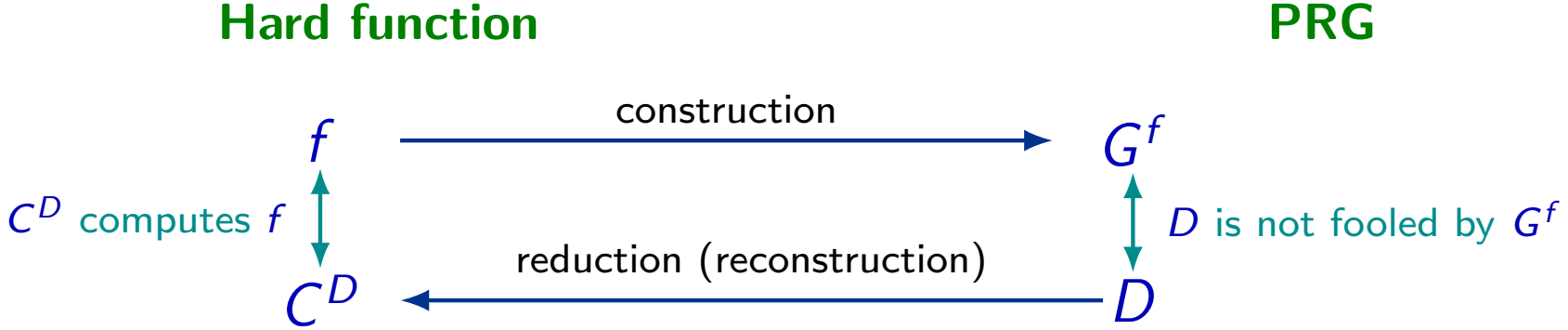
then $\exists$ *size* $2^{o(\ell)}$ *oracle-circuit* $C^{(\cdot)}$ *s.t.* $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.



Hardness $\Rightarrow$ PRGs (fully black-box view)

Theorem (hardness $\Rightarrow$ PRGs - fully-black-box view)

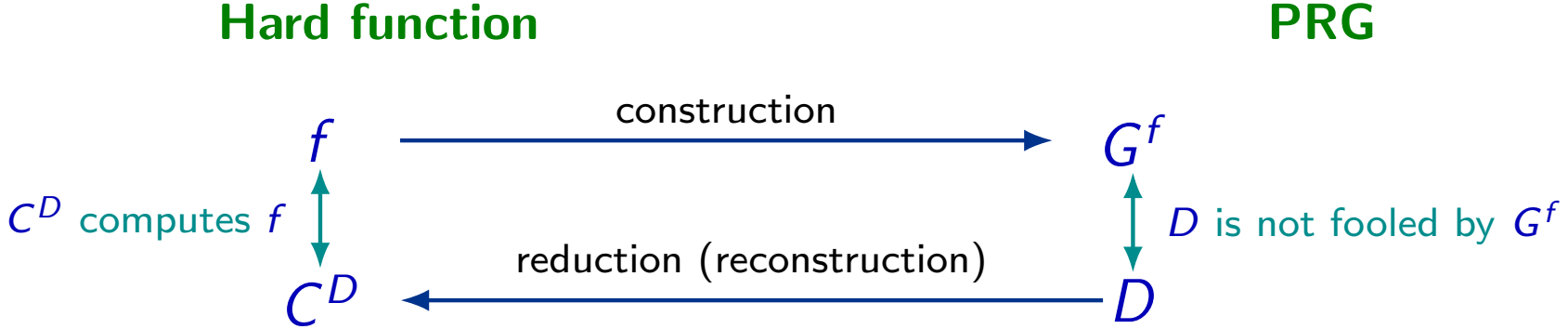
- $\exists$ *oracle TM* $G^{(\cdot)}$ *s.t.* $\forall f : \{0,1\}^\ell \to \{0,1\}$, *setting* $m = 2^{\Omega(\ell)}$: $G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *is computable in* E^f.
  Consequently, $f \in \mathsf{E} \Rightarrow G^f$ *is explicit.*
- $\forall D : \{0,1\}^m \to \{0,1\}$ *(not necessarily a small circuit), if*
$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon = 1/10,$$
  then $\exists$ *size* $2^{o(\ell)}$ *oracle-circuit* $C^{(\cdot)}$ *s.t.* $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.
  Consequently, D *is small and breaks* $G^f \Rightarrow f$ *is not hard.*



Hardness $\Rightarrow$ PRGs (fully black-box view)

Theorem (hardness $\Rightarrow$ PRGs - fully-black-box view)

- $\exists$ *oracle TM* $G^{(\cdot)}$ *s.t.* $\forall f : \{0,1\}^\ell \to \{0,1\}$, *setting* $m = 2^{\Omega(\ell)}$: $G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *is computable in* E^f.
  Consequently, $f \in \mathrm{E} \Rightarrow G^f$ *is explicit.*
- $\forall D : \{0,1\}^m \to \{0,1\}$ *(not necessarily a small circuit), if*
$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon = 1/10,$$
  then $\exists$ *size* $2^{o(\ell)}$ *oracle-circuit* $C^{(\cdot)}$ *s.t.* $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.
  Consequently, D *is small and breaks* $G^f \Rightarrow f$ *is not hard.*



Key point: We do not assume that f or D are efficient (except for the consequently clauses).

Trevisan's insight: black-box PRGs $\leftrightarrow$ extractors

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ *seeded extractor.*

Trevisan's insight: black-box PRGs $\leftrightarrow$ extractors

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ *seeded extractor.*

PRGs and extractors are syntactically different.

Trevisan's insight: black-box PRGs $\leftrightarrow$ extractors

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ *seeded extractor.*

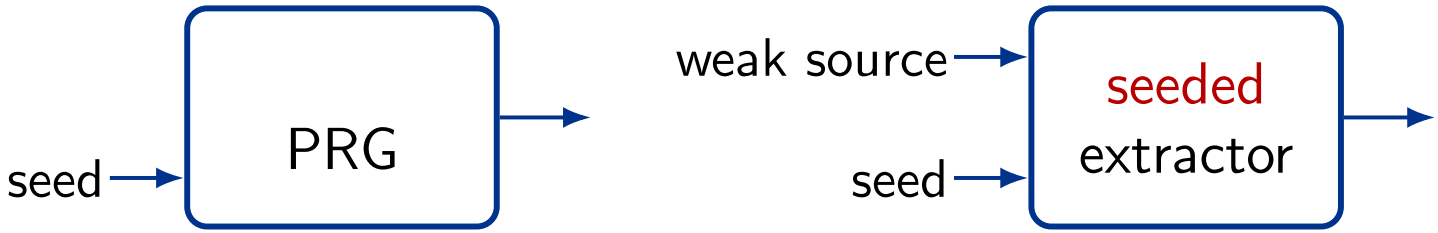
PRGs and extractors are syntactically different.

Trevisan's insight: black-box PRGs $\leftrightarrow$ extractors

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ *seeded extractor.*

PRGs and extractors are syntactically different.

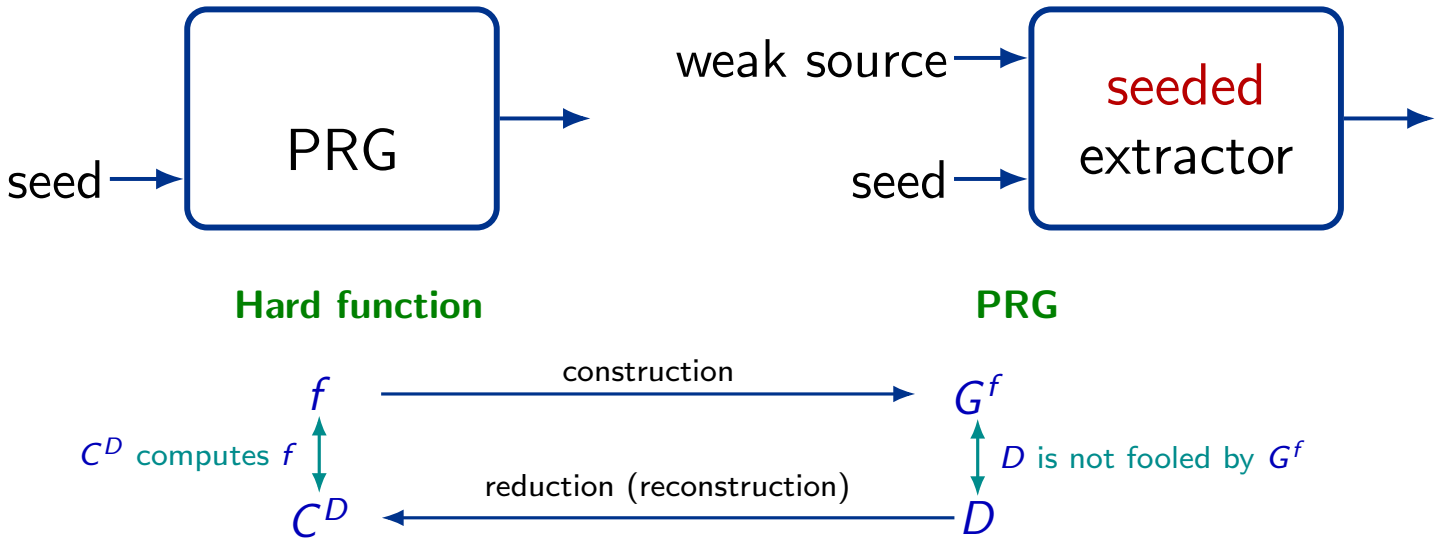


Trevisan's insight: black-box PRGs $\leftrightarrow$ extractors

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ *seeded extractor.*

PRGs and extractors are syntactically different.

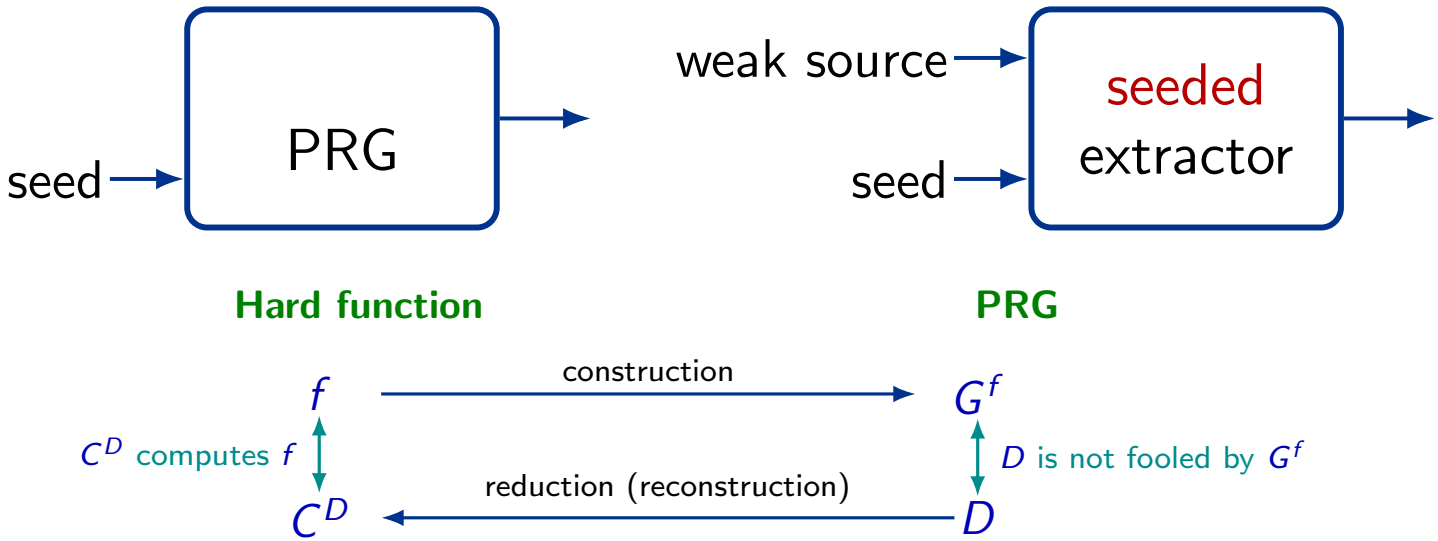


Trevisan's insight: black-box PRGs $\leftrightarrow$ extractors

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ *seeded extractor.*

PRGs and extractors are syntactically different.



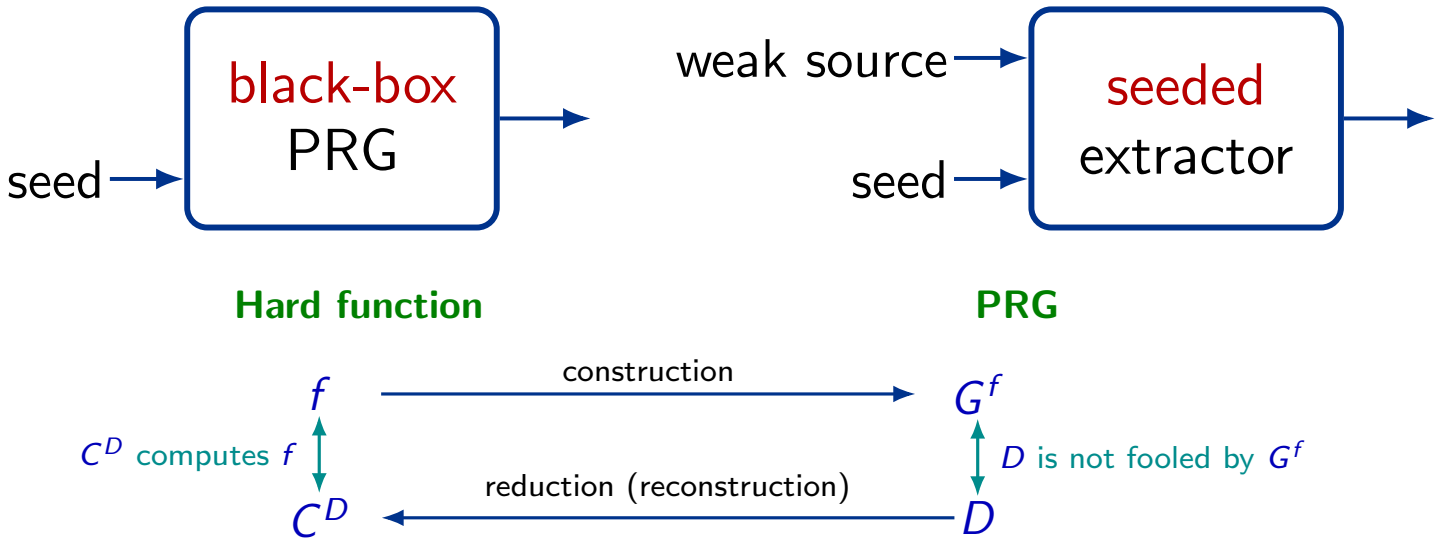
Key insight: A fully black-box PRG construction also receives the truth table of the hard function as input.

Trevisan's insight: black-box PRGs $\leftrightarrow$ extractors

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ *seeded extractor.*

PRGs and extractors are syntactically different.



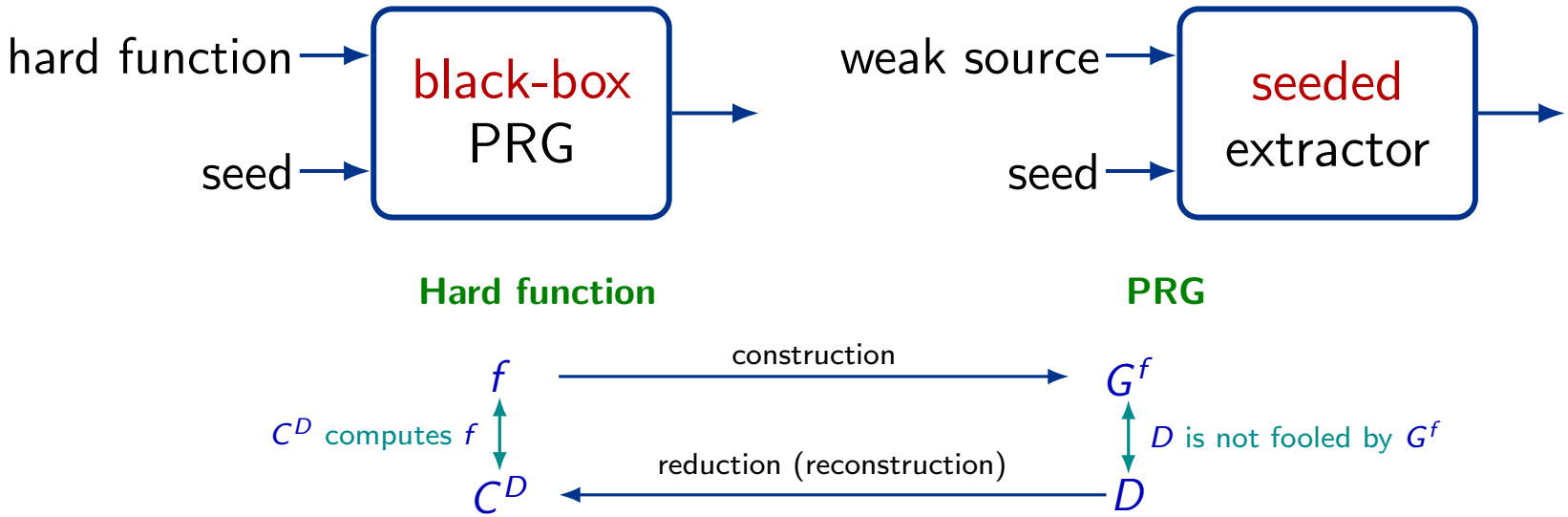
Key insight: A fully black-box PRG construction also receives the truth table of the hard function as input.

Trevisan's insight: black-box PRGs $\leftrightarrow$ extractors

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ seeded extractor.

PRGs and extractors are syntactically different.



Key insight: A fully black-box PRG construction also receives the truth table of the hard function as input.

Trevisan's argument

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ *seeded extractor.*

Trevisan's argument

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ *seeded extractor.*

Definition (Fully black-box PRG)

oracle TM $G^{(\cdot)}$ s.t. $\forall f : \{0,1\}^\ell \to \{0,1\}$, setting $m = 2^{\Omega(\ell)}$:
$G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ is computable in E^f.
s.t. $\forall D : \{0,1\}^m \to \{0,1\}$ (not necessarily a small circuit), if

$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon,$$

then $\exists$size $2^{o(\ell)}$ oracle-circuit $C^{(\cdot)}$ s.t. $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.

Trevisan's argument

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ *seeded extractor.*

Definition (Fully black-box PRG)

oracle TM $G^{(\cdot)}$ s.t. $\forall f : \{0,1\}^\ell \to \{0,1\}$, setting $m = 2^{\Omega(\ell)}$:
$G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ is computable in E^f.
s.t. $\forall D : \{0,1\}^m \to \{0,1\}$ (not necessarily a small circuit), if

$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon,$$

then $\exists$size $2^{o(\ell)}$ oracle-circuit $C^{(\cdot)}$ s.t. $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.

Define: $E(f,x) = G^f(x)$. (f of length $n = 2^\ell$ is "source" and x is "seed").

Trevisan's argument

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ *seeded extractor.*

Definition (Fully black-box PRG)

oracle TM $G^{(\cdot)}$ s.t. $\forall f : \{0,1\}^\ell \to \{0,1\}$, setting $m = 2^{\Omega(\ell)}$:
$G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ is computable in E^f.
s.t. $\forall D : \{0,1\}^m \to \{0,1\}$ (not necessarily a small circuit), if

$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon,$$

then $\exists$size $2^{o(\ell)}$ oracle-circuit $C^{(\cdot)}$ s.t. $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.

Define: $E(f,x) = G^f(x)$. (f of length $n = 2^\ell$ is "source" and x is "seed").

If E is not an extractor, $\exists$distribution F with $H_\infty(F) \geq k := n^{\Omega(1)}$, and
$D : \{0,1\}^m \to \{0,1\}$, s.t. $|\Pr[D(E(F,U_r)) = 1] - \Pr[D(U_m) = 1]| > \epsilon$.

Trevisan's argument

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ *seeded extractor.*

Definition (Fully black-box PRG)

oracle TM $G^{(\cdot)}$ s.t. $\forall f : \{0,1\}^\ell \to \{0,1\}$, setting $m = 2^{\Omega(\ell)}$:
$G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ is computable in E^f.
s.t. $\forall D : \{0,1\}^m \to \{0,1\}$ (not necessarily a small circuit), if

$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon,$$

then $\exists$size $2^{o(\ell)}$ oracle-circuit $C^{(\cdot)}$ s.t. $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.

Define: $E(f,x) = G^f(x)$. (f of length $n = 2^\ell$ is "source" and x is "seed").

If E is not an extractor, $\exists$distribution F with $H_\infty(F) \geq k := n^{\Omega(1)}$, and $D : \{0,1\}^m \to \{0,1\}$, s.t. $|\Pr[D(E(F,U_r)) = 1] - \Pr[D(U_m) = 1]| > \epsilon$.

$\Rightarrow$ For $\approx \epsilon$ fraction of $f \leftarrow F$, $|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]| > \epsilon$.

Trevisan's argument

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ *seeded extractor.*

Definition (Fully black-box PRG)

oracle TM $G^{(\cdot)}$ s.t. $\forall f : \{0,1\}^\ell \to \{0,1\}$, setting $m = 2^{\Omega(\ell)}$:
$G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ is computable in E^f.
s.t. $\forall D : \{0,1\}^m \to \{0,1\}$ (not necessarily a small circuit), if

$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon,$$

then $\exists$size $2^{o(\ell)}$ oracle-circuit $C^{(\cdot)}$ s.t. $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.

Define: $E(f,x) = G^f(x)$. (f of length $n = 2^\ell$ is "source" and x is "seed").

If E is not an extractor, $\exists$distribution F with $H_\infty(F) \geq k := n^{\Omega(1)}$, and
$D : \{0,1\}^m \to \{0,1\}$, s.t. $|\Pr[D(E(F,U_r)) = 1] - \Pr[D(U_m) = 1]| > \epsilon$.

$\Rightarrow$ For $\approx \epsilon$ fraction of $f \leftarrow F$, $|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]| > \epsilon$.

For every such f, $\exists$ size $2^{o(\ell)}$ oracle-circuit $C^{(\cdot)}$ s.t. $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.

Trevisan's argument

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ *seeded extractor.*

Definition (Fully black-box PRG)

oracle TM $G^{(\cdot)}$ s.t. $\forall f : \{0,1\}^\ell \to \{0,1\}$, setting $m = 2^{\Omega(\ell)}$:
$G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ is computable in E^f.
s.t. $\forall D : \{0,1\}^m \to \{0,1\}$ (not necessarily a small circuit), if

$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon,$$

then $\exists$size $2^{o(\ell)}$ oracle-circuit $C^{(\cdot)}$ s.t. $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.

Define: $E(f,x) = G^f(x)$. (f of length $n = 2^\ell$ is "source" and x is "seed").

If E is not an extractor, $\exists$distribution F with $H_\infty(F) \geq k := n^{\Omega(1)}$, and
$D : \{0,1\}^m \to \{0,1\}$, s.t. $|\Pr[D(E(F,U_r)) = 1] - \Pr[D(U_m) = 1]| > \epsilon$.

$\Rightarrow$ For $\approx \epsilon$ fraction of $f \leftarrow F$, $|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]| > \epsilon$.

For every such f, $\exists$ size $2^{o(\ell)}$ oracle-circuit $C^{(\cdot)}$ s.t. $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.

D is fixed $\Rightarrow$ every such f is "described" by C using $2^{o(\ell)} = m^{o(1)} = n^{o(1)}$ bits.

Trevisan's argument

Theorem (Trevisan (informal))

Fully-black box PRG construction $\Rightarrow$ *seeded extractor.*

Definition (Fully black-box PRG)

oracle TM $G^{(\cdot)}$ s.t. $\forall f : \{0,1\}^\ell \to \{0,1\}$, setting $m = 2^{\Omega(\ell)}$:
$G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ is computable in E^f.
s.t. $\forall D : \{0,1\}^m \to \{0,1\}$ (not necessarily a small circuit), if

$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon,$$

then $\exists$size $2^{o(\ell)}$ oracle-circuit $C^{(\cdot)}$ s.t. $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.

Define: $E(f,x) = G^f(x)$. (f of length $n = 2^\ell$ is "source" and x is "seed").

If E is not an extractor, $\exists$distribution F with $H_\infty(F) \geq k := n^{\Omega(1)}$, and
$D : \{0,1\}^m \to \{0,1\}$, s.t. $|\Pr[D(E(F,U_r)) = 1] - \Pr[D(U_m) = 1]| > \epsilon$.

$\Rightarrow$ For $\approx \epsilon$ fraction of $f \leftarrow F$, $|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]| > \epsilon$.

For every such f, $\exists$ size $2^{o(\ell)}$ oracle-circuit $C^{(\cdot)}$ s.t. $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.

D is fixed $\Rightarrow$ every such f is "described" by C using $2^{o(\ell)} = m^{o(1)} = n^{o(1)}$ bits.

Impossible, because an $f \leftarrow F$ is very unlikely to have such a short description C.

Reflection: usefulness of black-box view of PRGs

Trevisan's principle:

Reflection: usefulness of black-box view of PRGs

Trevisan's principle:

"PRGs" = "seeded extractors" + "efficient reduction".

Reflection: usefulness of black-box view of PRGs

Trevisan's principle:

"PRGs" = "seeded extractors" + "efficient reduction".

"Computational Construction" = "info-theory object" + "efficient reduction".

Reflection: usefulness of black-box view of PRGs

Trevisan's principle:

"PRGs" = "seeded extractors" + "efficient reduction".

"Computational Construction" = "info-theory object" + "efficient reduction".

"Hardness amplification" = "(list)-decoding" + "efficient reduction".

Reflection: usefulness of black-box view of PRGs

Trevisan's principle:

"PRGs" = "seeded extractors" + "efficient reduction".

"Computational Construction" = "info-theory object" + "efficient reduction".

"Hardness amplification" = "(list)-decoding" + "efficient reduction".

Many additional examples in recent work.

Reflection: usefulness of black-box view of PRGs

Trevisan's principle:

"PRGs" = "seeded extractors" + "efficient reduction".

"Computational Construction" = "info-theory object" + "efficient reduction".

"Hardness amplification" = "(list)-decoding" + "efficient reduction".

Many additional examples in recent work.

Somewhat surprisingly, this connection turned out to be helpful in both directions.

Reflection: usefulness of black-box view of PRGs

Trevisan's principle:

"PRGs" = "seeded extractors" + "efficient reduction".

"Computational Construction" = "info-theory object" + "efficient reduction".

"Hardness amplification" = "(list)-decoding" + "efficient reduction".

Many additional examples in recent work.

Somewhat surprisingly, this connection turned out to be helpful in both directions.

More generally:
Black-box PRGs have many additional consequences in various setups:

Reflection: usefulness of black-box view of PRGs

Trevisan's principle:

"PRGs" = "seeded extractors" + "efficient reduction".

"Computational Construction" = "info-theory object" + "efficient reduction".

"Hardness amplification" = "(list)-decoding" + "efficient reduction".

Many additional examples in recent work.

Somewhat surprisingly, this connection turned out to be helpful in both directions.

More generally:
Black-box PRGs have many additional consequences in various setups:

General principle: Fully black-box PRG constructions allow us to use many kinds of distinguishers and reinterpret the reconstruction accordingly.

Reflection: usefulness of black-box view of PRGs

Trevisan's principle:

"PRGs" = "seeded extractors" + "efficient reduction".

"Computational Construction" = "info-theory object" + "efficient reduction".

"Hardness amplification" = "(list)-decoding" + "efficient reduction".

Many additional examples in recent work.

Somewhat surprisingly, this connection turned out to be helpful in both directions.

More generally:
Black-box PRGs have many additional consequences in various setups:

General principle: Fully black-box PRG constructions allow us to use many kinds of distinguishers and reinterpret the reconstruction accordingly.

In the 2nd talk, we will also discuss some limitations of "black-box techniques".

Reflection: usefulness of black-box view of PRGs

Trevisan's principle:

"PRGs" = "seeded extractors" + "efficient reduction".

"Computational Construction" = "info-theory object" + "efficient reduction".

"Hardness amplification" = "(list)-decoding" + "efficient reduction".

Many additional examples in recent work.

Somewhat surprisingly, this connection turned out to be helpful in both directions.

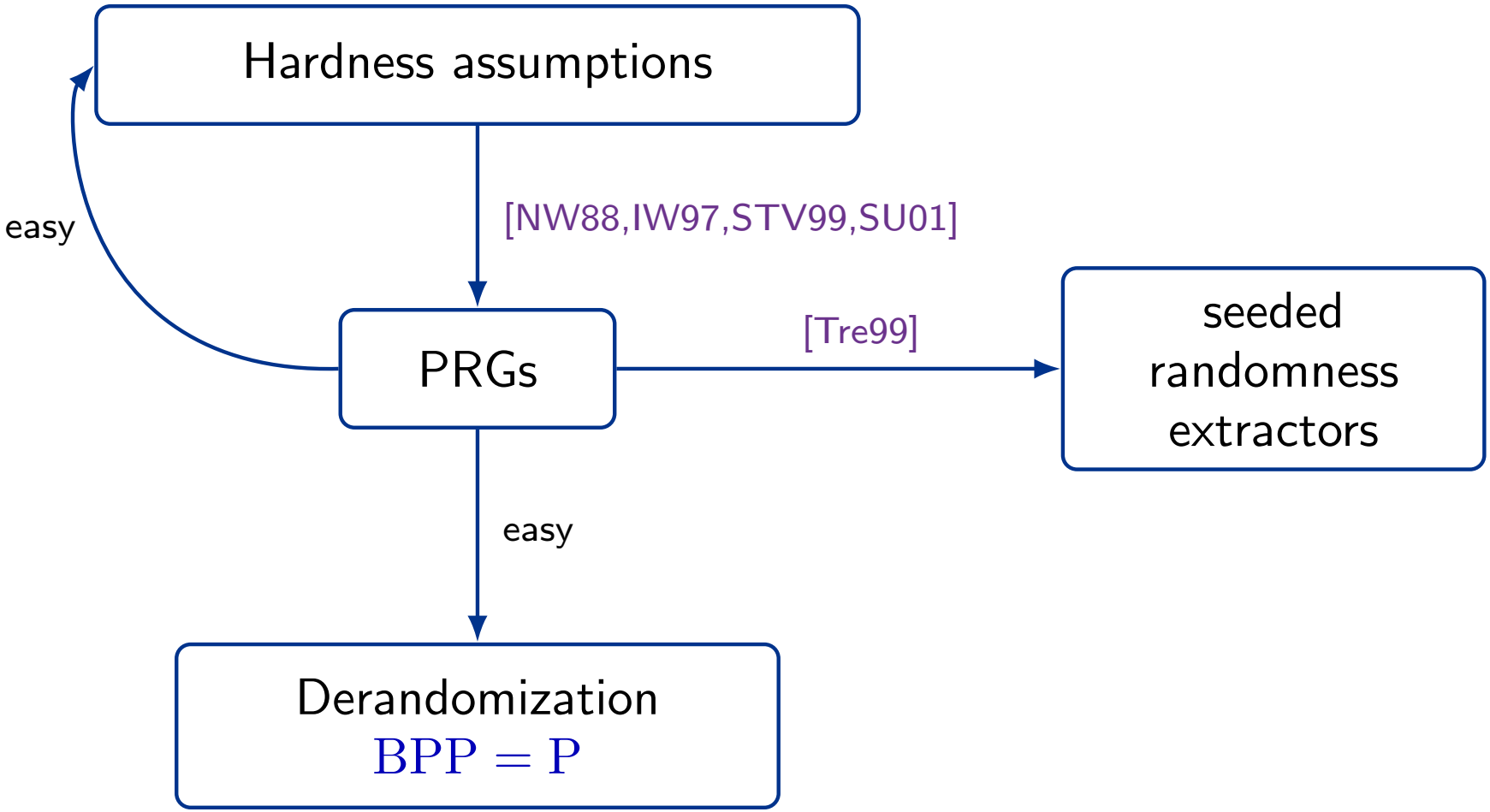
More generally:
Black-box PRGs have many additional consequences in various setups:

General principle: Fully black-box PRG constructions allow us to use many kinds of distinguishers and reinterpret the reconstruction accordingly.

In the 2nd talk, we will also discuss some limitations of "black-box techniques".

▸ Conclude talk

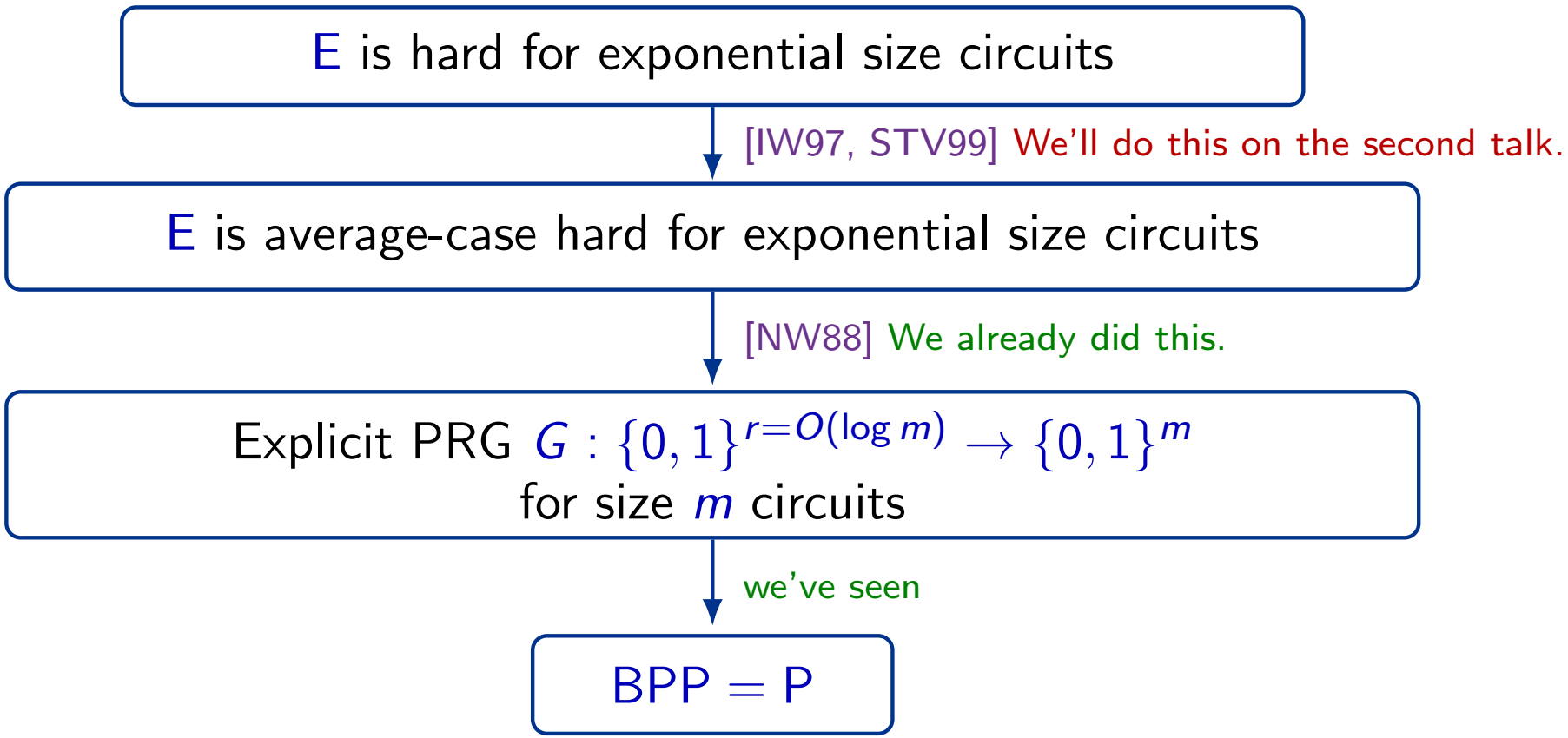
Summary (and where we are)



Hardness $\Rightarrow$ PRGs (Where we are)

Theorem (IW97)

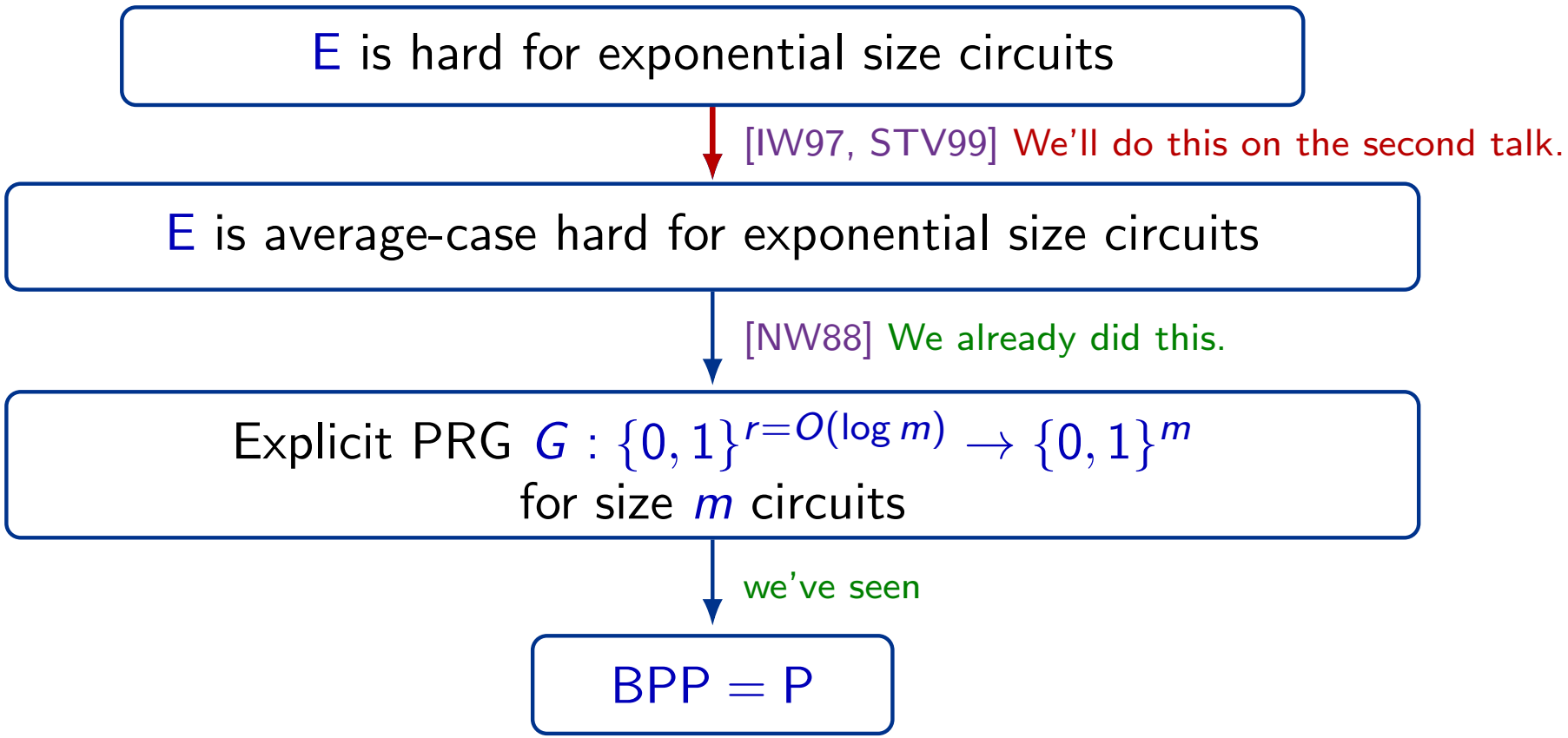
E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.



Hardness $\Rightarrow$ PRGs (Where we are)

Theorem (IW97)

E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.

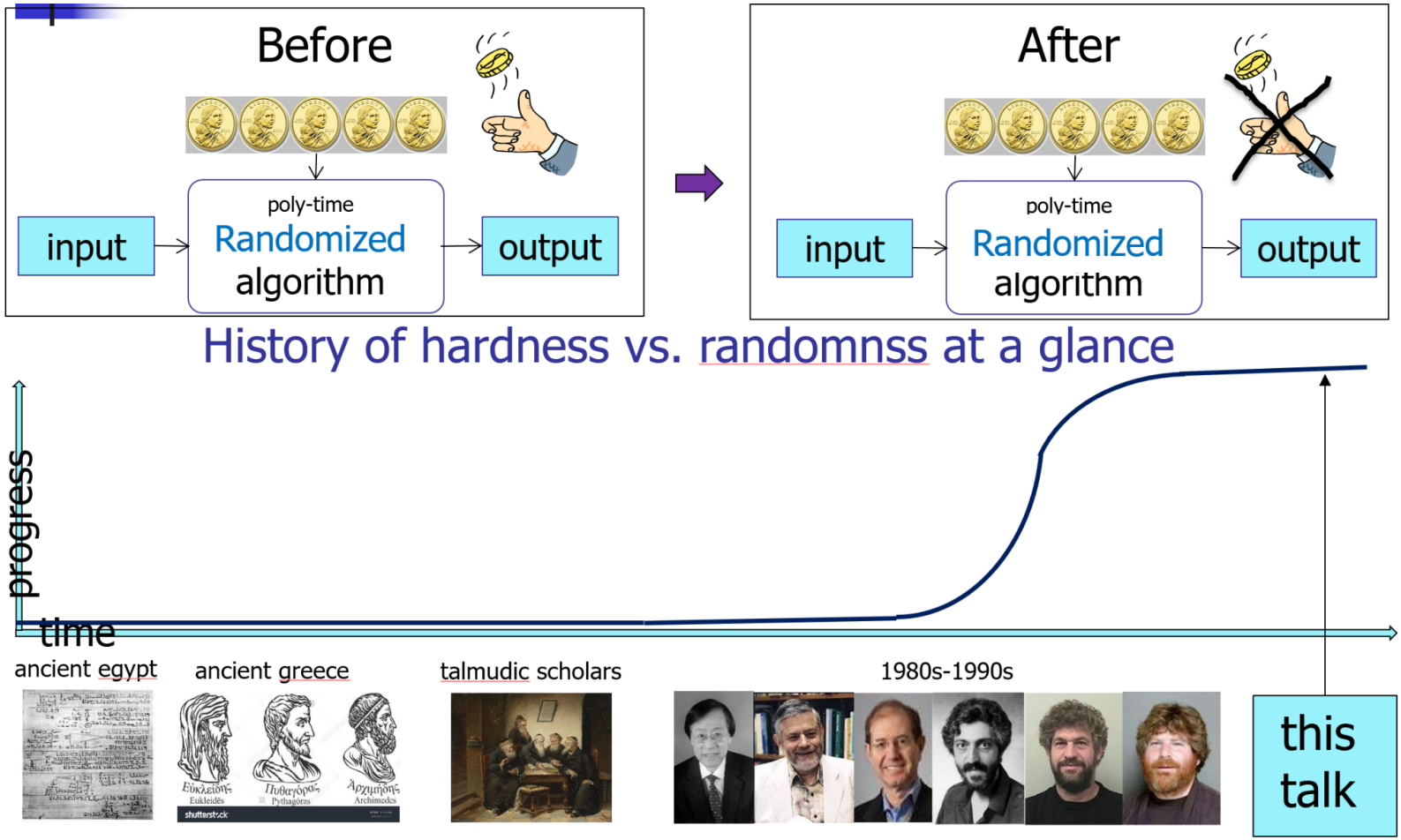


Current situation in hardness vs. randomness

Excerpt from a 2016 talk on hardness vs. randomness for Avi's birthday.

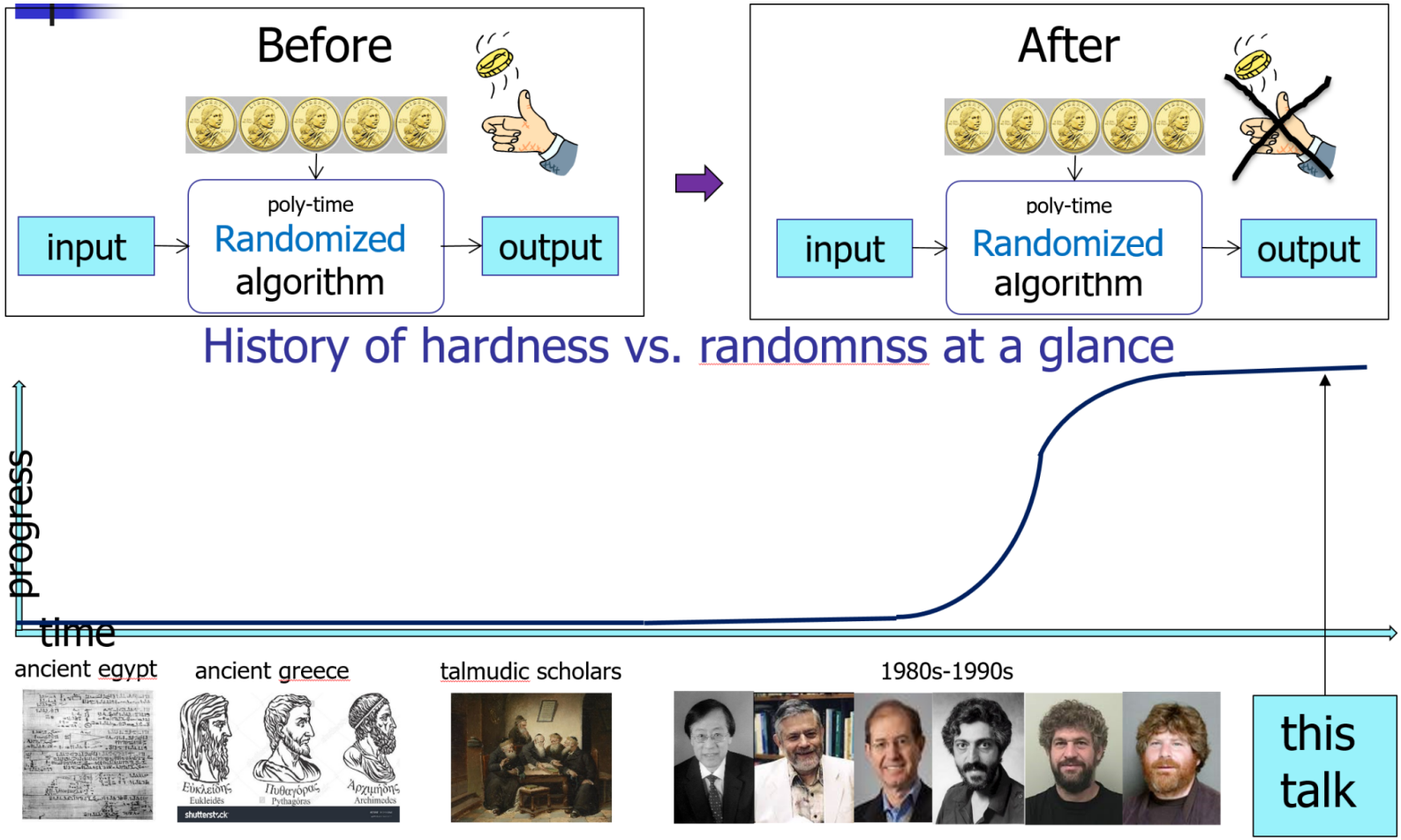
Current situation in hardness vs. randomness

Excerpt from a 2016 talk on hardness vs. randomness for Avi's birthday.



Current situation in hardness vs. randomness

Excerpt from a 2016 talk on hardness vs. randomness for Avi's birthday.



Amazing progress in recent years!

We'll hear about these developments in future talks and seminars.

End of 1st talk

Thank you!

An Introduction to the Hardness vs. Randomness Paradigm

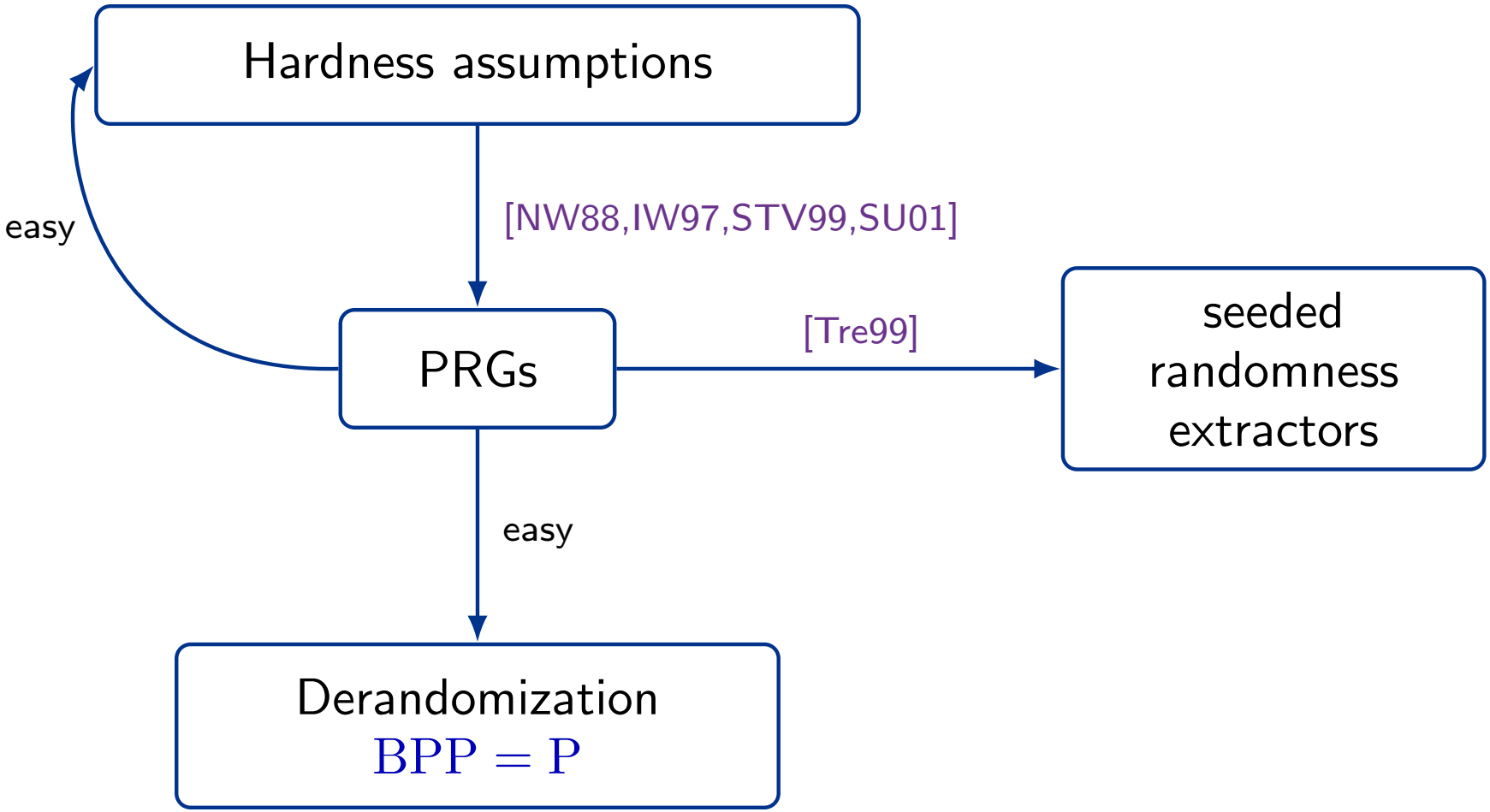
Tutorial for the Simons HDX and Pseudorandomness Workshop

Ronen Shaltiel

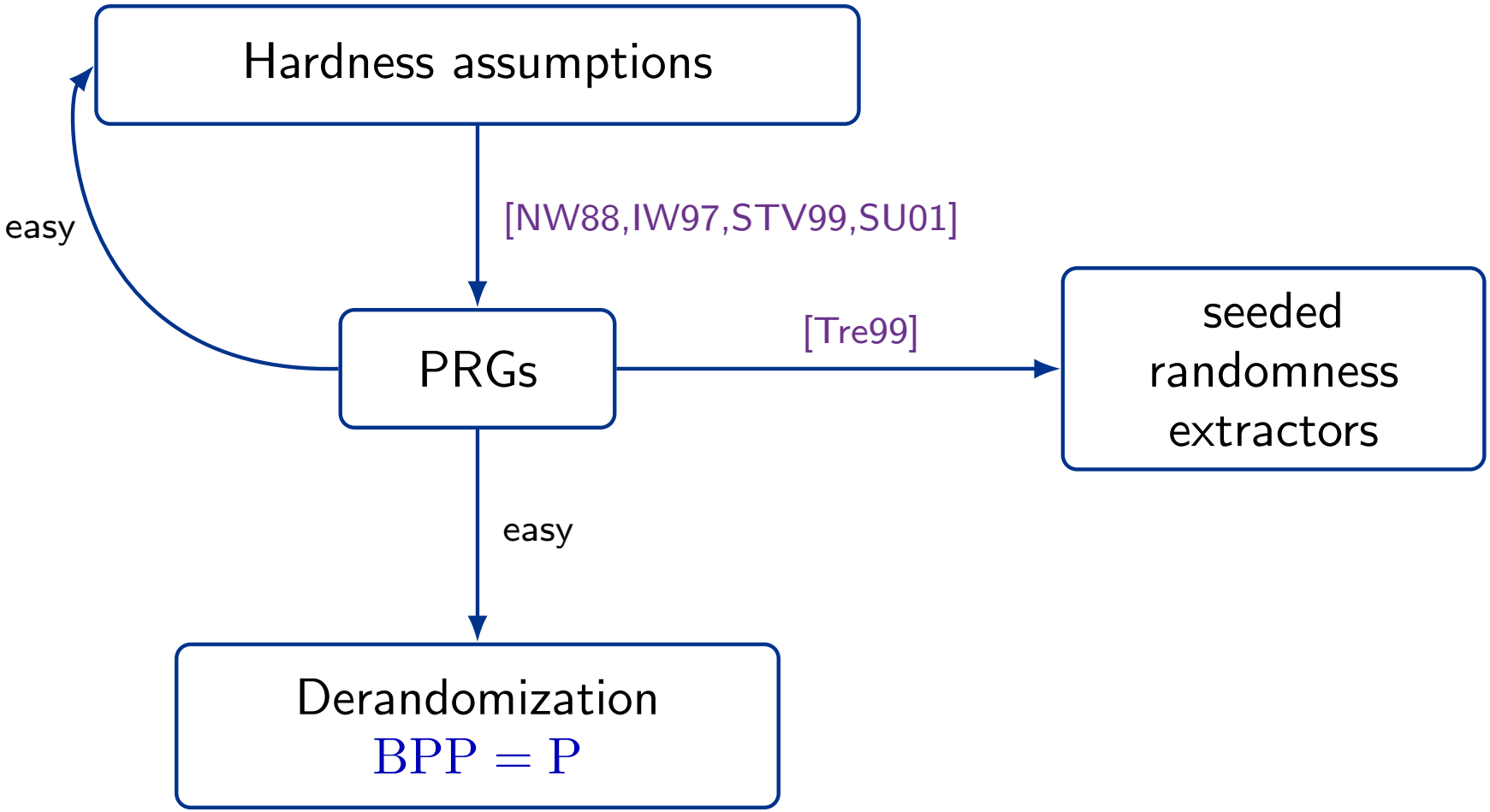
University of Haifa

August 2026

Overall summary for the two talks



Overall summary for the two talks



Then: extensions, generalizations, applications,
and some other research directions.

Reminder of key definitions from first talk

Definition (Pseudorandom generator (PRG))

$G : \{0,1\}^r \to \{0,1\}^m$ is an ϵ-PRG for $D : \{0,1\}^m \to \{0,1\}$ if

$$|\Pr[D(G(U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon.$$

Throughout this talk, $\epsilon = 1/10$.

Reminder of key definitions from first talk

Definition (Pseudorandom generator (PRG))

$G : \{0,1\}^r \to \{0,1\}^m$ is an ϵ-PRG for $D : \{0,1\}^m \to \{0,1\}$ if

$$|\Pr[D(G(U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon.$$

Throughout this talk, $\epsilon = 1/10$.

Definition (Hard functions)

$f : \{0,1\}^n \to \{0,1\}$ is

- s-hard if $\forall$size s circuit C, $\exists x \in \{0,1\}^n$ s.t. $C(x) \neq f(x)$.
- s-average-case-hard if $\forall$size s circuit C, $\Pr_{x \leftarrow U_n}[C(x) = f(x)] \leq \frac{1}{2} + \frac{1}{s}$.

Reminder of key definitions from first talk

Definition (Pseudorandom generator (PRG))

$G : \{0,1\}^r \to \{0,1\}^m$ is an ϵ-PRG for $D : \{0,1\}^m \to \{0,1\}$ if

$$|\Pr[D(G(U_r)) = 1] - \Pr[D(U_m) = 1]| \leq \epsilon.$$

Throughout this talk, $\epsilon = 1/10$.

Definition (Hard functions)

$f : \{0,1\}^n \to \{0,1\}$ is

- s-hard if $\forall$size s circuit C, $\exists x \in \{0,1\}^n$ s.t. $C(x) \neq f(x)$.
- s-average-case-hard if $\forall$size s circuit C, $\Pr_{x \leftarrow U_n}[C(x) = f(x)] \leq \frac{1}{2} + \frac{1}{s}$.

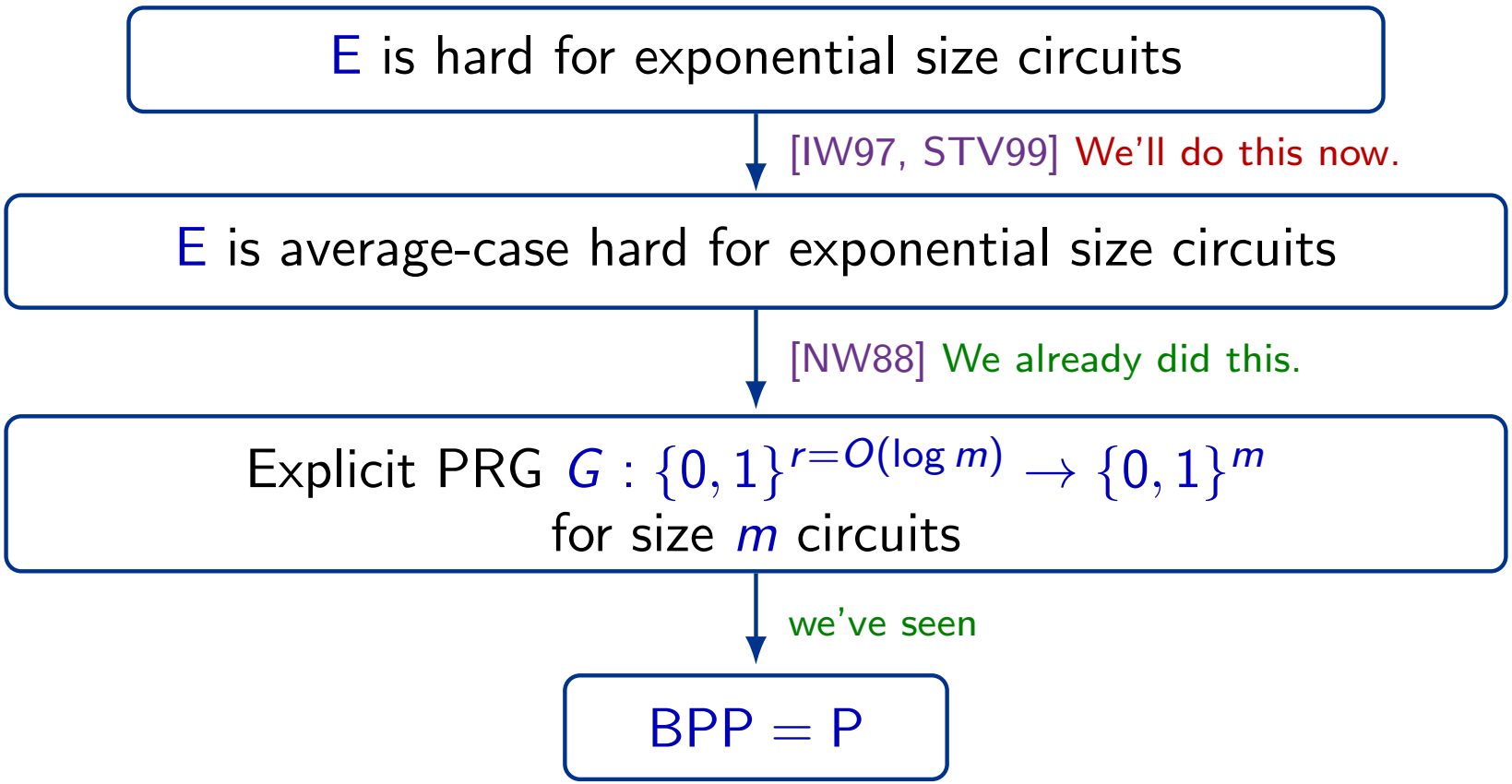
Definition (Hardness assumptions)

- E is hard for exponential size circuits if $\exists \beta > 0$, and $\exists L \in \mathrm{E} = \mathrm{DTIME}(2^{O(n)})$ s.t. $\forall n$, 1_L is $2^{\beta n}$-hard (on inputs of length n).
- E is average-case hard for exponential size circuits if $\exists \beta > 0$, and $\exists L \in \mathrm{E} = \mathrm{DTIME}(2^{O(n)})$ s.t. $\forall n$, 1_L is $2^{\beta n}$-average-case-hard.

Hardness $\Rightarrow$ PRGs (Where we are)

Theorem (IW97)

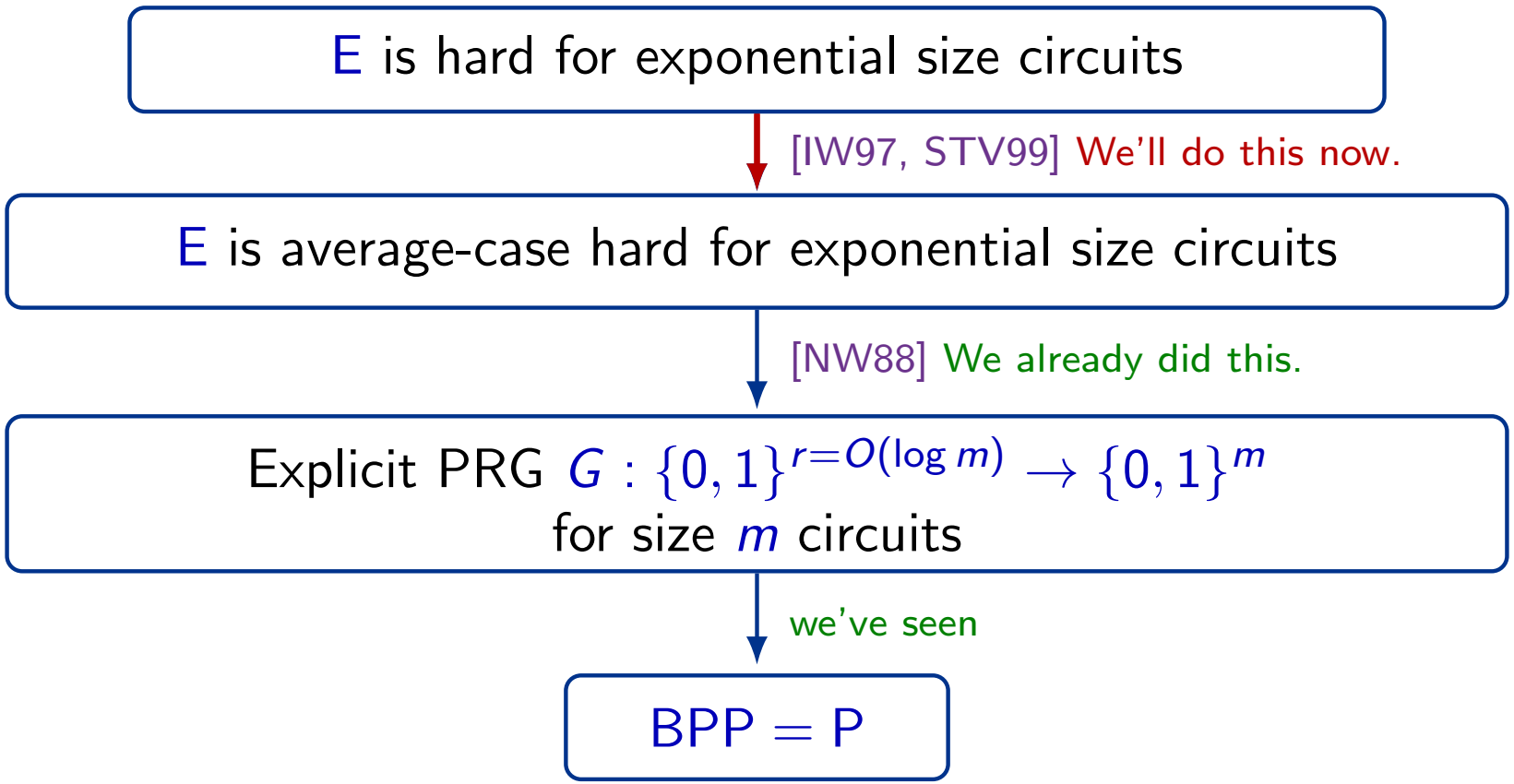
E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.



Hardness $\Rightarrow$ PRGs (Where we are)

Theorem (IW97)

E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.



Hardness amplification (worst-case to average case)

Theorem (IW97,STV99)

E *is hard for exponential size circuits* $\Rightarrow$ E *is average-case hard for exponential size circuits.*

Theorem (hardness amplification - more detailed version)

Hardness amplification (worst-case to average case)

Theorem (IW97,STV99)

E *is hard for exponential size circuits* $\Rightarrow$ E *is average-case hard for exponential size circuits.*

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^{\ell} \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

Hardness amplification (worst-case to average case)

Theorem (IW97,STV99)

E *is hard for exponential size circuits* $\Rightarrow$ E *is average-case hard for exponential size circuits.*

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^{\ell} \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*

Hardness amplification (worst-case to average case)

Theorem (IW97,STV99)

E *is hard for exponential size circuits* $\Rightarrow$ E *is average-case hard for exponential size circuits.*

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$

Hardness amplification (worst-case to average case)

Theorem (IW97,STV99)

E *is hard for exponential size circuits* $\Rightarrow$ E *is average-case hard for exponential size circuits.*

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathsf{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

Hardness amplification (worst-case to average case)

Theorem (IW97,STV99)

E *is hard for exponential size circuits* $\Rightarrow$ E *is average-case hard for exponential size circuits.*

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathsf{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

- First item is a "construction" of f' from f.

Hardness amplification (worst-case to average case)

Theorem (IW97,STV99)

E *is hard for exponential size circuits* $\Rightarrow$ E *is average-case hard for exponential size circuits.*

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

- First item is a "construction" of f' from f.
- Second item is a "reduction" (a.k.a. reconstruction algorithm) that:

Hardness amplification (worst-case to average case)

Theorem (IW97,STV99)

E *is hard for exponential size circuits* $\Rightarrow$ E *is average-case hard for exponential size circuits.*

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

- First item is a "construction" of f' from f.
- Second item is a "reduction" (a.k.a. reconstruction algorithm) that:
  - Takes a C' that computes f' too well on average,

Hardness amplification (worst-case to average case)

Theorem (IW97,STV99)

E *is hard for exponential size circuits* $\Rightarrow$ E *is average-case hard for exponential size circuits.*

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

- First item is a "construction" of f' from f.
- Second item is a "reduction" (a.k.a. reconstruction algorithm) that:
  - Takes a C' that computes f' too well on average,
  - and gives a C that computes f on the worst-case.

Hardness amplification (worst-case to average case)

Theorem (IW97,STV99)

E *is hard for exponential size circuits* $\Rightarrow$ E *is average-case hard for exponential size circuits.*

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

- First item is a "construction" of f' from f.
- Second item is a "reduction" (a.k.a. reconstruction algorithm) that:
  - Takes a C' that computes f' too well on average,
  - and gives a C that computes f on the worst-case.

Consequence: Worst-case to average-case hardness amplification.

Hardness amplification and list decoding [STV99]

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

Hardness amplification

Hardness amplification and list decoding [STV99]

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

Hardness amplification

f

Hardness amplification and list decoding [STV99]

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

Hardness amplification

Hardness amplification and list decoding [STV99]

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

Hardness amplification

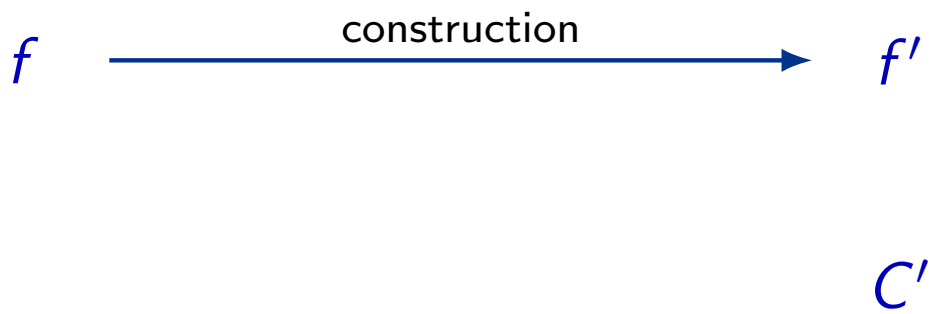
Hardness amplification and list decoding [STV99]

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

Hardness amplification



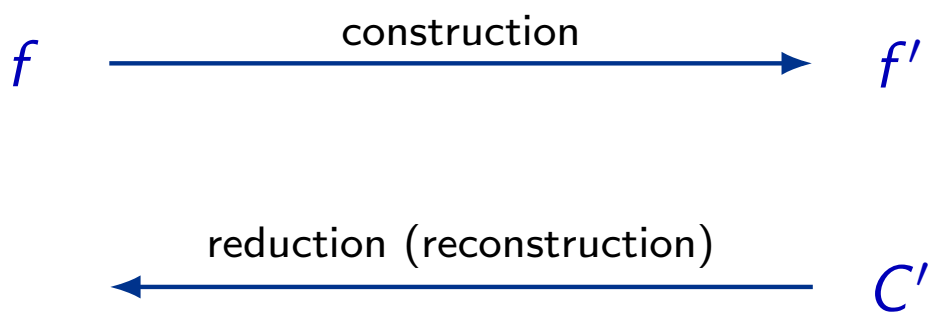
Hardness amplification and list decoding [STV99]

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

Hardness amplification



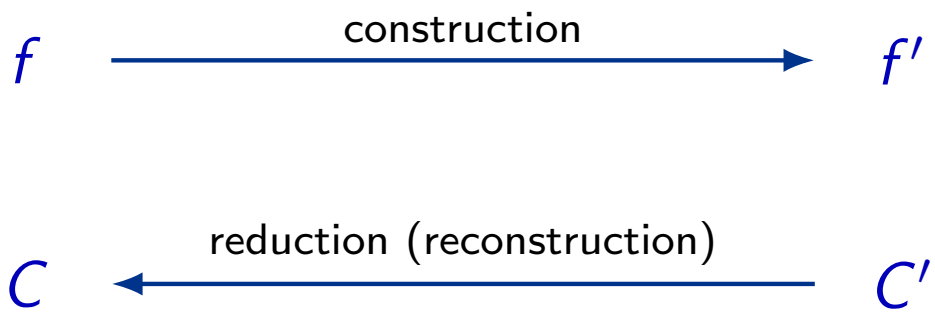
Hardness amplification and list decoding [STV99]

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

Hardness amplification



Hardness amplification and list decoding [STV99]

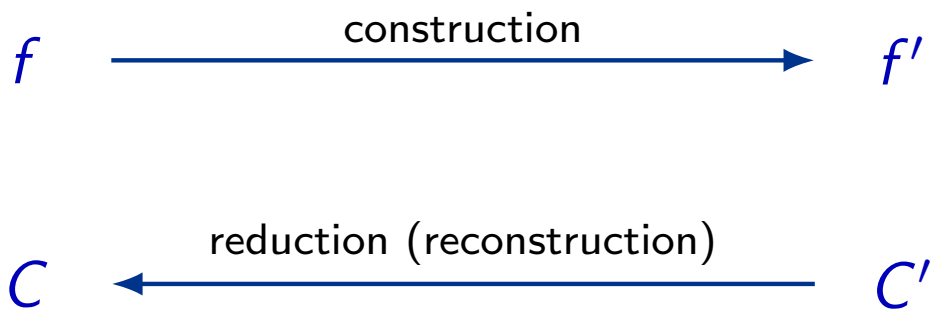
Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

Hardness amplification

f, f' are functions



Hardness amplification and list decoding [STV99]

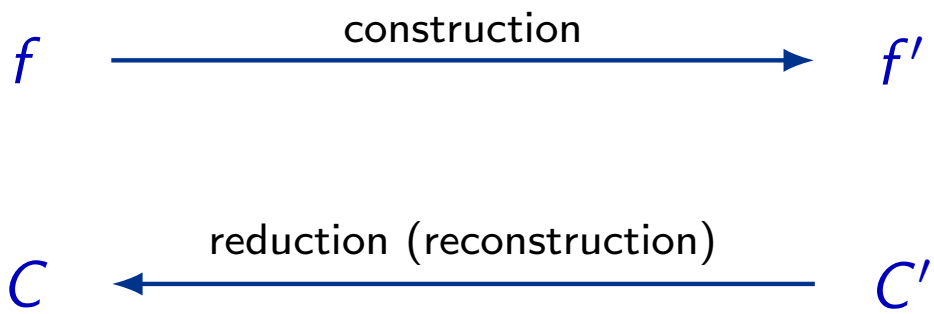
Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

Hardness amplification **(list)-decoding**

f, f' are functions



Hardness amplification and list decoding [STV99]

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

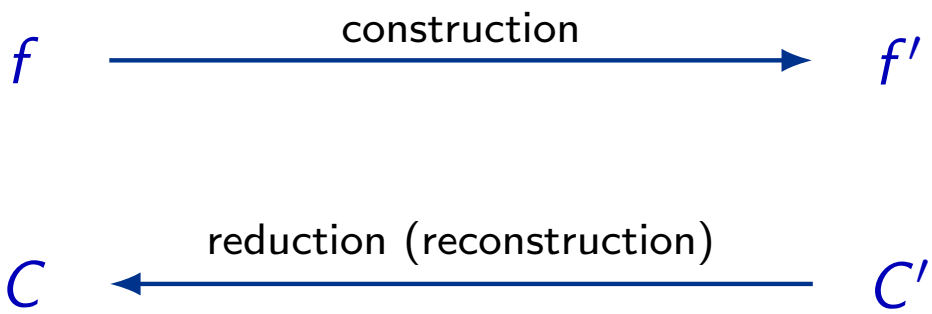
- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

Hardness amplification

f, f' are functions

(list)-decoding

f, f' are strings (truth tables)



Hardness amplification and list decoding [STV99]

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

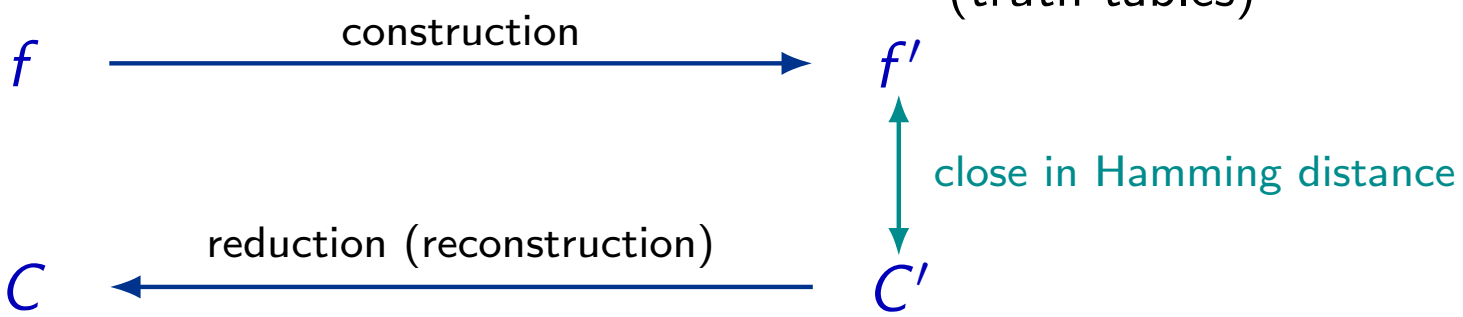
- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

Hardness amplification

f, f' are functions

(list)-decoding

f, f' are strings (truth tables)



Hardness amplification and list decoding [STV99]

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

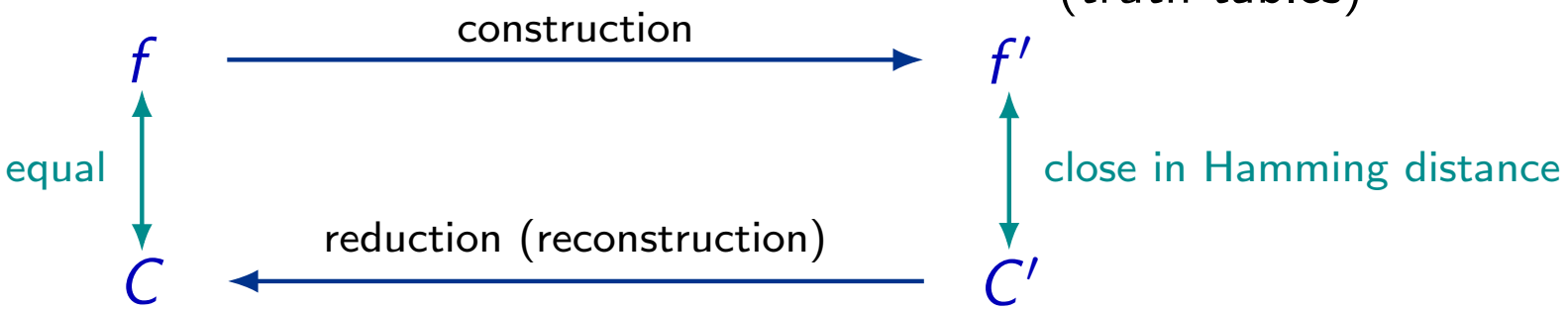
- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

Hardness amplification

f, f' are functions

(list)-decoding

f, f' are strings (truth tables)



Hardness amplification and list decoding [STV99]

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

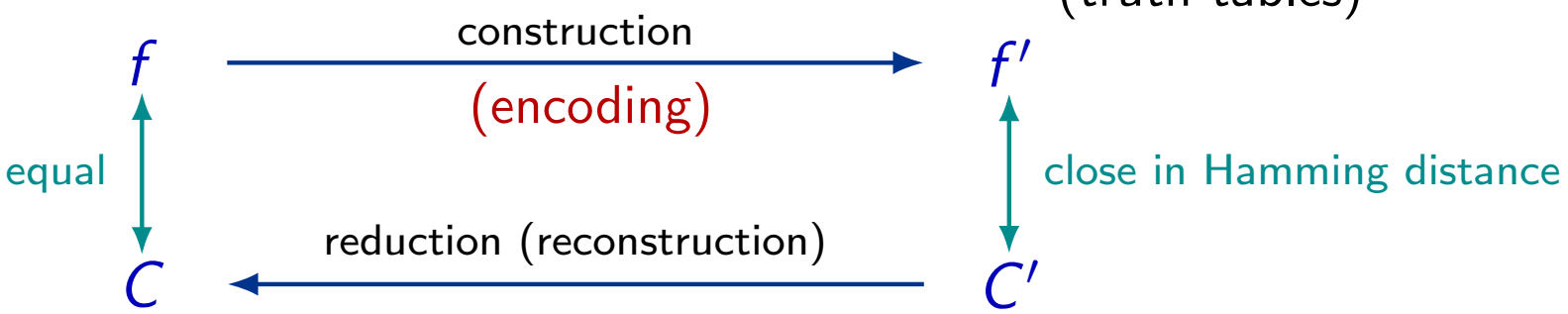
- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

Hardness amplification

f, f' are functions

(list)-decoding

f, f' are strings (truth tables)



Hardness amplification and list decoding [STV99]

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

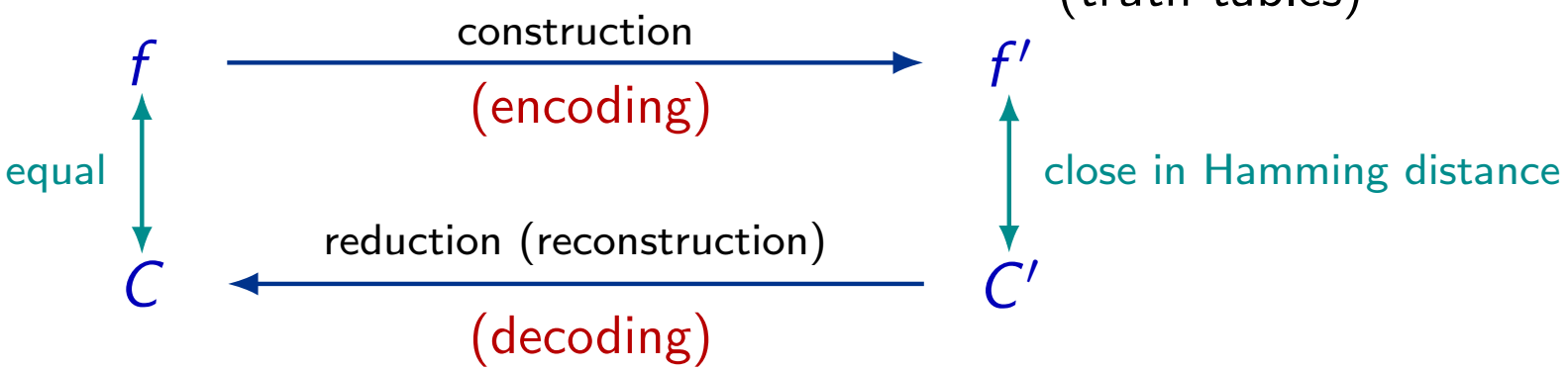
- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

Hardness amplification

f, f' are functions

(list)-decoding

f, f' are strings (truth tables)



Hardness amplification and list decoding [STV99]

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

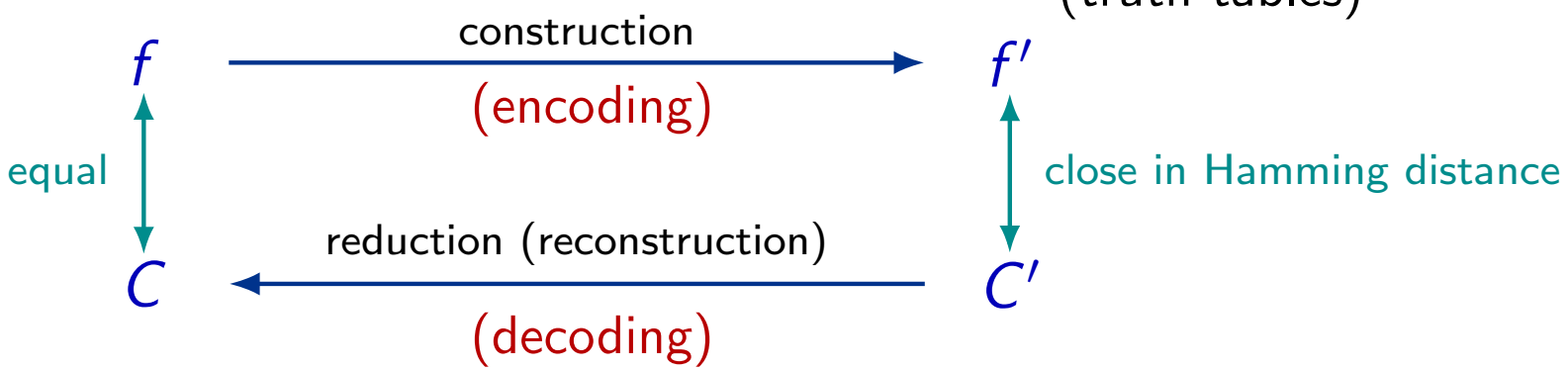
- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

Hardness amplification

f, f' are functions

(list)-decoding

f, f' are strings (truth tables)



"Hardness amplification" = "(list)-decoding" + "efficient reduction".

Hardness amplification and list decoding [STV99]

Theorem (hardness amplification - more detailed version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

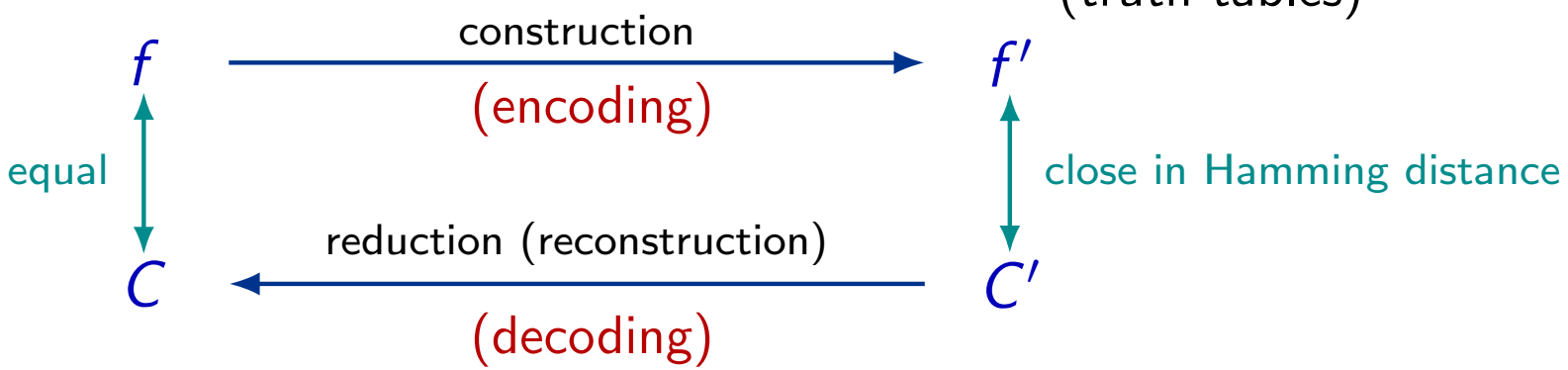
- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

Hardness amplification

f, f' are functions

(list)-decoding

f, f' are strings (truth tables)



"Hardness amplification" = "(list)-decoding" + "efficient reduction".

"Computational Construction" = "info-theory object" + "efficient reduction".

Overview of approach for hardness amplification

Theorem (hardness amplification - boolean version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

This is obtained in two steps:

Overview of approach for hardness amplification

Theorem (hardness amplification - boolean version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

This is obtained in two steps:

- Prove a version where target function $\hat{f}$ is non-boolean (large alphabet).

Overview of approach for hardness amplification

Theorem (hardness amplification - boolean version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

This is obtained in two steps:

- Prove a version where target function $\hat{f}$ is non-boolean (large alphabet).
- Obtain boolean function f' by Goldreich-Levin (code concatenation).

Overview of approach for hardness amplification

Theorem (hardness amplification - boolean version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists f' : \{0,1\}^{O(\ell)} \to \{0,1\}$ *s.t.*

- f' *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow f' \in \mathsf{E}$*).*
- $\exists$*size* s' *circuit* C' *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[C'(x) = f'(x)] > 1/2 + 1/s'$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(s', \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

This is obtained in two steps:

- Prove a version where target function $\hat{f}$ is non-boolean (large alphabet).
- Obtain boolean function f' by Goldreich-Levin (code concatenation).

Theorem (STV99: hardness amplification - non-boolean version)

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists \hat{f} : \{0,1\}^{O(\ell)} \to \{0,1\}^{O(\ell)}$ *s.t.*

- $\hat{f}$ *is computable in* E^f *(gives:* $f \in \mathsf{E} \Rightarrow \hat{f} \in \mathsf{E}$*).*
- $\exists$*size* $\hat{s}$ *circuit* $\hat{C}$ *s.t.* $\Pr_{x \leftarrow \{0,1\}^{O(\ell)}}[\hat{C}(x) = \hat{f}(x)] > \frac{1}{\hat{s}}$
  $\Rightarrow \exists$*size* $s = \mathrm{poly}(\hat{s}, \ell)$ *circuit* C *s.t.* $\forall x \in \{0,1\}^\ell$, $C(x) = f(x)$.

Low degree extension (Reed-Muller encoding) $f \to \hat{f}$

Theorem (Low degree extension)

Low degree extension (Reed-Muller encoding) $f \to \hat{f}$

Theorem (Low degree extension)

$\forall d$, *and* $\forall$*sufficiently large* ℓ. *Set* $h = 2^{\ell/d}$ *and* $q = \text{poly}(h)$.

Low degree extension (Reed-Muller encoding) $f \to \hat{f}$

Theorem (Low degree extension)

$\forall d$, and $\forall$sufficiently large ℓ. Set $h = 2^{\ell/d}$ and $q = \text{poly}(h)$.
Let $\mathbb{F}_q$ be the field with q elements (we assume that q is a power of 2).

Low degree extension (Reed-Muller encoding) $f \to \hat{f}$

Theorem (Low degree extension)

$\forall d$, and $\forall$sufficiently large ℓ. Set $h = 2^{\ell/d}$ and $q = \text{poly}(h)$.
Let $\mathbb{F}_q$ be the field with q elements (we assume that q is a power of 2).
Fix some $H \subseteq \mathbb{F}_q$ of size h. As $|H^d| = 2^\ell$ we can identify H^d with $\{0,1\}^\ell$.

Low degree extension (Reed-Muller encoding) $f \to \hat{f}$

Theorem (Low degree extension)

$\forall d$, *and* $\forall$*sufficiently large* ℓ. *Set* $h = 2^{\ell/d}$ *and* $q = \text{poly}(h)$.
Let $\mathbb{F}_q$ *be the field with* q *elements (we assume that* q *is a power of* 2*).*
Fix some $H \subseteq \mathbb{F}_q$ *of size* h. *As* $|H^d| = 2^\ell$ *we can identify* H^d *with* $\{0,1\}^\ell$.

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists \hat{f} : \mathbb{F}_q^d \to \mathbb{F}_q$ *s.t.*

- $\hat{f}$ *is a degree* $\approx h$ *(multivariate) polynomial.*
- $\forall x \in \{0,1\}^\ell$, $\hat{f}(x) = f(x)$.
- $\hat{f} \in E^f$ *(gives:* $f \in \mathsf{E} \Rightarrow \hat{f} \in \mathsf{E}$*).*

Low degree extension (Reed-Muller encoding) $f \to \hat{f}$

Theorem (Low degree extension)

$\forall d$, *and* $\forall$*sufficiently large* ℓ. *Set* $h = 2^{\ell/d}$ *and* $q = \text{poly}(h)$.
Let $\mathbb{F}_q$ *be the field with* q *elements (we assume that* q *is a power of* 2*).*
Fix some $H \subseteq \mathbb{F}_q$ *of size* h. *As* $|H^d| = 2^\ell$ *we can identify* H^d *with* $\{0,1\}^\ell$.

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists \hat{f} : \mathbb{F}_q^d \to \mathbb{F}_q$ *s.t.*

- $\hat{f}$ *is a degree* $\approx h$ *(multivariate) polynomial.*
- $\forall x \in \{0,1\}^\ell$, $\hat{f}(x) = f(x)$.
- $\hat{f} \in E^f$ *(gives:* $f \in \mathsf{E} \Rightarrow \hat{f} \in \mathsf{E}$*).*

$d = 2$

$h = 2^{\ell/d}$

$|H| = h$

$|H^d| = h^d = 2^\ell$

$q = \text{poly}(h)$

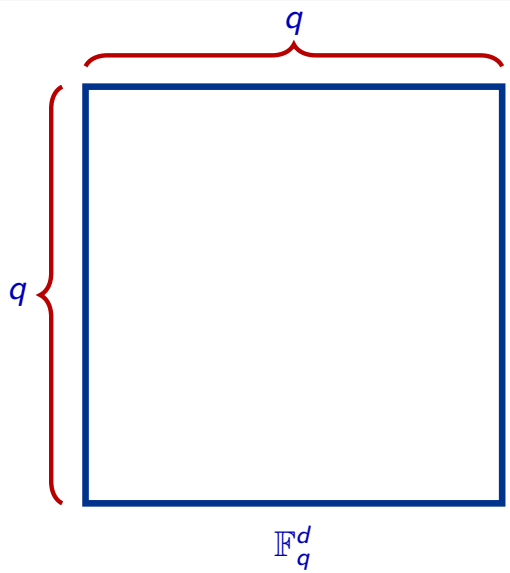
Low degree extension (Reed-Muller encoding) $f \to \hat{f}$

Theorem (Low degree extension)

$\forall d$, *and* $\forall$*sufficiently large* ℓ. *Set* $h = 2^{\ell/d}$ *and* $q = \text{poly}(h)$.
Let $\mathbb{F}_q$ *be the field with* q *elements (we assume that* q *is a power of* 2*).*
Fix some $H \subseteq \mathbb{F}_q$ *of size* h. *As* $|H^d| = 2^\ell$ *we can identify* H^d *with* $\{0,1\}^\ell$.

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists \hat{f} : \mathbb{F}_q^d \to \mathbb{F}_q$ *s.t.*

- $\hat{f}$ *is a degree* $\approx h$ *(multivariate) polynomial.*
- $\forall x \in \{0,1\}^\ell$, $\hat{f}(x) = f(x)$.
- $\hat{f} \in E^f$ *(gives:* $f \in \mathsf{E} \Rightarrow \hat{f} \in \mathsf{E}$*).*



$d = 2$

$h = 2^{\ell/d}$

$|H| = h$

$|H^d| = h^d = 2^\ell$

$q = \text{poly}(h)$

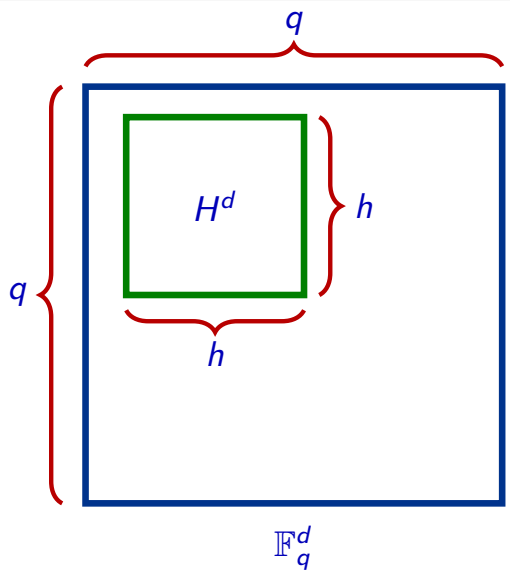
Low degree extension (Reed-Muller encoding) $f \to \hat{f}$

Theorem (Low degree extension)

$\forall d$, *and* $\forall$*sufficiently large* ℓ. *Set* $h = 2^{\ell/d}$ *and* $q = \text{poly}(h)$.
Let $\mathbb{F}_q$ *be the field with* q *elements (we assume that* q *is a power of* 2*).*
Fix some $H \subseteq \mathbb{F}_q$ *of size* h. *As* $|H^d| = 2^\ell$ *we can identify* H^d *with* $\{0,1\}^\ell$.

$\forall f : \{0,1\}^\ell \to \{0,1\}$, $\exists \hat{f} : \mathbb{F}_q^d \to \mathbb{F}_q$ *s.t.*

- $\hat{f}$ *is a degree* $\approx h$ *(multivariate) polynomial.*
- $\forall x \in \{0,1\}^\ell$, $\hat{f}(x) = f(x)$.
- $\hat{f} \in E^f$ *(gives:* $f \in \mathsf{E} \Rightarrow \hat{f} \in \mathsf{E}$*).*



$d = 2$

$h = 2^{\ell/d}$

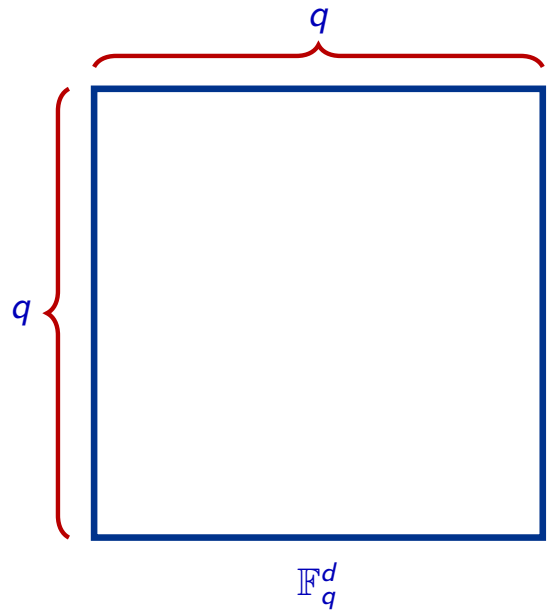
$|H| = h$

$|H^d| = h^d = 2^\ell$

$q = \text{poly}(h)$

Reduction (decoding): $\hat{C} \to C$

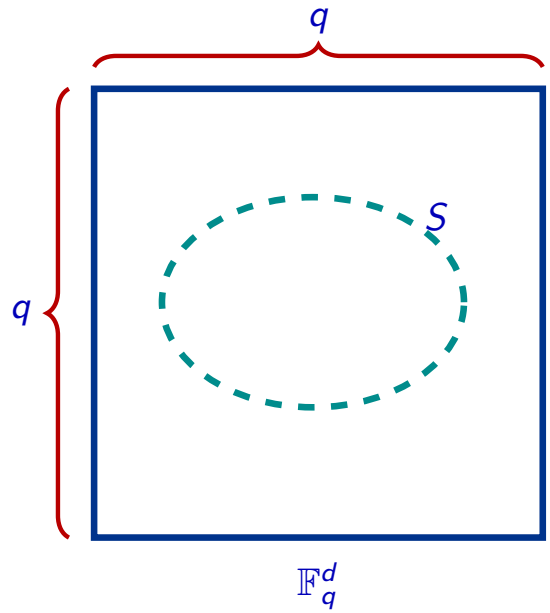
Starting point: A circuit $\hat{C}$ of size $\hat{s}$ s.t. $\Pr_{x \leftarrow \mathbb{F}_q^d}[\hat{C}(x) = \hat{f}(x)] > 1/\hat{s}$.



Reduction (decoding): $\hat{C} \to C$

Starting point: A circuit $\hat{C}$ of size $\hat{s}$ s.t. $\Pr_{x \leftarrow \mathbb{F}_q^d}[\hat{C}(x) = \hat{f}(x)] > 1/\hat{s}$.

Let $S = \left\{ x \in \mathbb{F}_q^d : \hat{C}(x) = \hat{f}(x) \right\}$.

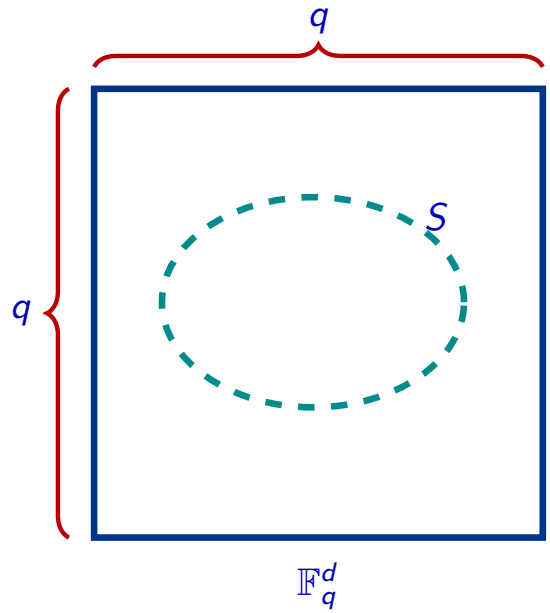


Reduction (decoding): $\hat{C} \to C$

Starting point: A circuit $\hat{C}$ of size $\hat{s}$ s.t. $\Pr_{x \leftarrow \mathbb{F}_q^d}[\hat{C}(x) = \hat{f}(x)] > 1/\hat{s}$.

Let $S = \left\{ x \in \mathbb{F}_q^d : \hat{C}(x) = \hat{f}(x) \right\}$.

Goal: Construct a circuit $C(x)$ that computes f.



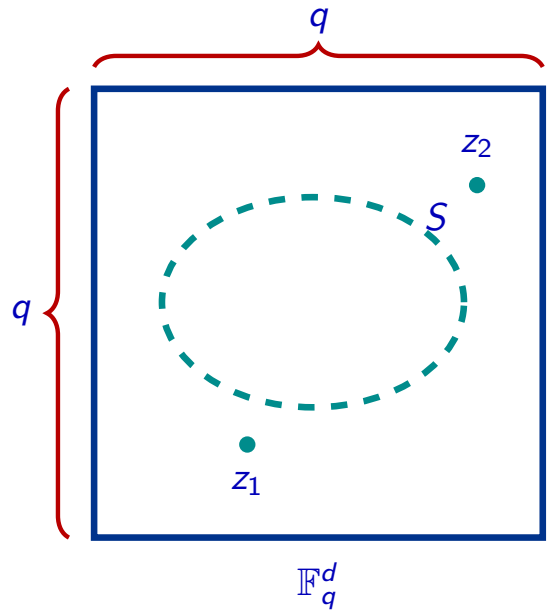
Reduction (decoding): $\hat{C} \to C$

Starting point: A circuit $\hat{C}$ of size $\hat{s}$ s.t. $\Pr_{x \leftarrow \mathbb{F}_q^d}[\hat{C}(x) = \hat{f}(x)] > 1/\hat{s}$.

Let $S = \left\{ x \in \mathbb{F}_q^d : \hat{C}(x) = \hat{f}(x) \right\}$.

Goal: Construct a circuit $C(x)$ that computes f.

C chooses two random points $z_1, z_2 \leftarrow \mathbb{F}_q^d$, which will later be hardwired as nonuniformity.



Reduction (decoding): $\hat{C} \to C$

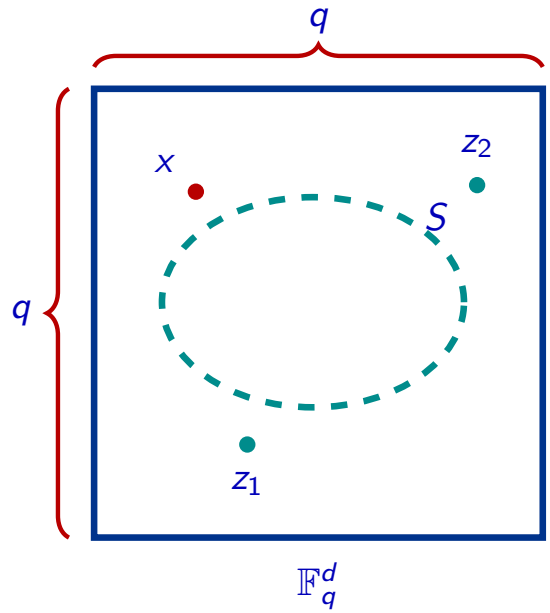
Starting point: A circuit $\hat{C}$ of size $\hat{s}$ s.t. $\Pr_{x \leftarrow \mathbb{F}_q^d}[\hat{C}(x) = \hat{f}(x)] > 1/\hat{s}$.

Let $S = \left\{ x \in \mathbb{F}_q^d : \hat{C}(x) = \hat{f}(x) \right\}$.

Goal: Construct a circuit $C(x)$ that computes f.

C chooses two random points $z_1, z_2 \leftarrow \mathbb{F}_q^d$, which will later be hardwired as nonuniformity.

Given input $x \in H^d \subseteq \mathbb{F}_q^d$,



Reduction (decoding): $\hat{C} \to C$

Starting point: A circuit $\hat{C}$ of size $\hat{s}$ s.t. $\Pr_{x \leftarrow \mathbb{F}_q^d}[\hat{C}(x) = \hat{f}(x)] > 1/\hat{s}$.

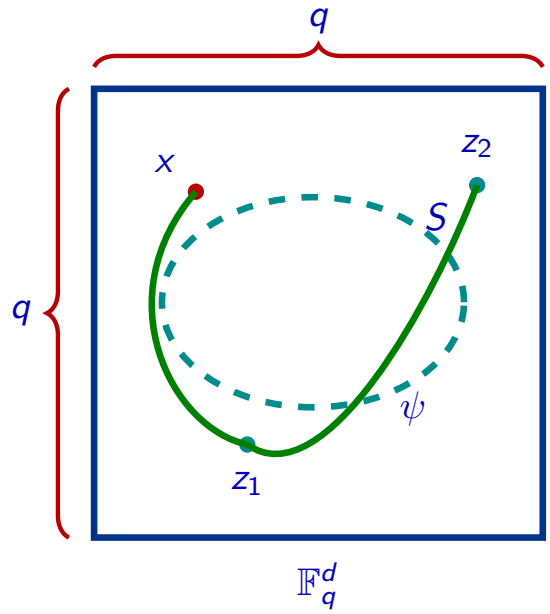
Let $S = \left\{ x \in \mathbb{F}_q^d : \hat{C}(x) = \hat{f}(x) \right\}$.

Goal: Construct a circuit $C(x)$ that computes f.

C chooses two random points $z_1, z_2 \leftarrow \mathbb{F}_q^d$, which will later be hardwired as nonuniformity.

Given input $x \in H^d \subseteq \mathbb{F}_q^d$,

C constructs the degree-2 curve $\psi : \mathbb{F}_q \to \mathbb{F}_q^d$ with $\psi(0) = x$, $\psi(1) = z_1$, and $\psi(2) = z_2$.



Reduction (decoding): $\hat{C} \to C$

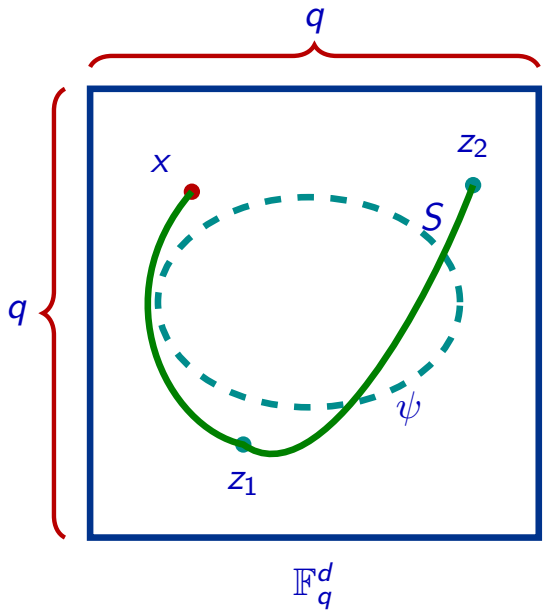
Starting point: A circuit $\hat{C}$ of size $\hat{s}$ s.t. $\Pr_{x \leftarrow \mathbb{F}_q^d}[\hat{C}(x) = \hat{f}(x)] > 1/\hat{s}$.

Let $S = \left\{ x \in \mathbb{F}_q^d : \hat{C}(x) = \hat{f}(x) \right\}$.

Goal: Construct a circuit $C(x)$ that computes f.

C chooses two random points $z_1, z_2 \leftarrow \mathbb{F}_q^d$, which will later be hardwired as nonuniformity.

Given input $x \in H^d \subseteq \mathbb{F}_q^d$,



C constructs the degree-2 curve $\psi : \mathbb{F}_q \to \mathbb{F}_q^d$ with $\psi(0) = x$, $\psi(1) = z_1$, and $\psi(2) = z_2$.

Observe that $p(t) = \hat{f}(\psi(t))$ is a univariate polynomial of degree $2h$, and $p(0) = \hat{f}(\psi(0)) = \hat{f}(x) = f(x)$.

Reduction (decoding): $\hat{C} \to C$

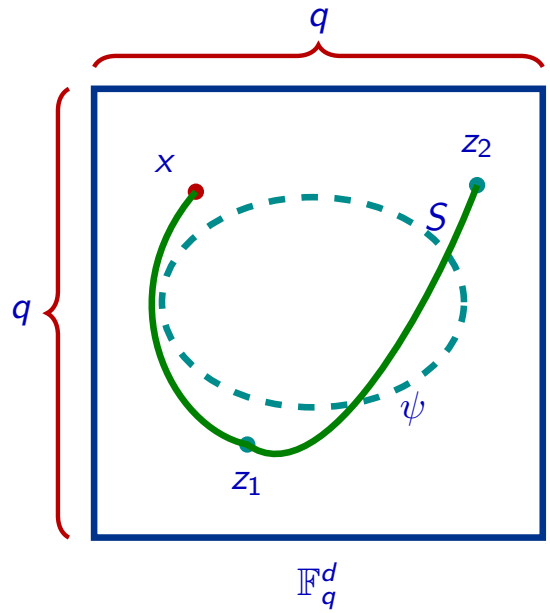
Starting point: A circuit $\hat{C}$ of size $\hat{s}$ s.t. $\Pr_{x \leftarrow \mathbb{F}_q^d}[\hat{C}(x) = \hat{f}(x)] > 1/\hat{s}$.

Let $S = \left\{ x \in \mathbb{F}_q^d : \hat{C}(x) = \hat{f}(x) \right\}$.

Goal: Construct a circuit $C(x)$ that computes f.

C chooses two random points $z_1, z_2 \leftarrow \mathbb{F}_q^d$, which will later be hardwired as nonuniformity.

Given input $x \in H^d \subseteq \mathbb{F}_q^d$,



C constructs the degree-2 curve $\psi : \mathbb{F}_q \to \mathbb{F}_q^d$ with $\psi(0) = x$, $\psi(1) = z_1$, and $\psi(2) = z_2$.

Observe that $p(t) = \hat{f}(\psi(t))$ is a univariate polynomial of degree $2h$, and $p(0) = \hat{f}(\psi(0)) = \hat{f}(x) = f(x)$.

The points $\big(\psi(1), \ldots, \psi(q-1)\big)$ are 2-wise independent.

Reduction (decoding): $\hat{C} \to C$

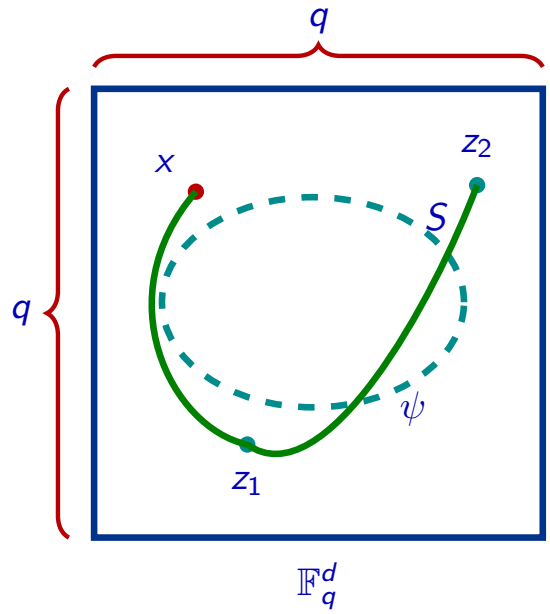
Starting point: A circuit $\hat{C}$ of size $\hat{s}$ s.t. $\Pr_{x \leftarrow \mathbb{F}_q^d}[\hat{C}(x) = \hat{f}(x)] > 1/\hat{s}$.

Let $S = \left\{x \in \mathbb{F}_q^d : \hat{C}(x) = \hat{f}(x)\right\}$.

Goal: Construct a circuit $C(x)$ that computes f.

C chooses two random points $z_1, z_2 \leftarrow \mathbb{F}_q^d$, which will later be hardwired as nonuniformity.

Given input $x \in H^d \subseteq \mathbb{F}_q^d$,



C constructs the degree-2 curve $\psi : \mathbb{F}_q \to \mathbb{F}_q^d$ with $\psi(0) = x$, $\psi(1) = z_1$, and $\psi(2) = z_2$.

Observe that $p(t) = \hat{f}(\psi(t))$ is a univariate polynomial of degree $2h$, and $p(0) = \hat{f}(\psi(0)) = \hat{f}(x) = f(x)$.

The points $\big(\psi(1), \ldots, \psi(q-1)\big)$ are 2-wise independent.

This gives that w.h.p. $(1/q)\,|\{t : \psi(t) \in S\}| \approx |S|/q^d > 1/\hat{s}$.

Reduction (decoding): $\hat{C} \to C$

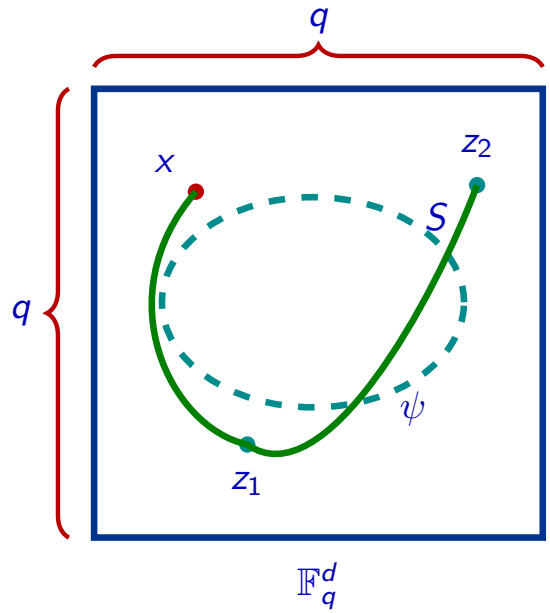
Starting point: A circuit $\hat{C}$ of size $\hat{s}$ s.t. $\Pr_{x \leftarrow \mathbb{F}_q^d}[\hat{C}(x) = \hat{f}(x)] > 1/\hat{s}$.

Let $S = \left\{ x \in \mathbb{F}_q^d : \hat{C}(x) = \hat{f}(x) \right\}$.

Goal: Construct a circuit $C(x)$ that computes f.

C chooses two random points $z_1, z_2 \leftarrow \mathbb{F}_q^d$, which will later be hardwired as nonuniformity.

Given input $x \in H^d \subseteq \mathbb{F}_q^d$,



C constructs the degree-2 curve $\psi : \mathbb{F}_q \to \mathbb{F}_q^d$ with $\psi(0) = x$, $\psi(1) = z_1$, and $\psi(2) = z_2$.

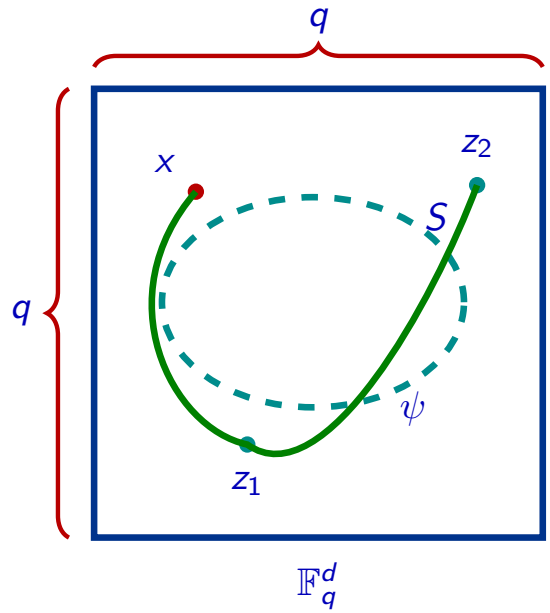
Observe that $p(t) = \hat{f}(\psi(t))$ is a univariate polynomial of degree $2h$, and $p(0) = \hat{f}(\psi(0)) = \hat{f}(x) = f(x)$.

The points $(\psi(1), \ldots, \psi(q-1))$ are 2-wise independent.

This gives that w.h.p. $(1/q)\,|\{t : \psi(t) \in S\}| \approx |S|/q^d > 1/\hat{s}$.

The evaluations $\hat{C}(\psi(1)), \ldots, \hat{C}(\psi(q-1))$ agree with the correct polynomial p on a $> 1/\hat{s}$ fraction of the points.

Reduction (decoding): $\hat{C} \to C$



Starting point: A circuit $\hat{C}$ of size $\hat{s}$ s.t. $\Pr_{x \leftarrow \mathbb{F}_q^d}[\hat{C}(x) = \hat{f}(x)] > 1/\hat{s}$.

Let $S = \left\{x \in \mathbb{F}_q^d : \hat{C}(x) = \hat{f}(x)\right\}$.

Goal: Construct a circuit $C(x)$ that computes f.

C chooses two random points $z_1, z_2 \leftarrow \mathbb{F}_q^d$, which will later be hardwired as nonuniformity.

Given input $x \in H^d \subseteq \mathbb{F}_q^d$,

C constructs the degree-2 curve $\psi : \mathbb{F}_q \to \mathbb{F}_q^d$ with $\psi(0) = x$, $\psi(1) = z_1$, and $\psi(2) = z_2$.

Observe that $p(t) = \hat{f}(\psi(t))$ is a univariate polynomial of degree $2h$, and $p(0) = \hat{f}(\psi(0)) = \hat{f}(x) = f(x)$.

The points $\left(\psi(1), \ldots, \psi(q-1)\right)$ are 2-wise independent.

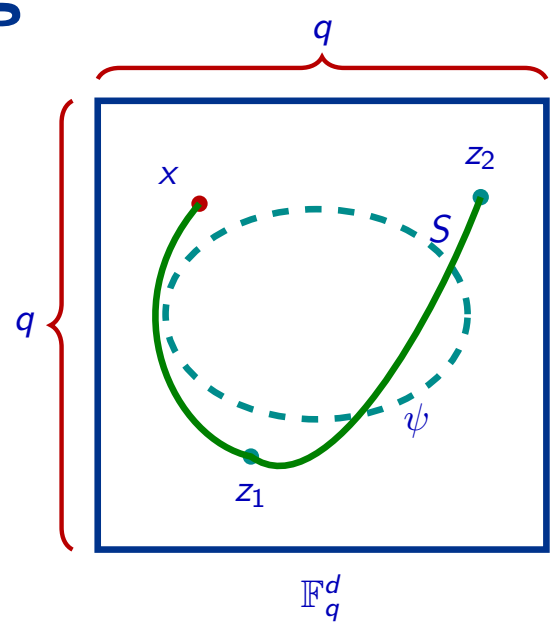
This gives that w.h.p. $(1/q)\,|\{t : \psi(t) \in S\}| \approx |S|/q^d > 1/\hat{s}$.

The evaluations $\hat{C}(\psi(1)), \ldots, \hat{C}(\psi(q-1))$ agree with the correct polynomial p on a $> 1/\hat{s}$ fraction of the points.

Obtaining p (which gives us $p(0) = f(x)$) is a Reed-Solomon decoding problem!

Using Reed-Solomon (list)-decoding

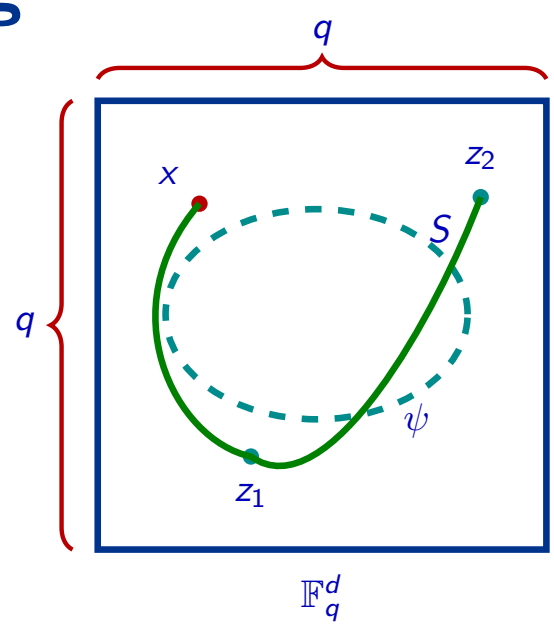
We only know that $a > 1/\hat{s}$ fraction of evaluations agree with p, where $a < 1/2$.



Using Reed-Solomon (list)-decoding

We only know that $a > 1/\hat{s}$ fraction of evaluations agree with p, where $a < 1/2$.

In this range unique-decoding is impossible.

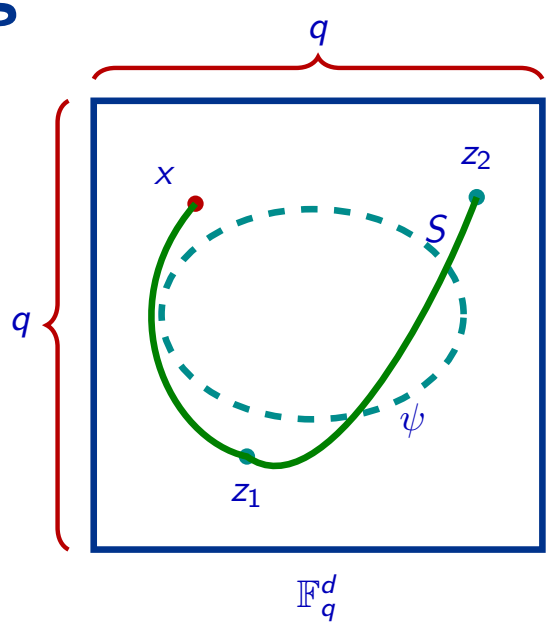


Using Reed-Solomon (list)-decoding

We only know that $a > 1/\hat{s}$ fraction of evaluations agree with p, where $a < 1/2$.

In this range unique-decoding is impossible.

List-decoding algorithm [Sud97] returns a list of $L = \text{poly}(\hat{s})$ candidate polynomials $p_1, \ldots, p_L$, such that one of them is the correct polynomial p.



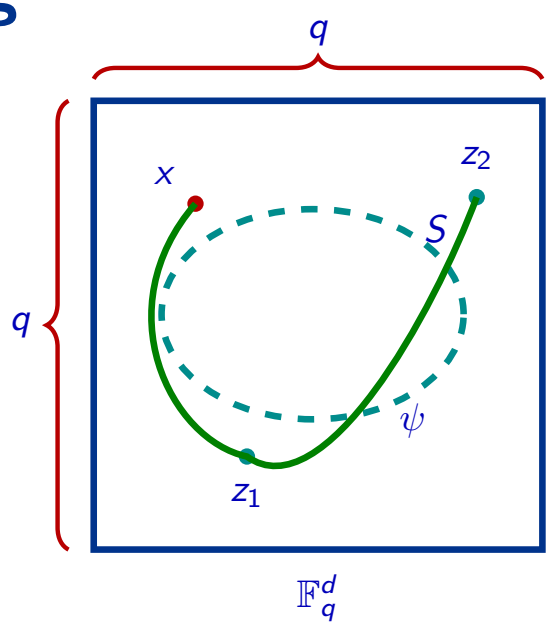
Using Reed-Solomon (list)-decoding

We only know that $a > 1/\hat{s}$ fraction of evaluations agree with p, where $a < 1/2$.

In this range unique-decoding is impossible.

List-decoding algorithm [Sud97] returns a list of $L = \text{poly}(\hat{s})$ candidate polynomials $p_1, \ldots, p_L$, such that one of them is the correct polynomial p.

How do we find the correct polynomial p?



Using Reed-Solomon (list)-decoding

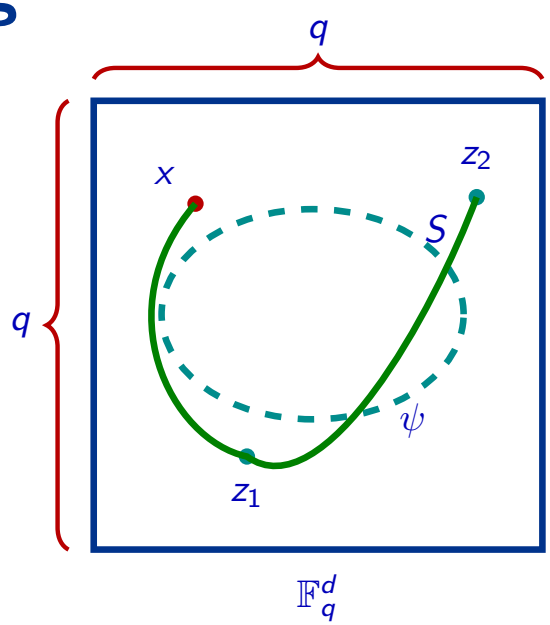
We only know that $a > 1/\hat{s}$ fraction of evaluations agree with p, where $a < 1/2$.

In this range unique-decoding is impossible.

List-decoding algorithm [Sud97] returns a list of $L = \text{poly}(\hat{s})$ candidate polynomials $p_1, \ldots, p_L$, such that one of them is the correct polynomial p.

How do we find the correct polynomial p?

The circuit C that we will construct will also be hardwired with $\hat{f}(z_1)$.



Using Reed-Solomon (list)-decoding

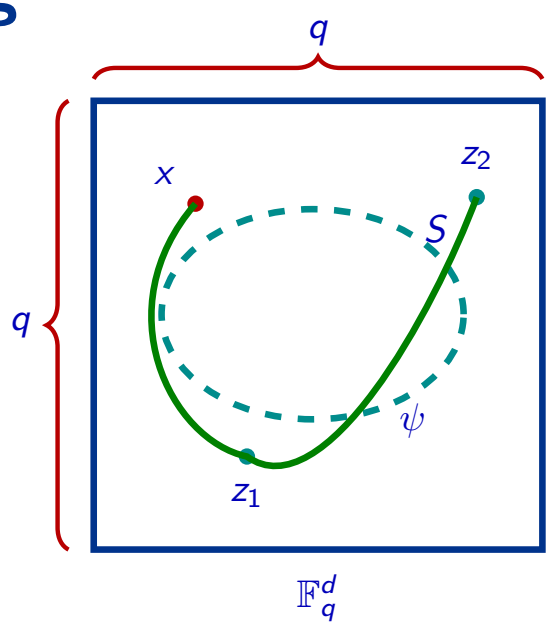
We only know that $a > 1/\hat{s}$ fraction of evaluations agree with p, where $a < 1/2$.

In this range unique-decoding is impossible.

List-decoding algorithm [Sud97] returns a list of $L = \text{poly}(\hat{s})$ candidate polynomials $p_1, \ldots, p_L$, such that one of them is the correct polynomial p.

How do we find the correct polynomial p?

The circuit C that we will construct will also be hardwired with $\hat{f}(z_1)$.



Claim: w.h.p. p is the only polynomial in the list that agrees with $\hat{f}$ on z_1.

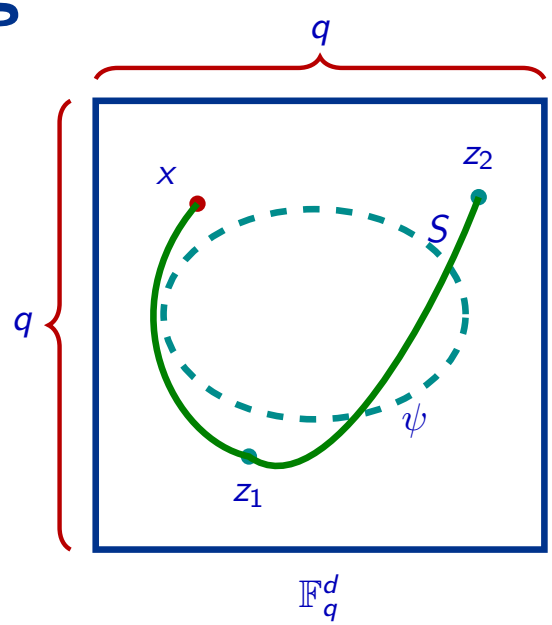
Using Reed-Solomon (list)-decoding

We only know that $a > 1/\hat{s}$ fraction of evaluations agree with p, where $a < 1/2$.

In this range unique-decoding is impossible.

List-decoding algorithm [Sud97] returns a list of $L = \text{poly}(\hat{s})$ candidate polynomials $p_1, \ldots, p_L$, such that one of them is the correct polynomial p.

How do we find the correct polynomial p?



The circuit C that we will construct will also be hardwired with $\hat{f}(z_1)$.

Claim: w.h.p. p is the only polynomial in the list that agrees with $\hat{f}$ on z_1.

Proof: Two different low degree polynomials differ w.h.p. on a random point.

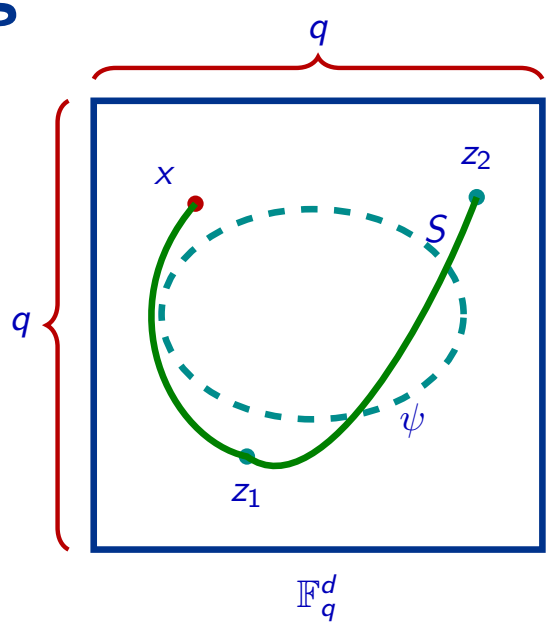
Using Reed-Solomon (list)-decoding

We only know that $a > 1/\hat{s}$ fraction of evaluations agree with p, where $a < 1/2$.

In this range unique-decoding is impossible.

List-decoding algorithm [Sud97] returns a list of $L = \text{poly}(\hat{s})$ candidate polynomials $p_1, \ldots, p_L$, such that one of them is the correct polynomial p.

How do we find the correct polynomial p?



The circuit C that we will construct will also be hardwired with $\hat{f}(z_1)$.

Claim: w.h.p. p is the only polynomial in the list that agrees with $\hat{f}$ on z_1.

Proof: Two different low degree polynomials differ w.h.p. on a random point.

This claim means that we can identify the correct polynomial p.

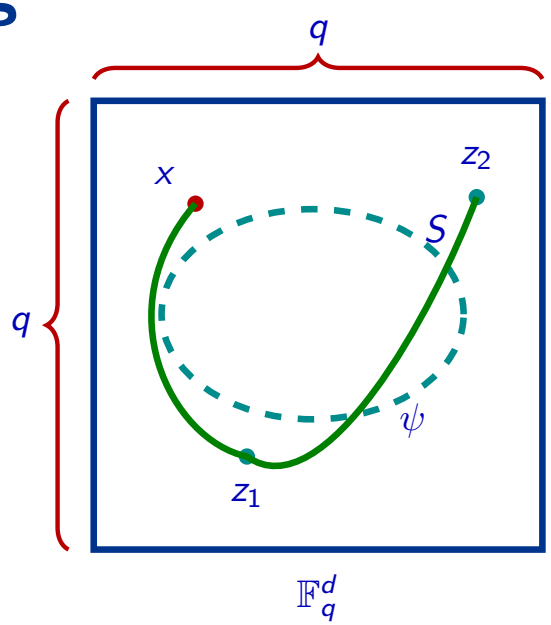
Using Reed-Solomon (list)-decoding

We only know that $a > 1/\hat{s}$ fraction of evaluations agree with p, where $a < 1/2$.

In this range unique-decoding is impossible.

List-decoding algorithm [Sud97] returns a list of $L = \text{poly}(\hat{s})$ candidate polynomials $p_1, \ldots, p_L$, such that one of them is the correct polynomial p.

How do we find the correct polynomial p?



The circuit C that we will construct will also be hardwired with $\hat{f}(z_1)$.

Claim: w.h.p. p is the only polynomial in the list that agrees with $\hat{f}$ on z_1.

Proof: Two different low degree polynomials differ w.h.p. on a random point.

This claim means that we can identify the correct polynomial p.

Overall, we can find p and obtain $f(x) = p(0)$ in time $q^{O(1)} = h^{O(1)} = 2^{O(\ell/d)}$.

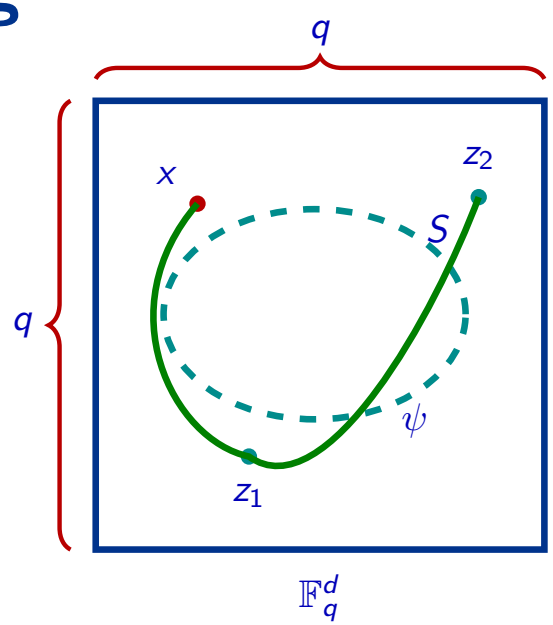
Using Reed-Solomon (list)-decoding

We only know that $a > 1/\hat{s}$ fraction of evaluations agree with p, where $a < 1/2$.

In this range unique-decoding is impossible.

List-decoding algorithm [Sud97] returns a list of $L = \text{poly}(\hat{s})$ candidate polynomials $p_1, \ldots, p_L$, such that one of them is the correct polynomial p.

How do we find the correct polynomial p?



The circuit C that we will construct will also be hardwired with $\hat{f}(z_1)$.

Claim: w.h.p. p is the only polynomial in the list that agrees with $\hat{f}$ on z_1.

Proof: Two different low degree polynomials differ w.h.p. on a random point.

This claim means that we can identify the correct polynomial p.

Overall, we can find p and obtain $f(x) = p(0)$ in time $q^{O(1)} = h^{O(1)} = 2^{O(\ell/d)}$.

(Important point is that time is $\text{poly}(q) \ll 2^{\ell}$ and not $\text{poly}(q^d) > 2^{\ell}$).

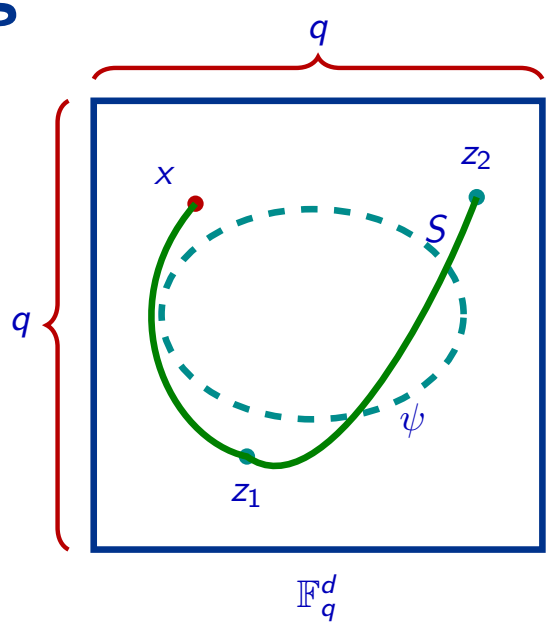
Using Reed-Solomon (list)-decoding

We only know that $a > 1/\hat{s}$ fraction of evaluations agree with p, where $a < 1/2$.

In this range unique-decoding is impossible.

List-decoding algorithm [Sud97] returns a list of $L = \text{poly}(\hat{s})$ candidate polynomials $p_1, \ldots, p_L$, such that one of them is the correct polynomial p.

How do we find the correct polynomial p?



The circuit C that we will construct will also be hardwired with $\hat{f}(z_1)$.

Claim: w.h.p. p is the only polynomial in the list that agrees with $\hat{f}$ on z_1.

Proof: Two different low degree polynomials differ w.h.p. on a random point.

This claim means that we can identify the correct polynomial p.

Overall, we can find p and obtain $f(x) = p(0)$ in time $q^{O(1)} = h^{O(1)} = 2^{O(\ell/d)}$.

(Important point is that time is $\text{poly}(q) \ll 2^{\ell}$ and not $\text{poly}(q^d) > 2^{\ell}$).

Choosing a sufficiently large d, can make time as small as required.

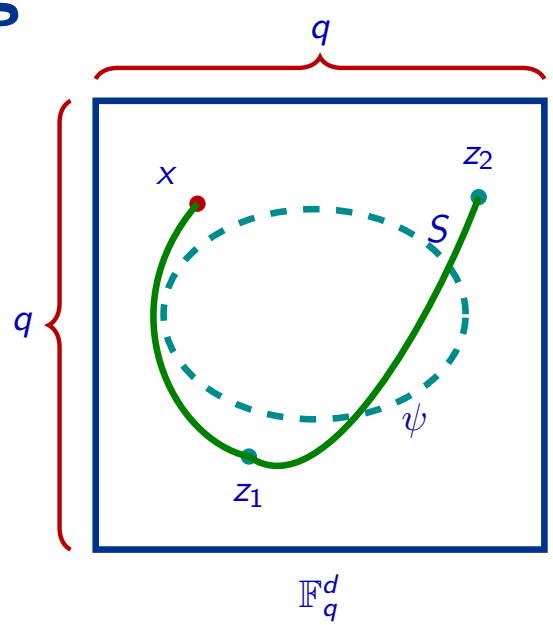
Using Reed-Solomon (list)-decoding

We only know that $a > 1/\hat{s}$ fraction of evaluations agree with p, where $a < 1/2$.

In this range unique-decoding is impossible.

List-decoding algorithm [Sud97] returns a list of $L = \text{poly}(\hat{s})$ candidate polynomials $p_1, \ldots, p_L$, such that one of them is the correct polynomial p.

How do we find the correct polynomial p?



The circuit C that we will construct will also be hardwired with $\hat{f}(z_1)$.

Claim: w.h.p. p is the only polynomial in the list that agrees with $\hat{f}$ on z_1.

Proof: Two different low degree polynomials differ w.h.p. on a random point.

This claim means that we can identify the correct polynomial p.

Overall, we can find p and obtain $f(x) = p(0)$ in time $q^{O(1)} = h^{O(1)} = 2^{O(\ell/d)}$.

(Important point is that time is $\text{poly}(q) \ll 2^{\ell}$ and not $\text{poly}(q^d) > 2^{\ell}$).

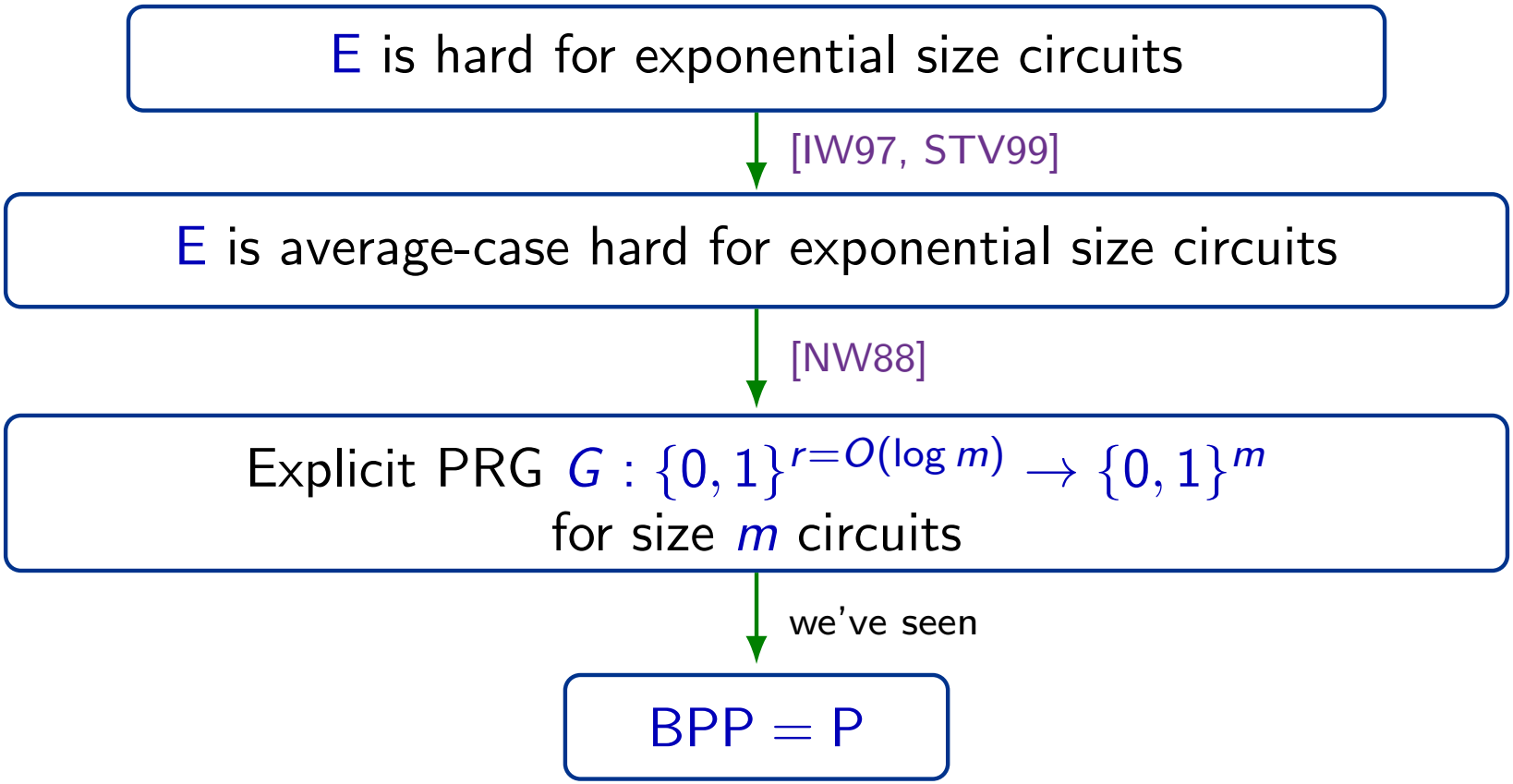
Choosing a sufficiently large d, can make time as small as required.

Comment: This presentation is different from [STV99] which uses lines (and not curves) and a more delicate argument.

Hardness assumptions $\Rightarrow$ derandomization: recap

Theorem (IW97)

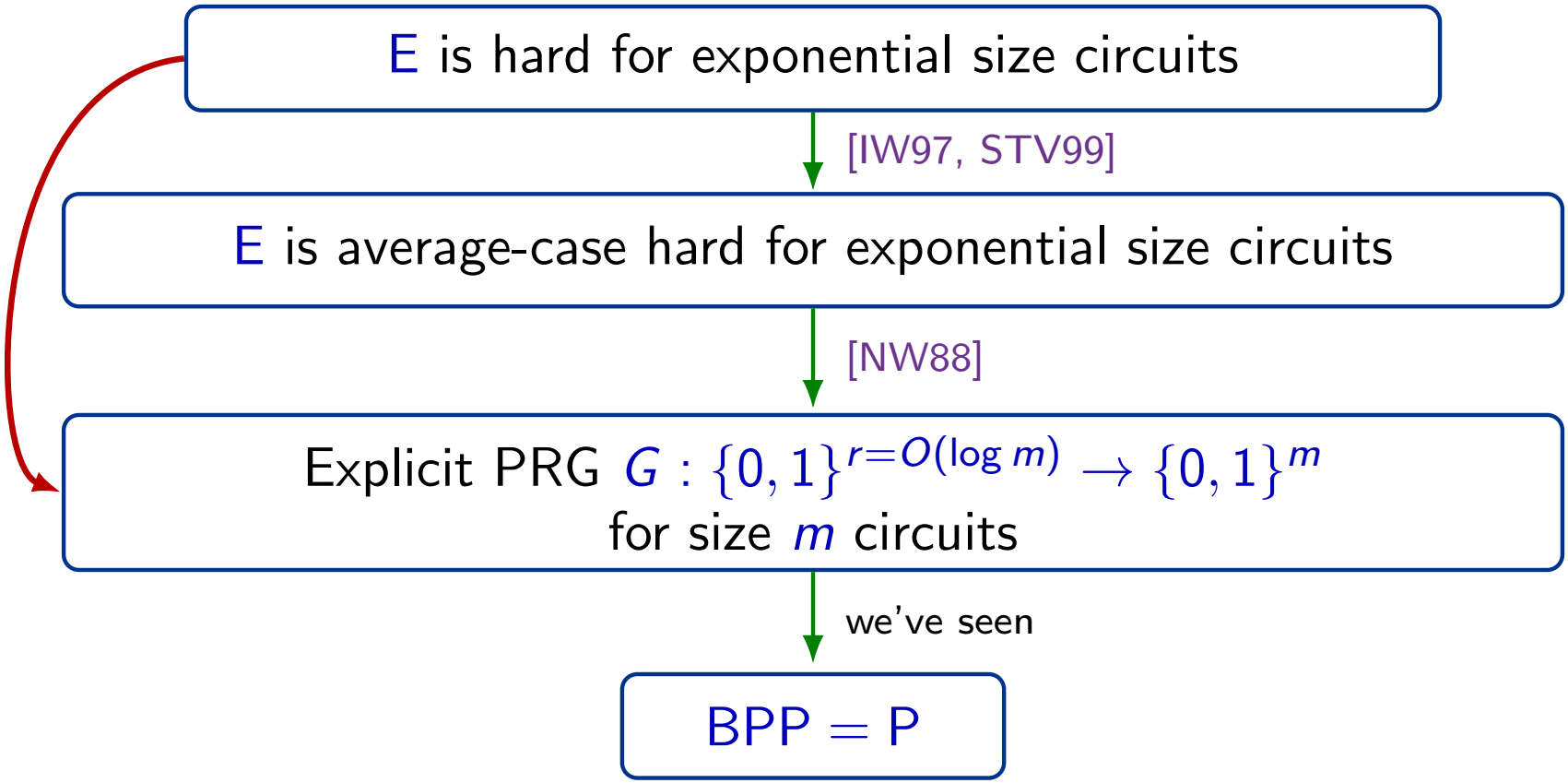
E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.



Hardness assumptions $\Rightarrow$ derandomization: recap

Theorem (IW97)

E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.

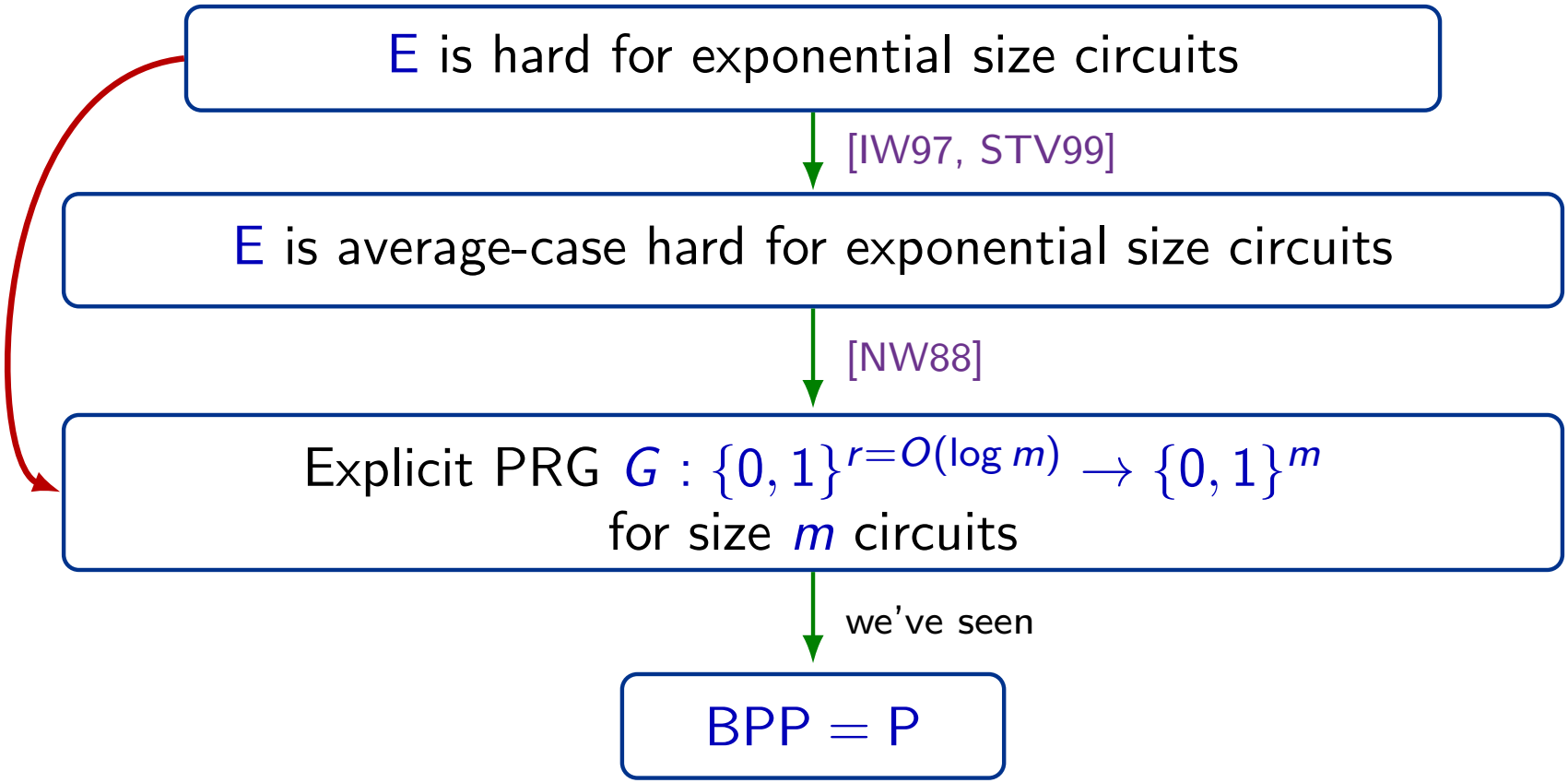


We'll now sketch a different PRG construction [SU01,Uma02].

Hardness assumptions $\Rightarrow$ derandomization: recap

Theorem (IW97)

E *is hard for exponential size circuits* $\Rightarrow$ BPP $=$ P.



We'll now sketch a different PRG construction [SU01,Uma02].

Does not use designs, and has shorter seed length in some scenarios.

The SU PRG (high level idea I)

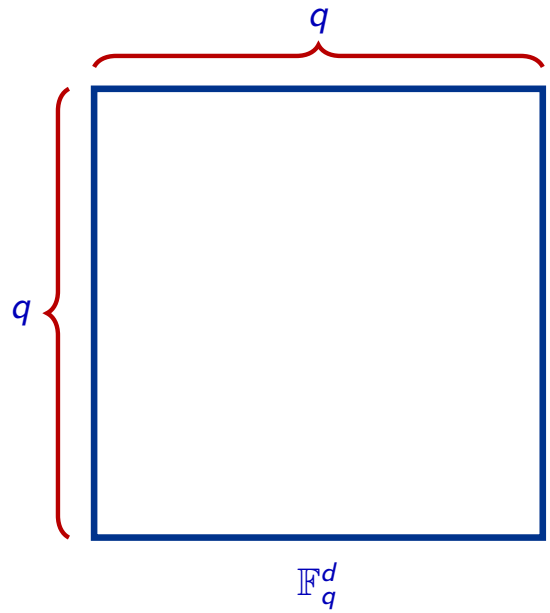
We've seen hardness amplification

$f : \{0,1\}^{\ell} \to \{0,1\} \longrightarrow \hat{f} : \mathbb{F}_q^d \to \mathbb{F}_q$.

The SU PRG (high level idea I)

We've seen hardness amplification

$f : \{0,1\}^{\ell} \to \{0,1\} \longrightarrow \hat{f} : \mathbb{F}_q^d \to \mathbb{F}_q$.

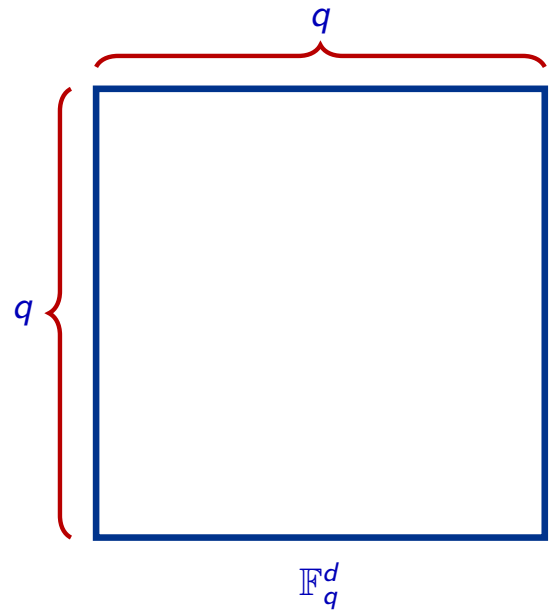


The SU PRG (high level idea I)

We've seen hardness amplification

$f : \{0,1\}^\ell \to \{0,1\} \longrightarrow \hat{f} : \mathbb{F}_q^d \to \mathbb{F}_q$.

Here $H^d \subseteq \mathbb{F}_q^d$ is identified with $\{0,1\}^\ell$.

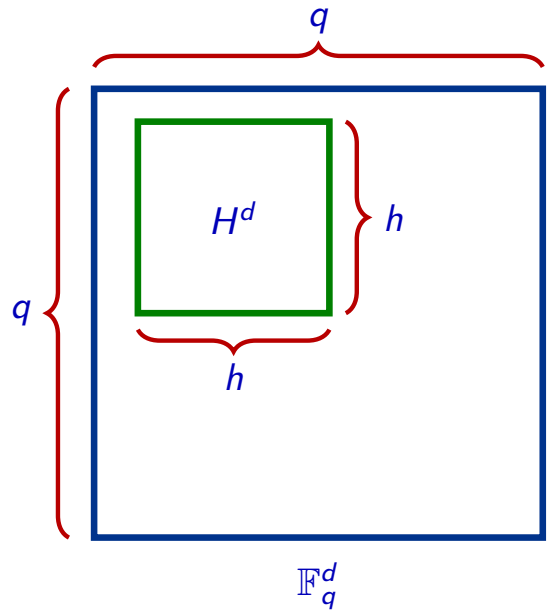


The SU PRG (high level idea I)

We've seen hardness amplification

$f : \{0,1\}^\ell \to \{0,1\} \longrightarrow \hat{f} : \mathbb{F}_q^d \to \mathbb{F}_q$.

Here $H^d \subseteq \mathbb{F}_q^d$ is identified with $\{0,1\}^\ell$.

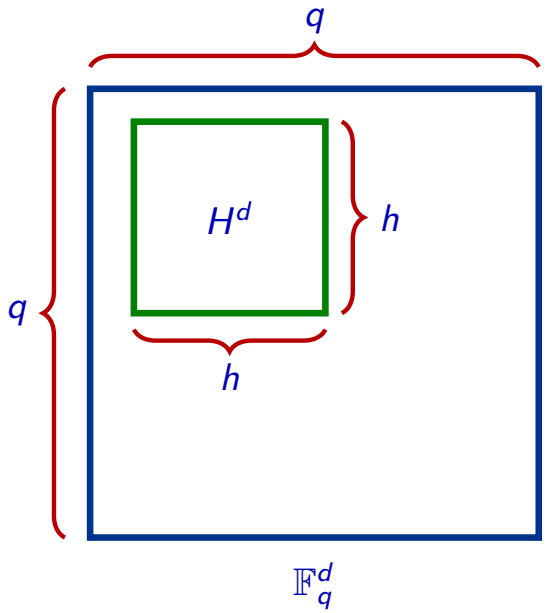


The SU PRG (high level idea I)

We've seen hardness amplification
$f : \{0,1\}^\ell \to \{0,1\} \longrightarrow \hat{f} : \mathbb{F}_q^d \to \mathbb{F}_q$.

Here $H^d \subseteq \mathbb{F}_q^d$ is identified with $\{0,1\}^\ell$.

We've seen a 1-bit stretch PRG, $G(x) = x, \hat{f}(x)$.
(Slightly cheating and ignoring the GL step.)

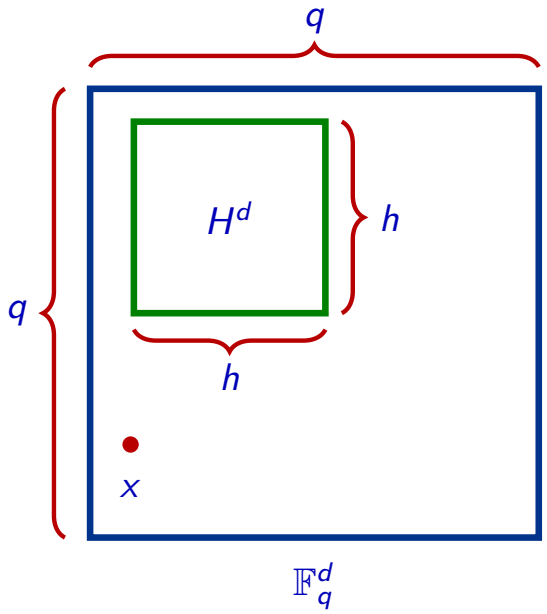


The SU PRG (high level idea I)

We've seen hardness amplification
$f : \{0,1\}^\ell \to \{0,1\} \longrightarrow \hat{f} : \mathbb{F}_q^d \to \mathbb{F}_q$.

Here $H^d \subseteq \mathbb{F}_q^d$ is identified with $\{0,1\}^\ell$.

We've seen a 1-bit stretch PRG, $G(x) = x, \hat{f}(x)$.
(Slightly cheating and ignoring the GL step.)



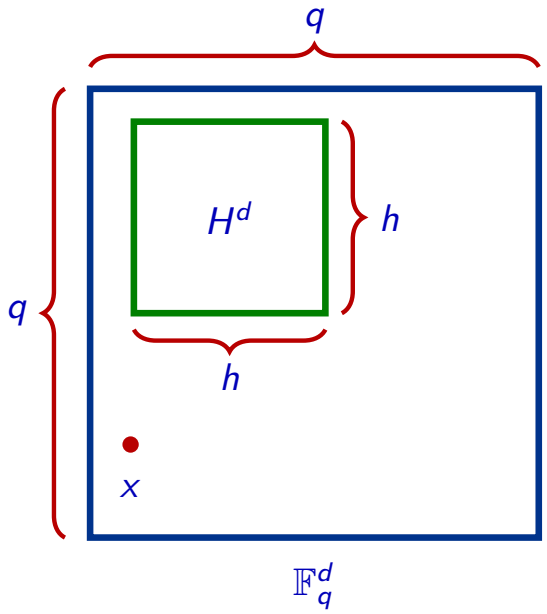
The SU PRG (high level idea I)

We've seen hardness amplification
$f : \{0,1\}^\ell \to \{0,1\} \longrightarrow \hat{f} : \mathbb{F}_q^d \to \mathbb{F}_q$.

Here $H^d \subseteq \mathbb{F}_q^d$ is identified with $\{0,1\}^\ell$.

We've seen a 1-bit stretch PRG, $G(x) = x, \hat{f}(x)$.
(Slightly cheating and ignoring the GL step.)

Question: How do we output more bits?



The SU PRG (high level idea I)

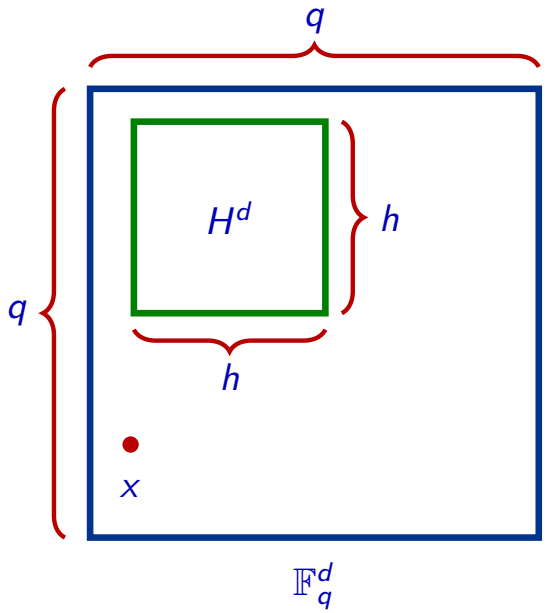
We've seen hardness amplification
$f : \{0,1\}^\ell \to \{0,1\} \longrightarrow \hat{f} : \mathbb{F}_q^d \to \mathbb{F}_q$.

Here $H^d \subseteq \mathbb{F}_q^d$ is identified with $\{0,1\}^\ell$.

We've seen a 1-bit stretch PRG, $G(x) = x, \hat{f}(x)$.
(Slightly cheating and ignoring the GL step.)

Question: How do we output more bits?

Idea: [TZS01] Fix some $d \times d$ invertible matrix A over $\mathbb{F}_q$ and define:



The SU PRG (high level idea I)

We've seen hardness amplification
$f : \{0,1\}^\ell \to \{0,1\} \longrightarrow \hat{f} : \mathbb{F}_q^d \to \mathbb{F}_q$.

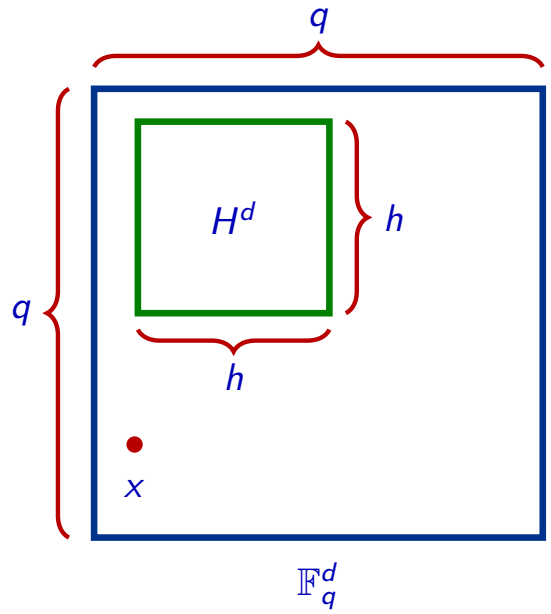
Here $H^d \subseteq \mathbb{F}_q^d$ is identified with $\{0,1\}^\ell$.

We've seen a 1-bit stretch PRG, $G(x) = x, \hat{f}(x)$.
(Slightly cheating and ignoring the GL step.)

Question: How do we output more bits?

Idea: [TZS01] Fix some $d \times d$ invertible matrix A over $\mathbb{F}_q$ and define:

$$G(x) = x, \hat{f}(x), \hat{f}(Ax), \hat{f}(A^2x), \ldots, \hat{f}(A^{m-1}x).$$



The SU PRG (high level idea I)

We've seen hardness amplification
$f : \{0,1\}^\ell \to \{0,1\} \longrightarrow \hat{f} : \mathbb{F}_q^d \to \mathbb{F}_q$.

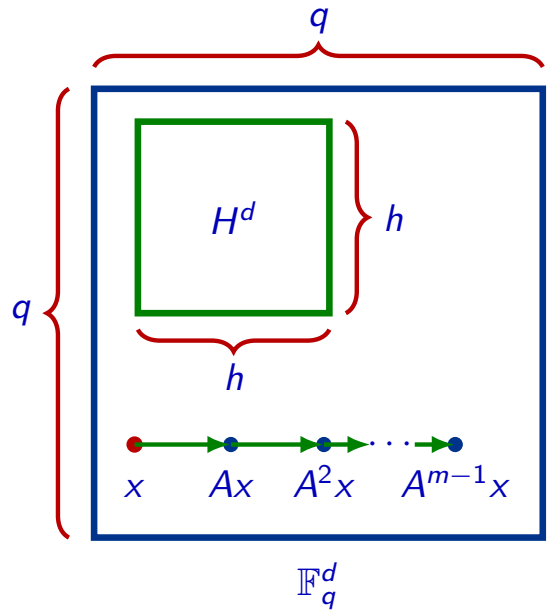
Here $H^d \subseteq \mathbb{F}_q^d$ is identified with $\{0,1\}^\ell$.

We've seen a 1-bit stretch PRG, $G(x) = x, \hat{f}(x)$.
(Slightly cheating and ignoring the GL step.)

Question: How do we output more bits?

Idea: [TZS01] Fix some $d \times d$ invertible matrix A over $\mathbb{F}_q$ and define:

$$G(x) = x, \hat{f}(x), \hat{f}(Ax), \hat{f}(A^2x), \ldots, \hat{f}(A^{m-1}x).$$



The SU PRG (high level idea I)

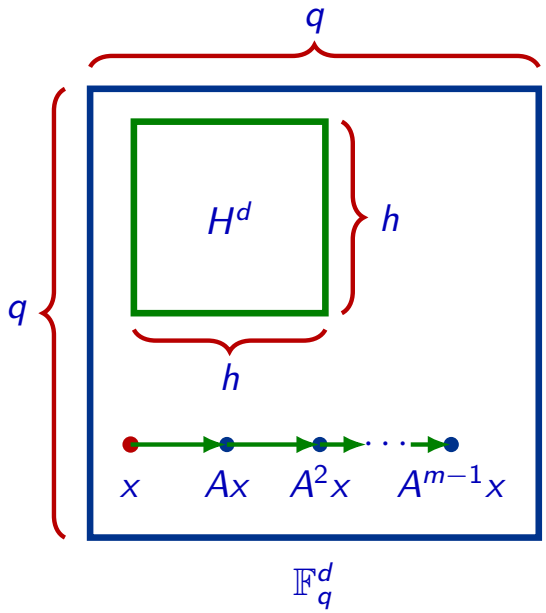
We've seen hardness amplification
$f : \{0,1\}^\ell \to \{0,1\} \longrightarrow \hat{f} : \mathbb{F}_q^d \to \mathbb{F}_q$.

Here $H^d \subseteq \mathbb{F}_q^d$ is identified with $\{0,1\}^\ell$.

We've seen a 1-bit stretch PRG, $G(x) = x, \hat{f}(x)$.
(Slightly cheating and ignoring the GL step.)

Question: How do we output more bits?

Idea: [TZS01] Fix some $d \times d$ invertible matrix A over $\mathbb{F}_q$ and define:



$$G(x) = x, \hat{f}(x), \hat{f}(Ax), \hat{f}(A^2x), \ldots, \hat{f}(A^{m-1}x).$$

Recall: Distinguisher $\Rightarrow$ Predictor: We need to redo [STV99], but instead of a circuit $\hat{C}(x)$ s.t.

$$\Pr_{x \leftarrow \mathbb{F}_q^d}[\hat{C}(x) = \hat{f}(x)] > 1/2 + 1/\hat{s},$$

we get a "predictor circuit" $\hat{P}$.

The SU PRG (high level idea I)

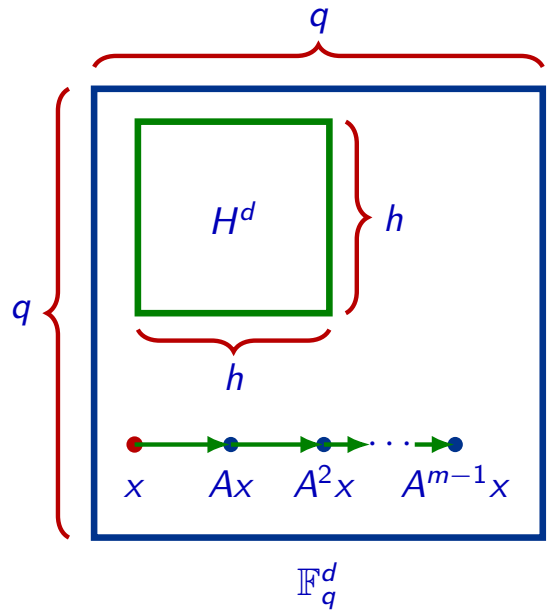
We've seen hardness amplification
$f : \{0,1\}^\ell \to \{0,1\} \longrightarrow \hat{f} : \mathbb{F}_q^d \to \mathbb{F}_q$.

Here $H^d \subseteq \mathbb{F}_q^d$ is identified with $\{0,1\}^\ell$.

We've seen a 1-bit stretch PRG, $G(x) = x, \hat{f}(x)$.
(Slightly cheating and ignoring the GL step.)

Question: How do we output more bits?

Idea: [TZS01] Fix some $d \times d$ invertible matrix A over $\mathbb{F}_q$ and define:



$$G(x) = x, \hat{f}(x), \hat{f}(Ax), \hat{f}(A^2x), \ldots, \hat{f}(A^{m-1}x).$$

Recall: Distinguisher $\Rightarrow$ Predictor: We need to redo [STV99], but instead of a circuit $\hat{C}(x)$ s.t.

$$\Pr_{x \leftarrow \mathbb{F}_q^d}[\hat{C}(x) = \hat{f}(x)] > 1/2 + 1/\hat{s},$$

we get a "predictor circuit" $\hat{P}$.

$$\Pr_{x \leftarrow \mathbb{F}_q^d}\left[\hat{P}\left(x, \hat{f}(A^{-(i-1)}x), \hat{f}(A^{-(i-2)}x), \ldots, \hat{f}(A^{-1}x)\right) = \hat{f}(x)\right] > 1/2 + 1/\hat{s}.$$

The SU PRG (high level idea I)

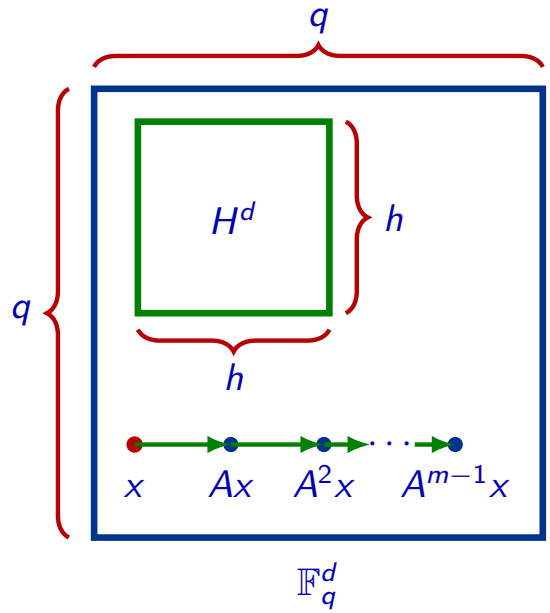
We've seen hardness amplification
$f : \{0,1\}^\ell \to \{0,1\} \longrightarrow \hat{f} : \mathbb{F}_q^d \to \mathbb{F}_q$.

Here $H^d \subseteq \mathbb{F}_q^d$ is identified with $\{0,1\}^\ell$.

We've seen a 1-bit stretch PRG, $G(x) = x, \hat{f}(x)$.
(Slightly cheating and ignoring the GL step.)

Question: How do we output more bits?

Idea: [TZS01] Fix some $d \times d$ invertible matrix A over $\mathbb{F}_q$ and define:



$$G(x) = x, \hat{f}(x), \hat{f}(Ax), \hat{f}(A^2x), \ldots, \hat{f}(A^{m-1}x).$$

Recall: Distinguisher $\Rightarrow$ Predictor: We need to redo [STV99], but instead of a circuit $\hat{C}(x)$ s.t.

$$\Pr_{x \leftarrow \mathbb{F}_q^d}[\hat{C}(x) = \hat{f}(x)] > 1/2 + 1/\hat{s},$$

we get a "predictor circuit" $\hat{P}$.

$$\Pr_{x \leftarrow \mathbb{F}_q^d}\left[\hat{P}\left(x, \hat{f}(A^{-(i-1)}x), \hat{f}(A^{-(i-2)}x), \ldots, \hat{f}(A^{-1}x)\right) = \hat{f}(x)\right] > 1/2 + 1/\hat{s}.$$

Idea: PRG = STV99 + predictor (from previous elements).

The SU PRG (high level idea II)

In [STV99] we use coding machinery to "error-correct" from small agreement to perfect agreement.

The SU PRG (high level idea II)

In [STV99] we use coding machinery to "error-correct" from small agreement to perfect agreement.

We are given a predictor $\hat{P}$ with success probability $1/\hat{s}$. Imagine that we already "error-corrected" it so that

$$\Pr_{x \leftarrow \mathbb{F}_q^d}\left[\hat{P}\left(x, \hat{f}(A^{-m}x), \hat{f}(A^{-(m-1)}x), \ldots, \hat{f}(A^{-1}x)\right) = \hat{f}(x)\right] = 1.$$

The SU PRG (high level idea II)

In [STV99] we use coding machinery to "error-correct" from small agreement to perfect agreement.

We are given a predictor $\hat{P}$ with success probability $1/\hat{s}$. Imagine that we already "error-corrected" it so that

$$\Pr_{x \leftarrow \mathbb{F}_q^d}\left[\hat{P}\left(x, \hat{f}(A^{-m}x), \hat{f}(A^{-(m-1)}x), \ldots, \hat{f}(A^{-1}x)\right) = \hat{f}(x)\right] = 1.$$

Goal: Construct a circuit $C(x)$ that computes $\hat{f}(x)$ correctly.

The SU PRG (high level idea II)

In [STV99] we use coding machinery to "error-correct" from small agreement to perfect agreement.

We are given a predictor $\hat{P}$ with success probability $1/\hat{s}$. Imagine that we already "error-corrected" it so that

$$\Pr_{x \leftarrow \mathbb{F}_q^d}\left[\hat{P}\left(x, \hat{f}(A^{-m}x), \hat{f}(A^{-(m-1)}x), \ldots, \hat{f}(A^{-1}x)\right) = \hat{f}(x)\right] = 1.$$

Goal: Construct a circuit $C(x)$ that computes $\hat{f}(x)$ correctly.

We will choose A to be a "traversing" matrix. Namely, for some $x_0 \in \mathbb{F}_q^d$,

$$\left\{x_0, Ax_0, A^2x_0, \ldots\right\} = \mathbb{F}_q^d \setminus \{0\}.$$

The SU PRG (high level idea II)

In [STV99] we use coding machinery to "error-correct" from small agreement to perfect agreement.

We are given a predictor $\hat{P}$ with success probability $1/\hat{s}$. Imagine that we already "error-corrected" it so that

$$\Pr_{x \leftarrow \mathbb{F}_q^d}\left[\hat{P}\left(x, \hat{f}(A^{-m}x), \hat{f}(A^{-(m-1)}x), \ldots, \hat{f}(A^{-1}x)\right) = \hat{f}(x)\right] = 1.$$

Goal: Construct a circuit $C(x)$ that computes $\hat{f}(x)$ correctly.

We will choose A to be a "traversing" matrix. Namely, for some $x_0 \in \mathbb{F}_q^d$,

$$\left\{x_0, Ax_0, A^2x_0, \ldots\right\} = \mathbb{F}_q^d \setminus \{0\}.$$

d-dimensional

The SU PRG (high level idea II)

In [STV99] we use coding machinery to "error-correct" from small agreement to perfect agreement.

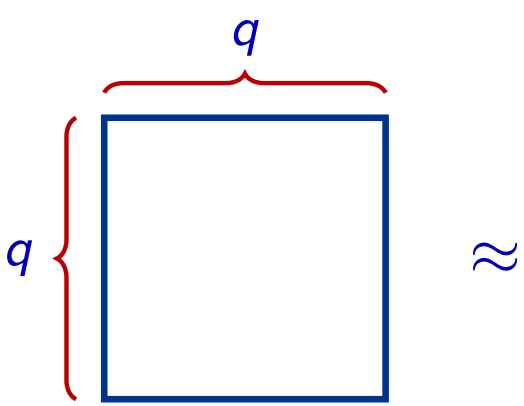
We are given a predictor $\hat{P}$ with success probability $1/\hat{s}$. Imagine that we already "error-corrected" it so that

$$\Pr_{x \leftarrow \mathbb{F}_q^d}\left[\hat{P}\left(x, \hat{f}(A^{-m}x), \hat{f}(A^{-(m-1)}x), \ldots, \hat{f}(A^{-1}x)\right) = \hat{f}(x)\right] = 1.$$

Goal: Construct a circuit $C(x)$ that computes $\hat{f}(x)$ correctly.

We will choose A to be a "traversing" matrix. Namely, for some $x_0 \in \mathbb{F}_q^d$,

$$\left\{x_0, Ax_0, A^2x_0, \ldots\right\} = \mathbb{F}_q^d \setminus \{0\}.$$



The SU PRG (high level idea II)

In [STV99] we use coding machinery to "error-correct" from small agreement to perfect agreement.

We are given a predictor $\hat{P}$ with success probability $1/\hat{s}$. Imagine that we already "error-corrected" it so that

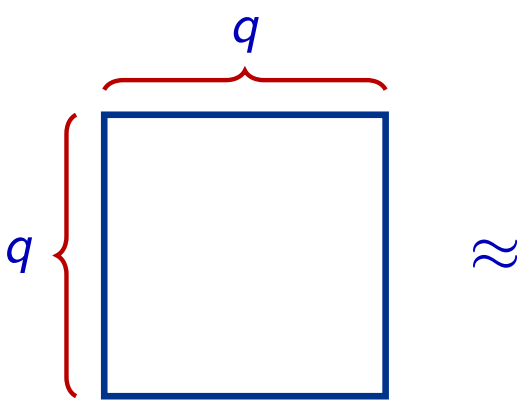
$$\Pr_{x \leftarrow \mathbb{F}_q^d}\left[\hat{P}\left(x, \hat{f}(A^{-m}x), \hat{f}(A^{-(m-1)}x), \ldots, \hat{f}(A^{-1}x)\right) = \hat{f}(x)\right] = 1.$$

Goal: Construct a circuit $C(x)$ that computes $\hat{f}(x)$ correctly.

We will choose A to be a "traversing" matrix. Namely, for some $x_0 \in \mathbb{F}_q^d$,

$$\left\{x_0, Ax_0, A^2x_0, \ldots\right\} = \mathbb{F}_q^d \setminus \{0\}.$$

d-dimensional **one dimensional**



The SU PRG (high level idea II)

In [STV99] we use coding machinery to "error-correct" from small agreement to perfect agreement.

We are given a predictor $\hat{P}$ with success probability $1/\hat{s}$. Imagine that we already "error-corrected" it so that

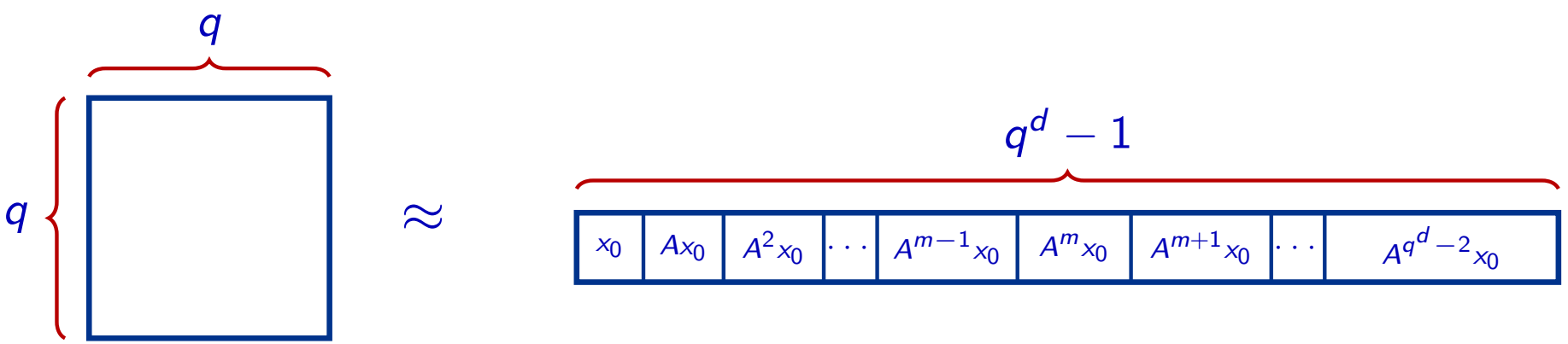
$$\Pr_{x \leftarrow \mathbb{F}_q^d}\left[\hat{P}\left(x, \hat{f}(A^{-m}x), \hat{f}(A^{-(m-1)}x), \ldots, \hat{f}(A^{-1}x)\right) = \hat{f}(x)\right] = 1.$$

Goal: Construct a circuit $C(x)$ that computes $\hat{f}(x)$ correctly.

We will choose A to be a "traversing" matrix. Namely, for some $x_0 \in \mathbb{F}_q^d$,

$$\left\{x_0, Ax_0, A^2x_0, \ldots\right\} = \mathbb{F}_q^d \setminus \{0\}.$$

d-dimensional **one dimensional**



The SU PRG (high level idea III)

We assume that we have an "error-corrected" predictor $\hat{P}$ s.t. for every $x \in \mathbb{F}_q^d$,

$$\hat{P}\left(x, \hat{f}(A^{-m}x), \hat{f}(A^{-(m-1)}x), \ldots, \hat{f}(A^{-1}x)\right) = \hat{f}(x).$$

The SU PRG (high level idea III)

We assume that we have an "error-corrected" predictor $\hat{P}$ s.t. for every $x \in \mathbb{F}_q^d$,

$$\hat{P}\left(x, \hat{f}(A^{-m}x), \hat{f}(A^{-(m-1)}x), \ldots, \hat{f}(A^{-1}x)\right) = \hat{f}(x).$$

C will be hardwired with $\hat{f}(x_0), \hat{f}(Ax_0), \ldots, \hat{f}(A^{m-1}x_0)$.

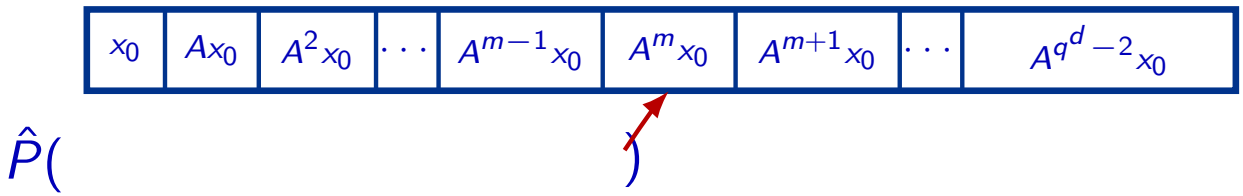
x_0	Ax_0	A^2x_0	$\cdots$	$A^{m-1}x_0$	$A^m x_0$	$A^{m+1}x_0$	$\cdots$	$A^{q^d-2}x_0$

The SU PRG (high level idea III)

We assume that we have an "error-corrected" predictor $\hat{P}$ s.t. for every $x \in \mathbb{F}_q^d$,

$$\hat{P}\Big(x, \hat{f}(A^{-m}x), \hat{f}(A^{-(m-1)}x), \ldots, \hat{f}(A^{-1}x)\Big) = \hat{f}(x).$$

C will be hardwired with $\hat{f}(x_0), \hat{f}(Ax_0), \ldots, \hat{f}(A^{m-1}x_0)$.



Obtain the next value: $\boxed{x_0, \ldots, A^{m-1}x_0} \xrightarrow{\hat{P}} \boxed{A^m x_0}$:

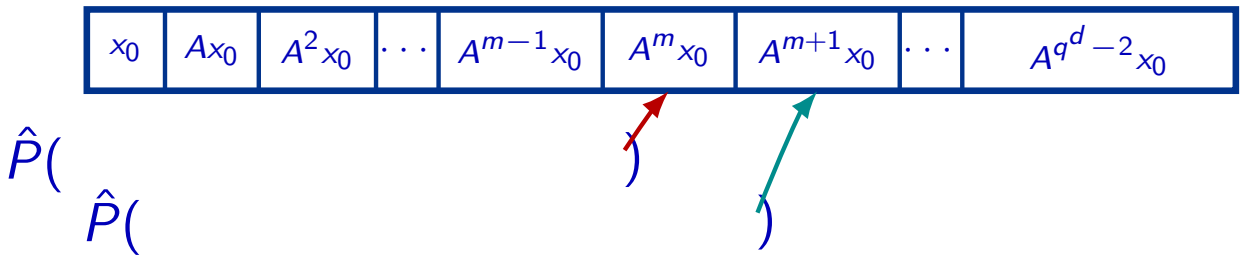
$$\hat{f}(A^m x_0) = \hat{P}\Big(A^m x_0, \hat{f}(x_0), \hat{f}(Ax_0), \ldots, \hat{f}(A^{m-1}x_0)\Big).$$

The SU PRG (high level idea III)

We assume that we have an "error-corrected" predictor $\hat{P}$ s.t. for every $x \in \mathbb{F}_q^d$,

$$\hat{P}\Big(x, \hat{f}(A^{-m}x), \hat{f}(A^{-(m-1)}x), \ldots, \hat{f}(A^{-1}x)\Big) = \hat{f}(x).$$

C will be hardwired with $\hat{f}(x_0), \hat{f}(Ax_0), \ldots, \hat{f}(A^{m-1}x_0)$.



Obtain the next value: $\boxed{x_0, \ldots, A^{m-1}x_0} \xrightarrow{\hat{P}} \boxed{A^m x_0}$:

$$\hat{f}(A^m x_0) = \hat{P}\Big(A^m x_0, \hat{f}(x_0), \hat{f}(Ax_0), \ldots, \hat{f}(A^{m-1}x_0)\Big).$$

Obtain the next value: $\boxed{Ax_0, \ldots, A^m x_0} \xrightarrow{\hat{P}} \boxed{A^{m+1}x_0}$:

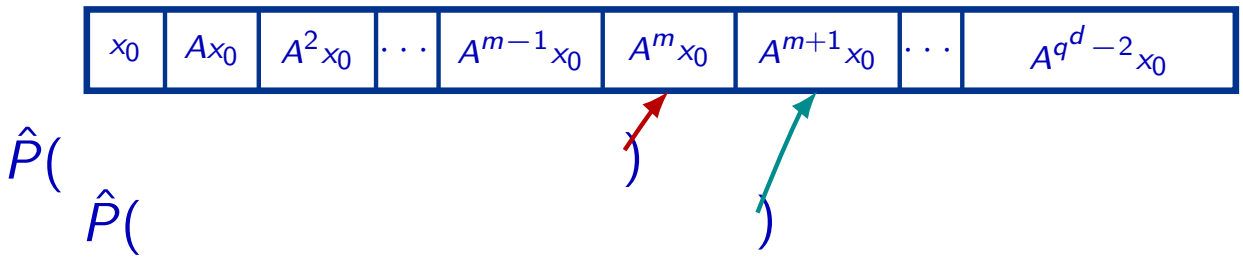
$$\hat{f}(A^{m+1}x_0) = \hat{P}\Big(A^{m+1}x_0, \hat{f}(Ax_0), \ldots, \hat{f}(A^m x_0)\Big).$$

The SU PRG (high level idea III)

We assume that we have an "error-corrected" predictor $\hat{P}$ s.t. for every $x \in \mathbb{F}_q^d$,

$$\hat{P}\left(x, \hat{f}(A^{-m}x), \hat{f}(A^{-(m-1)}x), \ldots, \hat{f}(A^{-1}x)\right) = \hat{f}(x).$$

C will be hardwired with $\hat{f}(x_0), \hat{f}(Ax_0), \ldots, \hat{f}(A^{m-1}x_0)$.



Obtain the next value: $\boxed{x_0, \ldots, A^{m-1}x_0} \xrightarrow{\hat{P}} \boxed{A^m x_0}$:

$$\hat{f}(A^m x_0) = \hat{P}\left(A^m x_0, \hat{f}(x_0), \hat{f}(Ax_0), \ldots, \hat{f}(A^{m-1}x_0)\right).$$

Obtain the next value: $\boxed{Ax_0, \ldots, A^m x_0} \xrightarrow{\hat{P}} \boxed{A^{m+1}x_0}$:

$$\hat{f}(A^{m+1}x_0) = \hat{P}\left(A^{m+1}x_0, \hat{f}(Ax_0), \ldots, \hat{f}(A^m x_0)\right).$$

$\Rightarrow$ Eventually we reconstruct $\hat{f}$ correctly on every point of $\mathbb{F}_q^d$, and hence reconstruct f correctly on every point of $\{0,1\}^\ell$.

The SU PRG (high level idea IV)

Difficulties:

The SU PRG (high level idea IV)

Difficulties:

- In [STV99], each decoding step requires hardwiring $\hat{f}$ on an additional point (to find the correct polynomial in the list).

The SU PRG (high level idea IV)

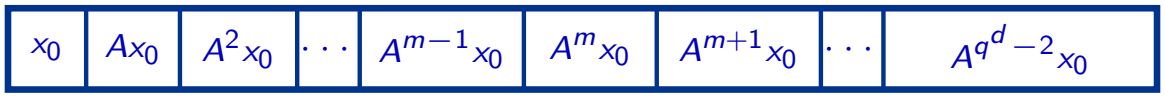
Difficulties:

- In [STV99], each decoding step requires hardwiring $\hat{f}$ on an additional point (to find the correct polynomial in the list).
- Doing it this way is too expensive.

The SU PRG (high level idea IV)

Difficulties:

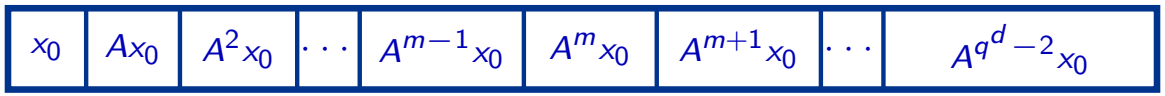
- In [STV99], each decoding step requires hardwiring $\hat{f}$ on an additional point (to find the correct polynomial in the list).
- Doing it this way is too expensive.



The SU PRG (high level idea IV)

Difficulties:

- In [STV99], each decoding step requires hardwiring $\hat{f}$ on an additional point (to find the correct polynomial in the list).
- Doing it this way is too expensive.

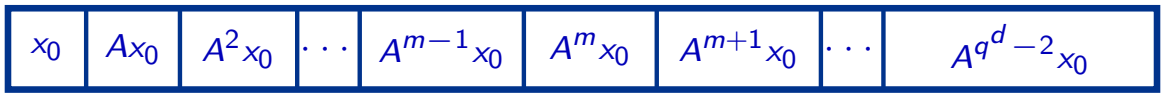


- Reconstructing $\hat{f}(x)$ for an x that is "far away" from x_0 takes $q^d > 2^\ell$ steps.

The SU PRG (high level idea IV)

Difficulties:

- In [STV99], each decoding step requires hardwiring $\hat{f}$ on an additional point (to find the correct polynomial in the list).
- Doing it this way is too expensive.

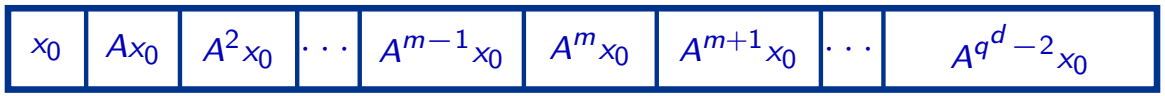


- Reconstructing $\hat{f}(x)$ for an x that is "far away" from x_0 takes $q^d > 2^\ell$ steps.
- This will lead to a circuit of size larger than $q^d > 2^\ell$ for a function on ℓ bits (and this is not helpful).

The SU PRG (high level idea IV)

Difficulties:

- In [STV99], each decoding step requires hardwiring $\hat{f}$ on an additional point (to find the correct polynomial in the list).
- Doing it this way is too expensive.

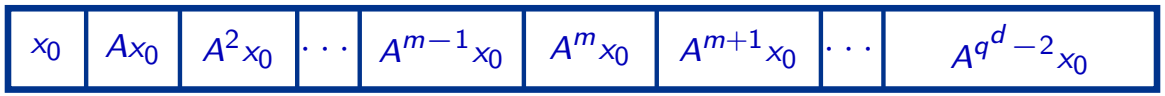


- Reconstructing $\hat{f}(x)$ for an x that is "far away" from x_0 takes $q^d > 2^\ell$ steps.
- This will lead to a circuit of size larger than $q^d > 2^\ell$ for a function on ℓ bits (and this is not helpful).
- These problems require more work and additional ideas in [SU01,Umans02] that we will not explain here.

The SU PRG (high level idea IV)

Difficulties:

- In [STV99], each decoding step requires hardwiring $\hat{f}$ on an additional point (to find the correct polynomial in the list).
- Doing it this way is too expensive.



- Reconstructing $\hat{f}(x)$ for an x that is "far away" from x_0 takes $q^d > 2^\ell$ steps.
- This will lead to a circuit of size larger than $q^d > 2^\ell$ for a function on ℓ bits (and this is not helpful).
- These problems require more work and additional ideas in [SU01,Umans02] that we will not explain here.
- We will now explain one clean part: how to find a traversing matrix.

The SU PRG (high level idea V)

Finding a traversing matrix:

The SU PRG (high level idea V)

Finding a traversing matrix:

We want to find a $d \times d$ matrix A over $\mathbb{F}_q$ such that, for some $x_0 \in \mathbb{F}_q^d$,

$$\left\{x_0, Ax_0, A^2x_0, \ldots\right\} = \mathbb{F}_q^d \setminus \{0\}\,.$$

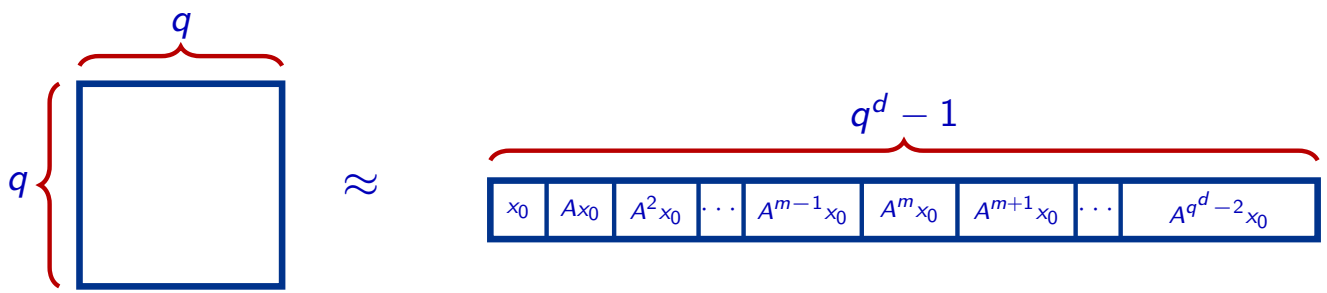
The SU PRG (high level idea V)

Finding a traversing matrix:

We want to find a $d \times d$ matrix A over $\mathbb{F}_q$ such that, for some $x_0 \in \mathbb{F}_q^d$,

$$\left\{x_0, Ax_0, A^2x_0, \ldots\right\} = \mathbb{F}_q^d \setminus \{0\}\,.$$

d-dimensional **one dimensional**



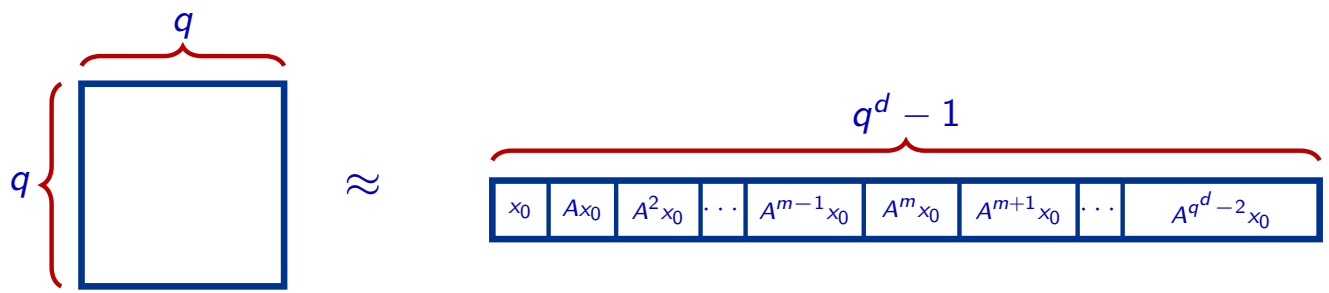
The SU PRG (high level idea V)

Finding a traversing matrix:

We want to find a $d \times d$ matrix A over $\mathbb{F}_q$ such that, for some $x_0 \in \mathbb{F}_q^d$,

$$\left\{x_0, Ax_0, A^2x_0, \ldots\right\} = \mathbb{F}_q^d \setminus \{0\}\,.$$

d-dimensional **one dimensional**



We identify the vector space $\mathbb{F}_q^d$ with the field $\mathbb{F}_{q^d}$.

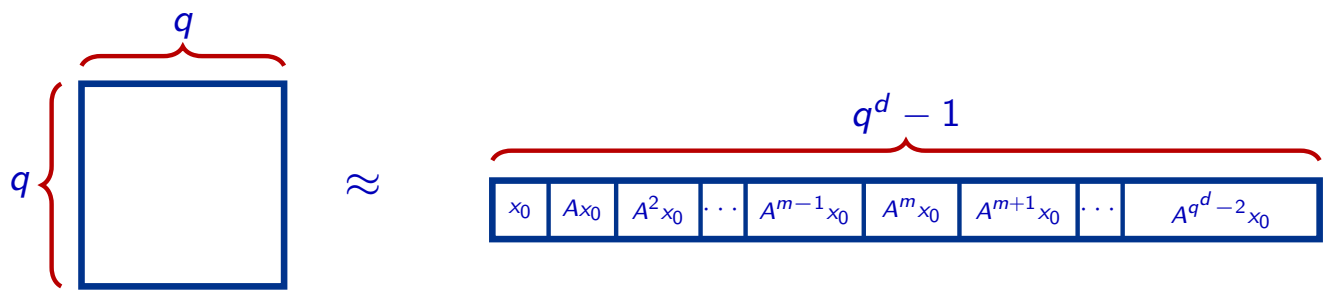
The SU PRG (high level idea V)

Finding a traversing matrix:

We want to find a $d \times d$ matrix A over $\mathbb{F}_q$ such that, for some $x_0 \in \mathbb{F}_q^d$,

$$\left\{x_0, Ax_0, A^2x_0, \ldots\right\} = \mathbb{F}_q^d \setminus \{0\}\,.$$

d-dimensional **one dimensional**



We identify the vector space $\mathbb{F}_q^d$ with the field $\mathbb{F}_{q^d}$.

The multiplicative group of $\mathbb{F}_{q^d}$ is cyclic $\Rightarrow$ it has a generator $g \in \mathbb{F}_{q^d}$.

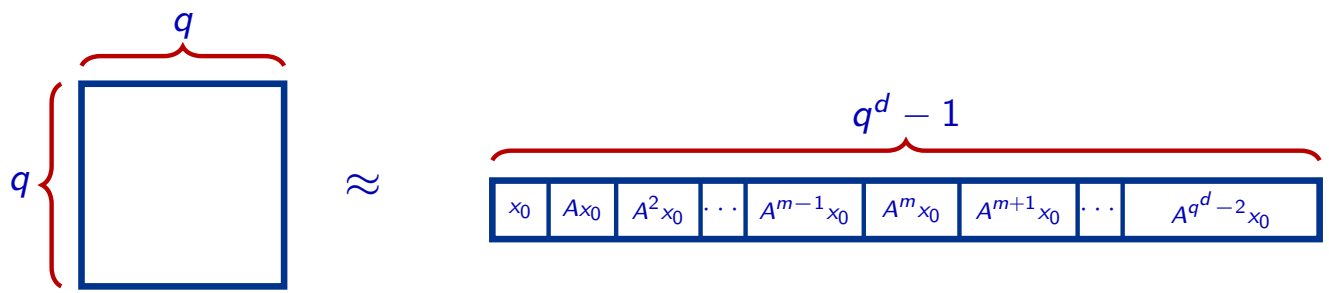
The SU PRG (high level idea V)

Finding a traversing matrix:

We want to find a $d \times d$ matrix A over $\mathbb{F}_q$ such that, for some $x_0 \in \mathbb{F}_q^d$,

$$\left\{x_0, Ax_0, A^2x_0, \ldots\right\} = \mathbb{F}_q^d \setminus \{0\}.$$

d-dimensional **one dimensional**



We identify the vector space $\mathbb{F}_q^d$ with the field $\mathbb{F}_{q^d}$.

The multiplicative group of $\mathbb{F}_{q^d}$ is cyclic $\Rightarrow$ it has a generator $g \in \mathbb{F}_{q^d}$.

The matrix A will be the $d \times d$ matrix that corresponds to the $\mathbb{F}_q$-linear transformation T over $\mathbb{F}_q^d$ defined by

$$T(x) = g \cdot x.$$

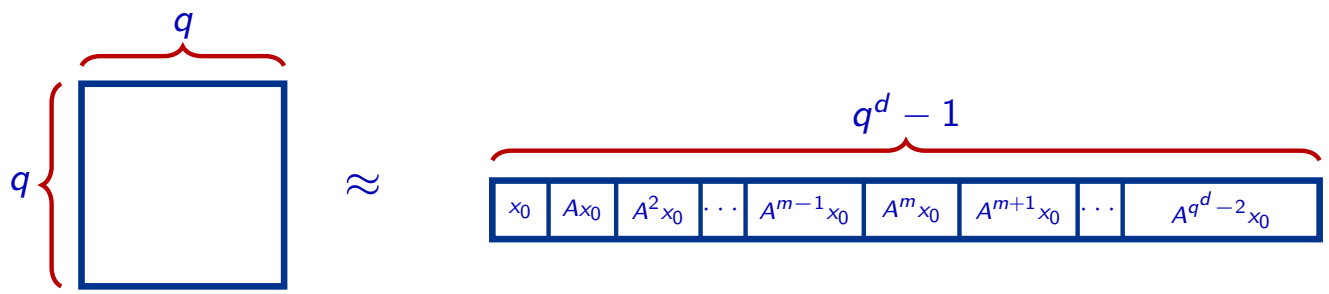
The SU PRG (high level idea V)

Finding a traversing matrix:

We want to find a $d \times d$ matrix A over $\mathbb{F}_q$ such that, for some $x_0 \in \mathbb{F}_q^d$,

$$\left\{x_0, Ax_0, A^2x_0, \ldots\right\} = \mathbb{F}_q^d \setminus \{0\}.$$

d-dimensional **one dimensional**



We identify the vector space $\mathbb{F}_q^d$ with the field $\mathbb{F}_{q^d}$.

The multiplicative group of $\mathbb{F}_{q^d}$ is cyclic $\Rightarrow$ it has a generator $g \in \mathbb{F}_{q^d}$.

The matrix A will be the $d \times d$ matrix that corresponds to the $\mathbb{F}_q$-linear transformation T over $\mathbb{F}_q^d$ defined by

$$T(x) = g \cdot x.$$

We get that for every nonzero $x_0 \in \mathbb{F}_q^d$,

$$\left\{x_0, Ax_0, \ldots, A^{q^d-2}x_0\right\} = \left\{x_0, g \cdot x_0, \ldots, g^{q^d-2} \cdot x_0\right\} = \mathbb{F}_q^d \setminus \{0\}.$$

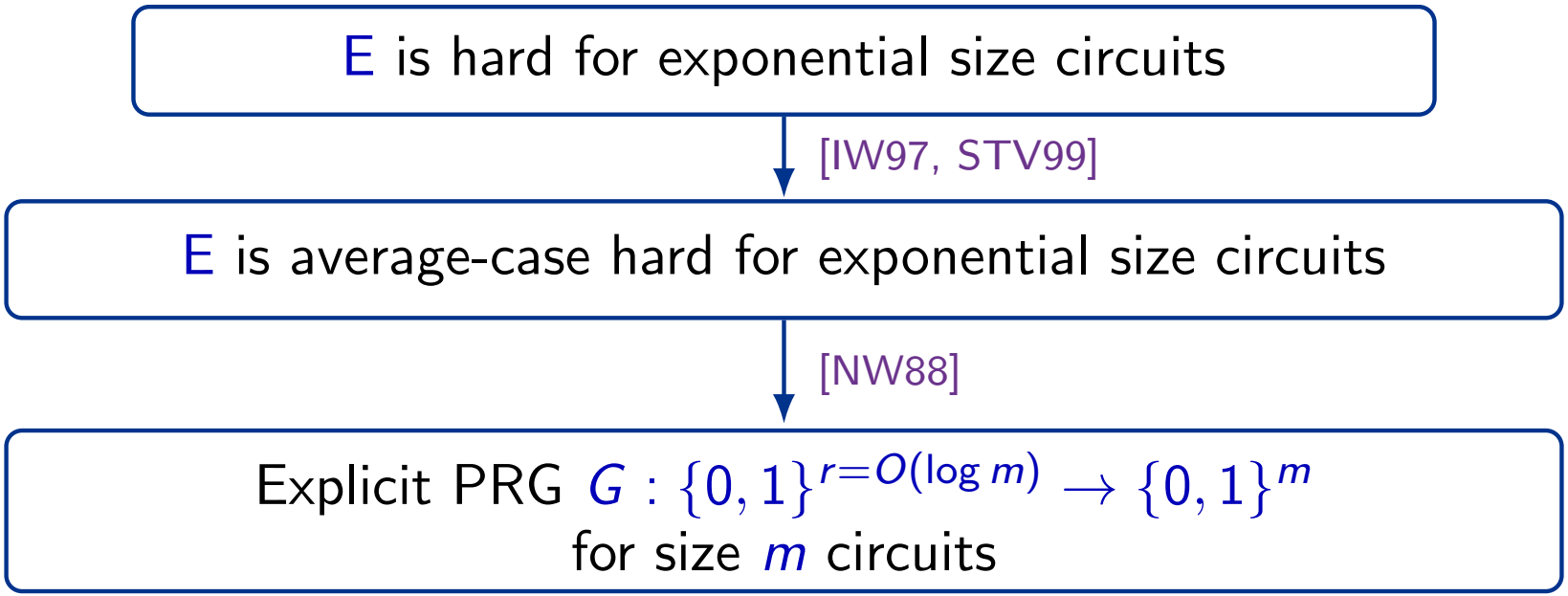
Hardness $\Rightarrow$ PRGs (review of argument):

Theorem (IW97)

E *is hard for exponential size circuits* $\Rightarrow$ *there is an explicit PRG*

$$G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$$

for size m *circuits.*



Hardness $\Rightarrow$ PRGs (fully black-box view)

Theorem (hardness $\Rightarrow$ PRGs - fully-black-box view)

- $\exists$ *oracle TM* $G^{(\cdot)}$ *s.t.* $\forall f : \{0,1\}^\ell \to \{0,1\}$, *setting* $m = 2^{\Omega(\ell)}$:
  $G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *is computable in* E^f.
  Consequently, $f \in \mathsf{E} \Rightarrow G^f$ *is explicit.*
- $\forall D : \{0,1\}^m \to \{0,1\}$ *(not necessarily a small circuit), if*

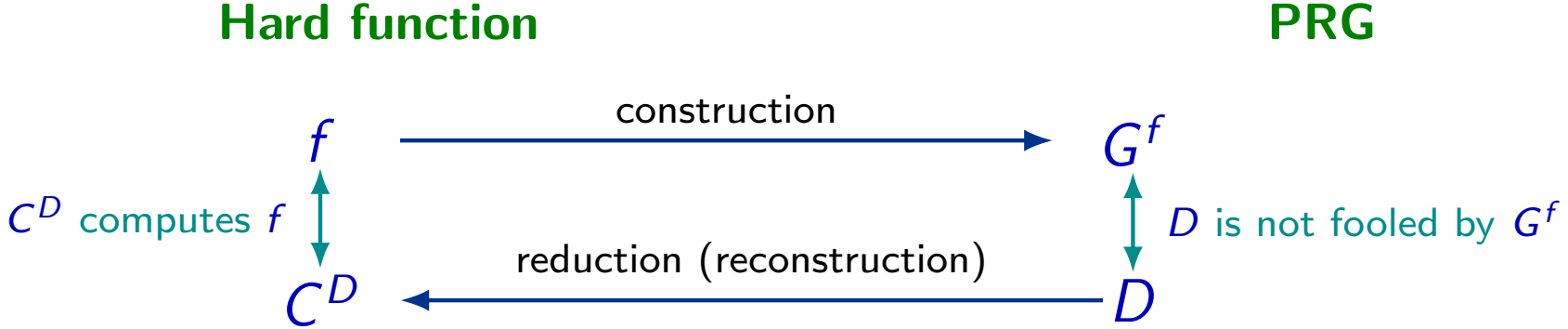
$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon = 1/10,$$

  then $\exists$ *size* $2^{o(\ell)}$ *oracle-circuit* $C^{(\cdot)}$ *s.t.* $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.
  Consequently, D *is small and breaks* $G^f \Rightarrow f$ *is not hard.*

Hardness $\Rightarrow$ PRGs (fully black-box view)

Theorem (hardness $\Rightarrow$ PRGs - fully-black-box view)

- $\exists$ *oracle TM* $G^{(\cdot)}$ *s.t.* $\forall f : \{0,1\}^\ell \to \{0,1\}$, *setting* $m = 2^{\Omega(\ell)}$: $G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ *is computable in* E^f.
  Consequently, $f \in \mathrm{E} \Rightarrow G^f$ *is explicit.*
- $\forall D : \{0,1\}^m \to \{0,1\}$ *(not necessarily a small circuit), if*
$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon = 1/10,$$
  then $\exists$ *size* $2^{o(\ell)}$ *oracle-circuit* $C^{(\cdot)}$ *s.t.* $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.
  Consequently, D *is small and breaks* $G^f \Rightarrow f$ *is not hard.*



Key point: We do not assume that f or D are efficient (except for the consequently clauses).

PRGs against other classes of circuits

Definition (Hardness assumptions for circuits of type X)

E is hard for exponential size circuits of type X if $\exists \beta > 0$, and $\exists L \in \mathrm{E} = \mathrm{DTIME}(2^{O(n)})$ s.t. $\forall n$, 1_L is $2^{\beta n}$-hard for circuits of type X on inputs of length n.

PRGs against other classes of circuits

Definition (Hardness assumptions for circuits of type X)

E is hard for exponential size circuits of type X if $\exists \beta > 0$, and $\exists L \in \mathrm{E} = \mathrm{DTIME}(2^{O(n)})$ s.t. $\forall n$, 1_L is $2^{\beta n}$-hard for circuits of type X on inputs of length n.

Theorem (KvM99 - "relativized IW97")

E is hard for exponential size circuits of type X $\Rightarrow$ there is an explicit PRG

$$G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$$

for size m circuits of type X.

PRGs against other classes of circuits

Definition (Hardness assumptions for circuits of type X)

E is hard for exponential size circuits of type X if $\exists \beta > 0$, and $\exists L \in \mathrm{E} = \mathrm{DTIME}(2^{O(n)})$ s.t. $\forall n$, 1_L is $2^{\beta n}$-hard for circuits of type X on inputs of length n.

Theorem (KvM99 - "relativized IW97")

E is hard for exponential size circuits of type X $\Rightarrow$ there is an explicit PRG

$$G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$$

for size m circuits of type X.

Consequences:

PRGs against other classes of circuits

Definition (Hardness assumptions for circuits of type X)

E is hard for exponential size circuits of type X if $\exists \beta > 0$, and $\exists L \in \mathrm{E} = \mathrm{DTIME}(2^{O(n)})$ s.t. $\forall n$, 1_L is $2^{\beta n}$-hard for circuits of type X on inputs of length n.

Theorem (KvM99 - "relativized IW97")

E is hard for exponential size circuits of type X $\Rightarrow$ there is an explicit PRG

$$G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$$

for size m circuits of type X.

Consequences:

- E hard for circuits with SAT oracle $\Rightarrow$ PRG for circuits with SAT-oracle.

PRGs against other classes of circuits

Definition (Hardness assumptions for circuits of type X)

E is hard for exponential size circuits of type X if $\exists \beta > 0$, and $\exists L \in \mathrm{E} = \mathrm{DTIME}(2^{O(n)})$ s.t. $\forall n$, 1_L is $2^{\beta n}$-hard for circuits of type X on inputs of length n.

Theorem (KvM99 - "relativized IW97")

E is hard for exponential size circuits of type X $\Rightarrow$ there is an explicit PRG

$$G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$$

for size m circuits of type X.

Consequences:

- E hard for circuits with SAT oracle $\Rightarrow$ PRG for circuits with SAT-oracle.
- E hard for nondeterministic circuits $\Rightarrow$ PRG for nondeterministic circuits. (Not a direct consequence, and requires more work [SU01,SU05]).

PRGs against other classes of circuits

Definition (Hardness assumptions for circuits of type X)

E is hard for exponential size circuits of type X if $\exists \beta > 0$, and $\exists L \in \mathrm{E} = \mathrm{DTIME}(2^{O(n)})$ s.t. $\forall n$, 1_L is $2^{\beta n}$-hard for circuits of type X on inputs of length n.

Theorem (KvM99 - "relativized IW97")

E is hard for exponential size circuits of type X $\Rightarrow$ there is an explicit PRG

$$G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$$

for size m circuits of type X.

Consequences:

- E hard for circuits with SAT oracle $\Rightarrow$ PRG for circuits with SAT-oracle.
- E hard for nondeterministic circuits $\Rightarrow$ PRG for nondeterministic circuits. (Not a direct consequence, and requires more work [SU01,SU05]).

Theorem (KvM99,MV99,SU01 Derandomization of AM)

E is hard for exponential size nondeterministic circuits $\Rightarrow$ AM = NP.

Reflection: PRGs for nondeterministic circuits

Theorem (KvM99,SU01,SU05 PRGs for nondeterministic circuits)

E *is hard for exponential size nondeterministic circuits* $\Rightarrow$ *there is an explicit PRG*

$$G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$$

for size m *nondeterministic circuits.*

Reflection: PRGs for nondeterministic circuits

Theorem (KvM99,SU01,SU05 PRGs for nondeterministic circuits)

E *is hard for exponential size nondeterministic circuits* $\Rightarrow$ *there is an explicit PRG*

$$G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$$

for size m *nondeterministic circuits.*

Crypto style PRGs (distinguisher can run PRG) do not exist against nondeterministic adversaries.
(Adversary can guess seed, and break PRG).

Reflection: PRGs for nondeterministic circuits

Theorem (KvM99,SU01,SU05 PRGs for nondeterministic circuits)

E *is hard for exponential size nondeterministic circuits* $\Rightarrow$ *there is an explicit PRG*

$$G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$$

for size m *nondeterministic circuits.*

Crypto style PRGs (distinguisher can run PRG) do not exist against nondeterministic adversaries.
(Adversary can guess seed, and break PRG).

Nisan-Wigderson style PRGs (distinguisher does not have resources to run PRG) exist against nondeterministic circuits.

Reflection: PRGs for nondeterministic circuits

Theorem (KvM99,SU01,SU05 PRGs for nondeterministic circuits)

E *is hard for exponential size nondeterministic circuits* $\Rightarrow$ *there is an explicit PRG*

$$G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$$

for size m *nondeterministic circuits.*

Crypto style PRGs (distinguisher can run PRG) do not exist against nondeterministic adversaries.
(Adversary can guess seed, and break PRG).

Nisan-Wigderson style PRGs (distinguisher does not have resources to run PRG) exist against nondeterministic circuits.

Hardness assumptions against nondeterministic circuits have many applications: [KvM99,AK99,MV99,SU01,BOV02,GW02,GST03,SU05,SU07,Dru13,AASY15,BV17, AIKS16,HNY17,DMOZ20,BDL22,CT22,BDGM23,BSS24,SS24,BSS25,Sha25].

Reflection: PRGs for nondeterministic circuits

Theorem (KvM99,SU01,SU05 PRGs for nondeterministic circuits)

E *is hard for exponential size nondeterministic circuits* $\Rightarrow$ *there is an explicit PRG*

$$G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$$

for size m *nondeterministic circuits.*

Crypto style PRGs (distinguisher can run PRG) do not exist against nondeterministic adversaries.
(Adversary can guess seed, and break PRG).

Nisan-Wigderson style PRGs (distinguisher does not have resources to run PRG) exist against nondeterministic circuits.

Hardness assumptions against nondeterministic circuits have many applications: [KvM99,AK99,MV99,SU01,BOV02,GW02,GST03,SU05,SU07,Dru13,AASY15,BV17, AIKS16,HNY17,DMOZ20,BDL22,CT22,BDGM23,BSS24,SS24,BSS25,Sha25].

More broadly: "explicit constructions" from hardness assumptions [KvM99]:

Reflection: PRGs for nondeterministic circuits

Theorem (KvM99,SU01,SU05 PRGs for nondeterministic circuits)

E *is hard for exponential size nondeterministic circuits* $\Rightarrow$ *there is an explicit PRG*

$$G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$$

for size m *nondeterministic circuits.*

Crypto style PRGs (distinguisher can run PRG) do not exist against nondeterministic adversaries.
(Adversary can guess seed, and break PRG).

Nisan-Wigderson style PRGs (distinguisher does not have resources to run PRG) exist against nondeterministic circuits.

Hardness assumptions against nondeterministic circuits have many applications: [KvM99,AK99,MV99,SU01,BOV02,GW02,GST03,SU05,SU07,Dru13,AASY15,BV17, AIKS16,HNY17,DMOZ20,BDL22,CT22,BDGM23,BSS24,SS24,BSS25,Sha25].

More broadly: "explicit constructions" from hardness assumptions [KvM99]: Extractors for samplable distributions [TV00,BDGM23,BSS25,Sha25,OS26].

Reflection: PRGs for nondeterministic circuits

Theorem (KvM99,SU01,SU05 PRGs for nondeterministic circuits)

E *is hard for exponential size nondeterministic circuits* $\Rightarrow$ *there is an explicit PRG*

$$G : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$$

for size m *nondeterministic circuits.*

Crypto style PRGs (distinguisher can run PRG) do not exist against nondeterministic adversaries.
(Adversary can guess seed, and break PRG).

Nisan-Wigderson style PRGs (distinguisher does not have resources to run PRG) exist against nondeterministic circuits.

Hardness assumptions against nondeterministic circuits have many applications: [KvM99,AK99,MV99,SU01,BOV02,GW02,GST03,SU05,SU07,Dru13,AASY15,BV17, AIKS16,HNY17,DMOZ20,BDL22,CT22,BDGM23,BSS24,SS24,BSS25,Sha25].

More broadly: "explicit constructions" from hardness assumptions [KvM99]: Extractors for samplable distributions [TV00,BDGM23,BSS25,Sha25,OS26].

Codes for computationally bounded channels [GS10,SS16,BDL22,SS24,BSS25].

Perspective: limitations of black-box constructions

Definition (Fully black-box PRG)

oracle TM $G^{(\cdot)}$ s.t. $\forall f : \{0,1\}^{\ell} \to \{0,1\}$, setting $m = 2^{\Omega(\ell)}$:
$G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ is computable in E^f.
s.t. $\forall D : \{0,1\}^m \to \{0,1\}$ (not necessarily a small circuit), if

$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon,$$

then $\exists$size $2^{o(\ell)}$ oracle-circuit $C^{(\cdot)}$ s.t. $\forall x \in \{0,1\}^{\ell}$, $C^D(x) = f(x)$.

Perspective: limitations of black-box constructions

Definition (Fully black-box PRG)

oracle TM $G^{(\cdot)}$ s.t. $\forall f : \{0,1\}^{\ell} \to \{0,1\}$, setting $m = 2^{\Omega(\ell)}$:
$G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ is computable in E^f.
s.t. $\forall D : \{0,1\}^m \to \{0,1\}$ (not necessarily a small circuit), if

$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon,$$

then $\exists$size $2^{o(\ell)}$ oracle-circuit $C^{(\cdot)}$ s.t. $\forall x \in \{0,1\}^{\ell}$, $C^D(x) = f(x)$.

The constructions we saw (and most in the literature) are fully black-box.

Perspective: limitations of black-box constructions

Definition (Fully black-box PRG)

oracle TM $G^{(\cdot)}$ s.t. $\forall f : \{0,1\}^{\ell} \to \{0,1\}$, setting $m = 2^{\Omega(\ell)}$:
$G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ is computable in E^f.
s.t. $\forall D : \{0,1\}^m \to \{0,1\}$ (not necessarily a small circuit), if

$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon,$$

then $\exists$size $2^{o(\ell)}$ oracle-circuit $C^{(\cdot)}$ s.t. $\forall x \in \{0,1\}^{\ell}$, $C^D(x) = f(x)$.

The constructions we saw (and most in the literature) are fully black-box.

Such constructions have the following limitations:

Perspective: limitations of black-box constructions

Definition (Fully black-box PRG)

oracle TM $G^{(\cdot)}$ s.t. $\forall f : \{0,1\}^\ell \to \{0,1\}$, setting $m = 2^{\Omega(\ell)}$:
$G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ is computable in E^f.
s.t. $\forall D : \{0,1\}^m \to \{0,1\}$ (not necessarily a small circuit), if

$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon,$$

then $\exists$size $2^{o(\ell)}$ oracle-circuit $C^{(\cdot)}$ s.t. $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.

The constructions we saw (and most in the literature) are fully black-box.

Such constructions have the following limitations:

- Cannot start from hardness assumptions against uniform algorithms.
  (such a construction $\Rightarrow$ extractor with "output length" $>$ "entropy").

Perspective: limitations of black-box constructions

Definition (Fully black-box PRG)

oracle TM $G^{(\cdot)}$ s.t. $\forall f : \{0,1\}^\ell \to \{0,1\}$, setting $m = 2^{\Omega(\ell)}$:
$G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ is computable in E^f.
s.t. $\forall D : \{0,1\}^m \to \{0,1\}$ (not necessarily a small circuit), if

$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon,$$

then $\exists$size $2^{o(\ell)}$ oracle-circuit $C^{(\cdot)}$ s.t. $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.

The constructions we saw (and most in the literature) are fully black-box.

Such constructions have the following limitations:

- Cannot start from hardness assumptions against uniform algorithms.
  (such a construction $\Rightarrow$ extractor with "output length" $>$ "entropy").
- Must lose in circuit size when converting hardness $\Rightarrow$ PRG
  [SV08,AASY15,GSV18,SV22].

Perspective: limitations of black-box constructions

Definition (Fully black-box PRG)

oracle TM $G^{(\cdot)}$ s.t. $\forall f : \{0,1\}^\ell \to \{0,1\}$, setting $m = 2^{\Omega(\ell)}$:
$G^f : \{0,1\}^{r=O(\log m)} \to \{0,1\}^m$ is computable in E^f.
s.t. $\forall D : \{0,1\}^m \to \{0,1\}$ (not necessarily a small circuit), if

$$\left|\Pr[D(G^f(U_r)) = 1] - \Pr[D(U_m) = 1]\right| > \epsilon,$$

then $\exists$size $2^{o(\ell)}$ oracle-circuit $C^{(\cdot)}$ s.t. $\forall x \in \{0,1\}^\ell$, $C^D(x) = f(x)$.

The constructions we saw (and most in the literature) are fully black-box.

Such constructions have the following limitations:

- Cannot start from hardness assumptions against uniform algorithms.
  (such a construction $\Rightarrow$ extractor with "output length" $>$ "entropy").
- Must lose in circuit size when converting hardness $\Rightarrow$ PRG [SV08,AASY15,GSV18,SV22].

Do we have constructions that aren't fully black box?

Perspective: beyond black-box techniques I

Using the *efficiency* of f in the reduction proving pseudorandomness.

Perspective: beyond black-box techniques I

Using the *efficiency* of f in the reduction proving pseudorandomness.

Theorem (IW98 uniform hardness vs. randomness tradeoffs)

$\mathrm{EXP} \neq \mathrm{BPP} \Rightarrow \mathrm{BPP} \subseteq i.o. - \mathit{Heuristic} - \mathrm{SUBEXP}$.

Perspective: beyond black-box techniques I

Using the *efficiency* of f in the reduction proving pseudorandomness.

Theorem (IW98 uniform hardness vs. randomness tradeoffs)

$\mathrm{EXP} \neq \mathrm{BPP} \Rightarrow \mathrm{BPP} \subseteq i.o. - \mathit{Heuristic} - \mathrm{SUBEXP}.$

Extensions and generalizations [TV02,GST03,SU07,CRTY20,...]

Perspective: beyond black-box techniques I

Using the *efficiency* of f in the reduction proving pseudorandomness.

Theorem (IW98 uniform hardness vs. randomness tradeoffs)

$\mathrm{EXP} \neq \mathrm{BPP} \Rightarrow \mathrm{BPP} \subseteq i.o. - \mathit{Heuristic} - \mathrm{SUBEXP}.$

Extensions and generalizations [TV02,GST03,SU07,CRTY20,...]

Idea: We often transform a hard function into a low-degree polynomial.

.

Perspective: beyond black-box techniques I

Using the *efficiency* of f in the reduction proving pseudorandomness.

Theorem (IW98 uniform hardness vs. randomness tradeoffs)

$\mathrm{EXP} \neq \mathrm{BPP} \Rightarrow \mathrm{BPP} \subseteq i.o. - \mathit{Heuristic} - \mathrm{SUBEXP}.$

Extensions and generalizations [TV02,GST03,SU07,CRTY20,...]

Idea: We often transform a hard function into a low-degree polynomial.
We've seen the [STV99] low-degree extension:

.

Perspective: beyond black-box techniques I

Using the *efficiency* of f in the reduction proving pseudorandomness.

Theorem (IW98 uniform hardness vs. randomness tradeoffs)

$\mathrm{EXP} \neq \mathrm{BPP} \Rightarrow \mathrm{BPP} \subseteq i.o. - \mathit{Heuristic} - \mathrm{SUBEXP}.$

Extensions and generalizations [TV02,GST03,SU07,CRTY20,...]

Idea: We often transform a hard function into a low-degree polynomial.
We've seen the [STV99] low-degree extension:
(Reed-Muller encoding of the truth table): $f \to \hat{f}$.

.

Perspective: beyond black-box techniques I

Using the *efficiency* of f in the reduction proving pseudorandomness.

Theorem (IW98 uniform hardness vs. randomness tradeoffs)

$\mathrm{EXP} \neq \mathrm{BPP} \Rightarrow \mathrm{BPP} \subseteq i.o. - \mathit{Heuristic} - \mathrm{SUBEXP}.$

Extensions and generalizations [TV02,GST03,SU07,CRTY20,...]

Idea: We often transform a hard function into a low-degree polynomial.
We've seen the [STV99] low-degree extension:
(Reed-Muller encoding of the truth table): $f \to \hat{f}$.
Can we imitate the IP = PSPACE proof and use arithmetization?

.

Perspective: beyond black-box techniques I

Using the *efficiency* of f in the reduction proving pseudorandomness.

Theorem (IW98 uniform hardness vs. randomness tradeoffs)

$\mathrm{EXP} \neq \mathrm{BPP} \Rightarrow \mathrm{BPP} \subseteq i.o. - \mathit{Heuristic} - \mathrm{SUBEXP}.$

Extensions and generalizations [TV02,GST03,SU07,CRTY20,...]

Idea: We often transform a hard function into a low-degree polynomial.
We've seen the [STV99] low-degree extension:
(Reed-Muller encoding of the truth table): $f \to \hat{f}$.
Can we imitate the $\mathrm{IP} = \mathrm{PSPACE}$ proof and use arithmetization?
That way, $\hat{f}$ will "exploit" the easiness of f.

Perspective: beyond black-box techniques I

Using the *efficiency* of f in the reduction proving pseudorandomness.

Theorem (IW98 uniform hardness vs. randomness tradeoffs)

$\mathrm{EXP} \neq \mathrm{BPP} \Rightarrow \mathrm{BPP} \subseteq i.o. - \mathit{Heuristic} - \mathrm{SUBEXP}.$

Extensions and generalizations [TV02,GST03,SU07,CRTY20,...]

Idea: We often transform a hard function into a low-degree polynomial.
We've seen the [STV99] low-degree extension:
(Reed-Muller encoding of the truth table): $f \to \hat{f}$.
Can we imitate the $\mathrm{IP} = \mathrm{PSPACE}$ proof and use arithmetization?
That way, $\hat{f}$ will "exploit" the easiness of f.

Breakthrough result: [CT21]: Use machinery from interactive proofs [GKR08] to "arithmetize" f, in a form that is compatible with the [IW98] reconstruction
$\Rightarrow$ "uniform, instance-wise hardness vs. randomness tradeoffs".

Perspective: beyond black-box techniques I

Using the *efficiency* of f in the reduction proving pseudorandomness.

Theorem (IW98 uniform hardness vs. randomness tradeoffs)

$\mathrm{EXP} \neq \mathrm{BPP} \Rightarrow \mathrm{BPP} \subseteq i.o. - \mathit{Heuristic} - \mathrm{SUBEXP}.$

Extensions and generalizations [TV02,GST03,SU07,CRTY20,...]

Idea: We often transform a hard function into a low-degree polynomial.
We've seen the [STV99] low-degree extension:
(Reed-Muller encoding of the truth table): $f \to \hat{f}$.
Can we imitate the $\mathrm{IP} = \mathrm{PSPACE}$ proof and use arithmetization?
That way, $\hat{f}$ will "exploit" the easiness of f.

Breakthrough result: [CT21]: Use machinery from interactive proofs [GKR08] to "arithmetize" f, in a form that is compatible with the [IW98] reconstruction
$\Rightarrow$ "uniform, instance-wise hardness vs. randomness tradeoffs".

Extensions and generalizations [CRT22,SvM23,BCT25,...].

Perspective: beyond black-box techniques II

Using the *efficiency* of D in the reduction establishing pseudorandomness.

Perspective: beyond black-box techniques II

Using the *efficiency* of D in the reduction establishing pseudorandomness.
Proofs that use the efficiency of D in related contexts [GST05,Atserias06,Hirahara18].

Perspective: beyond black-box techniques II

Using the *efficiency* of D in the reduction establishing pseudorandomness.
Proofs that use the efficiency of D in related contexts [GST05,Atserias06,Hirahara18].

Idea: Maybe it's easier to use the easiness of D when D is weak.

Perspective: beyond black-box techniques II

Using the *efficiency* of D in the reduction establishing pseudorandomness.
Proofs that use the efficiency of D in related contexts [GST05,Atserias06,Hirahara18].

Idea: Maybe it's easier to use the easiness of D when D is weak.
Specifically, if D is a small space algorithm.

Perspective: beyond black-box techniques II

Using the *efficiency* of D in the reduction establishing pseudorandomness.
Proofs that use the efficiency of D in related contexts [GST05,Atserias06,Hirahara18].

Idea: Maybe it's easier to use the easiness of D when D is weak.
Specifically, if D is a small space algorithm.
We can try to make the distinguisher-to-predictor transformation deterministic and space-efficient.

Perspective: beyond black-box techniques II

Using the *efficiency* of D in the reduction establishing pseudorandomness.
Proofs that use the efficiency of D in related contexts [GST05,Atserias06,Hirahara18].

Idea: Maybe it's easier to use the easiness of D when D is weak.
Specifically, if D is a small space algorithm.
We can try to make the distinguisher-to-predictor transformation deterministic and space-efficient. Using these ideas:

Perspective: beyond black-box techniques II

Using the *efficiency* of D in the reduction establishing pseudorandomness.
Proofs that use the efficiency of D in related contexts [GST05,Atserias06,Hirahara18].

Idea: Maybe it's easier to use the easiness of D when D is weak.
Specifically, if D is a small space algorithm.
We can try to make the distinguisher-to-predictor transformation deterministic and space-efficient. Using these ideas:

Recently: Hardness vs. randomness tradeoffs against small space algorithms that exploit the efficiency of D. [DT23,PRZ23,DPT24,LPT24,DPTW25,...]

Perspective: beyond black-box techniques II

Using the *efficiency* of D in the reduction establishing pseudorandomness.
Proofs that use the efficiency of D in related contexts [GST05,Atserias06,Hirahara18].

Idea: Maybe it's easier to use the easiness of D when D is weak.
Specifically, if D is a small space algorithm.
We can try to make the distinguisher-to-predictor transformation deterministic and space-efficient. Using these ideas:

Recently: Hardness vs. randomness tradeoffs against small space algorithms that exploit the efficiency of D. [DT23,PRZ23,DPT24,LPT24,DPTW25,...]

Theorem (DPTW25 Hardness vs. randomness for space)

$\forall \epsilon > 0$, $\mathsf{NSPACE}(n) \not\subseteq i.o.TISP(2^{n^{2-\epsilon}}, n^{O(1)}) \Rightarrow \mathsf{BPSPACE}(n) \subseteq \mathsf{SPACE}(n^{1+\epsilon})$.

Perspective: beyond black-box techniques II

Using the *efficiency* of D in the reduction establishing pseudorandomness.
Proofs that use the efficiency of D in related contexts [GST05,Atserias06,Hirahara18].

Idea: Maybe it's easier to use the easiness of D when D is weak.
Specifically, if D is a small space algorithm.
We can try to make the distinguisher-to-predictor transformation deterministic and space-efficient. Using these ideas:

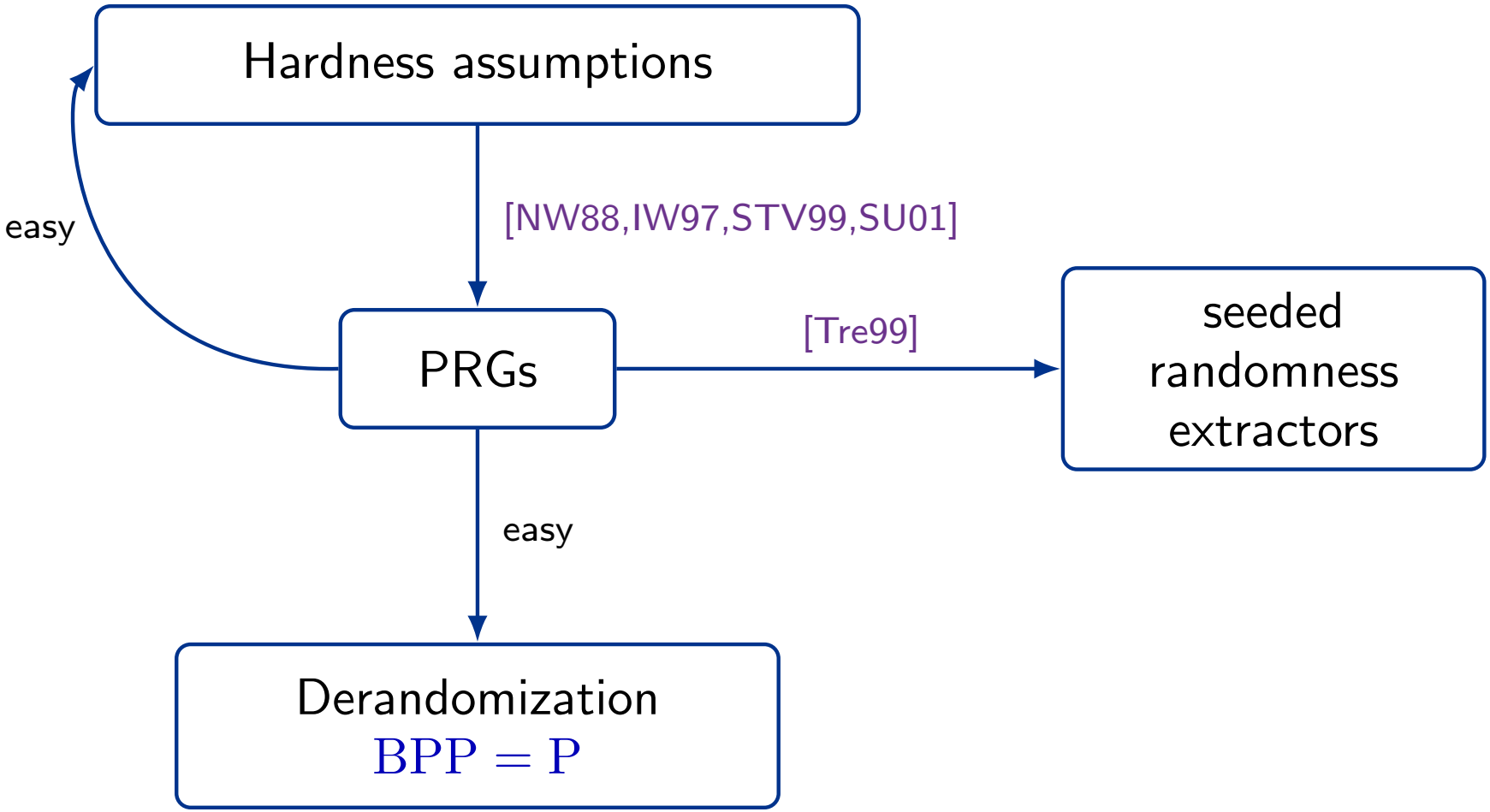
Recently: Hardness vs. randomness tradeoffs against small space algorithms that exploit the efficiency of D. [DT23,PRZ23,DPT24,LPT24,DPTW25,...]

Theorem (DPTW25 Hardness vs. randomness for space)

$\forall \epsilon > 0$, $\mathsf{NSPACE}(n) \not\subseteq i.o.TISP(2^{n^{2-\epsilon}}, n^{O(1)}) \Rightarrow \mathsf{BPSPACE}(n) \subseteq \mathsf{SPACE}(n^{1+\epsilon})$.

Other non-black-box uses of D: [LP22,LPT24].

Overall summary for the two talks



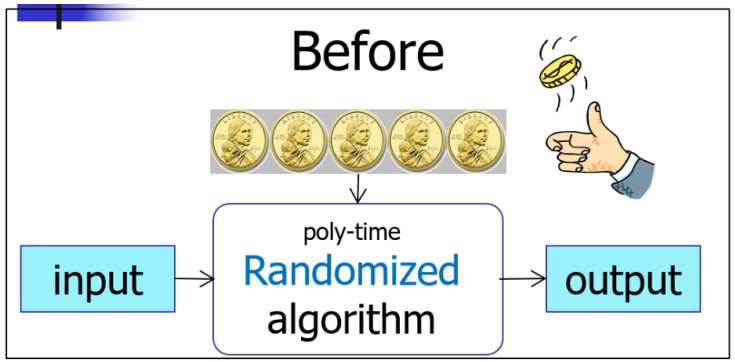
Then: extensions, generalizations, applications,
and some other research directions.

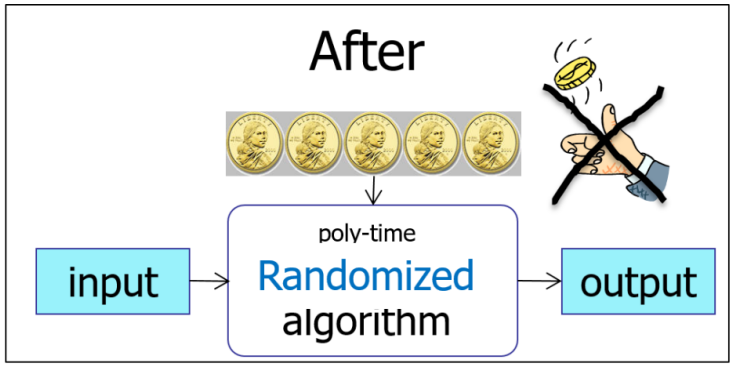
Current situation in hardness vs. randomness

Excerpt from a 2016 talk on hardness vs. randomness for Avi's birthday.

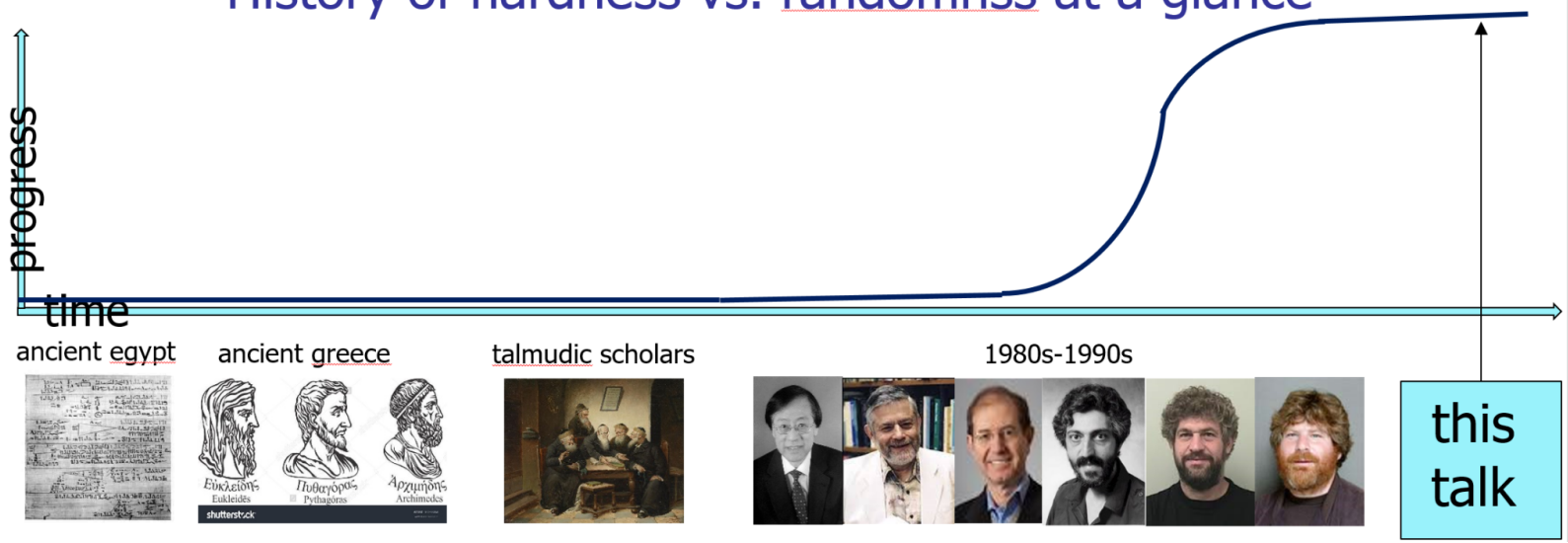
Current situation in hardness vs. randomness

Excerpt from a 2016 talk on hardness vs. randomness for Avi's birthday.



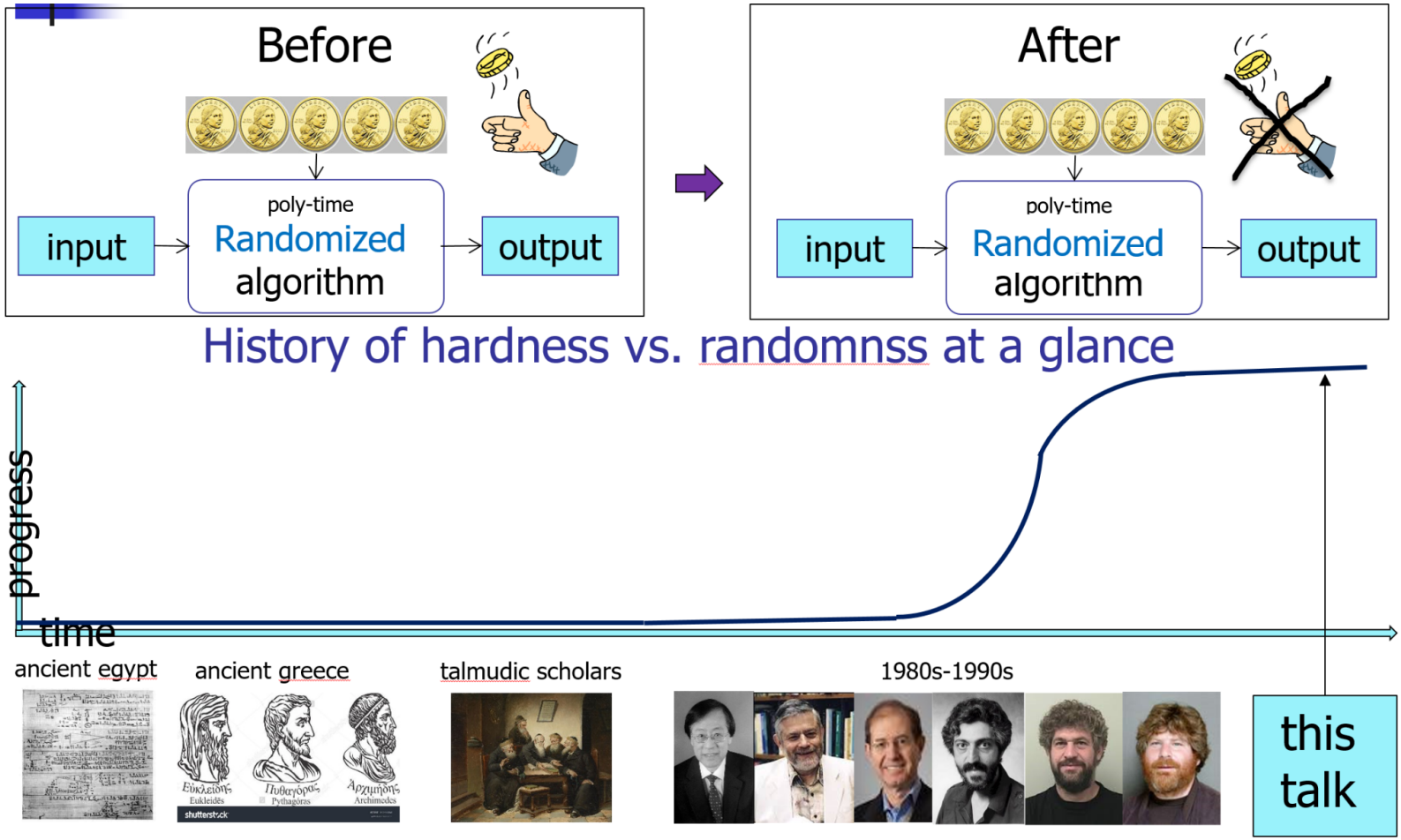


History of hardness vs. randomnss at a glance



Current situation in hardness vs. randomness

Excerpt from a 2016 talk on hardness vs. randomness for Avi's birthday.



Amazing progress in recent years!

We'll hear about these developments in future talks and seminars.