Extractors for Weak Random Sources and their Applications

David Zuckerman
University of Texas at Austin

Randomization extremely useful

- Algorithms (e.g. primality testing, factoring polynomials, approximation algorithms, min cut, load balancing)
- Monte Carlo simulations
- Network constructions
- Cryptography
- Distributed computing

But: Computers don't use truly random bits

Pseudo-random generators:

$$\underset{\text{random}}{01110100} \longrightarrow \boxed{\text{PRG}} \longrightarrow \underset{\text{``random'' enough?}}{11011100110110001001111}$$

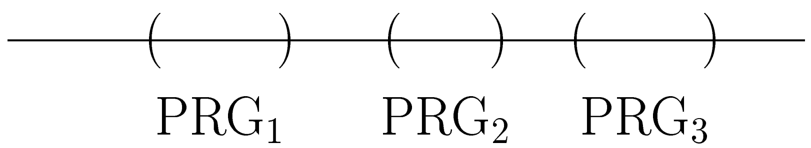
Pseudorandomness: Classical Approach
(e.g. [Knuth])

- Most popular PRG: Linear Congruential

$$x_{n+1} = (ax_n + b) \bmod m$$

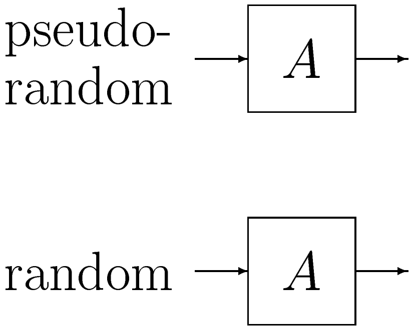
- Tests for pseudorandomness
  - long period
  - frequency tests, e.g. χ^2.
- Problems with PRG's
  - Tests ad hoc, e.g. $x_{n+1} = (x_n + 1) \bmod m$
  - More sophisticated uses of randomization $\Rightarrow$ more errors with PRG's reported, for example [Joh]:

95% confidence intervals



Pseudorandomness: Modern Approach

Goal:



$\approx$ same behavior
$\forall$ poly-time A

If e.g. FACTORING hard, then $\exists$ good PRG's [BM,Y,...]

Drawbacks:

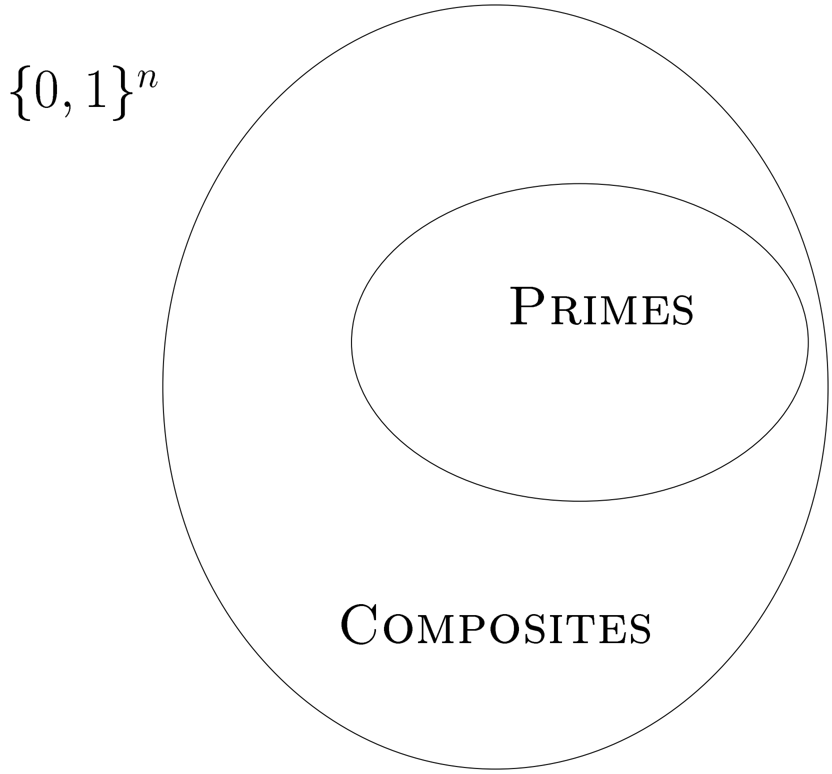
- Based on unproven assumptions
- Impractical

Extracting Randomness

Toy Example: Generate random n-bit prime.

Assume: Deterministic primality test.

Fact: $\Theta(1/n)$ fraction of n-bit numbers are prime.



Naive: Pick $x_1, x_2, \ldots \in_R \{0,1\}^n$ until some x_i prime.

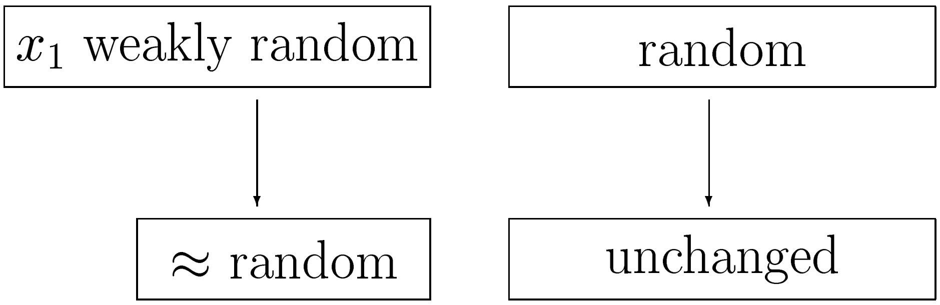
E[# of random bits] $= O(n^2)$.

Improvement idea: If x_1 composite, extract randomness left in x_1.

Easy if knew Composites.

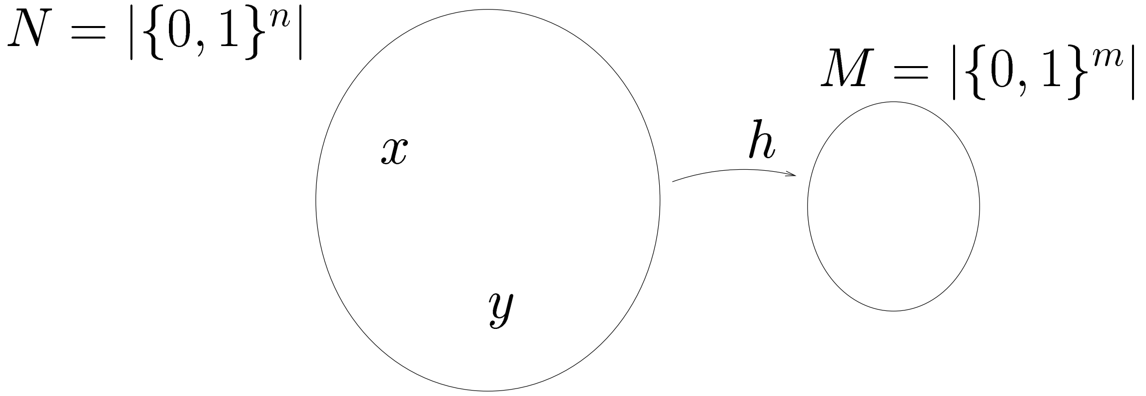
$$\begin{array}{rcl} 4 & \longrightarrow & 1 \\ 6 & \longrightarrow & 2 \\ 8 & \longrightarrow & 3 \\ 9 & \longrightarrow & 4 \\ 10 & \longrightarrow & 5 \\ & \vdots & \end{array}$$

Method: Add random catalyst.



$\approx$ uncorrelated

Definitions



$H = \{h\}$ universal family of hash functions:

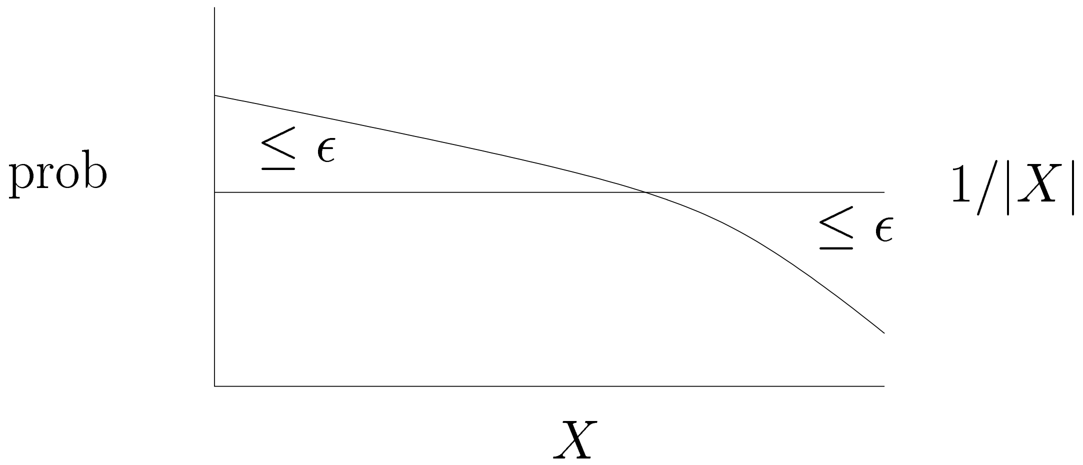
$$(\forall x \neq y) \Pr_{h \in_R H}[h(x) = h(y)] \leq 1/M$$

Example: $F =$ field on 2^n elements

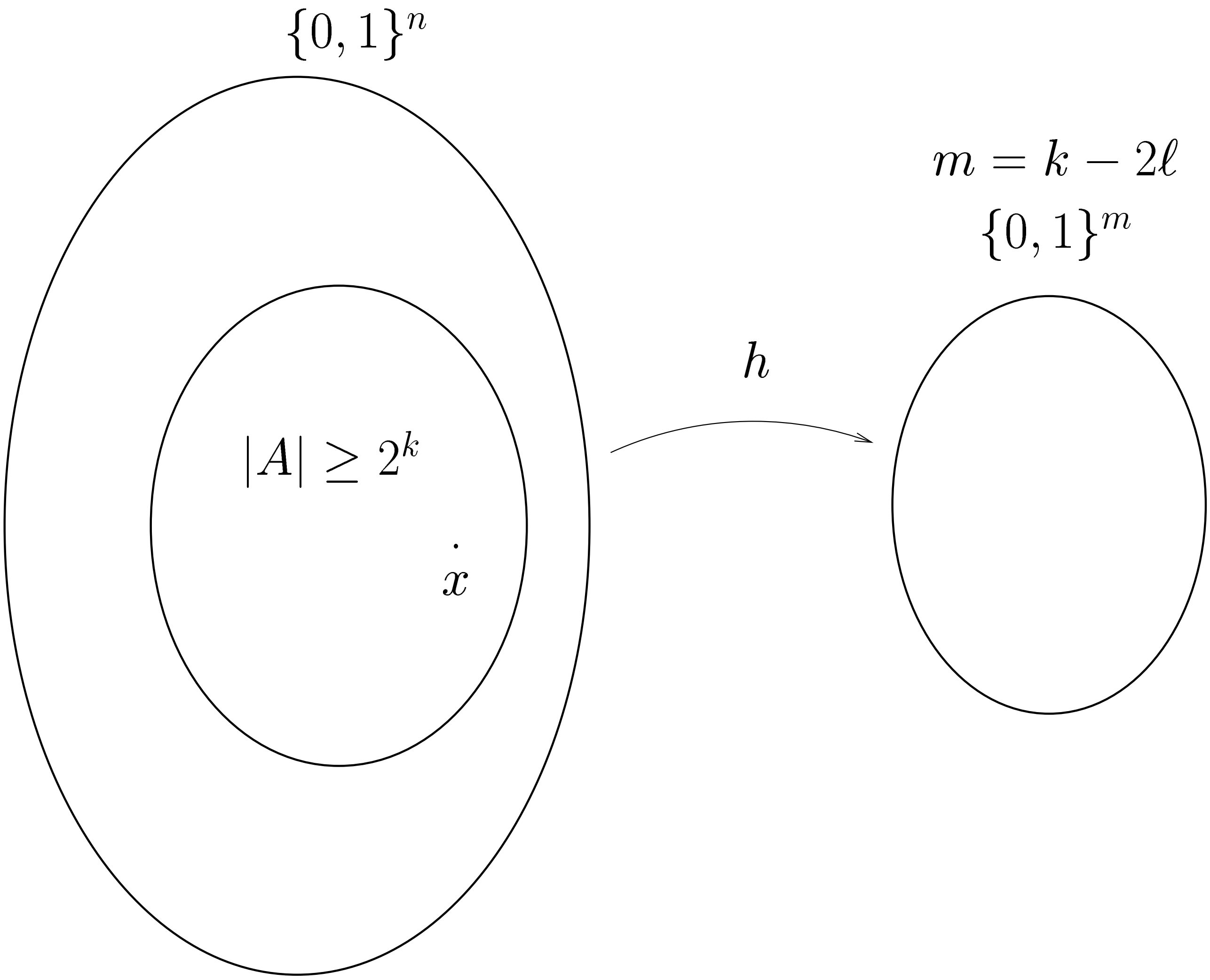
$$h_a(x) = \text{last } m \text{ bits of } (a \cdot x)$$
$$|H| = 2^n$$

Distribution D on set X is ϵ-*uniform* iff

$$(\forall S \subseteq X) \left| D(S) - \frac{|S|}{|X|} \right| \leq \epsilon$$

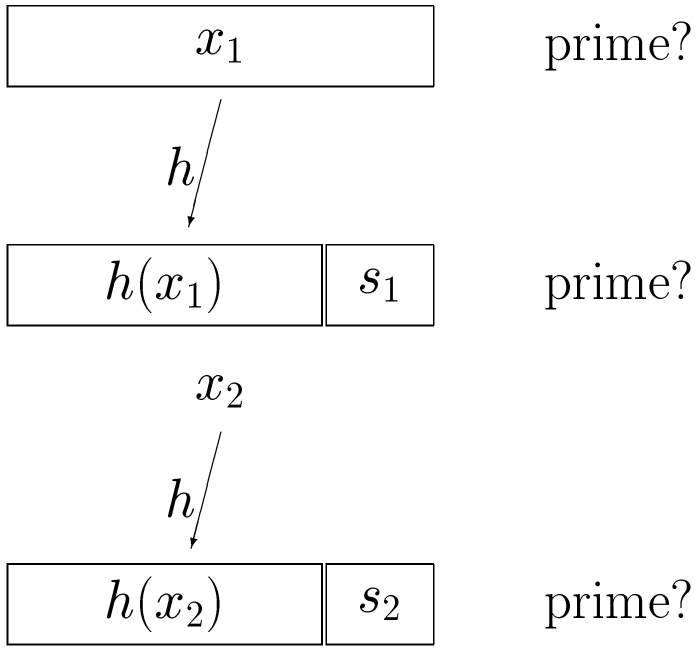


Leftover Hash Lemma [ILL]: For $x \in_R A$, $h \in_R H$ distribution of $h \circ h(x)$ is $2^{-\ell}$-uniform.



Recycling Randomness [IZ]

Pick $h \in_R H$.



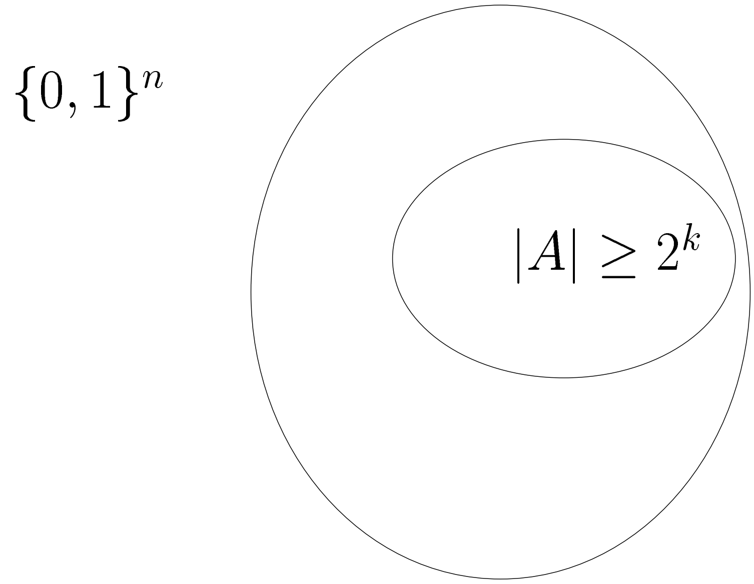
E[# of random bits used] $= O(n \log n)$.

Theorem [IZ]: This method works whenever information used about each random number is small.

Partial explanation of success of practical PRG's

- Linear congruential, shift register generators are hash functions.

Weak Random Sources



$X \in_R A$, $|A| \geq 2^k$
More general: $(\forall x) \Pr[X = x] \leq 2^{-k}$.

Examples:

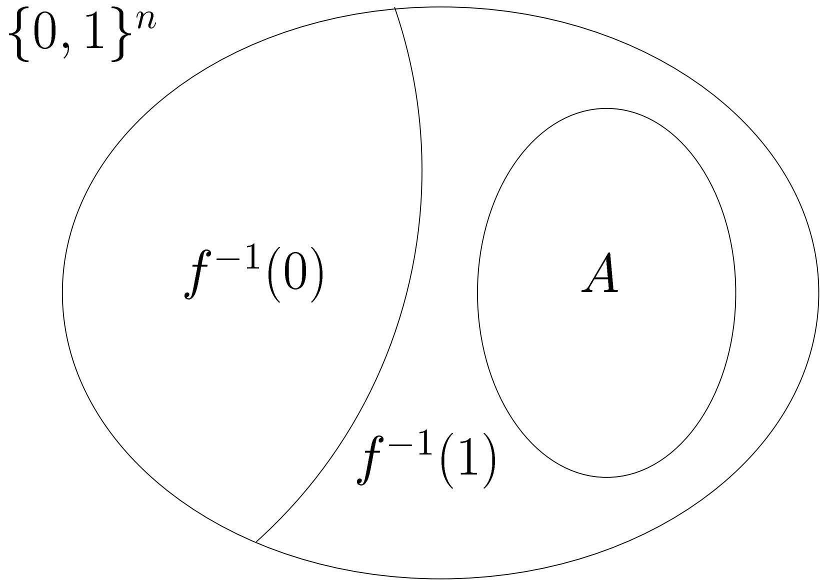
1. k random bits; others a function of these.
2. Each bit a little random.

No random catalyst available.

Goal: Algorithms that work for *any* weak source.

Can't extract 1 good bit

Suppose $f : \{0,1\}^n \longrightarrow \{0,1\}$ extracts.

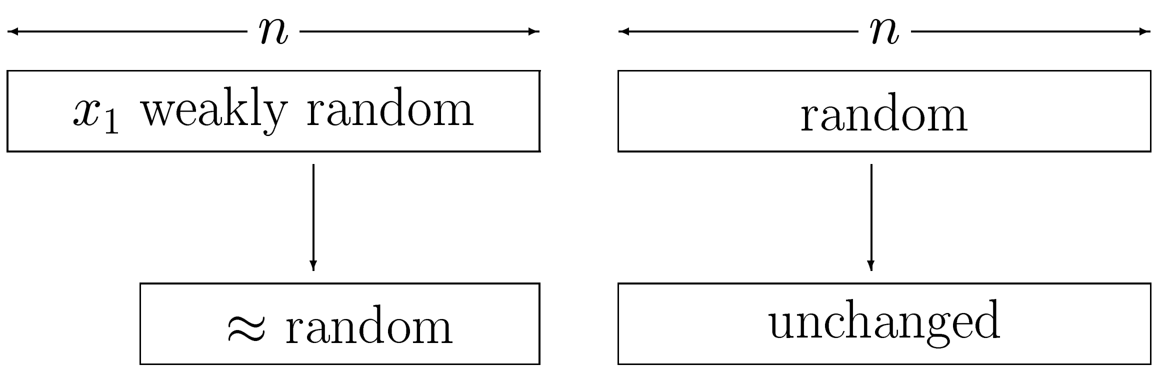


Can still simulate randomized algorithms using such a source! [Z]

Based on using small random catalyst.

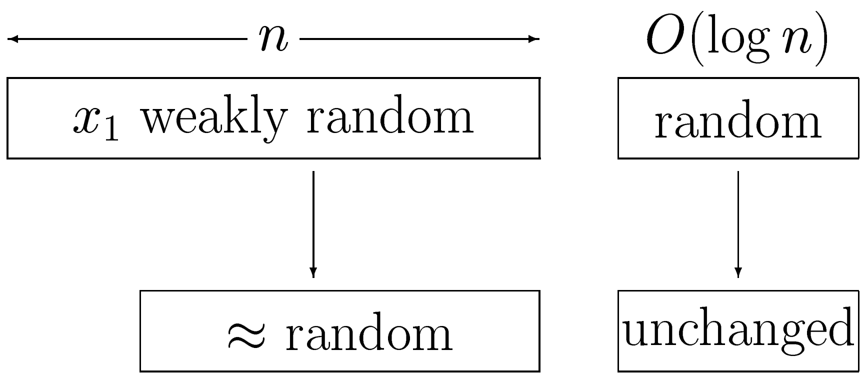
Extractor: Small random catalyst

Leftover Hash Lemma:



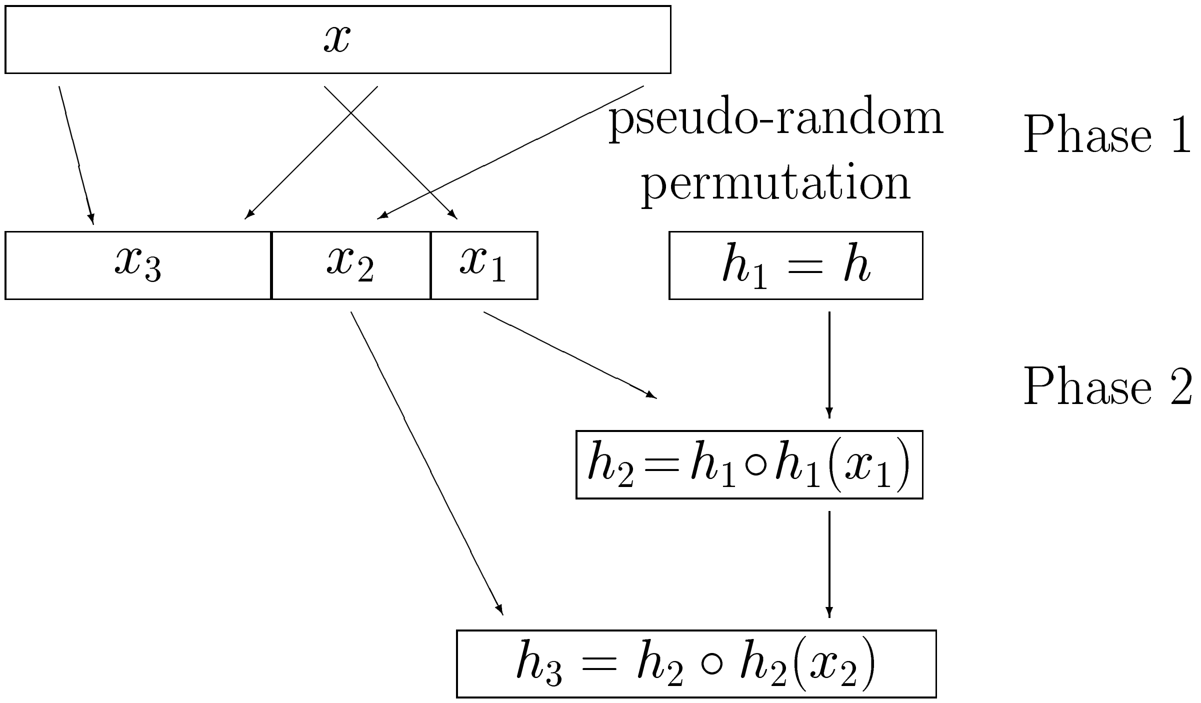
$\approx$ uncorrelated

Extractor:



Extractors [Z,NZ]

Construction of $d = O(\log^3 n)$ bit catalyst:



This paradigm [Z,NZ] [WZ,SZ,SSZ,Z,Ta-S]

Good for higher min-entropy.

New paradigm [Tre] [RRV,ISW]

Good for lower min-entropy.

Best Known Extractors

error $\epsilon = 1/n^{\Theta(1)}$, $\alpha > 0$ arbitrary

due to	min-entropy k	d random bits	output m
goal	any k	$O(\log n)$	k
[Z]	$\Omega(n)$	$O(\log n)$	$(1-\alpha)k$
[TUZ]	$n^{O(1/(\log\log n)^2)}$	$O(\log n)$	$(1-\alpha)k$
[RSW]	any k	$O(\log n)$	$k/\log n$
[RSW]	any k	$O(\log n(\log\log n)^2)$	$(1-\alpha)k$
+[WZ]	any k	$O(\log^2 n(\log\log n)^2)$	k

Best Known Dispersers

due to	min-ent. k	d random bits	output m
goal	any k	$O(\log n)$	k
[TUZ]	any k	$O(\log n)$	$k - O(\log n)$

Applications of Extractors

Random sampling using few random bits [Z]

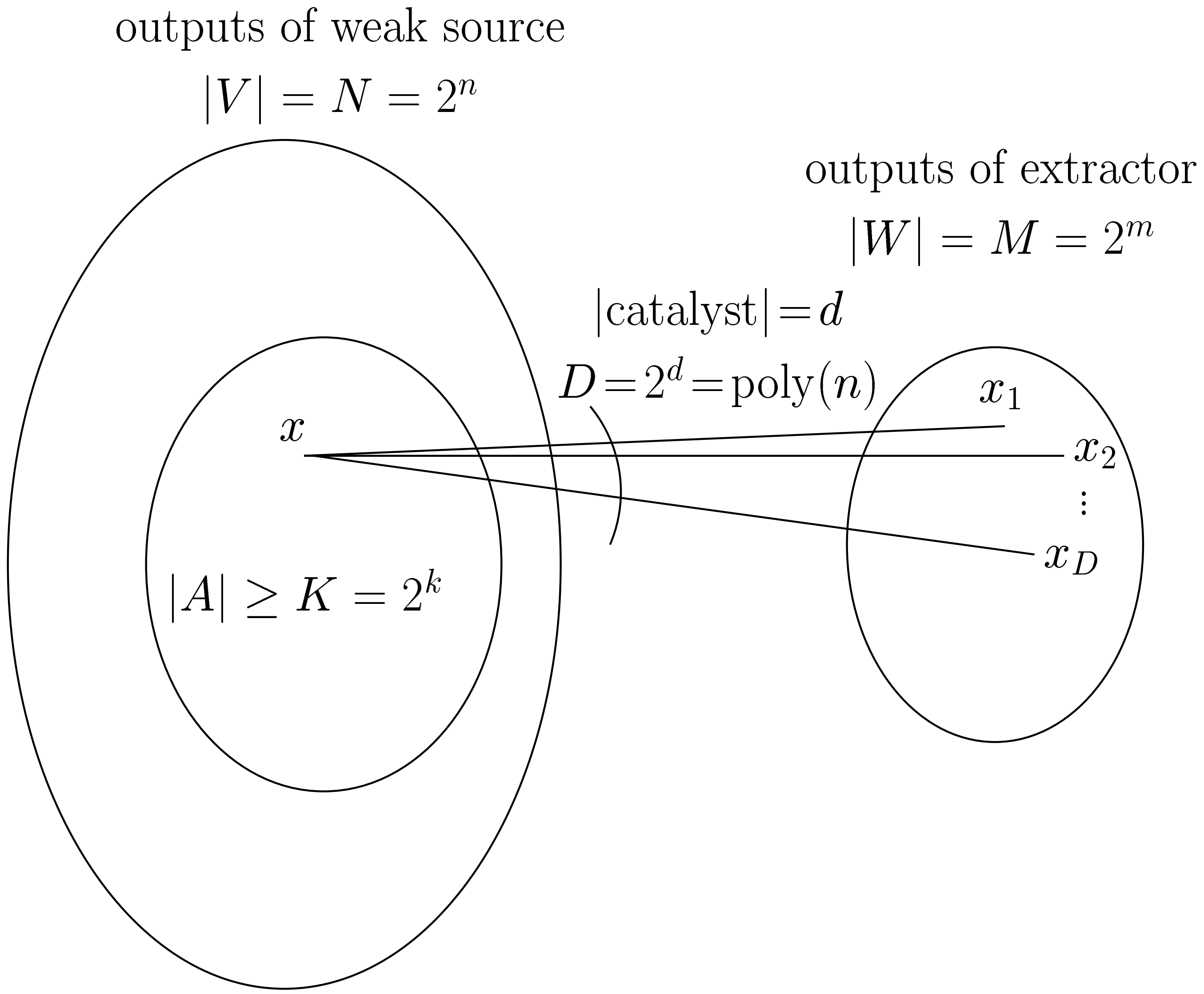
Simulating randomized algorithms using weak random sources [Z]

Expanders that beat the eigenvalue bound [WZ]

Error-correcting codes with good list decoding properties [TZ]

$\vdots$

Graph-Theoretic View of Extractor

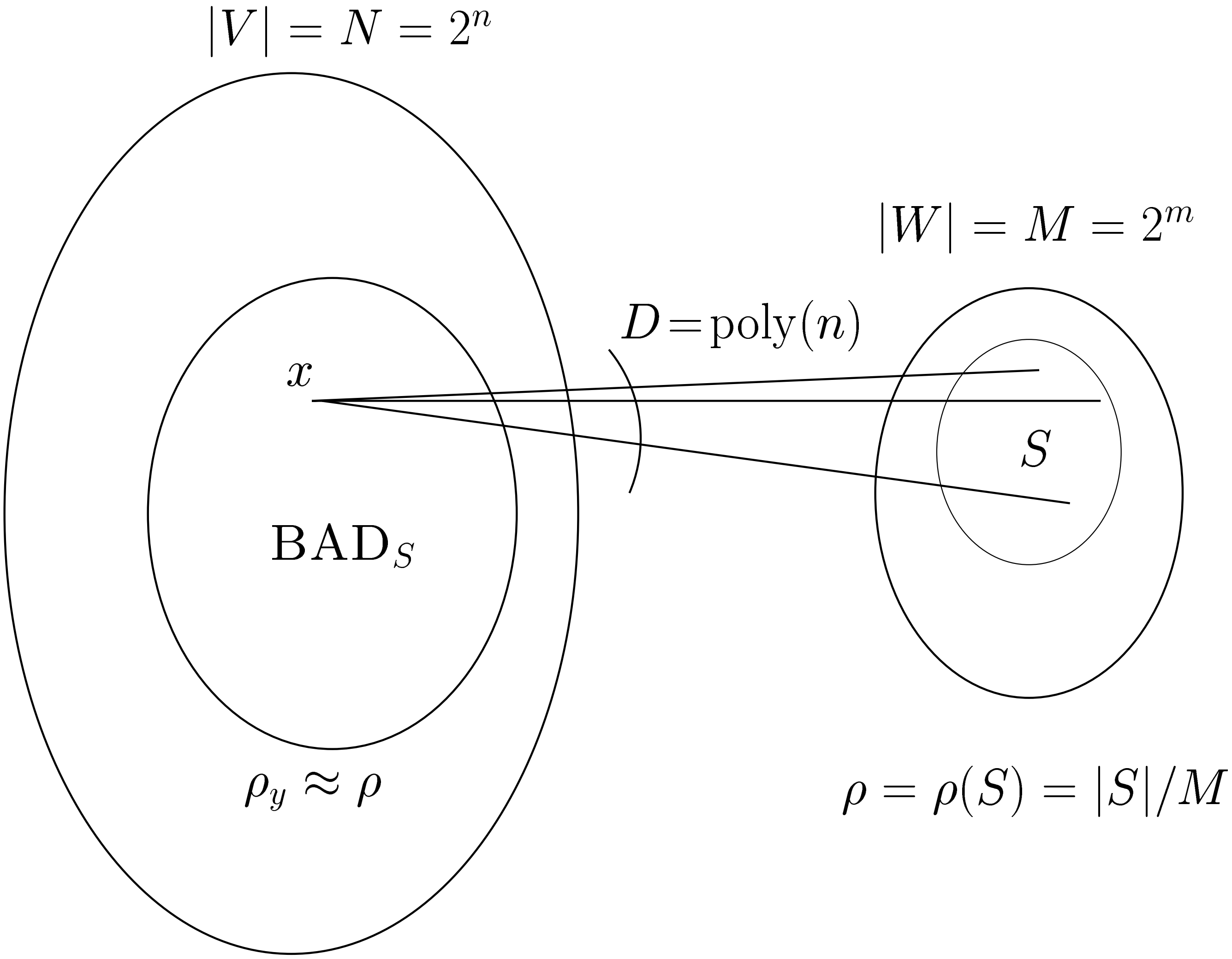


$$\Gamma(x) = \{h_1(x), \ldots, h_D(x)\} = \{x_1, \ldots, x_D\}$$

Extractor property:

$x \in_R A, x_i \in_R \Gamma(x) \Rightarrow x_i$ ϵ-uniform on W

Alternate View



$$\rho_x = \frac{|\Gamma(x) \cap S|}{D}$$

$\mathrm{BAD}_S = \{x \in V : |\rho_x - \rho| > \epsilon\}$ is $< 2K$ because

$\mathrm{BAD}_S^+ = \{x \in V : \rho_x > \rho + \epsilon\}$ is $< K$
$\mathrm{BAD}_S^- = \{x \in V : \rho_x < \rho - \epsilon\}$ is $< K$

Random Sampling

Given black box $\boxed{x \in S?}$ $\quad S \subseteq \{0,1\}^m$
Task: Estimate $\rho = |S|/2^m$.

$$\text{random bits} \longrightarrow \boxed{\text{sampler}} \xrightarrow[x_1, \ldots, x_D]{} \boxed{x_i \in S?} \longrightarrow \boxed{\text{est}} \longrightarrow \hat{\rho}$$

Oblivious estimator $\hat{\rho} = |\{x_1, \ldots, x_D\} \cap S|/D$.

Goal: estimate accurately whp: $\Pr[|\hat{\rho} - \rho| \geq \epsilon] \leq \gamma$

Think of $\epsilon = 1/\mathrm{poly}(m)$, $\gamma = 2^{-m/200}$.

At least m random bits necessary.

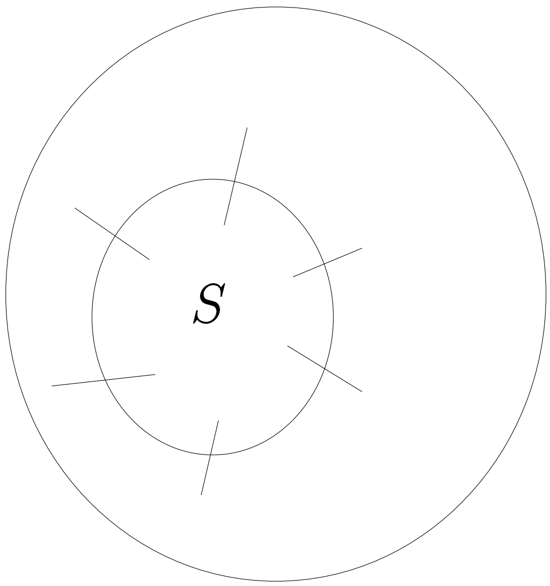
Can achieve using $1.01m$ random bits! [Z]

Pick $x \in_R \{0,1\}^n$, $\Gamma(x) = \{x_1, \ldots, x_D\}$, $\hat{\rho} = \rho_x$

$\Pr_x[|\rho_x - \rho| \geq \epsilon] \leq |\mathrm{BAD}_S| \cdot 2^{-n}$

Works when x from weak source $\qquad 2^{-k'}$
– can simulate randomized algorithms [Z]

Expander Graphs



- sorting
- network constructions
- complexity theory
- cryptography
- removing randomness

$|\Gamma(S) - S| \geq c|S|$, $|S| \leq N/2$

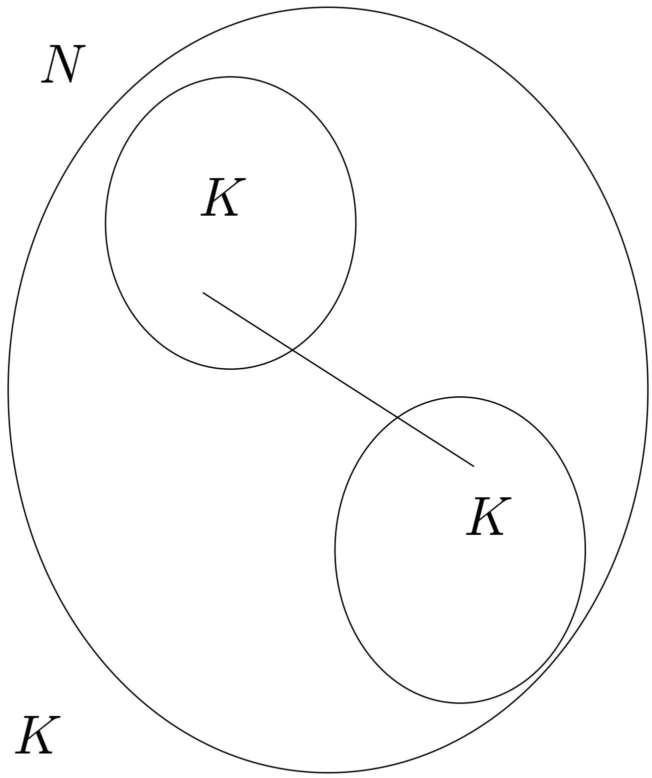
Goal: Expanders with constant degree

Random constant-degree graphs are expanders.

Goal: Deterministic constructions.

Second eigenvalue method successful [M,GG,...,LPS,M]

Highly Expanding Graphs



Think of $K = \sqrt{N}$.

$|A| \geq K \Rightarrow |\Gamma(A)| > N - K$

$d_{\text{avg}} > (N-K)/K \approx N/K$.

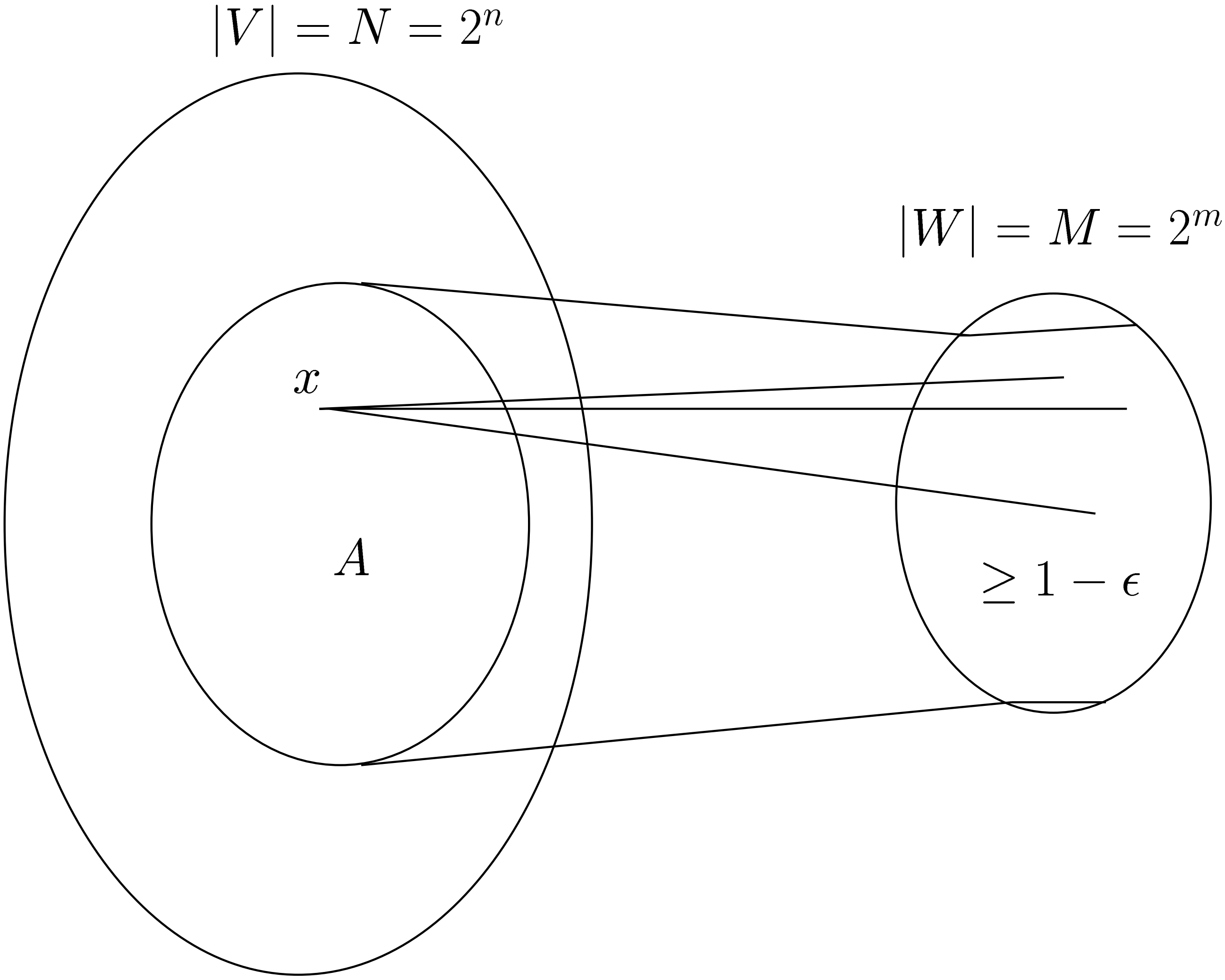
Second eigenvalue method: $d_{\text{avg}} \approx (N/K)^2$

[WZ] construction: $d_{\text{avg}} = N^{1+o(1)}/K$

Gives optimal constructions of

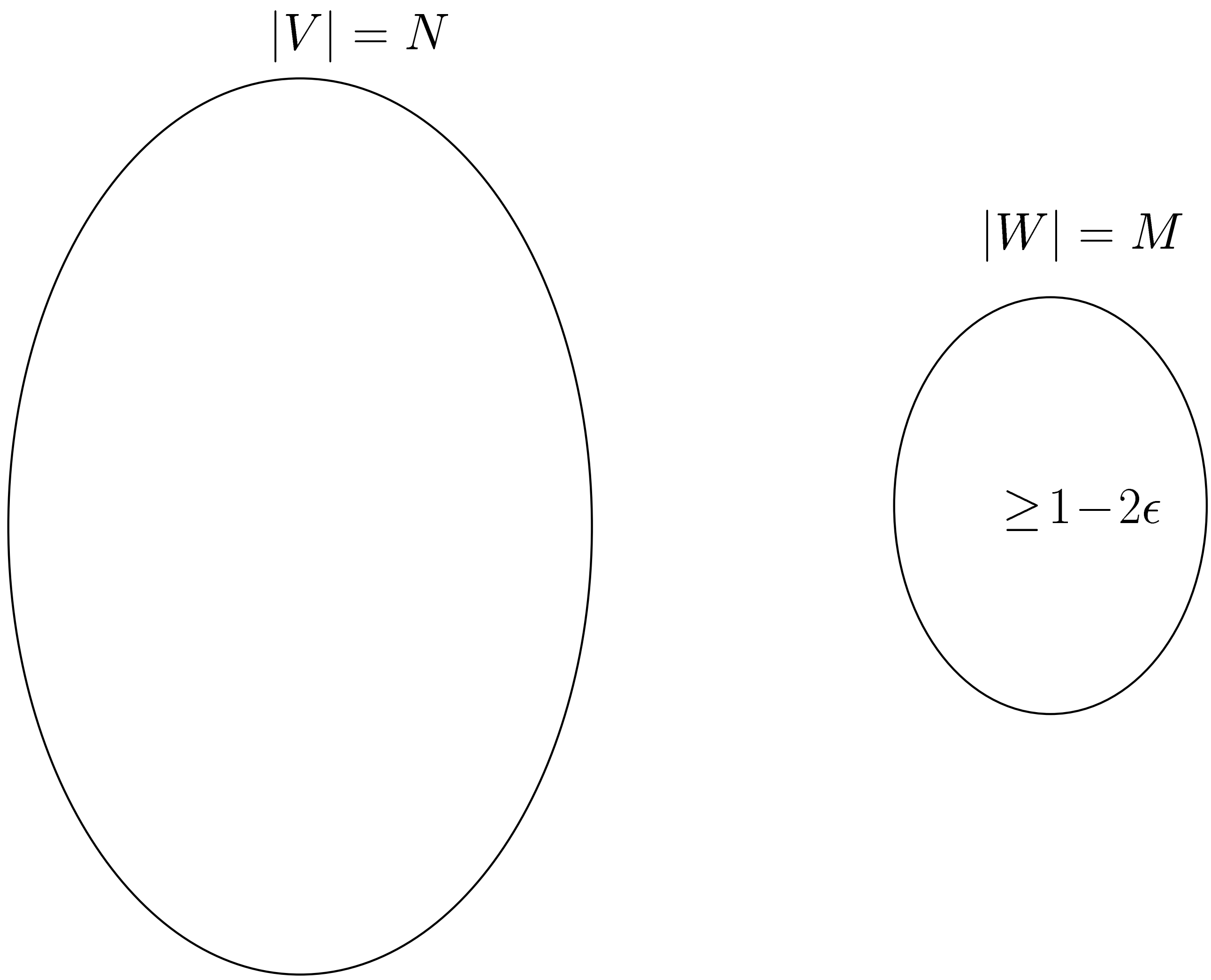
- superconcentrators [WZ]
- non-blocking networks [WZ,FFP]
- sorting and selecting in rounds [WZ,Pip]

Extractor $\Rightarrow$Disperser



$$|A| \geq K \Rightarrow |\Gamma(A)| \geq 1 - \epsilon$$

Disperser $\Rightarrow K$-expanding graphs [WZ]



$|A|, |B| \geq K \Rightarrow A$, B have common neighbor.

$G =$ paths of length 2 on vertex set V.

Some care needed to keep degree small.

Error-Correcting Codes

Lossy channel: some packets get lost
Use code $C : \Sigma^r \to \Sigma^s$

Example: $\Sigma = \{0, 1, \ldots, 9\}$
$C(627) = 666222777$
$\text{dist}(C) = \min_{x \neq y} \text{distance}(C(x), C(y)) = 3$
Can recover from $\text{dist}(C) - 1$ lost packets.

Noisy channel: packets can get changed
Can uniquely decode $< \text{dist}(C)/2$ errors

Reed-Solomon codes: $\Sigma = \{0, 1, \ldots, q-1\}$, q prime
Input to C: polynomials of degree $< r$ modulo q.
$C(p) = (p(1), p(2), \ldots, p(s))$
$\text{dist}(C) = s - r$

List Decoding

Very noisy channel: decoding with $t \geq \mathrm{dist}(C)/2$.
List Decoding: [Sud,...]: List messages within distance t
Not too many messages; can find them.

List decoding has numerous applications

- Cryptography: hard core predicates [GL]
- Average case hardness of permanent [Lip,CPS,GRS]

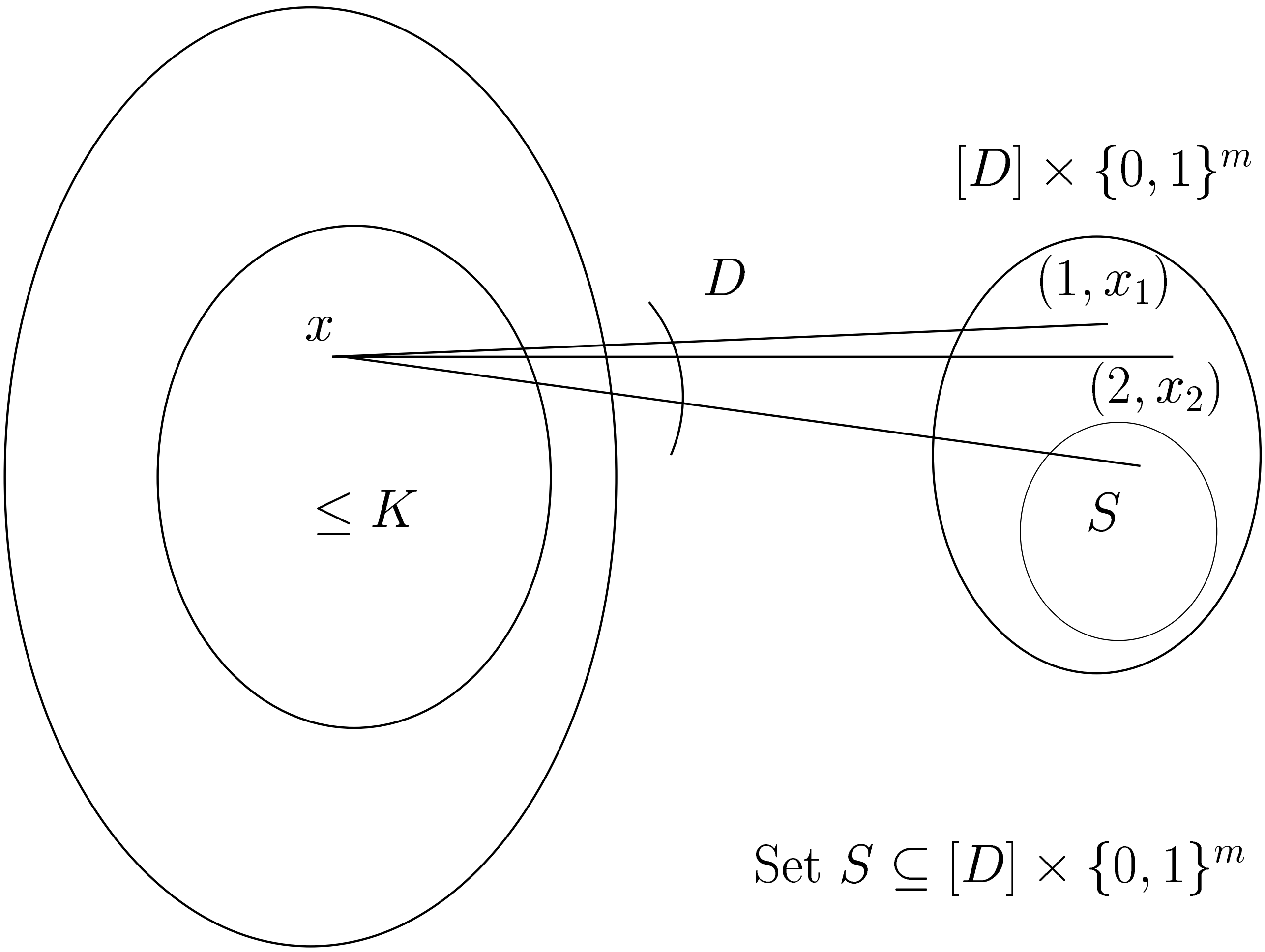
Extremely noisy channel:

$$\begin{array}{lcl} \text{Probably } w_1 \in \{2,3,7\} & \longleftrightarrow & (1,2),(1,3),(1,7) \\ \text{Probably } w_2 \in \{3,4,6,8\} & \longleftrightarrow & (2,3),(2,4),(2,6),(2,8) \\ \vdots & & \vdots \\ & & S \subseteq [s] \times \Sigma,\ [s] = \{1,2,\ldots,s\} \end{array}$$

$\exists$ few codewords $C(x) = (w_1, \ldots, w_s)$ agreeing with S in $(\rho(S) + \epsilon)s$ places?
Can we find them?

Reed-Solomon codes don't have good properties.

Extractor Codes [TZ]



$$\boxed{C(x) = (x_1, \ldots, x_D)}$$

Extractor property: $\leq K$ codewords have agreement

$$\geq (\rho(S) + \epsilon)D.$$

Better than RS codes for large sets S!

Algorithms to decode for certain extractors (such as [Tre]).

Other Applications of Extractors

Pseudo-random generator for space bounded computation [NZ]. Example: generate random n-bit prime using $O(n)$ random bits, without deterministic primality test.

Pseudo-random generators from hard functions [STV].

Another proof that BPP $\subseteq$ PH [GZ].

Unapproximability [Z,U].

Time-Space Tradeoffs [Sip].

Leader election [Z].

Further Directions

Pseudorandomness

Error Correcting Codes

Explicit Combinatorial Constructions

Fault Tolerance

Cryptography

Unapproximability