Codes in Theoretical Computer Science

David Zuckerman
University of Texas at Austin

Applications of Codes

Most natural: fault tolerance.

Code Property	Pseudorandomness	Other Areas
MDS	k-wise independence	secret sharing
minimum distance	almost k-wise indep. hard function $\to$ PRG	commun. complexity hashing
local decoding	average case hardness	probabilistically checkable proofs
list decoding	improved avg. hardness extractors	hardcore bits

New Codes

- expander codes (asymptotically good/efficiency)
- extractor codes (list decodability)
- tree codes (interactive communication)

Pseudorandomness

- Many randomized algorithms (e.g. primality testing, factoring polynomials, approximation algorithms, min cut, load balancing)
- Problem: Computers don't use truly random bits

Pseudo-random generators:

$$\underset{\text{random}}{01110100} \longrightarrow \boxed{\text{PRG}} \longrightarrow \underset{\text{“random” enough?}}{11011100110110001001111}$$

Goal:

$$\begin{array}{l} \text{pseudo-random} \longrightarrow \boxed{A} \longrightarrow \\ \\ \text{random} \longrightarrow \boxed{A} \longrightarrow \end{array} \quad \approx \text{same behavior } \forall \text{ poly-time } A$$

Explicit Combinatorial Constructions

Almost all $f : \{0,1\}^{O(\log n)} \to \{0,1\}^n$ "fool" all poly-time A.

Challenge: Find efficient one. Will show NP $\neq$ P.

Similar: Almost all functions require superpolynomial size circuits.

Challenge: Find one in NP. Will show NP $\neq$ P.

Similar: Random codes have nice properties.

Challenge: Fine efficient ones.

Short of ideal PRG's

$\exists$ hard functions $\Rightarrow$general PRG's.

Fool large classes of poly-time A, e.g. space-bounded A.

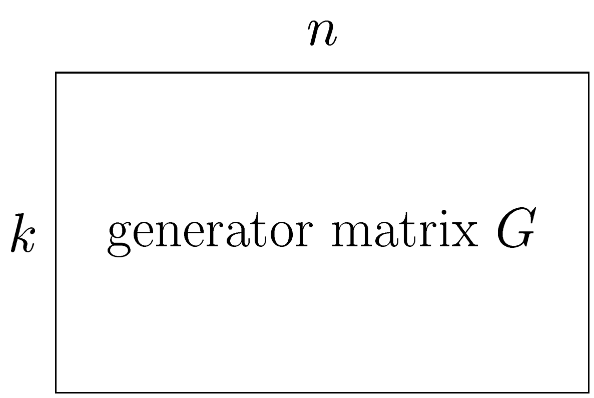
Various tools:

- (almost) k-wise independence
- expanders
- extractors

k-wise independence

$$X_1, X_2, \ldots, X_n$$

Every subset of k variables independent.



Every k columns of G linearly independent.

$\iff$ minimum distance$(C^{\perp}) > k$.

Take random codeword.

Example: $k = 2$: Hadamard codes [Lancaster].

Sample space size $= n + 1$.

Example: MDS codes: $[n, k, d = n - k + 1]_q$ code
Example: Reed-Solomon codes [Joffe].

Sample space size $= q^k = n^k$.

Cryptography: Secret Sharing [Shamir]

Goal: distribute secret $\in \mathbb{F}_q$ among $n-1$ players:

1. any k players can recover secret.
2. any $k-1$ players gain no information.

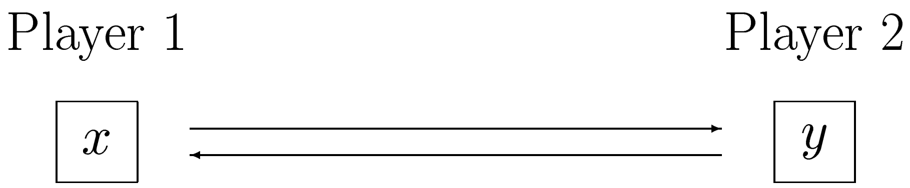
Use $[n,k]_q$ MDS code.

- Choose random codeword x with $x_n =$ secret.
- Player i gets x_i.

Proof outline:

1. k coordinates determine codeword.
2. All q possibilities for x_n equally likely.

Communication Complexity



Does $x = y$?

Minimize: # bits of communication

$x, y \in \{0,1\}^k$.

Deterministic: $= k + 1$.

Randomized: $\log k + O(1)$.

Method: choose $[n = O(k), k, \delta n]$ code C.

Player 1 sends random $i \in [n]$, $C(x)_i$.

$$\Pr[C(x)_i \neq C(y)_i] \geq \delta$$

Almost k-wise independence [Naor-Naor]

ϵ-biased sample spaces suffice.

$\forall S \subseteq [m], S \neq \emptyset,$

$$1/2 - \epsilon \leq \Pr[\oplus_{i \in S} X_i = 1] \leq 1/2 + \epsilon$$

sample space

X_1

$\vdots$

X_m

generator matrix

$[n, m, (1/2 - \epsilon)n]_2$ code

$$\oplus_{i \in S} X_i \leftrightarrow \text{codeword } v_S$$

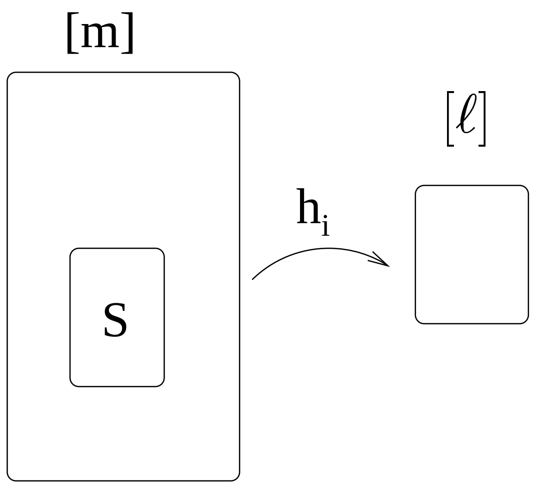
$$\Pr[\oplus_{i \in S} X_i = 1] = \text{relative weight}(v_S) \geq 1/2 - \epsilon.$$

Also need maximum relative distance $\leq 1/2 + \epsilon$.

Sample space size = code length.

Existence:	$O(m/\epsilon^2)$	
Explicit:	$O(m/\epsilon^4)$	[Naor-Naor]
	$O(m^2/\epsilon^2)$	RS concat Hadamard
		[Alon-Goldreich-Hastad-Peralta]

Hashing [Friedman]



$\ell = O(1)$

Goal: Construct $h_1, \ldots, h_{O(\log m)}$:

$$(\forall S : |S| = \ell)(\exists i) h_i \text{ 1-1 on } S.$$

$\ell = 2$: $h_i(x) = \text{binary}(x)_i$.

Doesn't generalize directly: 001,010,100.

Observation: suffices to map 1-1 to any $q = O(1)$.

Method: choose
$[n = O(\log_q m), \log_q m, d > (1 - 1/\binom{\ell}{2})n]_q$ code C.

$$h_i(x) = C(x)_i$$

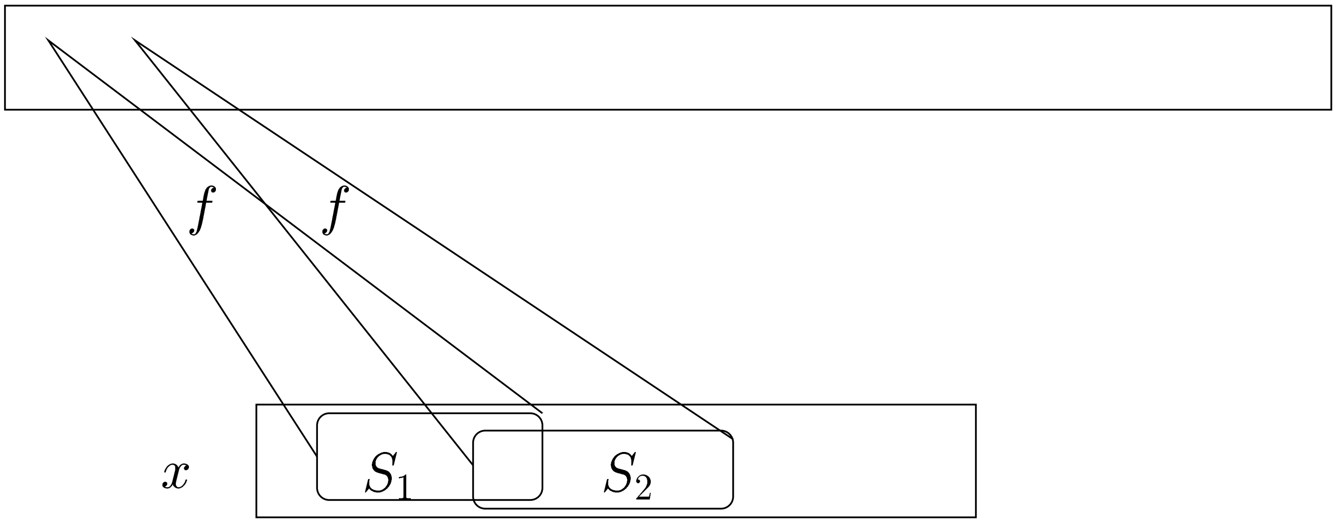
Application: $O(n \log n)$ size monotone formula for threshold $\ell = O(1)$.

PRG's from Hard Functions [Nisan-Wigderson]

PRG's $\Rightarrow$lower bounds.

Nisan-Wigderson: Lower bounds $\Rightarrow$PRG's.

Suppose $f : \{0,1\}^m \to \{0,1\}$ hard on average.



Designs:

$$\begin{aligned} |S_i| &= m \\ |S_i \cap S_j| &\leq \ell \end{aligned}$$

$\iff$ Binary code:

$$\begin{aligned} \text{weight(codewords)} &= m \\ \text{minimum distance} &\geq 2(m-\ell) \end{aligned}$$

Worst-Case → Average-Case Hardness

[Lipton, Beaver-Feigenbaum]

Given: worst case hard $f : \{0,1\}^m \to \{0,1\}$.

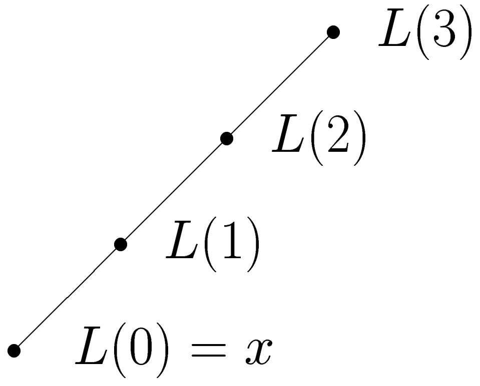
Encode f as a Reed-Muller code $\bar{f} : \mathbb{F}_q^m \to \mathbb{F}_q$
$\iff$ multilinear extension to $\mathbb{F}_q$; degree$(\bar{f}) \leq m$.

$$f(x) = \sum_{S \subseteq [m]} f_S \prod_{i \in S} x_i$$

Efficiently compute $\bar{f}$ on $\geq 1 - \frac{1}{3(m+1)}$ fraction of inputs
$\Rightarrow$efficiently compute $\bar{f}$ efficiently everywhere.

random line L

$\deg(\bar{f} \circ L) \leq m$



$\bar{f}(L(0))$ interpolated from $\bar{f}(L(1)), \ldots, \bar{f}(L(m+1))$.

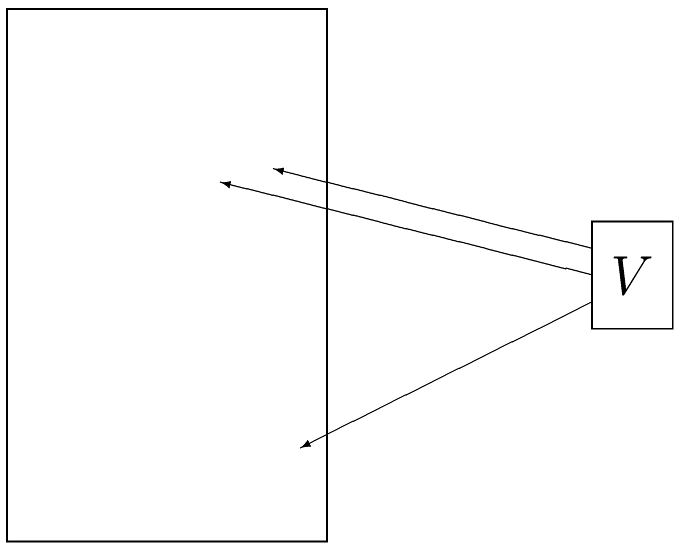
Local Decodability: Any bit of codeword obtained in time poly$(\log n)$.

Probabilistically Checkable Proofs

[Arora-Safra, Arora-Lund-Motwani-Sudan-Szegedy]

NP: short proofs of membership.

Surprising: $\exists$ robust proofs.



3 random (correlated) probes $\Rightarrow$constant confidence proof is correct.

Uses local decodability and RM codes.

Improvements use long code: $[2^{2^k}, k]_2$ code of all binary functions [Bellare-Goldreich-Sudan].

List Decoding

Output list of close codewords.

List decoding possible for $> 1/2$ minimum distance.

Combinatorial: few close codewords.
Algorithmic: find them.

Reed-Solomon codes:

- Algorithmic decoding within distance $n - \sqrt{kn}$ [Guruswami-Sudan, Sudan].

Random $[n, k]_n$ codes:

- Combinatorial: poly(n) codewords within distance $n - k \log n$.

Binary codes:

- Combinatorial: $[n, k, (1/2 - \epsilon)n]_2$ codes have $\leq 1/\epsilon$ codewords within distance $(1/2 - \sqrt{\epsilon})n$ [Johnson bound].
- Algorithmic: local list decode Hadamard [Goldreich-Levin], RM concat Hadamard [Sudan-Trevisan-Vadhan].

Improved Average Case Hardness

[Sudan-Trevisan-Vadhan]

Encode worst-case hard f by $\bar{f}$

code: RM concat Hadamard.

Efficiently compute $\bar{f}$ with probability $1/2 + \epsilon$

$\Rightarrow$efficiently compute $\bar{f}$ everywhere.

Local list decoding of RM concat Hadamard: output compact representation of codewords (circuits, machines).

Cryptography: Hardcore Predicates

[Goldreich-Levin]

One-way function $f : \{0,1\}^m \to \{0,1\}^m$.

- f poly-time.
- f^{-1} hard.

Example: $f(x) = g^x \bmod p$ (g generator, p prime).

Want: Computing $b(x) \in \{0,1\}$ hard, given $f(x)$.

C locally list decodable $[N, 2^m]_2$ code, e.g. Hadamard.

Theorem: $b(x,i) = C(x)_i$ hard to compute, given $f(x), i$.
($C(x)_i = x \cdot i$ for Hadamard code.)

Proof Outline:
Compute $b(x,i) = C(x)_i \Rightarrow$few candidates for x.
Test if $f(x_{candidate}) = f(x)$.

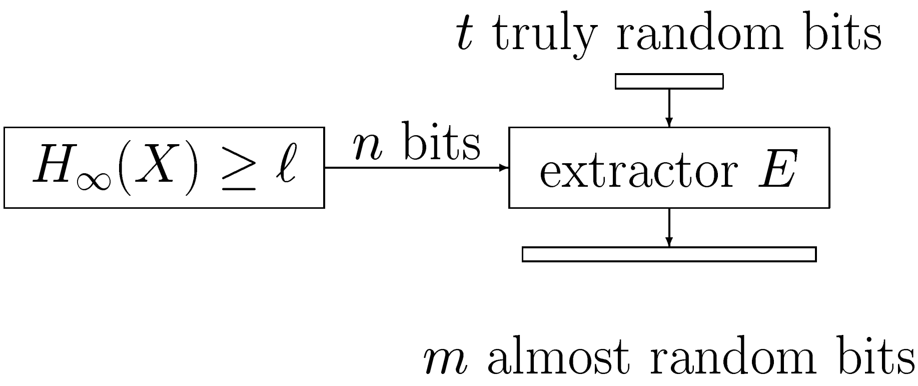
Extractors for Weak Random Sources

[Nisan-Z]

PRG's: few good random bits available

Extractors: also many weakly random bits available

$$H_\infty(X) = \log_2 \frac{1}{\max \Pr[X = x]}$$



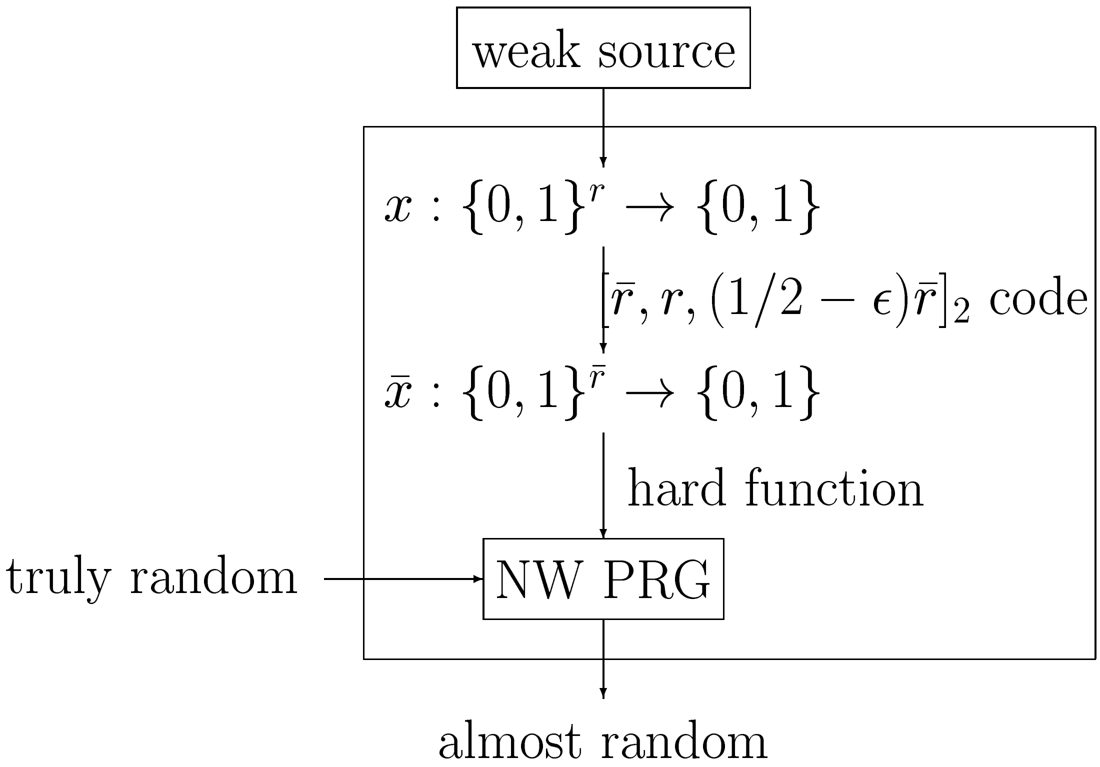
Ideal: $t = O(\log n)$, $m = \ell$.

Numerous applications:
Simulating randomized algorithms using weak source [Z]
PRG's for space-bounded computation [NZ]
Random sampling using few random bits [Z]
Expanders that beat the eigenvalue bound [WZ]

$\vdots$

PRG's + Codes → Extractor [Trevisan]

Weakly random function hard.

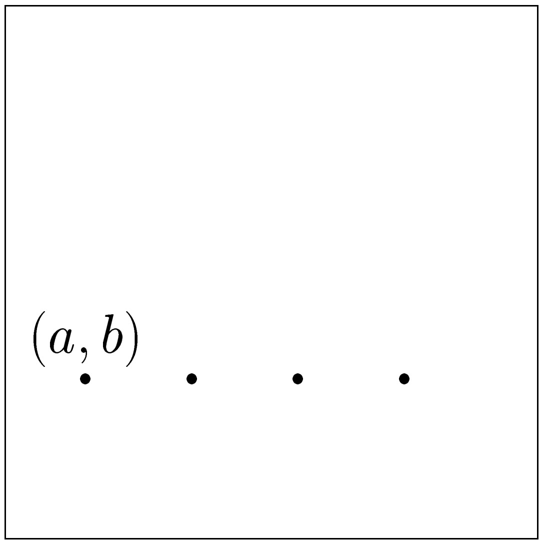


Uses combinatorial list decodability of binary code.

Reed Muller Codes → Extractor

[Ta-Shma-Z-Safra]

C an $[r, \log q, (1/2-\epsilon)r]_2$ code (need combinatorial list decodability).



$$\mathrm{BE}(\overbrace{\text{poly } p,\ \deg(p) < \sqrt{n}}^{\text{weak source}};\overbrace{(a,b)\in\mathbb{F}_q^2, i\in[r]}^{\text{truly random}})$$

$$= \langle C(p(a,b))_i, C(p(a+1,b))_i, \ldots, C(p(a+m-1,b))_i\rangle$$

Proof uses list decodability of Reed-Solomon and algebraic structure.

Strong extensions: [Shaltiel-Umans].

Extractor Codes [Ta-Shma-Z]

Extractor code: $C(x) = \langle E(x, y_1), \ldots, E(x, y_T))\rangle$

Good for list decoding in soft-decision model for highly noisy channels– better than Reed-Solomon.

Many ($\geq q^{k/2}$) RS codewords have all coordinates quadratic residues.

Explicit extractor $\Rightarrow$explicit encoding.

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

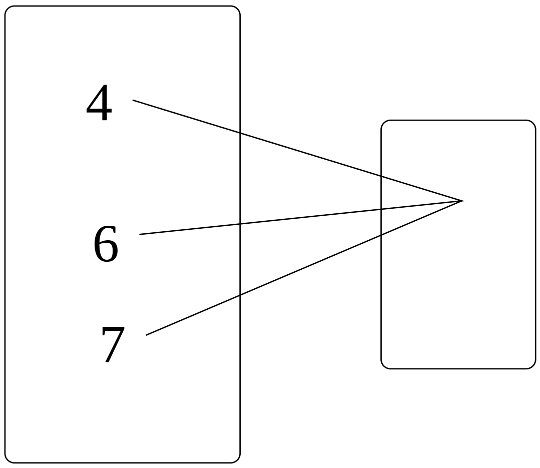
Johnson bound: binary codes with good distance
$\Rightarrow$combinatorial list decodability.

follows from

Good collision probability $\Rightarrow$extractor property.

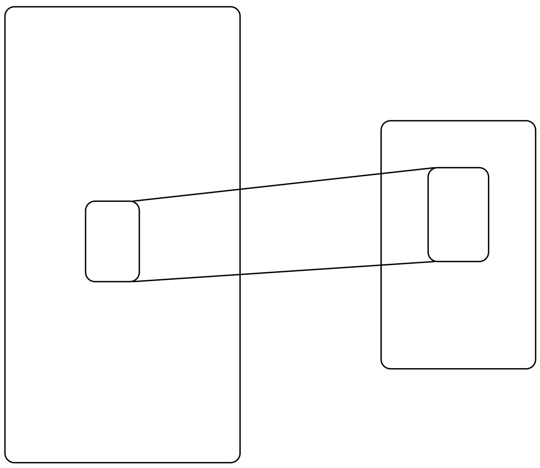
Expander Codes [Sipser-Spielman]

Bipartite graph $\rightarrow$ parity check matrix [Gallager,Tanner]:



$x_4 + x_6 + x_7 = 0.$

Expander: each subset has many neighbors.

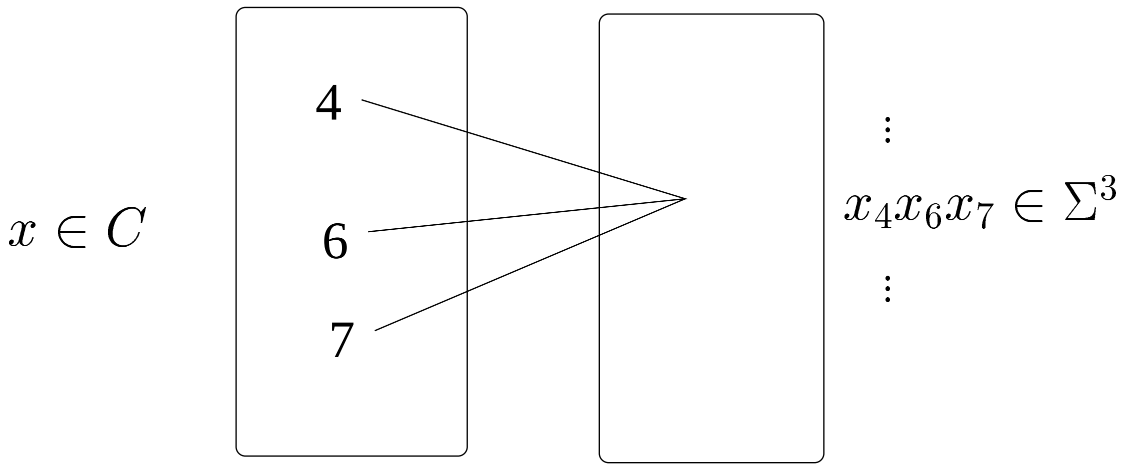


[Spielman]: Linear-time encodable and decodable codes.

[Luby-Mitzenmacher-Shokrollahi-Spielman]: Tornado codes.

More Expander-Based Codes

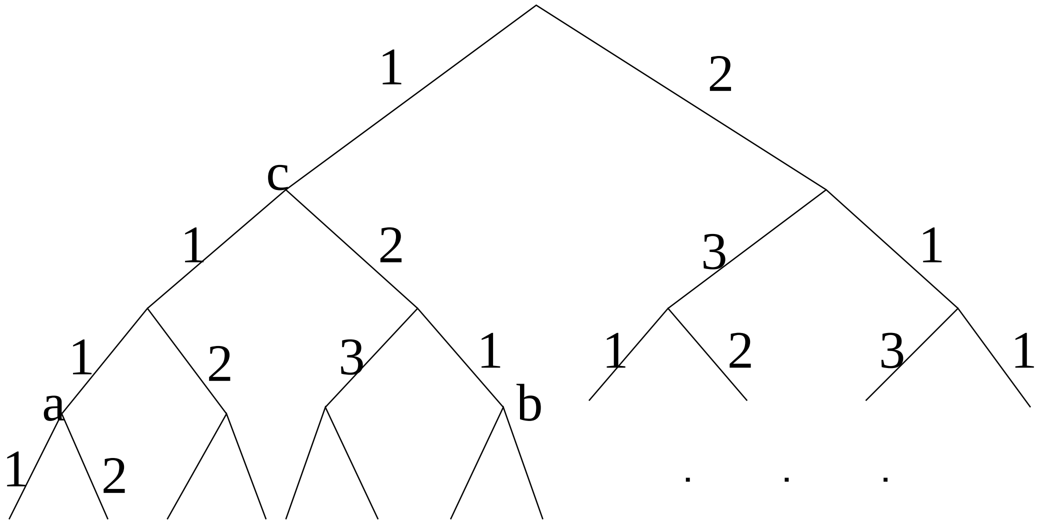
[Alon-Bruck-Naor-Naor-Roth]:
Code C over Σ on left $\rightarrow$ code C' over Σ^3 on right



[Guruswami-Indyk]:

- Efficiently encodable, (list) decodable.
- Improved rates.

Tree Codes [Schulman]



path distance(a, b) = relative Hamming distance of paths to least common ancestor c.
Example: 1/2.

min path distance = $\min_{a,b}$ path distance(a, b)

Existence: min path distance $\Omega(1)$, alphabet size $O(1)$.

Construction: open.

Application: Interactive communication.

Conclusions

- Codes extremely useful.
- Often because small coordinate sample distinguishes codeword.
- Most useful properties: MDS, minimum distance, local decoding, list decoding.
- Extractors, expanders $\rightarrow$ codes.

Further Directions

- Other applications?
- Other useful properties of codes?
- Better codes through pseudorandomness?

Acknowledgement

Besides original papers, a key source was a talk by Madhu Sudan, available at
http://theory.lcs.mit.edu/~madhu/coding/course.html.