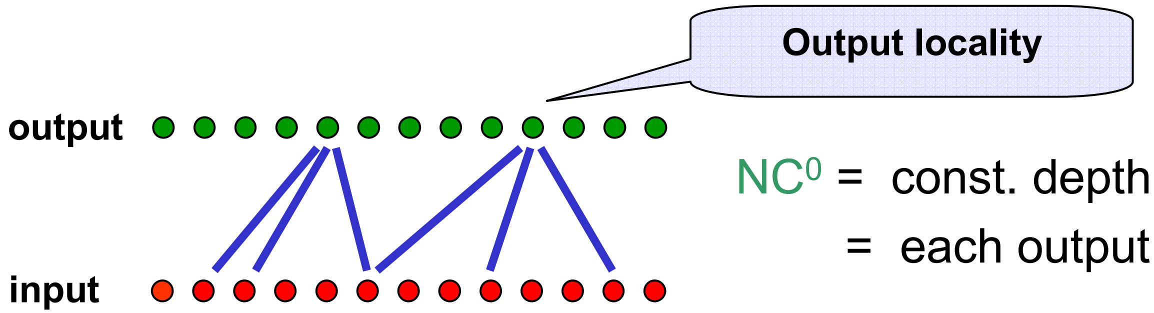
Outline

1. (Long) Introduction
2. Randomized Polynomials (w/applications to round-efficient MPC)
3. Randomized Encodings w/applications to NC^0 Cryptography
4. Constant Input Locality
5. Computational Randomized Encodings (w/applications)
6. NC^0 Linear Stretch PRG (w/applications)

Cryptography with Constant Input Locality

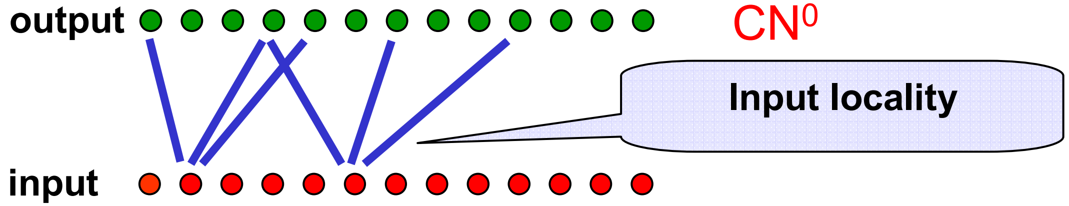
Till now we considered only NC^0 functions…



NC^0 = const. depth circuits with bounded fan-in

= each output bit depends on $O(1)$ input bits

Q: Can cryptographic primitives be realized by functions in which each input bit affects a constant number of output bits?



CN^0

Motivation I: Avalanche Property

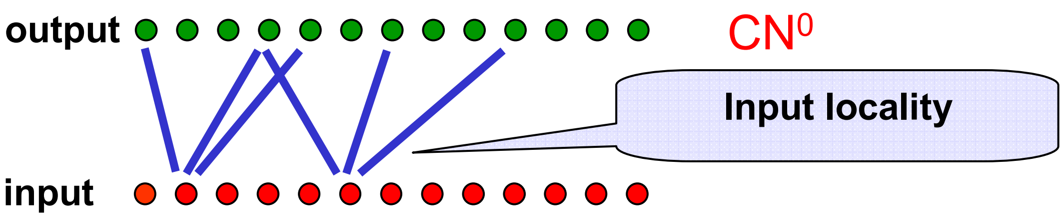
Confusion/Diffusion, Avalanche [Shannon 49, Feistel 73]:

input-output dependencies of a **block cipher** should be "complex"

"*The important fact is that all output digits have potentially become very involved functions of all input digits*" [Feistel 73]

Easily justified in block ciphers (or pseudorandom functions/permutations).

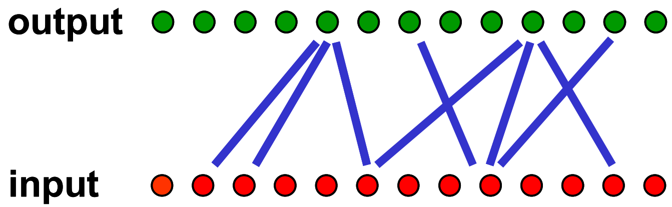
Is it also true for other primitives?



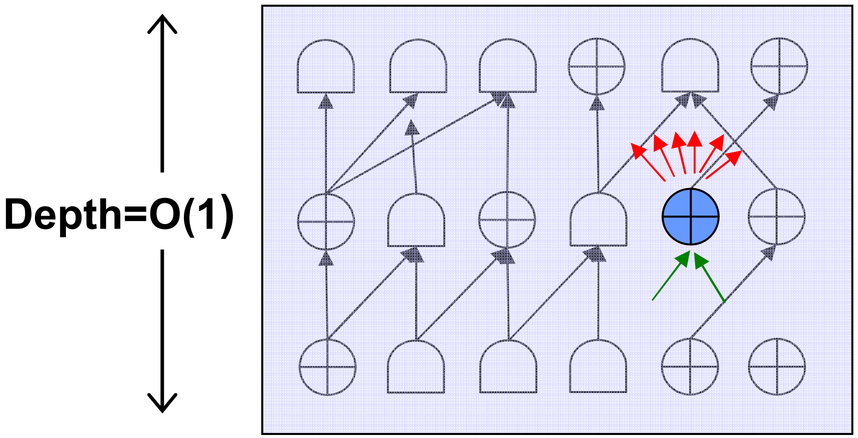
Motivation II: Fast Crypto Hardware

Functions of **const. output locality** & **input locality**

$NC^0 \cap NC^0$



Circuits of **const.** depth, **const.** fan-in, **const.** fan-out **unbounded** fan-out



Motivation III: Complexity Theory

Bounded-occurrence
k-Constraint Satisfaction Problem

$$\left.\begin{array}{l} - \ X_1 + X_3 \cdot X_5 = 0 \\ - \ X_2 \cdot X_3 \cdot X_4 = 1 \\ . \\ . \\ . \\ - \ X_2 + X_3 + X_4 = 1 \end{array}\right\}$$

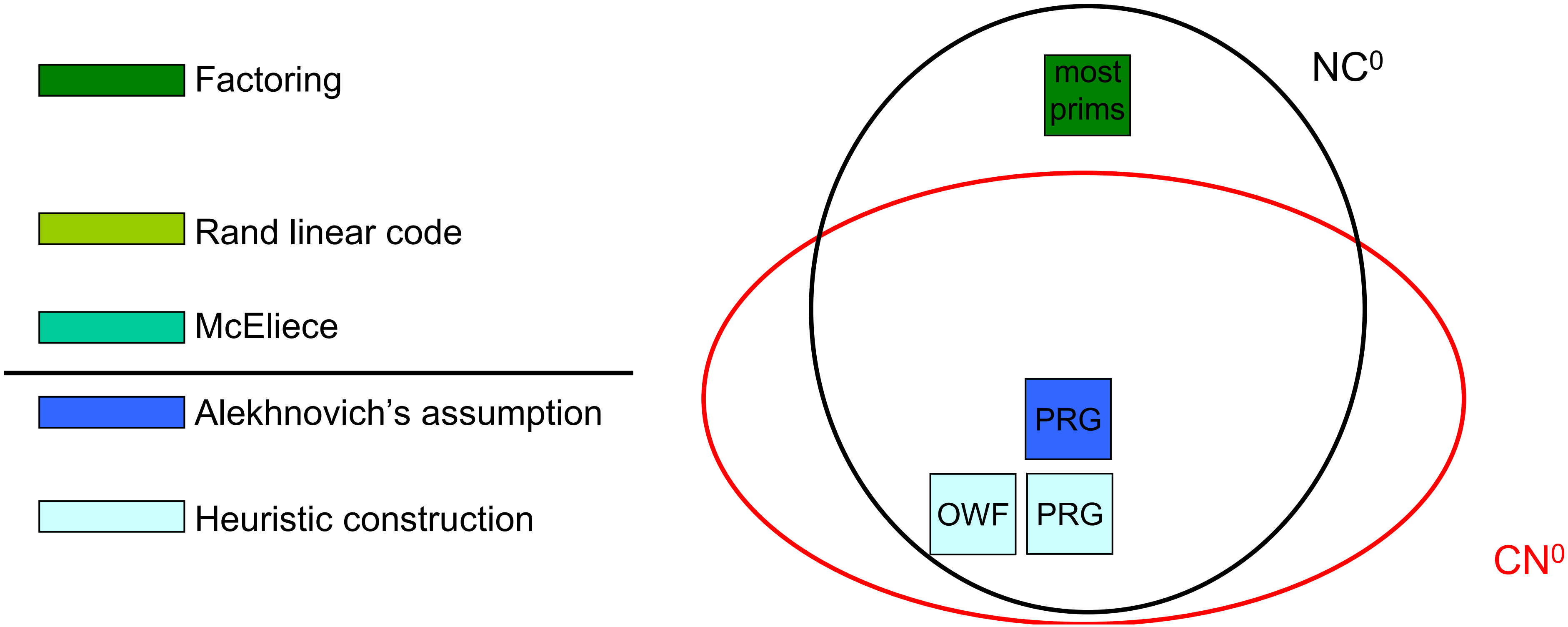
- List of constraints over n variables $x_1,\ldots,x_n$
- Each constraint involves **k=O(1)** variables
- Each variable appears in **O(1)** constraints

- Goal: Find a satisfying assignment
- Fact: Still Hard in many aspects:
  - Cook-Levin Theorem [C71,L73]: NP-hard
  - [C71]: **Still** NP-hard
  - PCP Theorem [ALMSS,AS 92]: NP-hard to approximate
  - [PY88]: **Still** NP-hard to approximate
  - OWF in NC^0 [AIK 04]: "Cryptographically-hard"
    Still "Cryptographically-hard" ?
  - OWF in $NC^0 \cap CN^0$ $\Rightarrow$ **YES**

Previous Work

- [Goldreich 00] **Heuristic** OWF in $NC^0 \cap CN^0$
- [Mossel
- [AIK 04]
  - Primitives in NC^0 from standard assumptions (e.g., factoring, DLOG, lattices)
  - $\Rightarrow$ OWFs, PRGs, Encryption, Signatures, Hash… in NC^0 from factoring
- [AIK 06] Linear PRG in $NC^0 \cap CN^0$ from Assumption of [Alekhnovich 03]

Crypto in CN^0 under standard assumptions?



Main Result

A characterization of crypto tasks computable in CN^0

Possible in $CN^0 \cap NC^0$

- **One-Way Functions** *
- **Pseudorandom Generators** *
- **Commitment Schemes** *
- **Semantically-Secure Encryption (symmetric* , public-key**)**

Impossible in CN^0

- **Message Authentication Codes**
- **Signatures**
- **Non-Malleable Encryption (symmetric, public-key)**

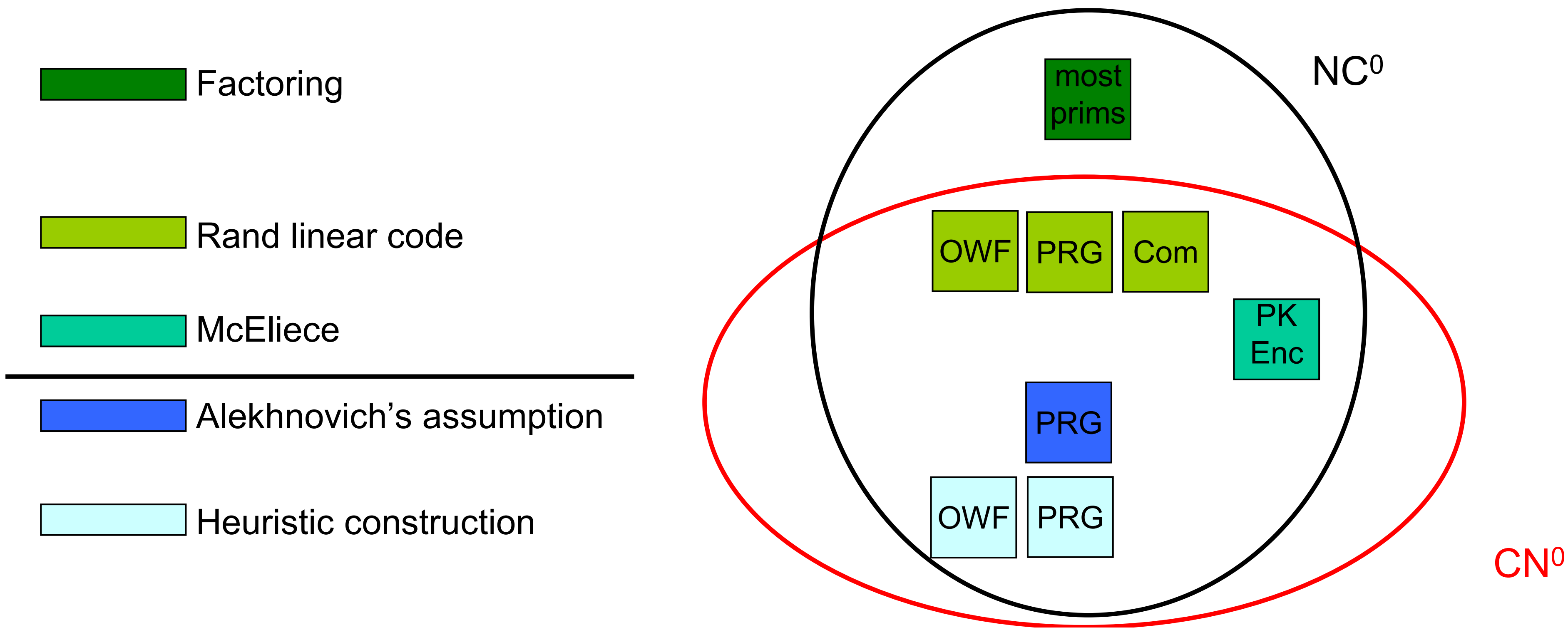
* If hard to decode random binary linear code / learn parity w/noise

** If hard to break McEliece cryptosystem

Previous Work

- [Goldreich 00] **Heuristic** OWF in $NC^0 \cap CN^0$
- [Mossel
- [AIK 04]
  - Primitives in NC^1 from standard assumptions (e.g., factoring, DLOG, lattices)
  - $\Rightarrow$ OWFs, PRGs, Encryption, Signatures, Hash… in NC^0 from factoring
- [AIK 06] Linear PRG in $NC^0 \cap CN^0$ from Assumption of [Alekhnovich 03]

Crypto in CN^0 under standard assumptions?

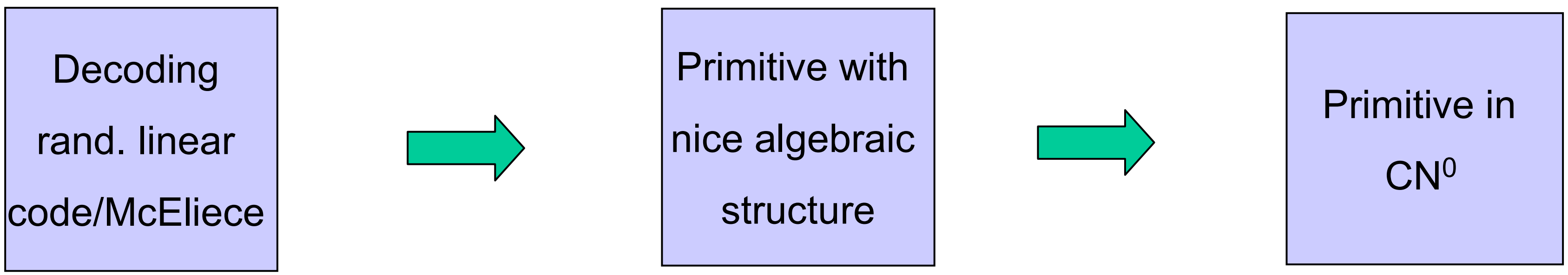


Positive Results

Proof Outline:

- Use the randomized encoding paradigm
- New Construction:

  encoding in CN^0 for functions with "nice algebraic structure"
- Assumption: Hardness of decoding random linear code / McEliece
- Assumption $\Rightarrow$ crypto primitives with "nice algebraic structure"



Encoding in CN^0 – Toy Example

$f(x) = (x_1 + x_2, \quad x_1 + x_3, \quad x_1 + x_4, \quad x_1 + x_5)$

Goal: Reduce locality of x_1 without increasing locality of other vars

Attempt 1 (chain):

$g(x) = (x_1 + x_2, \quad -x_2 + x_3, \quad -x_3 + x_4, \quad -x_4 + x_5)$

- Deterministic encoding !
- Problem: Increased the locality of other vars

Attempt 2 (replace):

$$g(x,r) = (r_1 + x_2, \quad r_2 + x_3, \quad r_3 + x_4, \quad r_4 + x_5$$
$$x_1 - r_1, \quad x_1 - r_2, \quad x_1 - r_3, \quad x_1 - r_4)$$

- Problem: Didn't reduce the locality of x_1

Solution: Combine 1+2 (replace and chain)

$$g(x,r) = (\mathbf{r_1} + x_2, \quad r_2 + x_3, \quad r_3 + x_4, \quad r_4 + x_5$$
$$x_1 - \mathbf{r_1}, \quad \mathbf{r_1} - r_2, \quad r_2 - r_3, \quad r_3 - r_4)$$

- Locality: x_1 is 1, x_2,x_3,x_4,x_5 did not increase, r_i's is 3

Encoding in CN^0 – Toy Example

$f(x) = (x_1 + x_2, \quad x_1 + x_3, \quad x_1 + x_4, \quad x_1 + x_5)$

Goal: Reduce locality of x_1 without increasing locality of other vars

Solution: Combine 1+2 (replace and chain)

$g(x,r) = (r_1 + x_2, \quad r_2 + x_3, \quad r_3 + x_4, \quad r_4 + x_5$

$x_1 - r_1, \quad r_1 - r_2, \quad r_2 - r_3, \quad r_3 - r_4 \;)$

- Locality: x_1 is 1, x_2,x_3,x_4,x_5 did not increase, r_i's is 3

Encoding in CN^0 – Toy Example

$f(x) = (x_1 + x_2, \quad x_1 + x_3, \quad x_1+x_4 \quad x_1 + x_5)$

Goal: Reduce locality of x_1 without increasing locality of other vars

Solution: Combine 1+2 (replace and chain)

$$g(x,r) = \begin{pmatrix} r_1 + x_2, & r_2 + x_3, & r_3 + x_4, & r_4 + x_5 \\ x_1 - r_1, & r_1 - r_2, & r_2 - r_3, & r_3 - r_4 \end{pmatrix}$$

•Correctness: To decode, add the corresponding entries.

•Privacy: g(x,r) distributed uniformly under correctness constraint.

By iterating the basic gadget for every variable ⇒

Corollary: every linear function can be encoded by function w/input locality 3

Encoding in CN^0 – Generalization

- Suppose that f is given in some additive form.
- $f(x)= (\boxed{x_1x_2}+x_2x_3x_5, \quad \boxed{x_1x_2}+x_2x_4x_5, \quad \boxed{x_1x_2}+x_1x_3x_4, \quad \boxed{x_1x_2}+x_2x_5)$
- rank(x_i)= # of distinct terms in which x_i appears

rank(x_1)= 2

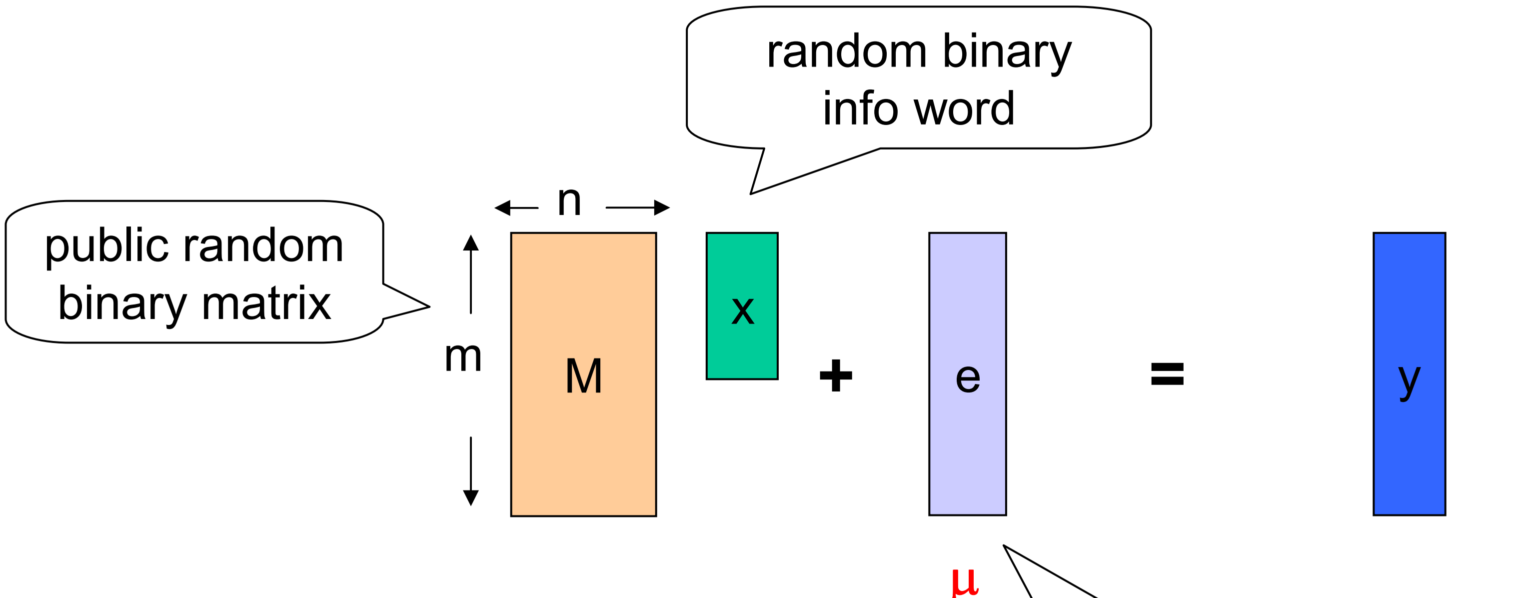
- Thm. f can be encoded by g such that:
  - input locality of x_i is rank(x_i)
  - input locality of random inputs is at most 3.
  - output locality is not increased.

- Tightness: Some functions cannot be encoded with locality < rank(x_i)

$\Rightarrow$ Some functions cannot be encoded in CN^0 (even w/non-efficient encoding).

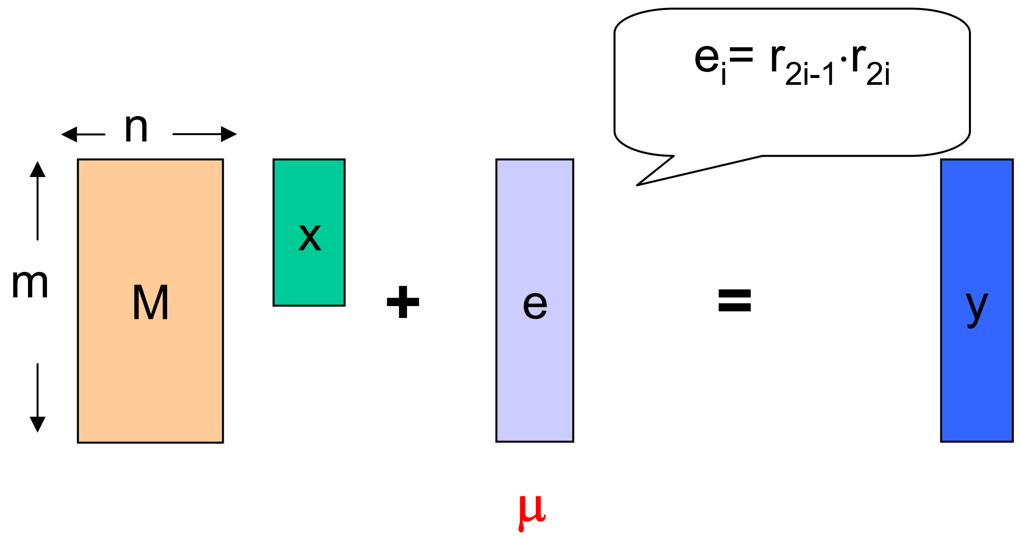
– Unlike NC^0 : “every f has (non-efficient) encoding in NC^0 “ [AIK04]

Decoding Random Linear Code



- Problem: Given M,y find x
- Params: m, μ. E.g., $m=10n$, $\mu = 1/4$.
- Assumption: Problem is computationally hard
- Well studied in Coding Theory/Learning Theory [Kearns98, BKW00, Lyu05, FGKP06]
- Assumption does not hold $\Rightarrow$ major breakthrough in Coding Theory
- Similar assumptions in [GKL93, BFKL93, Chab94, HB01, Reg05, JW05, KS06]

Decoding Random Linear Code



• Problem has nice algebraic structure:

linear function + some low-degree noise

• Can be used to construct primitives with low rank and low degree

- e.g., OWF, PRG, Commitment