The complexity of sampling:

a new paradigm in theoretical computer science

Emanuele Viola

Northeastern University

Fall 2011

New paradigm
[V FOCS 2010; SIAM J. Computing]

- Classical: Efficient Computation

  f : INPUT → OUTPUT

- New: Efficient Sampling

  f : RANDOM BITS → OUTPUT DISTRIBUTION

- Uncharted territory

- Progress on long-standing problems

Outline

- Randomness extractors
- Data structures
- Pseudorandom generators
- Error-correcting codes

Randomness extractors

- Randomness useful in computation, crucial in crypto
  - Monte-Carlo, passwords, ...

- But available sources weak: (correlation, bias, ...)
  - Thermal noise, Keystroke statistics, ...

- Want: weak source → **Extractor** → good

  ≈ uniform

Von Neumann extractor '51

- Source: n bits $Y_1\ Y_2\ \dots\ Y_n$
  independent, identical, unknown bias: $\Pr[\,Y_i = 1] = p$

- Extractor($Y_1\ Y_2\ \dots\ Y_n$) $\approx$ uniform

  Pair bits: 01 $\rightarrow$ 1,
  10 $\rightarrow$ 0,
  00, 11 $\rightarrow$ skip

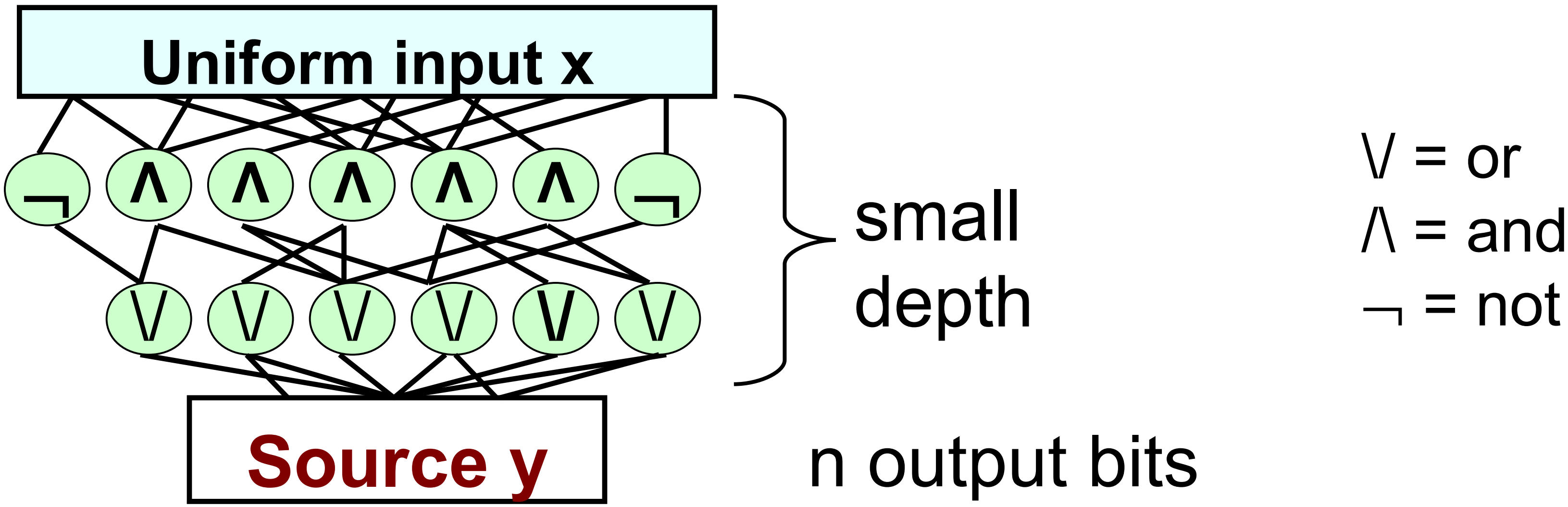
  $\Pr[1] = \Pr[0] = p(1-p)$

- Intel 80802 Firmware Hub chip

Randomness extractors

- How to handle more general sources?
- In practice: Crypto Hash Functions (e.g. SHA-2)
  No provable guarantee
- Major line of research in theoretical computer science
  ['85 - present]
- Led to goal: extract from sources sampled efficiently
  “reasonable model for sources arising in nature”
  [Trevisan Vadhan 2000]

Our extractor for small-depth circuits
[V; FOCS 2011]



- Theorem From n bits with entropy k: Extract k(k/n)
- First extractor for circuits; generalizes previous models

Key proof idea

- Extractor ⇔ sampling is difficult

$E : \{0,1\}^n \to \{0,1\}$ (balanced) ⇔ circuits cannot sample $E^{-1}(0)$ (uniformly, given random bits)

- To extract, use (and extend) techniques for sampling

[V], [Lovett V]

Key proof idea

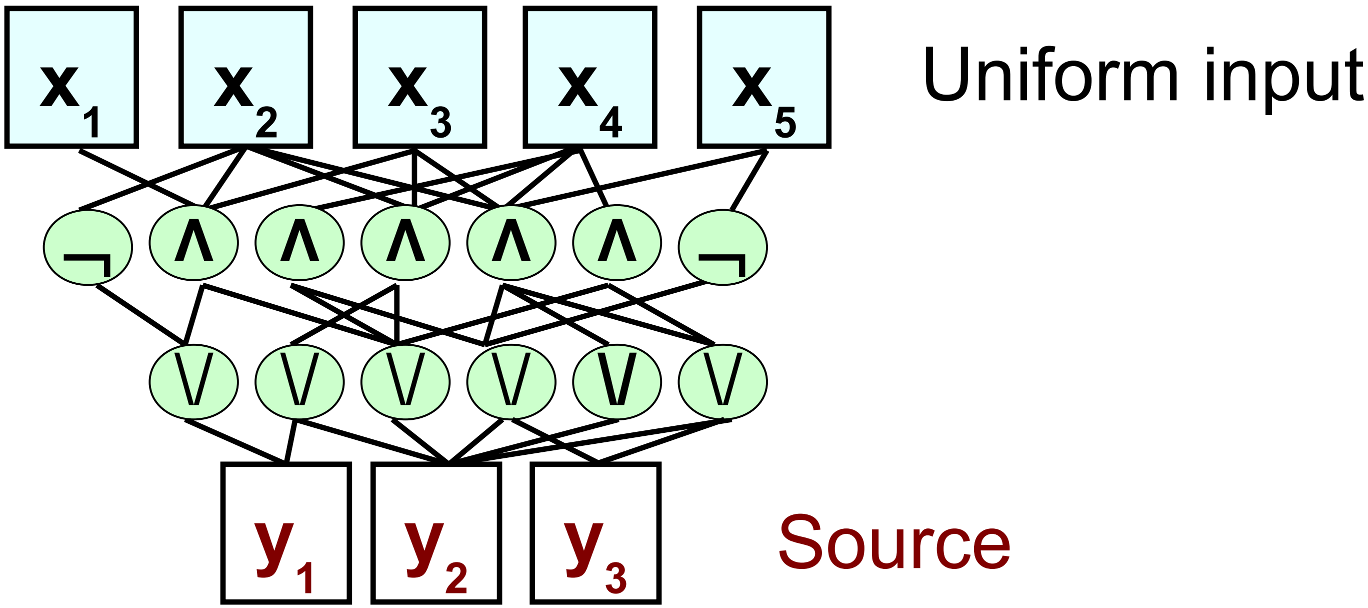
- Extractor ⇔ Circuit lower bound for sampling

$E : \{0,1\}^n \to \{0,1\}$ (balanced) ⇔ circuits cannot sample $E^{-1}(0)$ (uniformly, given random bits)

New approach!

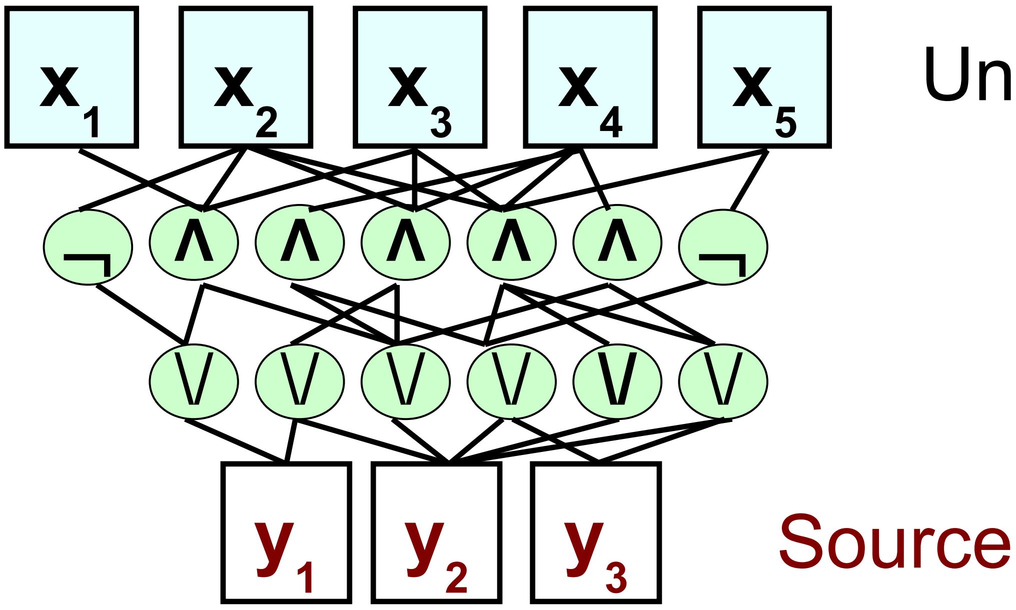
- To extract, use (and extend) techniques for sampling [V], [Lovett V]

Proof



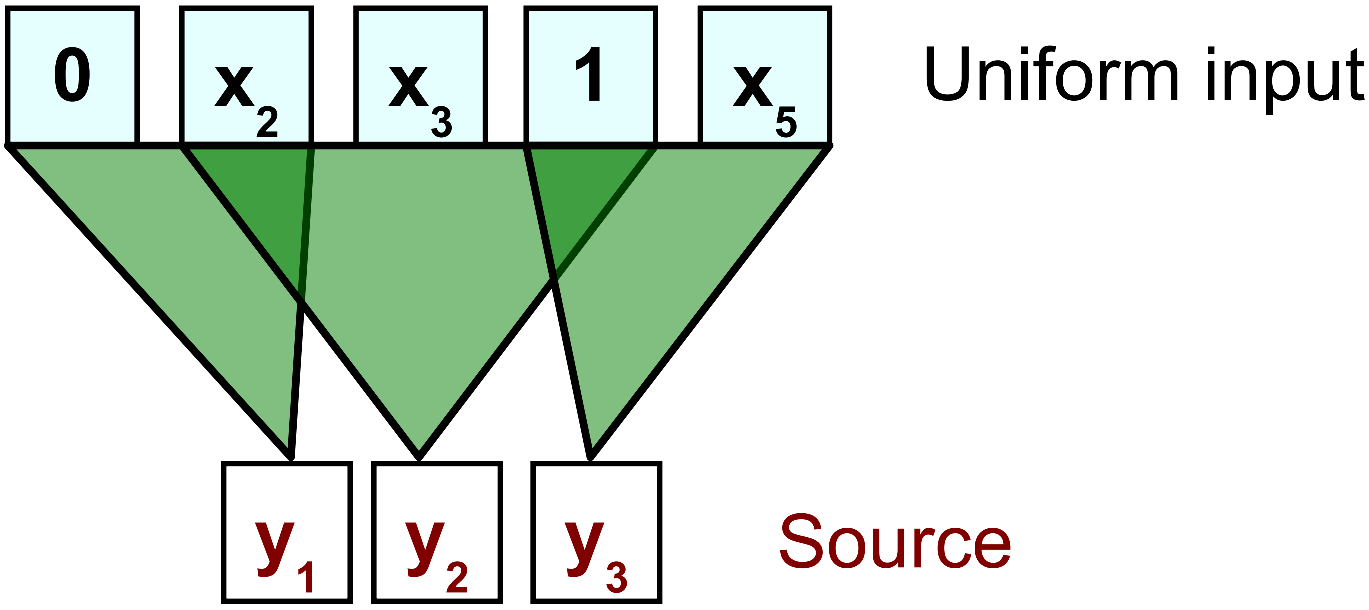
- Want: extract randomness from y_1 y_2 y_3
- We reduce to source: each y_i depends on one x_h then apply extractor from literature

Proof



- Step 1: Fix (Condition) few random x_i

Proof

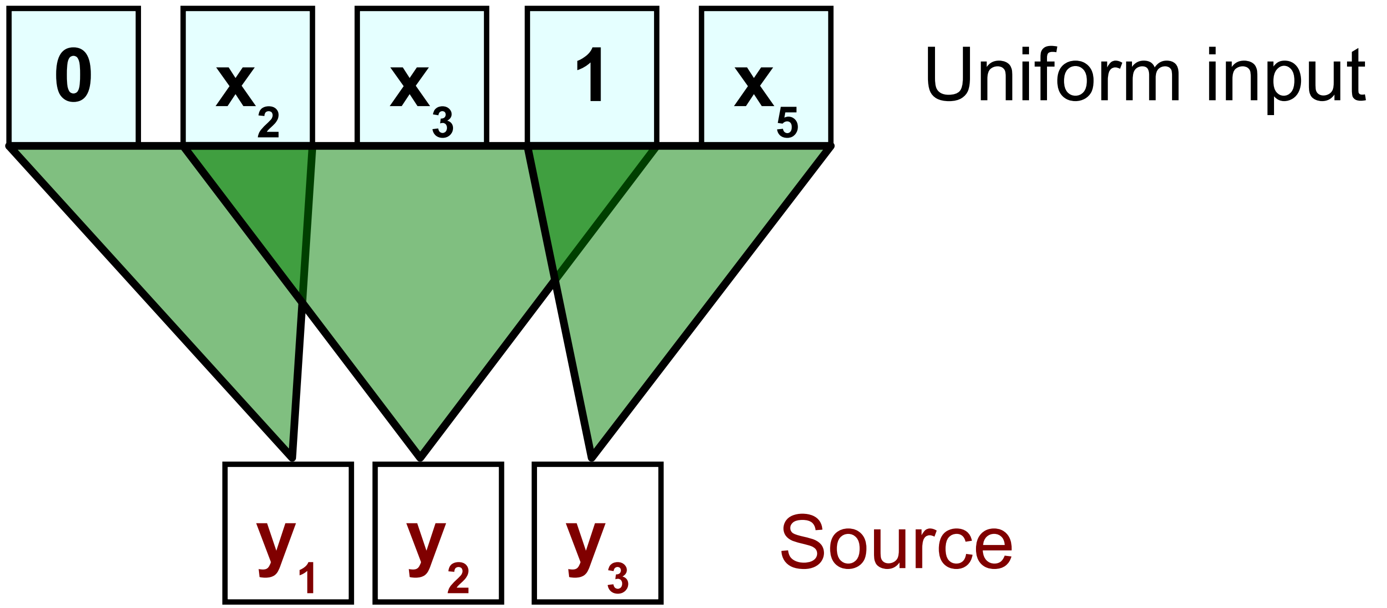


- Step 1: Fix (Condition) few random x_i

  [Hastad] Source turns local: y_i depends on few x_j

  [V] No entropy loss (Noise isoperimetric inequality)

Proof

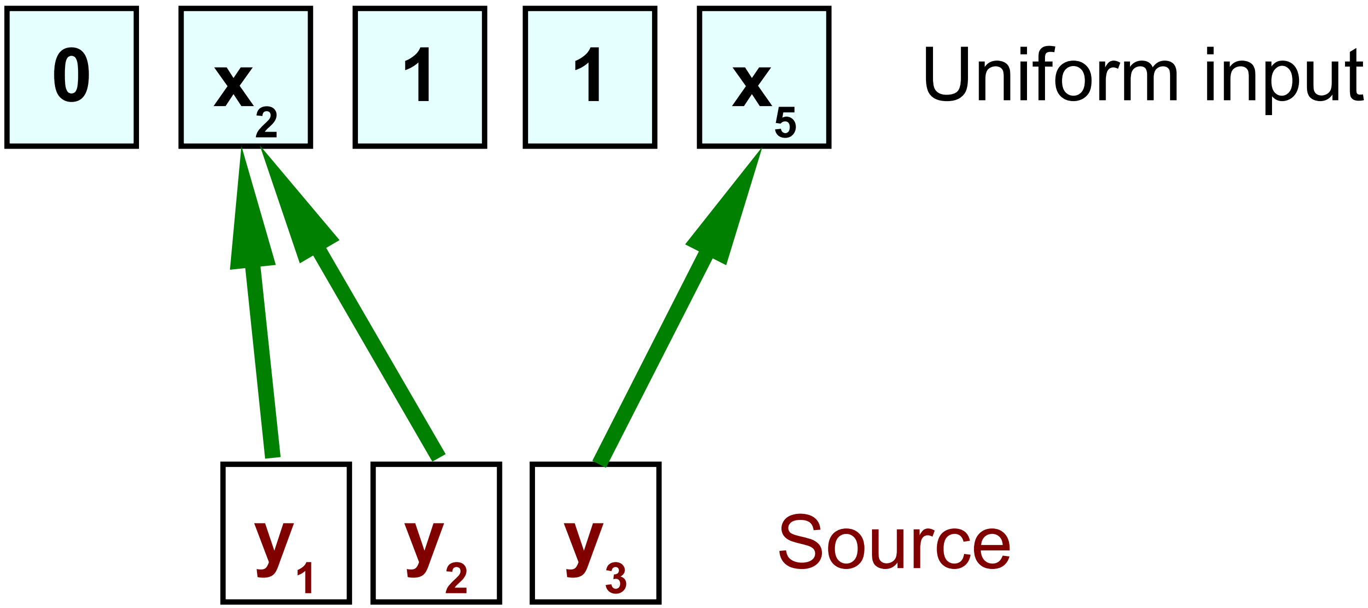


- Step 2: (Iteratively)
  [V] Pick high-entropy y_i.

  Local ⇒ some x_j high influence. Fix relevant rest

  ⇒ y_i depends on x_j only, and retains entropy

Proof



- Now each y_i depends on one $\mathbf{x}_h$
- Apply extractor from literature

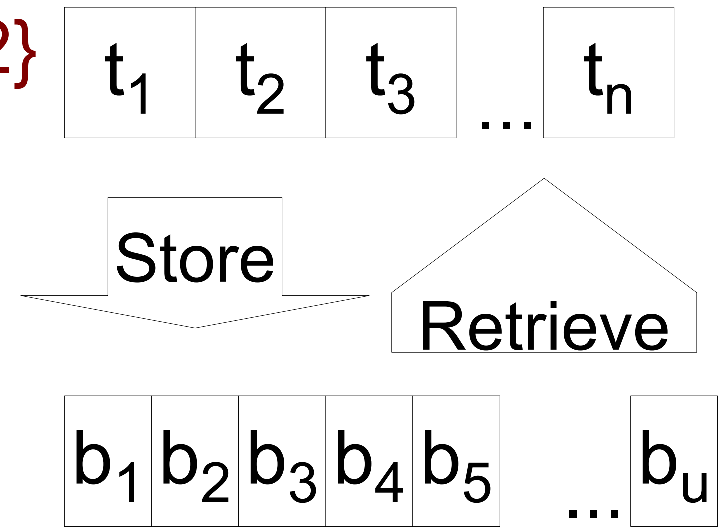
♦

Outline

- Randomness extractors
- Data structures
- Pseudorandom generators
- Error-correcting codes

Bits vs. trits

- Store n "trits" $t_1, t_2, \ldots, t_n \in \{0,1,2\}$



In u bits $b_1, b_2, \ldots, b_u \in \{0,1\}$

- Want:

Small space u $\left(\text{optimal} = \lceil n \lg_2 3 \rceil\right)$

Fast retrieval: Get t_i by probing few bits $\left(\text{optimal} = 2\right)$

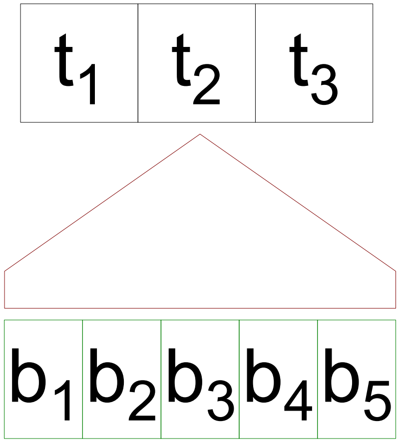
Two solutions

- Arithmetic coding:

  Store bits of $(t_1, \ldots, t_n) \in \{0, 1, \ldots, 3^n - 1\}$

  Optimal space: $\lceil n \lg_2 3 \rceil \approx n \cdot 1.584$

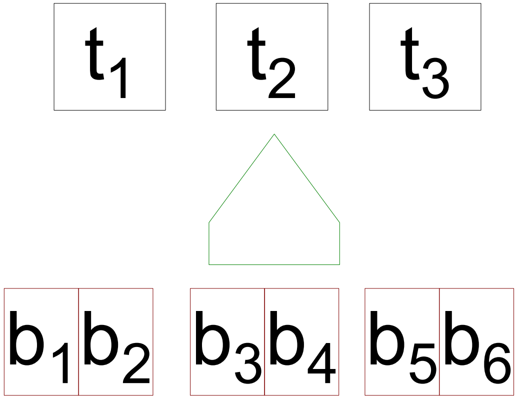
  Bad retrieval: To get t_i probe all > n bits



- Two bits per trit

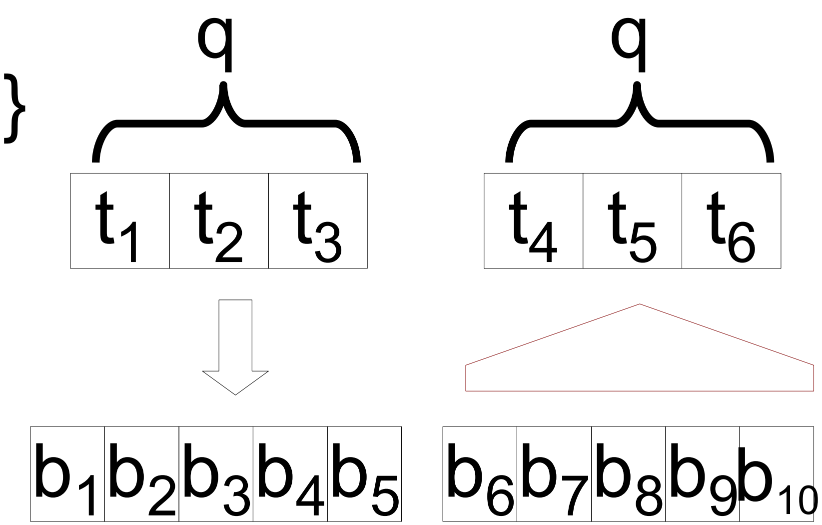
  Bad space: $n \cdot 2$

  Optimal retrieval: Probe 2 bits



Polynomial tradeoff

- Divide n trits $t_1, \ldots, t_n \in \{0,1,2\}$ in blocks of q
- Arithmetic-code each block



Space: $\lceil q \lg_2 3 \rceil n/q < (q \lg_2 3 + 1)\, n/q$

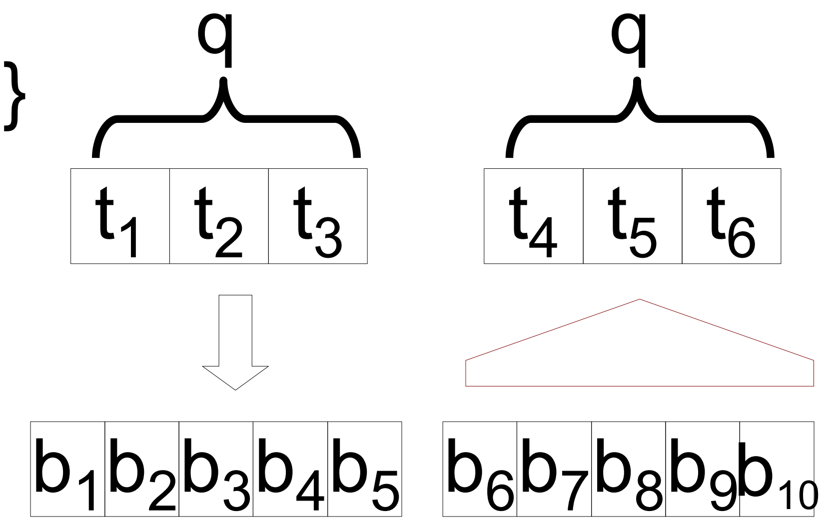
$= n \lg_2 3 + n/q$

Retrieval: Probe O(q) bits

polynomial tradeoff between probes, redundancy

Polynomial tradeoff

- Divide n trits $t_1, \ldots, t_n \in \{0,1,2\}$ in blocks of q
- Arithmetic-code each block



Space: $\lceil q \lg_2 3 \rceil n/q = (q \lg_2 3 + 1/q^{\Theta(1)})\, n/q$

$= n \lg_2 3 + n/q^{\Theta(1)}$

Retrieval: Probe O(q) bits

polynomial tradeoff between probes, redundancy

[V] logarithmic forms

Exponential tradeoff

- [Pătraşcu Thorup 08]

  Space: $n \lg_2 3 + n/2^{\Omega(q)}$

  Retrieval: Probe q bits

exponential tradeoff between probes, redundancy

- E.g., optimal space $\lceil n \lg_2 3 \rceil$, probe $O(\lg n)$

- Exponential tradeoff tight?

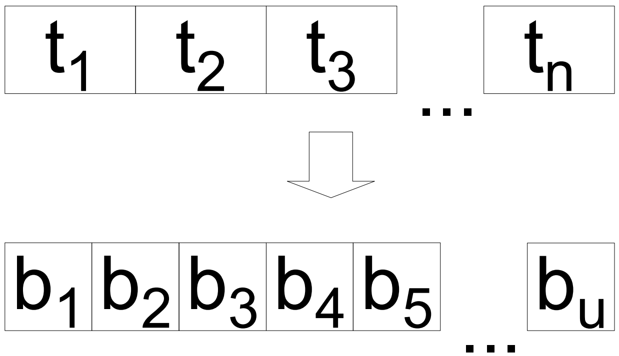
  "beyond the scope of current techniques"

Our results
[V; STOC 2009 Special Issue, SIAM J. Computing]

- Theorem: Tradeoff tight

  Store n trits $t_1, \ldots, t_n \in \{0,1,2\}$

  in u bits $b_1, \ldots, b_u \in \{0,1\}$.

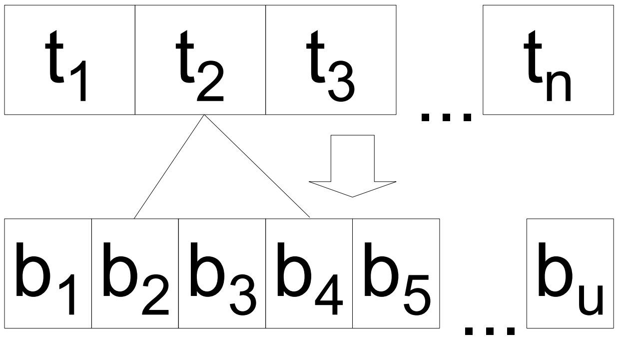


  Retrieval: probe q bits $\Rightarrow$ space $u > n \lg_2 3 + n/2^{O(q)}$.

- Matches [Pătraşcu Thorup]: space $< n \lg_2 3 + n/2^{\Omega(q)}$

Proof via sampling

- Store n trits in $u = n \lg_2 3 + r$ bits
  get trit by probing q bits



- Sample trits from bits, locality q
  distance $< 1 - 2^{-r}$ from uniform

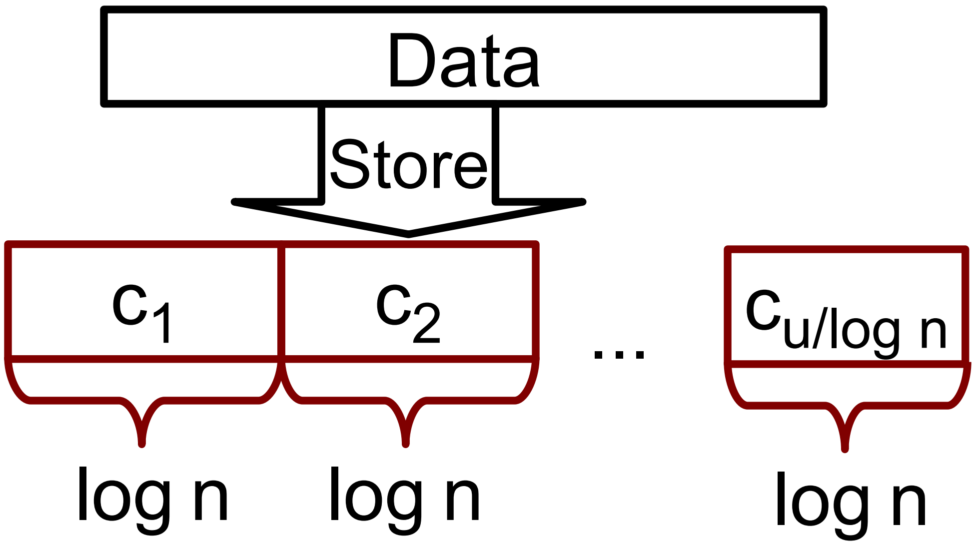
Proof: With prob. $> 2^{-r}$ uniform over trits' encodings ♦

- $\Rightarrow \exists$ trit $\approx$ uniform. Impossible: $1/3 \neq \text{INTEGER} / 2^u$ ♦

Cell-probe model

- So far: q = number of bit probes

- Cell model:
  q = number of probes
  in cells of log(n) bits



- Think of cell as long in C language

Our results
[Pătraşcu V; SODA 2010]

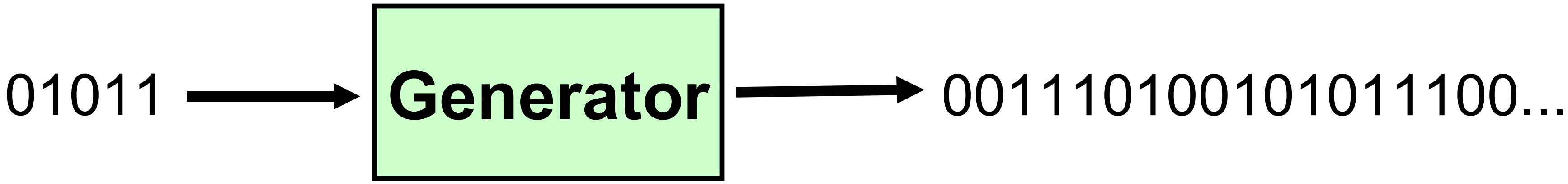
- "Bread and butter" of data structures:
  Store n bits $x_1, x_2, \ldots, x_n$ in cells
  Retrieve **PrefixSum(i)** $:= \sum_{k \leq i} x_k \in \{0, 1, 2, \ldots, n\}$

	Space	Probes
	$n \log(n)$	1
	n	$n/\log(n)$
[Pătraşcu]	$< n + n / \log^{\Omega(q)} n$	q
[Pătraşcu V]	$> n + n / \log^{O(q)} n$	q

Outline

- Randomness extractors
- Data structures
- Pseudorandom generators
- Error-correcting codes

Pseudorandom generator



- Stretch short seed into output that “looks random”
- Uses: Monte Carlo, cryptography, …
- Simple yet unexplored connection to sampling: only care about output distribution

Pseudorandom generators

Type	Looks random to:
In practice	?
Cryptographic	Efficient test Based on unproven assumptions
Unconditional	Small-depth [Nisan] [V] Central limit [DGJSV] Polynomials [Bogdanov V] [Lovett] [V] ...

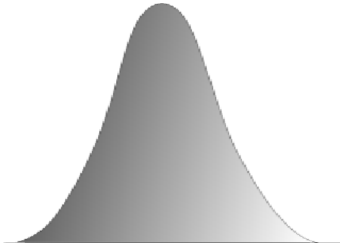
Pseudorandom generators

Type	Looks random to:
In practice	?
Cryptographic	Efficient test Based on unproven assumptions
Unconditional **Next**	Small-depth [Nisan] [V] Central limit [DGJSV] Polynomials [Bogdanov V] [Lovett] [V] ...

[Diakonikolas Gopalan Jaiswal Servedio V FOCS '09, SIAM J. Comp.]

- Central-limit theorem:

$x_1, x_2, \ldots, x_n$ independent $\Rightarrow \sum x_i \approx$ normal



- Bounded-independence Central limit Theorem:

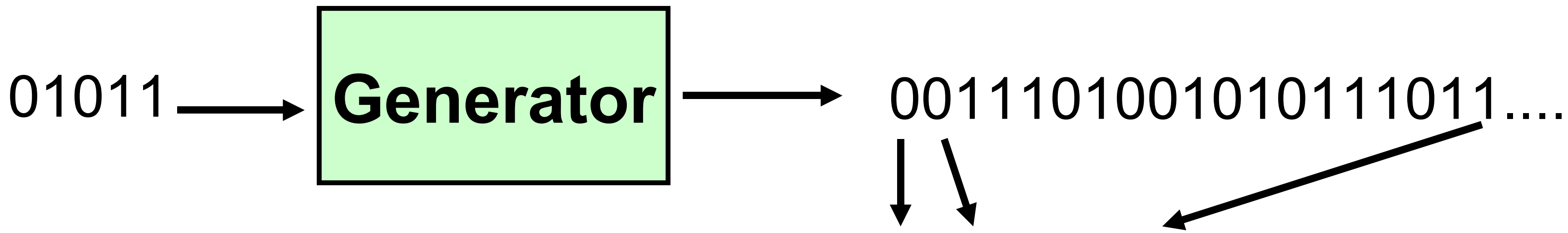
$x_1, x_2, \ldots, x_n$ k-wise independent $\Rightarrow \sum x_i \approx$ normal

$$\forall t \quad \left| \Pr[\sum x_i < t] - \Pr[\text{normal} < t] \right| < 1/\sqrt{k}$$

[Bogdanov V FOCS 2007 Special Issue, SIAM J. Comp.]

[V 2008. Best Paper Award, J. Comp. Complexity]

- Theorem:
  Pseudorandom generator for low-degree polynomials



$x_1 \cdot x_2 + x_{19}$

- Open for 15 years

- Led to progress on Gowers' norm [Green Tao]

Proof idea

- For degree d:

  Let L look random to degree 1 [Naor Naor]

  bit-wise XOR d independent copies of L:

  Generator := $L^1 + \ldots + L^d$

Proof idea

- Induction: Assume for degree d,
  prove for degree-(d+1) polynomial p

  Inductive step: Case-analysis based on

  Bias(p) := $| \text{Prob}_{\text{uniform } X} [p(X)=1] - \text{Prob}_{\text{uniform } X} [p(X)=0] |$

- Bias(p) small $\Rightarrow$ Pseudorandom bias small
  use expander graph given by extra generator

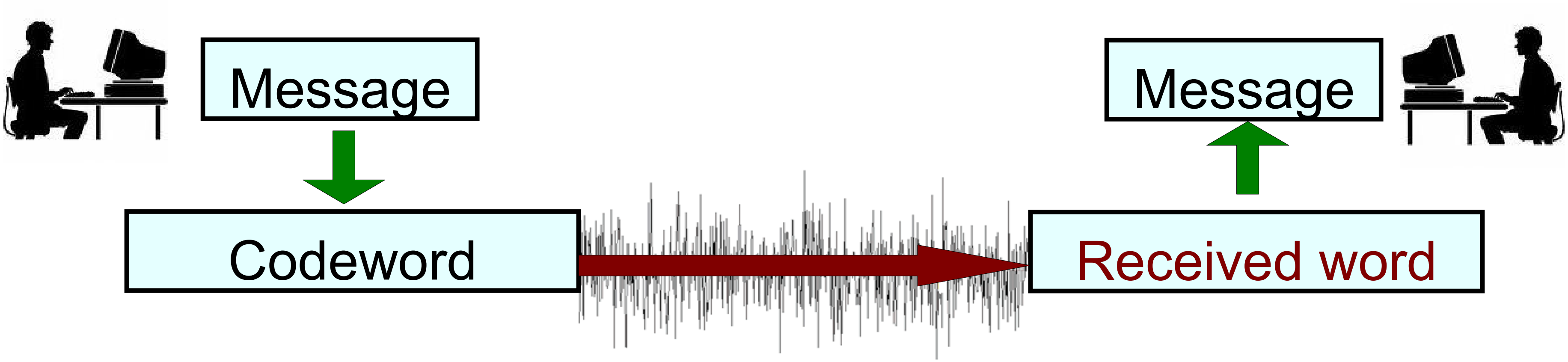
- Bias(p) large $\Rightarrow$
  (1) self-correct: p close to degree-d polynomial
  This result used in [Green Tao]
  (2) apply induction

Outline

- Randomness extractors
- Data structures
- Pseudorandom generators
- Error-correcting codes

Error-correcting codes

- To communicate over noisy channel



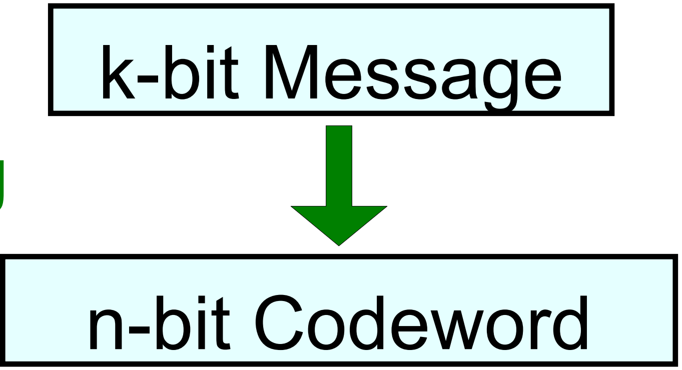
- Need compact, fast, low-energy codes for:

Portable communication electronics

Micro/nano systems

Error-correction within chips

Codes, parameters



- Focus on complexity of encoding

- Asymptotically good:

  code length n = O(k) (rate $\Omega(1)$)

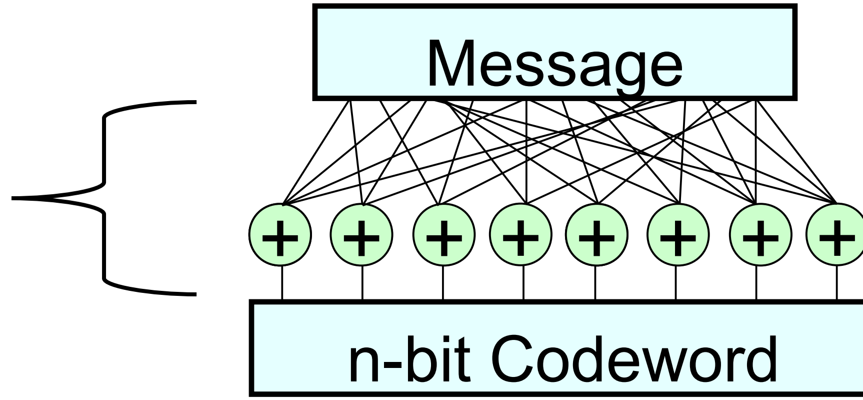
  minimum distance $\Omega(n)$ ($\Omega(n)$ adversarial errors)

- Alphabet = {0,1}

Previous codes

- Linear codes

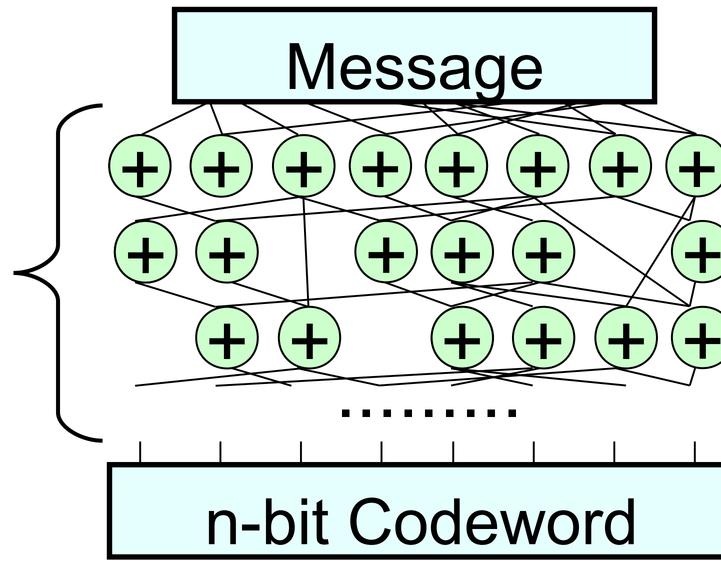
  Wires $O(n^2)$ Depth 1



- [Spielman 95]

  Wires $O(n)$ Depth $O(\log n)$

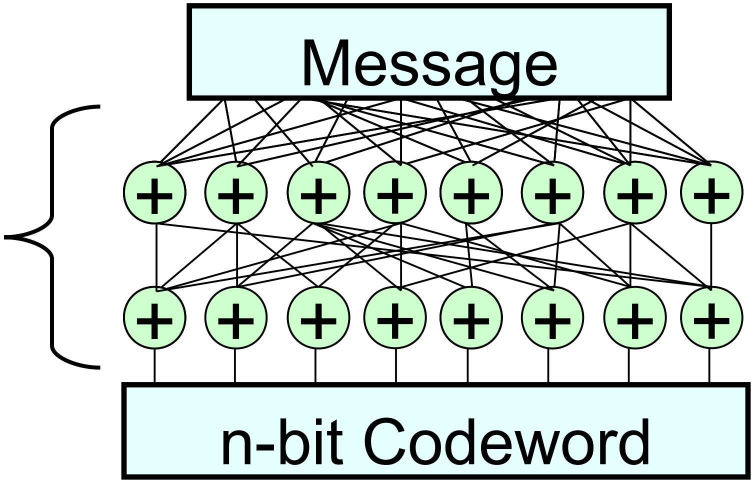
  (fan-in 2)



- Can we have both wires ≈ n and depth $O(1)$?

Our results
[Gal Hansen Koucky Pudlak V; 2011]

- Wires $\Theta(n \log^2 n)$ Depth 2



- Also new circuit lower bound beating $\Omega(n \log^{3/2} n)$

[Pudlak Rodl '96]

- Open prob: explicit construction, efficient decoding, ...

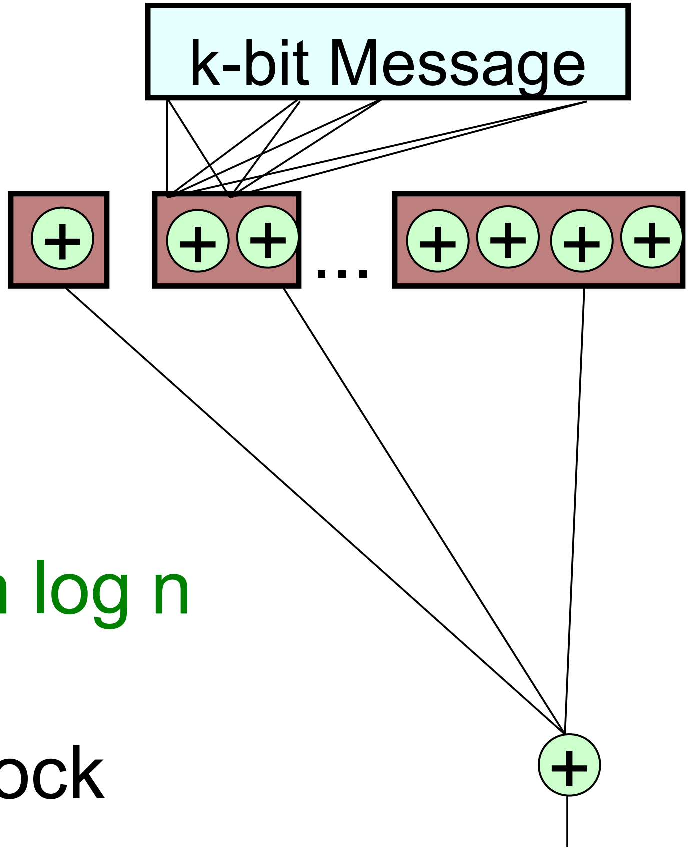
Proof idea

- Just sample uniform bit from message weight $w > 0$

- i-th middle block ($i < \log k$)
  Balanced if $w = \Theta(k/2^i)$

  Each gate k / w wires to "hit"
  $\log \binom{k}{w}$ gates to union bound
  Wires in block: $(k/w) \log \binom{k}{w} < n \log n$

- Each output: XOR one bit per block



Conclusion

New paradigm: Sample, not compute

- Randomness extractor — Circuit sources
- Data structure — Storing trits, prefix sums
- Pseudorandom generator — Central limit; Polynomials
- Error-correcting code — Quasi-linear size, depth 2
- Many new directions and open problems!

- ∑∏√ ∩∪ ⊃⊇⊄⊂⊆∨∧
- ≥≤ ∀∃ ΩΘω αβεγδ
- →⇓⇒⇑⇐⇔
- ≠≈
- Θω
- ∈ ∉
- ±
- ∑∏√∩∉ ∪⊃⊇⊄⊂⊆∈⇓⇒⇑⇐⇔∨∧≥≤∀∃Ωαβεγδ→
- ≠≈TAΘ
  Recall: edit style changes ALL settings.
- Click on “line” for just the one you highlight

Proof outline

- Circuit source

↓ [Lovett V]

- local source Y = f(X)   Each output bit of f(X) depends on few input bits

↓ **NEXT SLIDE**

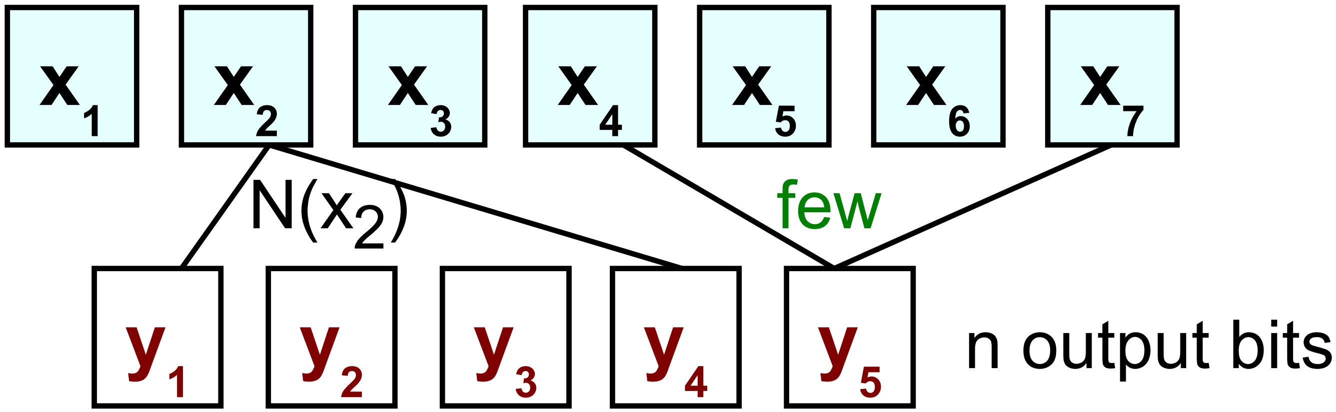
- Bit-source $Y = \quad Y_1 \; 0 \; Y_2 \; 0 \; Y_2 \; 1 \; 1 \; Y_3 \; Y_1 \; 0$

  $\Pr[Y_i = 1] = ½$

↓ Previous extractors

- Uniform

Local → bit source



- Entropy Y high $\Rightarrow \exists\, \mathbf{y}_i$ with high variance (~unbiased)
- Locality + Isoperimetry $\Rightarrow \exists\, \mathbf{x}_j$ with high influence
- Set uniformly $N(N(\mathbf{x}_j)) \setminus \{\mathbf{x}_j\}$ (N(v) = neighbors of v)

  with high prob. $N(\mathbf{x}_j)$ non-constant, depends on $\mathbf{x}_j$ only

  $\Rightarrow$ bit-source
- Repeat

♦