Reliable Memories From Unreliable Components: Theory and Connections With Codes On Graphs

Bane Vasić [1], **Predrag Ivaniš** [2]

[1] *Department of ECE, University of Arizona, Tucson*

[2] *School of Electrical Engineering, University of Belgrade*

E-mail: [1]vasic@ece.arizona.edu, [2]predrag.ivanis@etf.rs

Abstract

In this talk we introduce fault-tolerant memory architecture based on low-density parity check codes and iterative decoders. We also present a theoretical analysis of decoders failures.

Key words: Low-density parity check codes, Memory architectures, Unreliable components

Synopsis

Increased integration factor of integrated circuits coupled with stringent energy-efficiency constraints necessities a new design paradigm for Very Large Scale Integration (VLSI) technologies in which fully reliable operation of hardware components is not guaranteed. In this presentation we discuss an error control coding (ECC) as a method for ensuring fault-tolerance of systems build of unreliable hardware with a special focus on fault-tolerant memories. This approach, in contrast to the widely used von Neumann's triple modular redundancy, was introduced in the late sixties and early seventies by Taylor [1] and Kuznetsov [2], while the equivalence between Gallager B decoder built from unreliable logic gates and Taylor-Kuznetsov fault-tolerant memory architectures was first observed by our research group in [3] and [4], and further developed in [5] into a theoretical framework for analysis and design of faulty decoders of low-density parity check (LDPC) codes.

We start by introducing physical reasons for semiconductor devices failures. These reasons depend on the technology used but can be broadly divided into: permanent, intermittent and transient. We focus on the third type or faults, transient faults (TFs), which also referred to as soft errors and are mainly due to single or multiple event upsets or timing errors. These errors have probabilistic behavior and can be described statistically.

After that, we discuss Taylor's proof that a memory built from unreliable components can achieve storage capacity *C* for all memory redundancies greater that 1/C. In the Taylor-Kuznetsov (TK) model, the information is stored in a coded form, i.e., the stored vector is a codeword of some *(n, k)* block ECC. The memory in which the bits are stored is also assumed to be unreliable. It is connected to a correcting circuit, which periodically updates the memory using given decoding scheme. The redundancy necessary to ensure memory reliability grows linearly with the memory size. A memory failure occurs if the error pattern in the memory is uncorrectable by a perfect decoder in some maximum number of iterations.

In TK memory architecture the user information is first en-coded by a regular binary LDPC code of length *n* and dimension *k*. The stored codeword, denoted as $\boldsymbol{x}=(x_1,x_2,...,x_n)$, consist of *n* variable bits x_i, (i=1,...,n) which are involved in exactly *J* parity-check equations.. The *j*-th component of the vector $c=xH^T$, called a syndrome, corresponds to the value of *j*-th parity-check sum, and can be satisfied if $c_j=0$ and unsatisfied if $c_j=1$. After encoding, the *J* identical copies of every coded bit x_i, $\{x_i^{(1)}, x_i^{(1)}, ... , x_i^{(J)}\}$ are stored in *J* registers. Registers are unreliable. In each iteration, the estimates of each of these copies are obtained by using one combination of *J-1* checks by the following steps: (i) calculating the parity checks for each bit-copy (exclude one distinct parity check from the original set of check for each bit-copy), (ii) flipping the value of the bit-copy if majority of the parity checks are unsatisfied. The decision element in this case is a majority logic gate whose output is *1* if half or more of the parity checks are non-zero. We present a theoretical analysis of faulty decoders

Acknowledgment

This work was supported by the Seventh Framework Pro-gramme of the European Union, under Grant Agreement number 309129 (i-RISC project). It is also funded in part by the NSF under grants CCF-0963726 and CCF-1314147.

References

[1] M. Taylor, "Reliable information storage in memories designed from unreliable components," Bell System Technical Journal, 47, pp. 2299-2337, 1968.

[2] A. Kuznetsov, "Information storage in a memory assembled from unreliable components," Problems of Information Transmission, 9, pp. 254-264, 1973.

[3] B. Vasic, S. K. Chilappagari, S. Sankaranarayanan, and R. Radhakrishnan, "Failures of the Gallager B decoder: analysis and applications," in Proceedings of 2nd Information Theory and Applications Workshop (ITA 2006), San Diego, CA, Feb. 2006, paper 160, [Online Available:] http://ita.ucsd.edu/workshop/06/papers/160.pdf

[4] S. K. Chilappagari, B. Vasic, "Fault tolerant memories based on expander graphs," in Proc. of IEEE Information Theory Workshop, pp. 126–131, Tahoe City, CA, USA, Sep. 2–6 2007..

[5] B. Vasic and S. K. Chilappagari, "An information theoretical framework for analysis and design of nanoscale fault-tolerant memories based on low-density parity-check codes," IEEE Transactions on Circuits and Systems I, Regular Papers, vol. 54, no. 11, pp. 2438-2446, Nov. 2007.

Reliable Memories
Made of (Only!) Faulty Logic Gates

Bane Vasić

Department of Electrical and Computer Engineering
Department of Mathematics
University of Arizona, Tucson
USA

Predrag Ivaniš
Faculty of Electrical Engineering
University of Belgrade
Serbia

Supported by the NSF under Grant CCF-0963726 and CCF-1314147 and the Seventh Framework Programme of the European Union, under Grant Agreement number 309129 (i-RISC project)

THE UNIVERSITY OF ARIZONA®
TUCSON ARIZONA

Collaborators

- David Declercq, ENSEA
- Valentin Savin, CEA
- Elsa Dupraz, ENSEA
- Srdjan Brkić, ETF Belgrade
- Omran Al Rasheed, ETF Belgrade
- Alexander Kuznetsov, Seagate Research
- Erozan Kurtas, Standard & Poor’s
- Shashi Kiran Chilappagari, Marvell Semiconductor
- Milos Ivkovic, Cornell
- Rathnakumar Radhakrishnan, Marvell Semiconductor
- Anantha Raman Krishnan, Western Digital
- Dariush Divsalar, Fabrizio Polara, Ken Andrews, NASA-JPL
- Misha Chertkov, Los Alamos National Laboratory

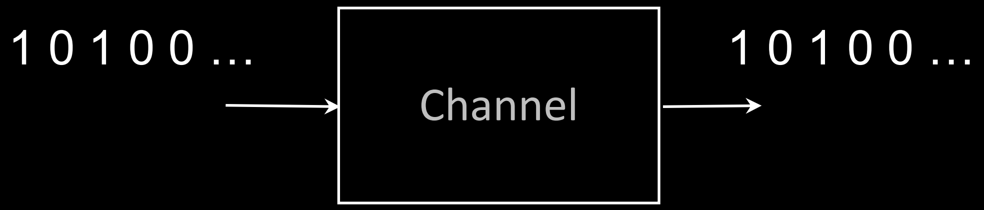
Outline

- Prologue:
  - Basics of iterative decoding and low-density parity check (LDPC) codes

- Drama - fault-tolerant memories
  - Motivation
  - History
  - Fault -tolerant architecture based on expander LDPC codes

- Epilogue:
  - Guaranteed error correction by LDPC codes

Information transmission

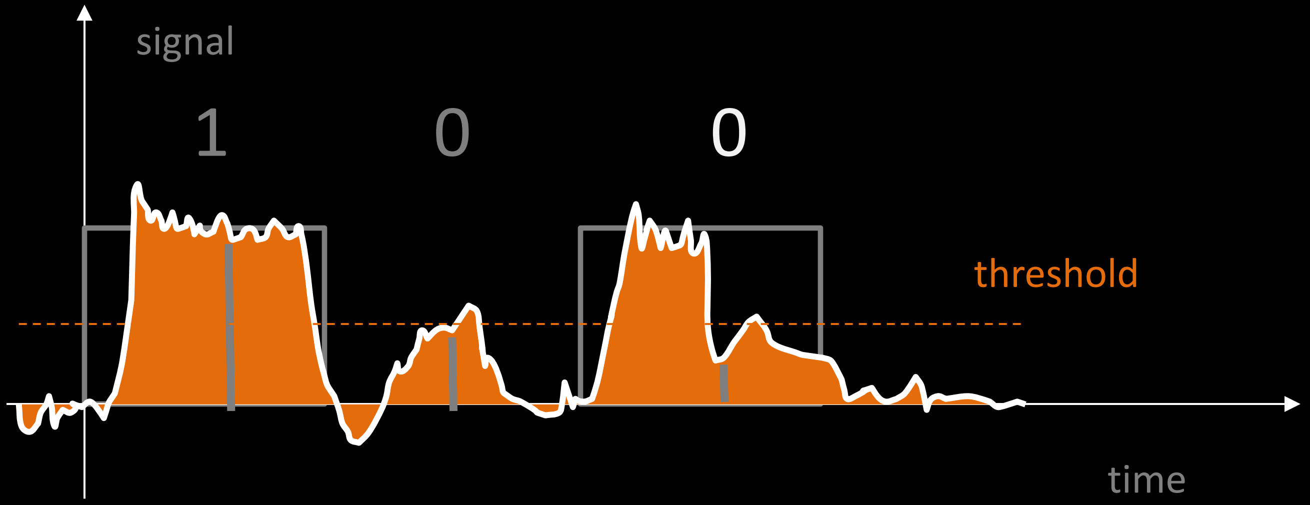


1 0 1 0 0 …

Channel

1 0 1 0 0 …

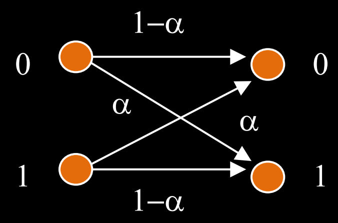
Information transmission



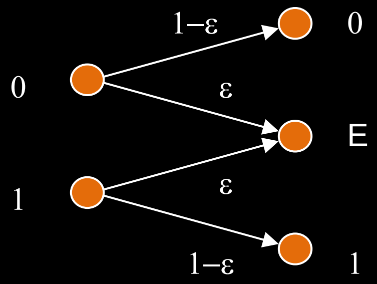
THE UNIVERSITY OF ARIZONA.
TUCSON ARIZONA

Simple memoryless channels

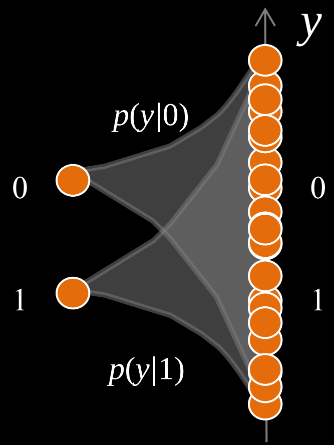
- Binary symmetric channel (BSC)



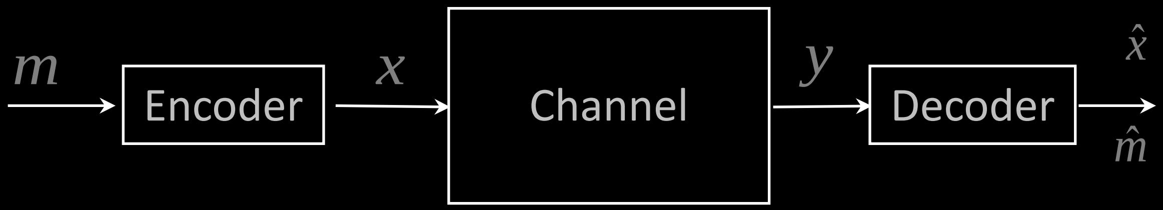
- Binary erasure channel (BEC)



- Binary input additive white Gaussian noise (AWGN) channel, σ^2

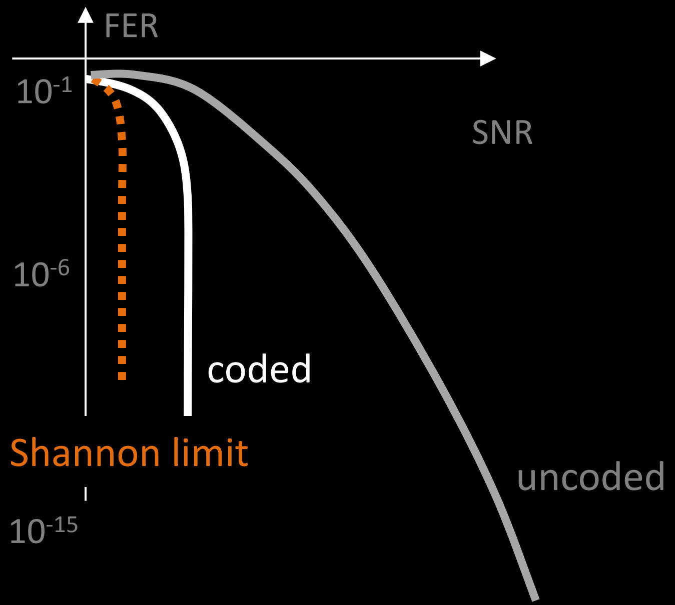


Communication channels

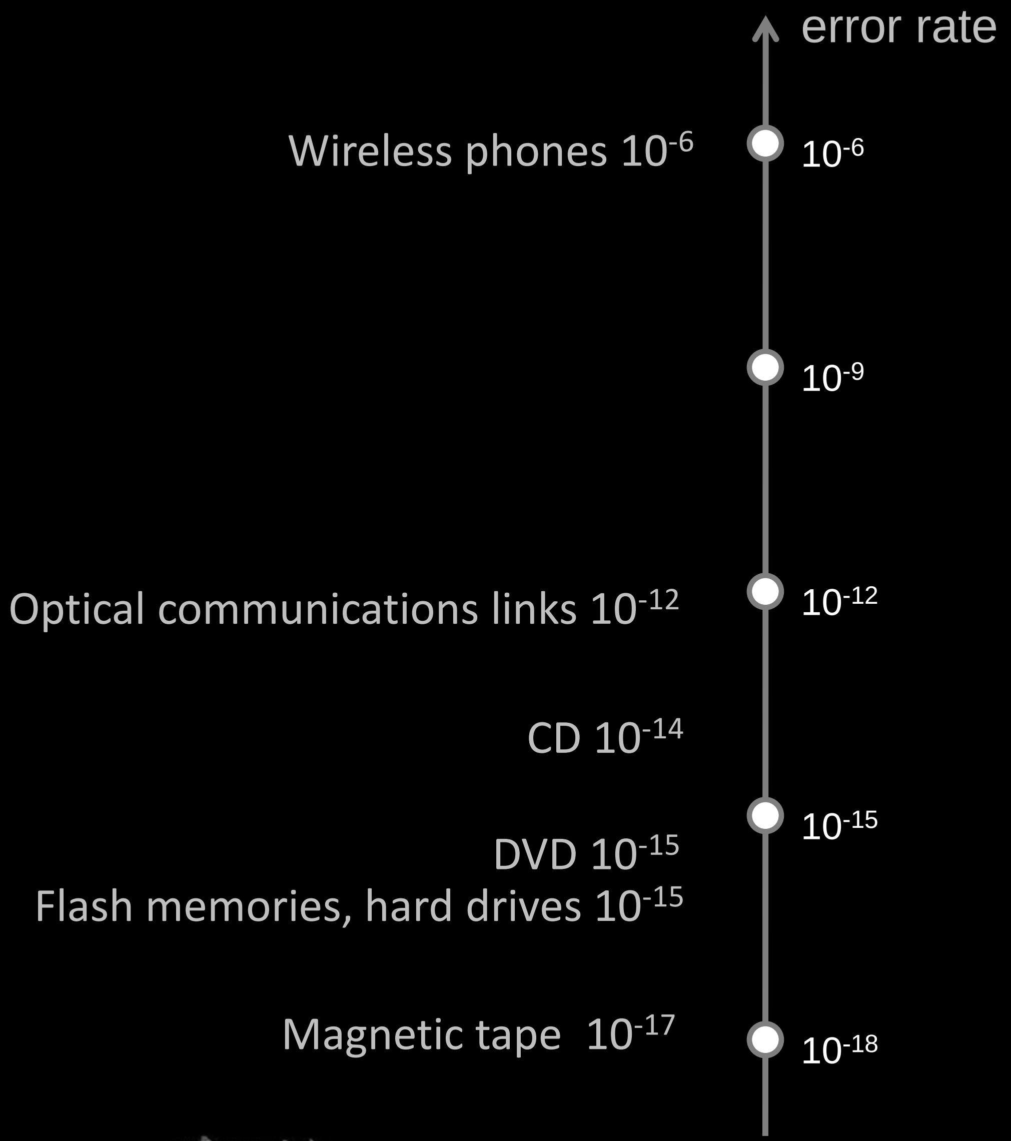


- Decoder tries to find x (or m) from y so that the probability of bit/codeword error is minimal
- In other words, decoder tries to find a codeword closest to y.

Error rate performance



Error rates in communications systems



error rate

Wireless phones 10^{-6} — 10^{-6}

10^{-9}

Optical communications links 10^{-12} — 10^{-12}

CD 10^{-14}

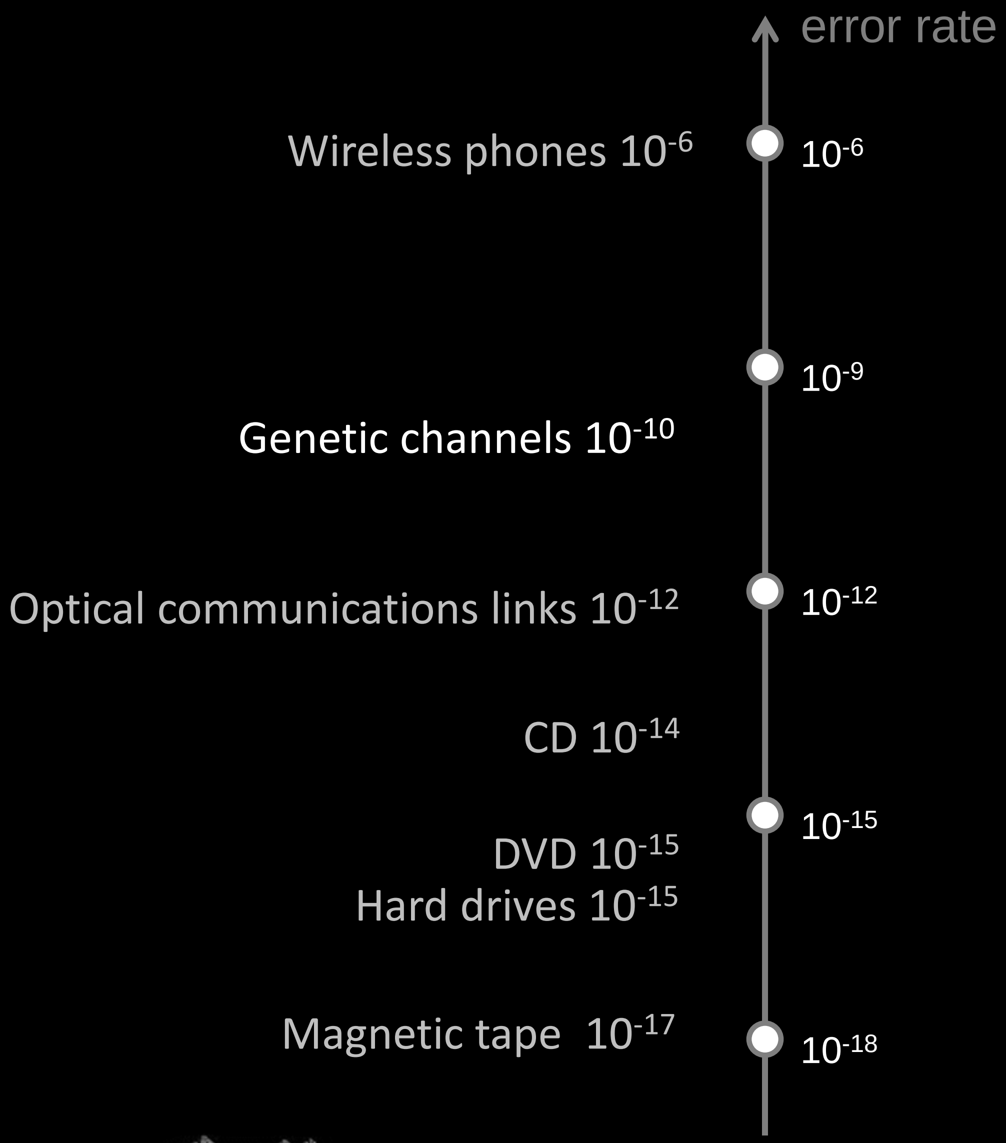
10^{-15}

DVD 10^{-15}

Flash memories, hard drives 10^{-15}

Magnetic tape 10^{-17} — 10^{-18}

And genetic channels



error rate

Wireless phones 10^{-6}

10^{-6}

10^{-9}

Genetic channels 10^{-10}

Optical communications links 10^{-12}

10^{-12}

CD 10^{-14}

10^{-15}

DVD 10^{-15}
Hard drives 10^{-15}

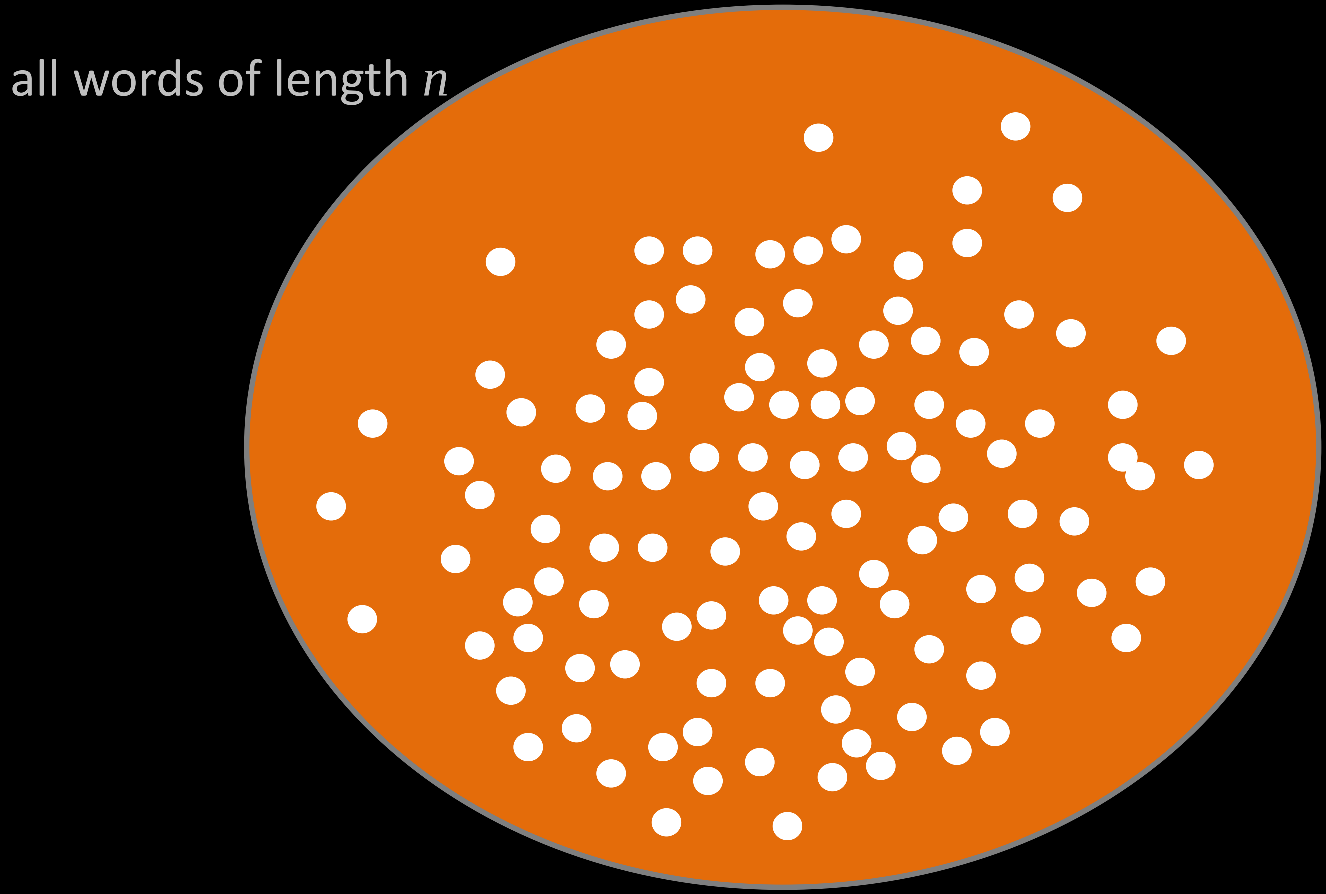
Magnetic tape 10^{-17}

10^{-18}

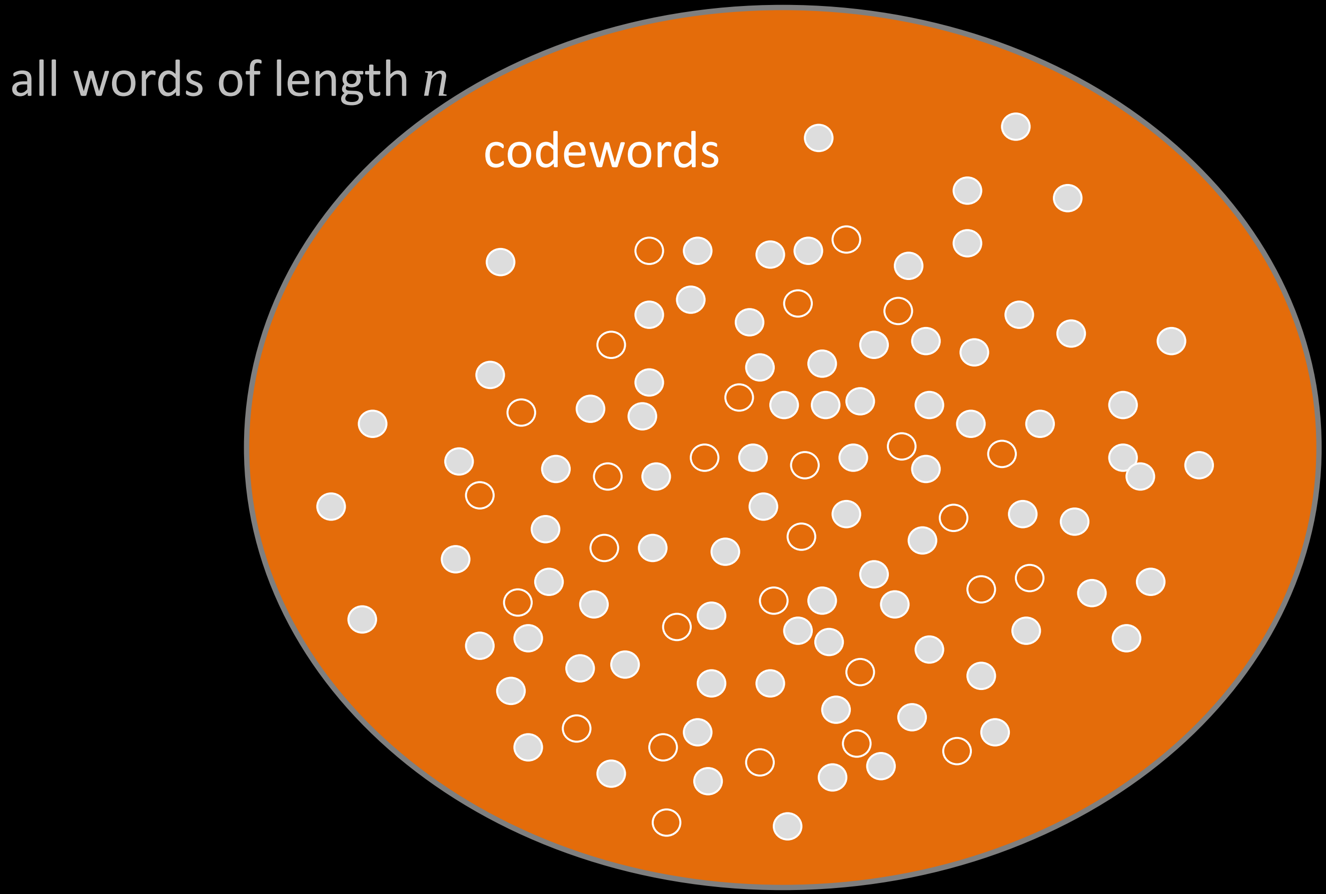
Global Power Consumption

- 200 Mega Watt per Hour worldwide for data error correction!
- Data centers only, not counting mobile devices and computers.

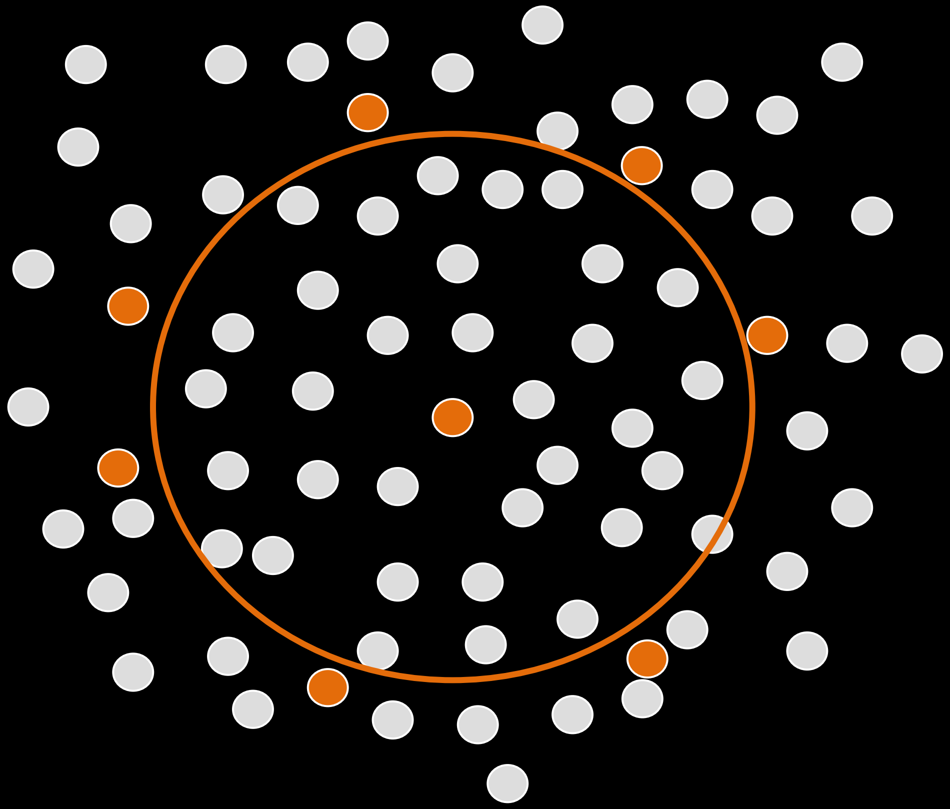
Protecting information by coding



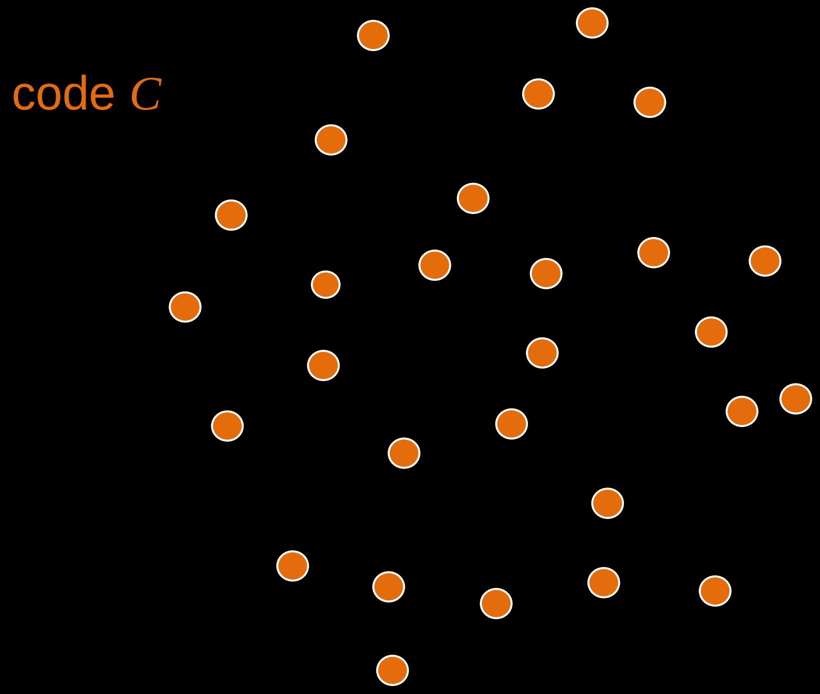
Protecting information by coding



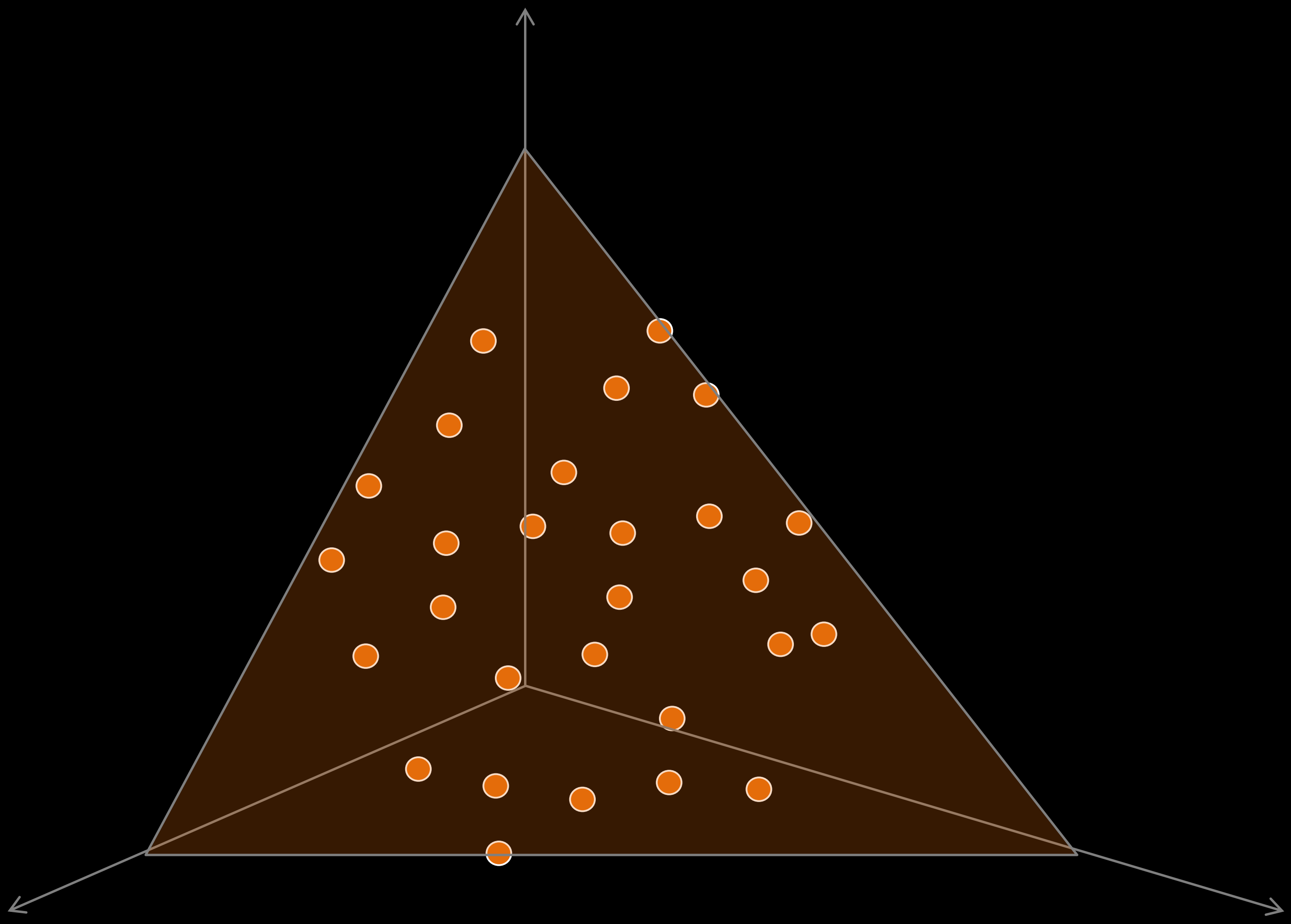
Minimum distance



Protecting information by coding



Linear codes



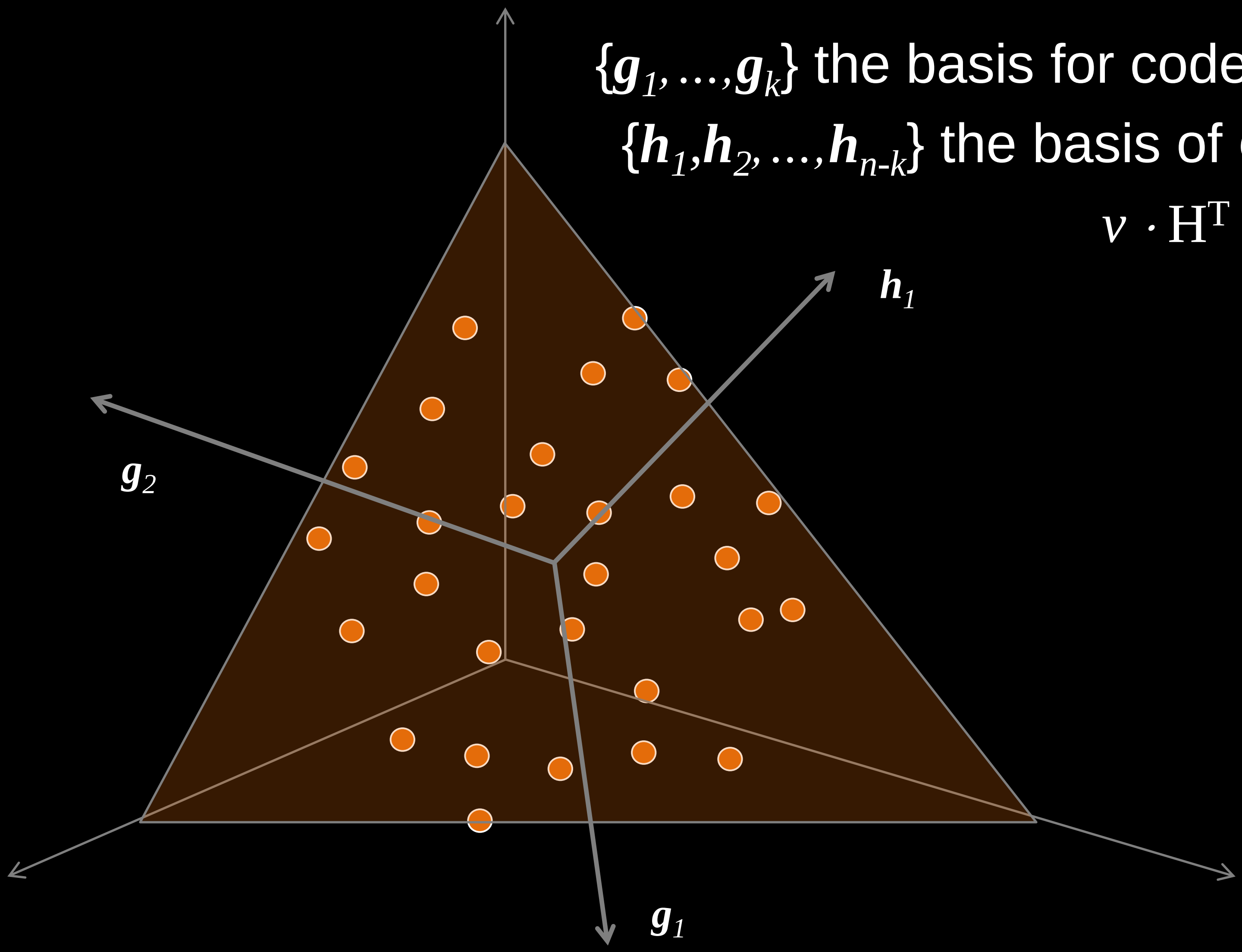
THE UNIVERSITY OF ARIZONA®
TUCSON ARIZONA

Dimension of a linear code

{$\boldsymbol{g}_1, \ldots, \boldsymbol{g}_k$} the basis for code C

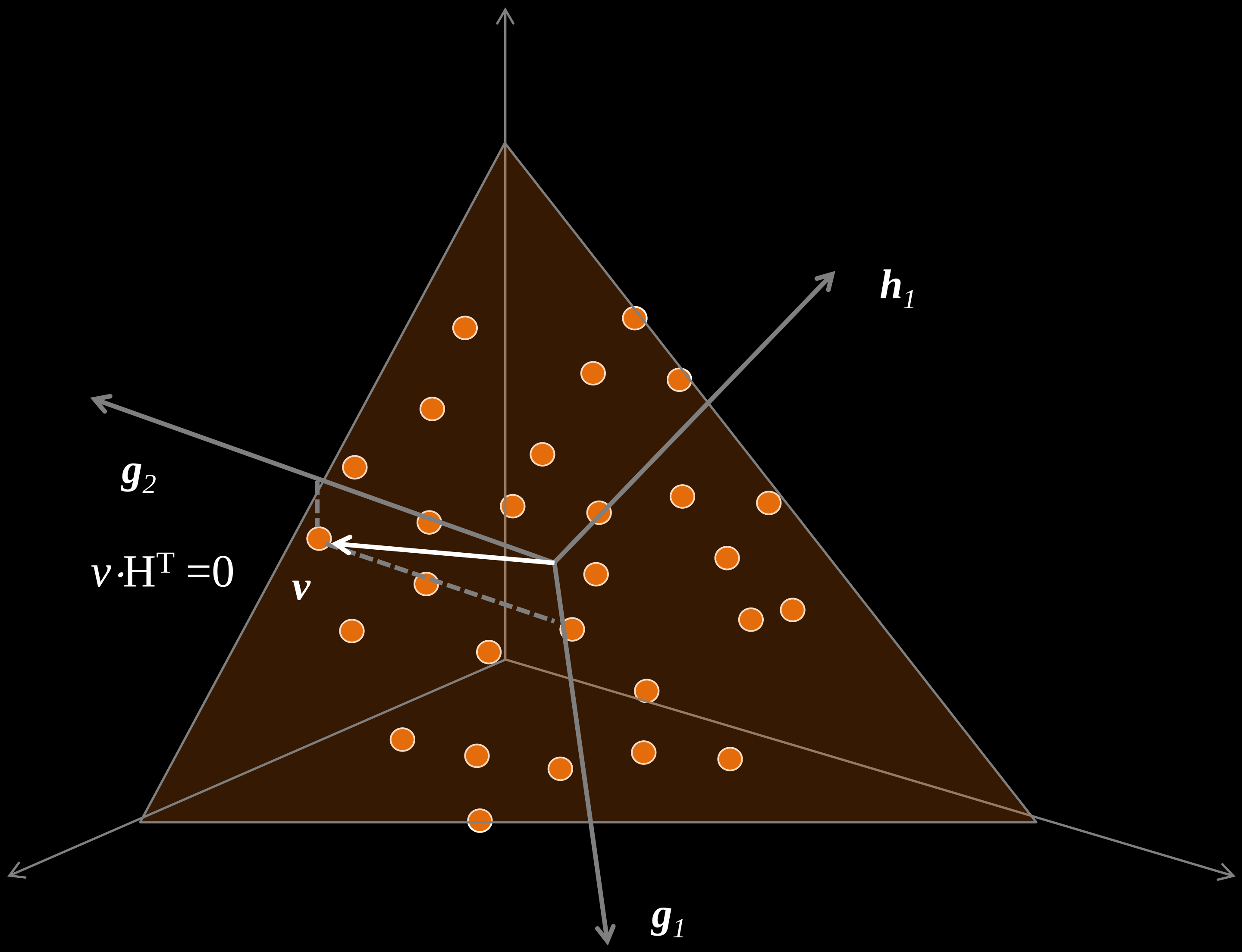
{$\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_{n-k}$} the basis of $C^\perp$

$v \cdot H^T = 0$

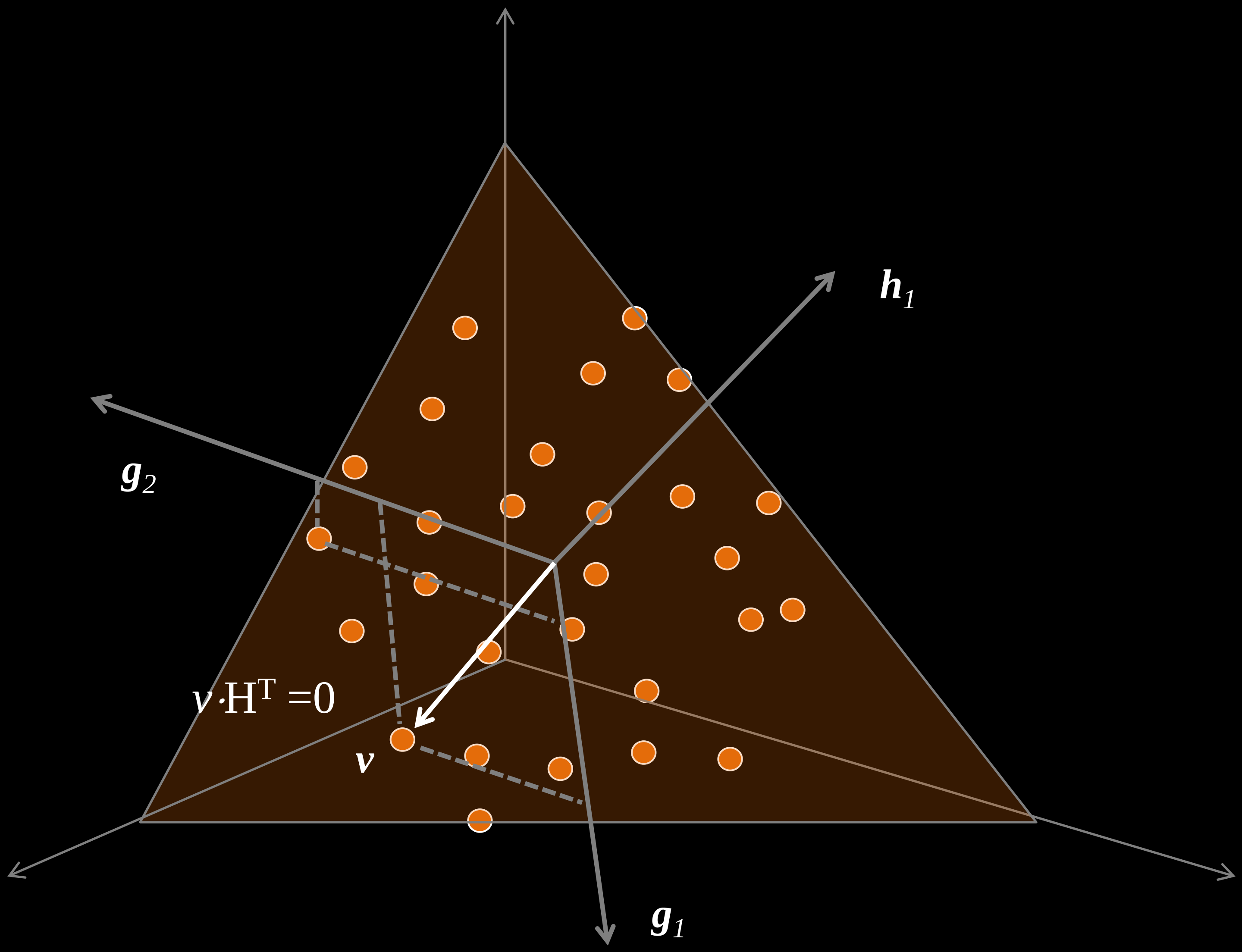


THE UNIVERSITY OF ARIZONA®
TUCSON ARIZONA

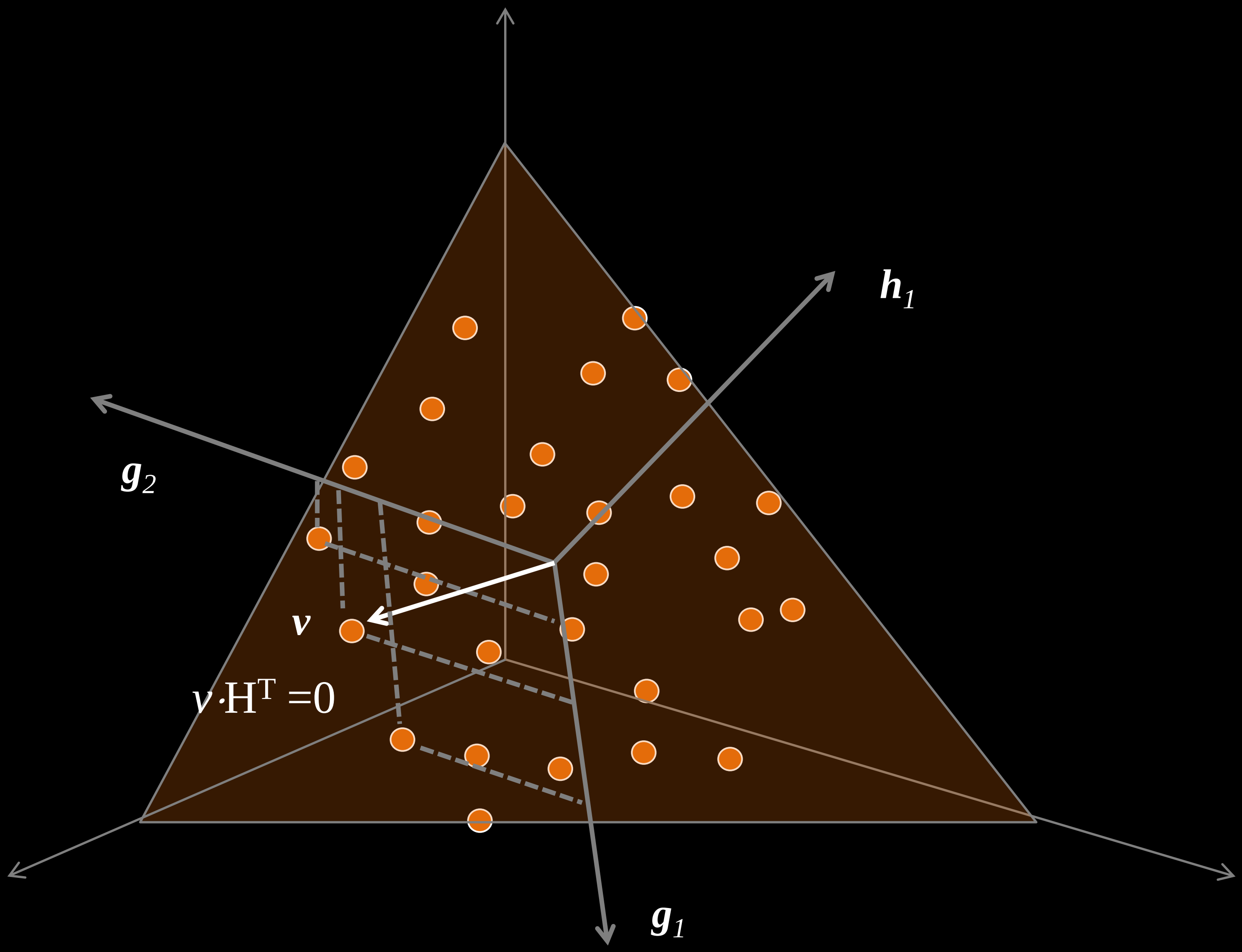
Parity check



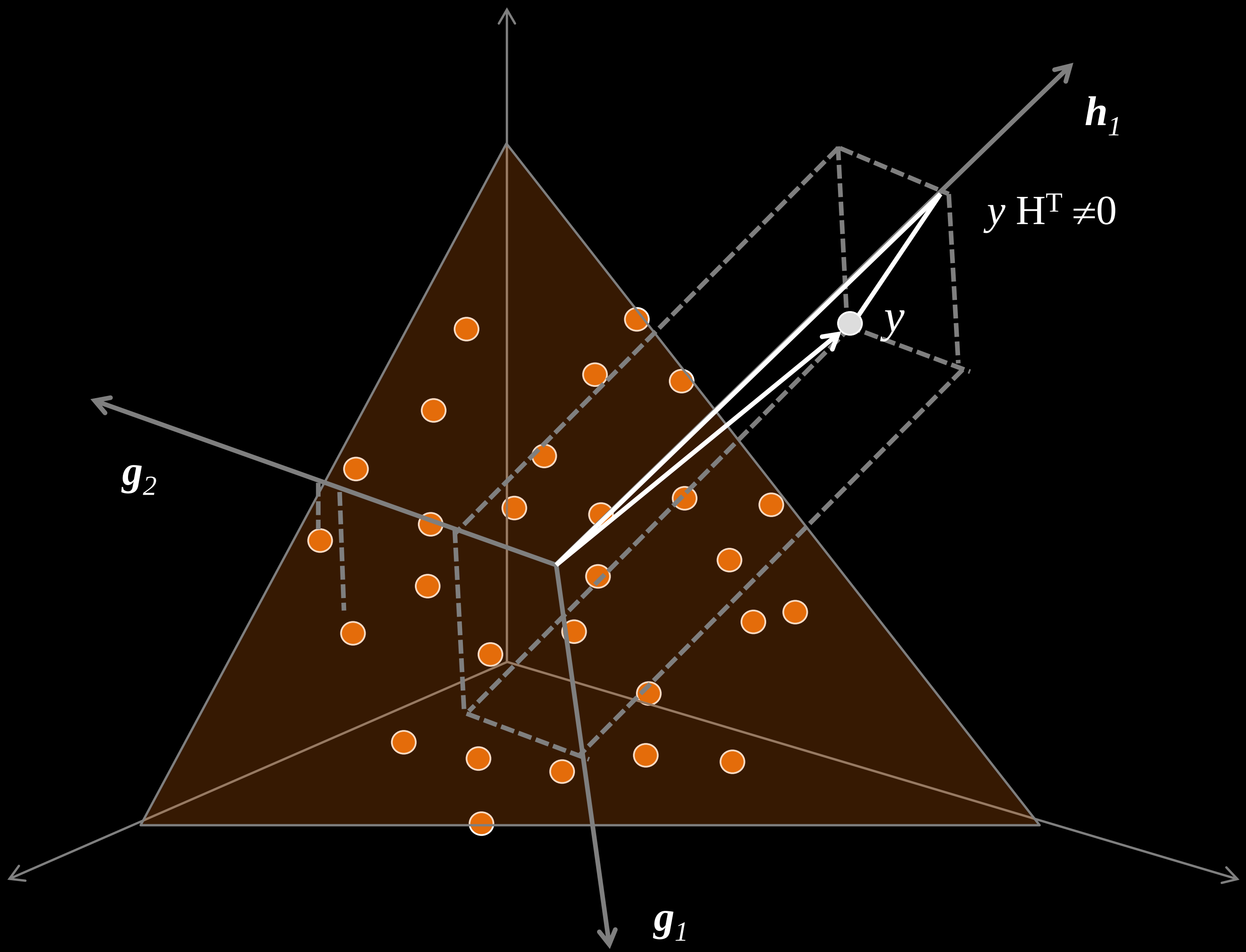
Parity check



Parity check



Syndrome



$\boldsymbol{h}_1$
$y\,\mathrm{H}^T \neq 0$
y
$\boldsymbol{g}_2$
$\boldsymbol{g}_1$

Linear constraints

- A codeword *v* satisfies $v \times H^T = 0$
- n-k equations in n variables
- Example:

$$H = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

$$c_1 : \quad v_1 + v_4 + v_6 + v_7 = 0$$

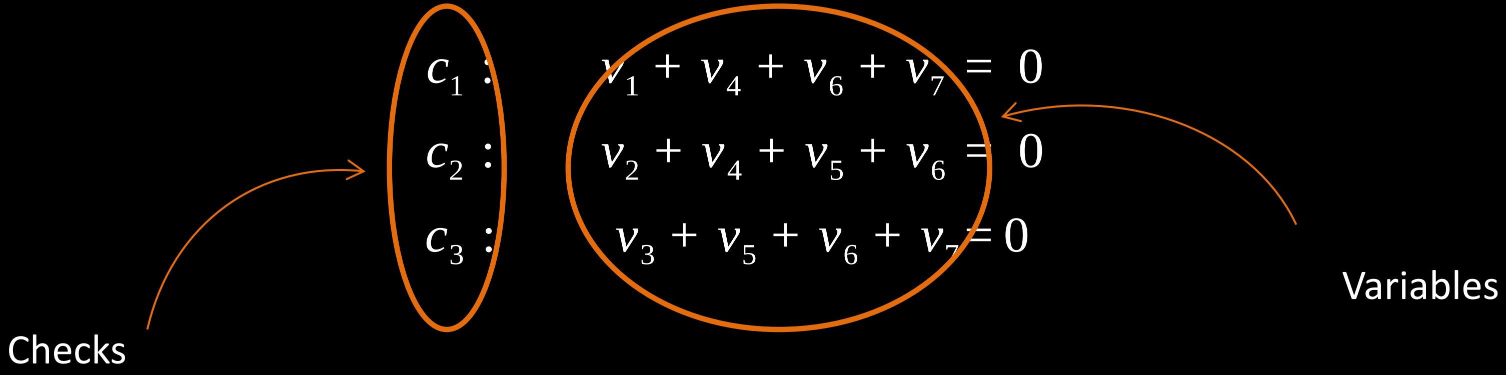
$$c_2 : \quad v_2 + v_4 + v_5 + v_6 = 0$$

$$c_3 : \quad v_3 + v_5 + v_6 + v_7 = 0$$

Graphical model for a linear block code

$$
\begin{aligned}
c_1 &: \quad v_1 + v_4 + v_6 + v_7 = 0 \\
c_2 &: \quad v_2 + v_4 + v_5 + v_6 = 0 \\
c_3 &: \quad v_3 + v_5 + v_6 + v_7 = 0
\end{aligned}
$$

Graphical model for a linear block code



$$
\begin{aligned}
c_1 &: \quad v_1 + v_4 + v_6 + v_7 = 0 \\
c_2 &: \quad v_2 + v_4 + v_5 + v_6 = 0 \\
c_3 &: \quad v_3 + v_5 + v_6 + v_7 = 0
\end{aligned}
$$

Checks

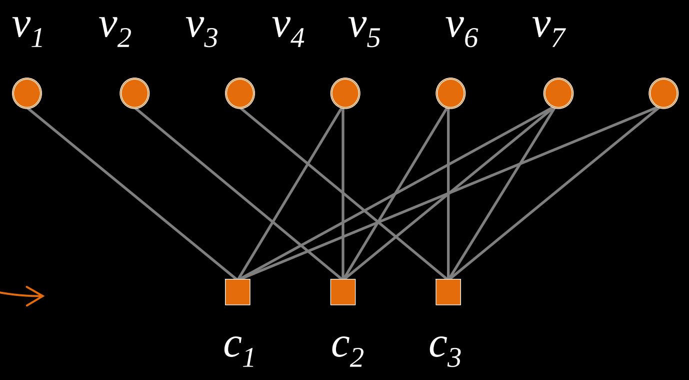
Variables

Graphical model for a linear block code

$$
\begin{aligned}
c_1 &: \quad v_1 + v_4 + v_6 + v_7 = 0 \\
c_2 &: \quad v_2 + v_4 + v_5 + v_6 = 0 \\
c_3 &: \quad v_3 + v_5 + v_6 + v_7 = 0
\end{aligned}
$$

Checks

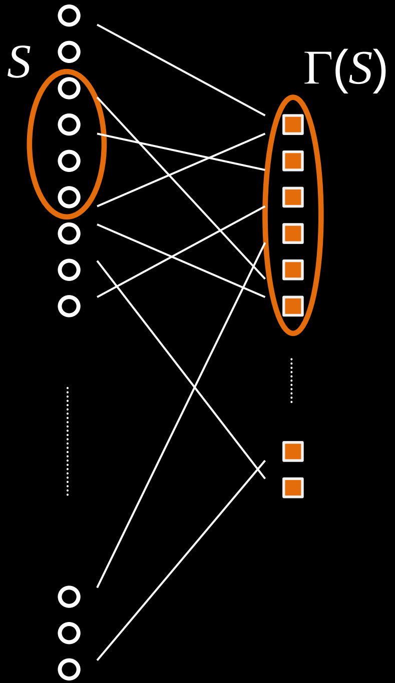
Variables



Expanders

- Definition: A bipartite graph with n variable nodes is called an $(□,\beta)$-expander if for any subset S of the variable nodes of size at most $□n$ the number of (check node) neighbors of S is at least $\beta\, a_S\, |S|$, where a_S is the average degree of the nodes in S.

$$|S| \leq □\ n \ \Rightarrow\ |\Gamma(S)| \geq \beta\, a_S\, |S|$$



- Remark: if there are many edges going out of a subset of message nodes, then there should be many different (unshared) neighbors.

Decoding on graphs

- Two basic types of algorithms:
  - Bit flipping
    - If more checks are unsatisfied than satisfied, flip the bit
    - Continue until all checks are satisfied
  - Message passing
    - A variable node sends his value to all neighboring checks
    - A checks computes XOR of all incoming messages and sends this along the edges
    - But it excludes the message on the edge the result is send along!
    - Variable takes a majority vote of incoming messages and sends this along
    - If tie, sends its original value

Fault-tolerant memories

Motivation

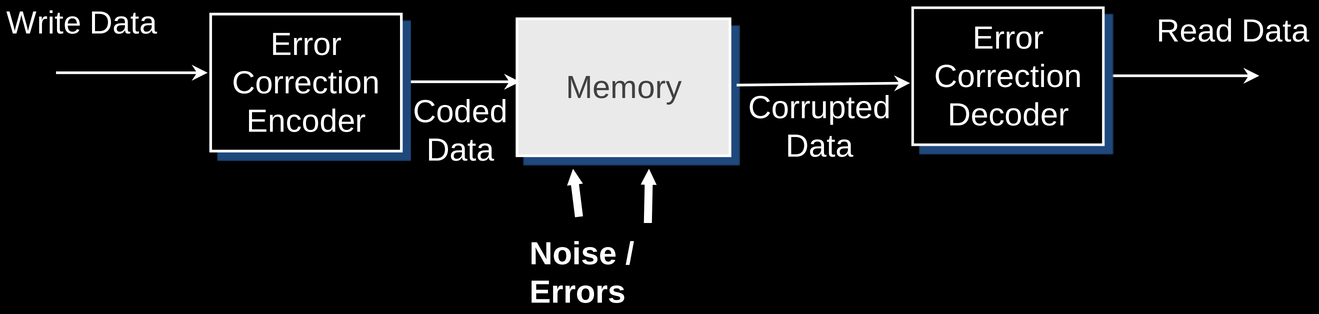
- Space systems
  - Radiation of protons, electrons, alpha particles (trapped radiation belts, solar proton events), galactic cosmic rays, etc.
  - The frequency of *single-event upsets* (SEU) per second due to cosmic ray ions can be as large as 0.033 !
  - Europa-Orbiter, currently scheduled to carry only 150 Mbits of rad-hard memory!
- Air systems
  - Neutrons, the radiation effect maximizes at ~55,000 feet high latitudes.
- Terrestrial systems
  - Low-level background radiation from galactic cosmic rays.
- Emerging nano-scale devices
  - Much less reliable than classical Silicon-based VLSI circuits.
  - *Permanent* failures (defects) and *transient* failures.
- Distributed computing over a network with unreliable links.

Why are Memories Interesting?

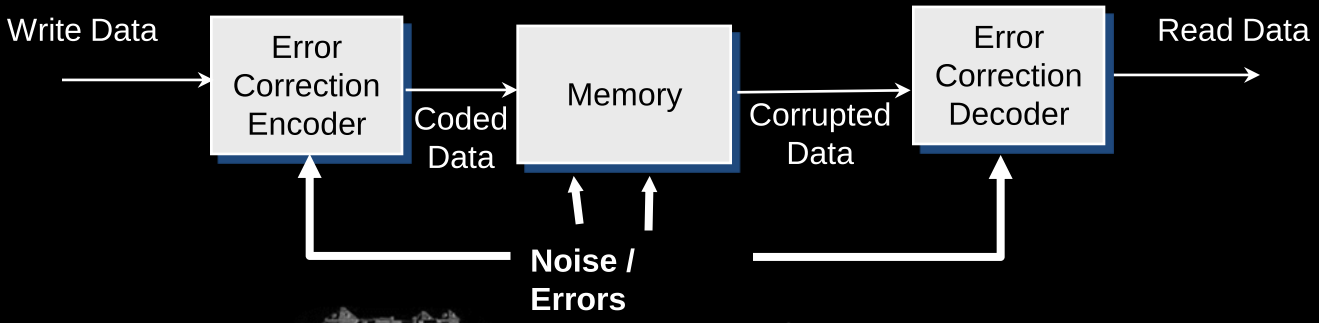
- A place where the information spends most time.
- Decoders also use memory.
- Boolean functions can be reliably computed indirectly using memories.
- The essential component to simulate a Turing machine by cellular automata made of on unreliable components is storing reliably a single bit on an infinite plane.

Faulty Memory Systems

- In a classical setting - full control over error correction encoders and decoders (errors/noise affects only the memory).



- In our setting: *both* the storage elements and logic gates are faulty.
  - The process of error correction not error-free as assumed in classical information theory.



Importance

- Is a reliable memory realizable using only unreliable components?
- We say that reliable information storage is possible in a memory if the probability of memory failure can be made arbitrarily small.
- Stronger encoder and decoder are also more complex, and may not necessarily improve the performance of a system.
- The same question can be asked for computation of Boolean functions.

A Fundamental Open Problem

- Given *n* memory cells and *m* universal logic gates which fail following a known random mechanism, what is the optimal memory architecture which stores the maximum number of information bits for the longest period of time with arbitrary low probability of error?
- The answer strongly depends on failure mechanism.
- We consider *von Neumann failure mechanism* (the elements fail independently with known probability, i.e., the components are reliably unreliable).
  - It is sufficient to assume that only gates and memory elements fail, and that the links are perfect, Dobrushin and Ortyukov (1977).

Two Approaches to Improve Reliability

- (i) von Neumann multiplexing (1952):

  The logic gate resources invested into multiplexing. Build highly redundant reliable networks that simulate the function of universal logic gates, and then use such better gates to build an error correction encoder and decoder.

- (ii) Taylor-Kuznetsov error correction (1965, 1971):

  Logic gate resources invested into building more powerful error correcting code (i.e., decoder) capable of handling both memory elements as well as logic gates errors.

J. von Neumann, "Probabilistic logics and the synthesis of reliable organisms from unreliable components," In C. E. Shannon and J. McCarthy, Eds. *Automata Studies*. Princeton: Princeton University Press, pp. 43-98, 1956.

M. Taylor, "Reliable Information Storage in Memories Designed from Unreliable Components", *Bell System Technical Journal*, 47, pp 2299-2337, 1968.

A. Kuznetsov, "Information Storage in a Memory Assembled from Unreliable Components," *Problems of Information Transmission,* 9, pp. 254-264, 1973.

Assumptions

- The information is always kept in coded form.
- If the message bits are extracted by faulty gates, then the probability of the message bit error is determined by these faulty gates, and cannot be made arbitrary small.
- Non-zero capacity is possible only until the moment the stored information is kept in a coded form.
- A memory is considered stable if it would be possible to correct errors using a perfect decoder.

A Note on Computational Capacity

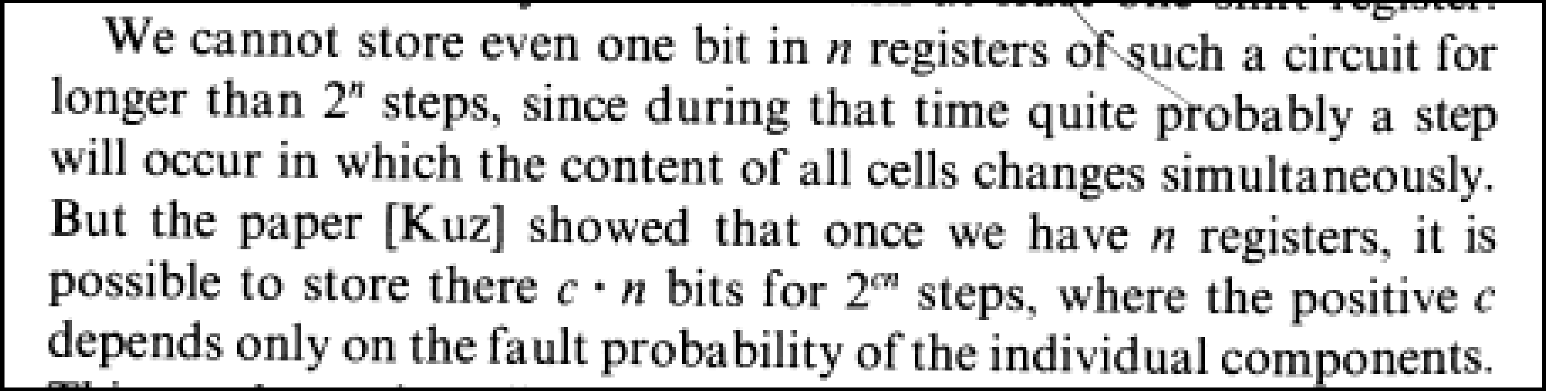
- Memory redundancy is related to *computational capacity* of Boolean functions.
- Difficult to evaluate computational (and storage) capacity (note the similarity with Kolmogorov-Chaitin-Solomonoff computational complexity).
- Pippenger (1988)
  - Proved that general computation with non-zero computational capacity is not possible.

Taylor (1965) - Summary

- Realized that von Neumann multiplexing is a special type of error correction code (repetition code).
- Proved that a memory has an associated information storage capacity, *C*, such that arbitrarily reliable information storage is possible for all memory redundancies greater than 1/*C*.
- Showed that faulty memory systems have *<u>nonzero</u>* (!!!) computational capacity (storage capacity).
- The methodology of the proof, however, does not allow explicit calculation of the storage capacity.
- Proposed construction of fault-tolerant memories based on low-density parity-check (LDPC) codes.
- Considered von Neumann type of error.

Results of Kuznetsov and Other Important Results

- Kuznetsov – many refinements, the results is what we call Taylor-Kuznetsov (TK) scheme.



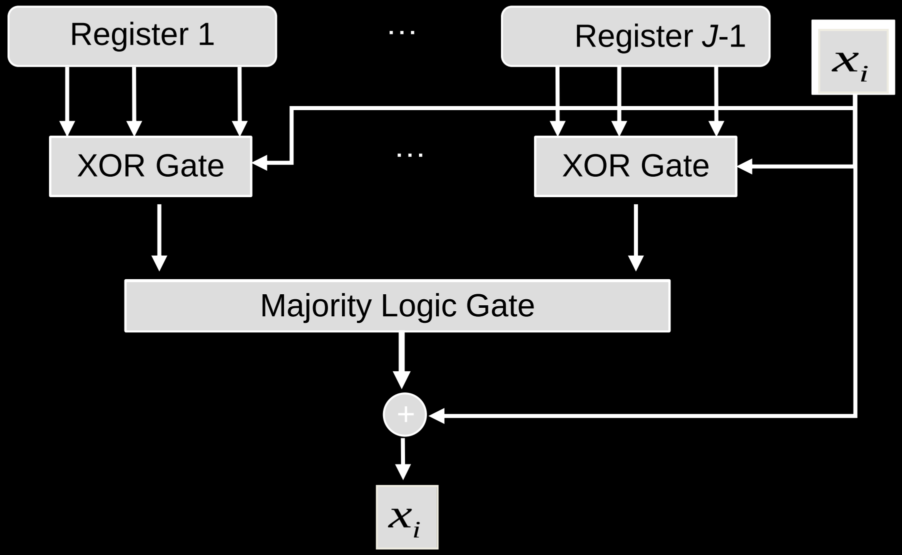
We cannot store even one bit in n registers of such a circuit for longer than 2^n steps, since during that time quite probably a step will occur in which the content of all cells changes simultaneously. But the paper [Kuz] showed that once we have n registers, it is possible to store there $c \cdot n$ bits for 2^{cn} steps, where the positive c depends only on the fault probability of the individual components.

- Hadjicostis - generalized TK scheme to fault tolerant linear finite-state machines.
- Spielman – considered general model of computation, combined von Neumann multiplexing with Reed-Solomon (RS) codes, an architecture with log-linear complexity.

Taylor-Kuznetsov Scheme

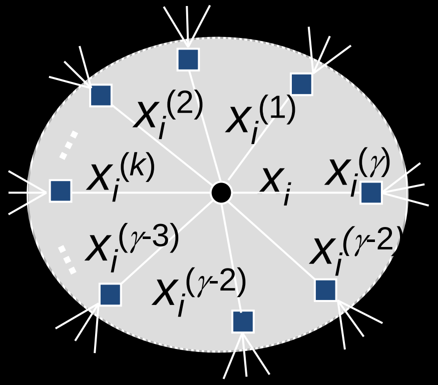
- After encoding, every coded bit x_i is replaced with γ bit-copies of itself $\{x_i^{(1)}, x_i^{(2)}, . . . x_i^{(\gamma)}\}$ stored in γ registers (all bit-copies initially have the same value).

1) Evaluate parity checks for each bit-copy (exclude one distinct parity check from the original set of checks for each bit-copy).
2) Flip the value of a particular bit-copy if half or more of the parity checks are unsatisfied.
3) Iterate 1) and 2).

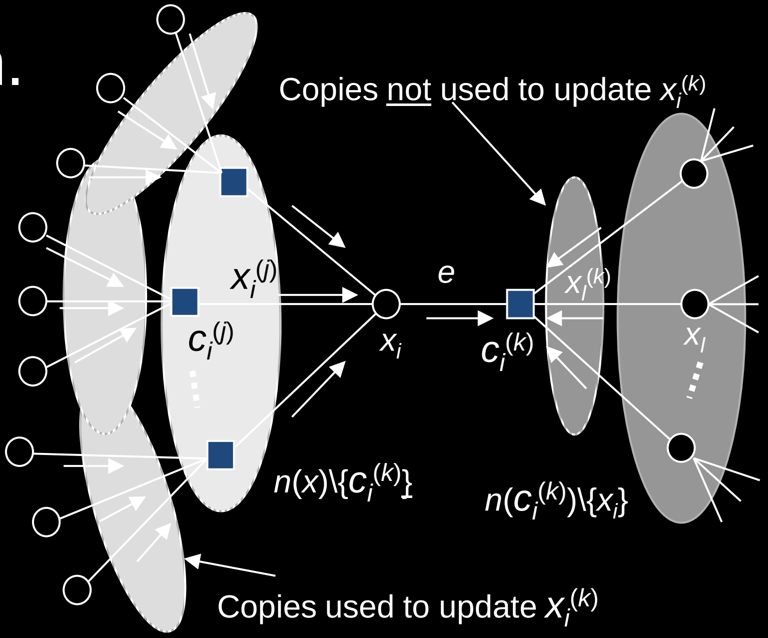


Another Look at Taylor-Kuznetsov Scheme

- Assign the γ identical copies of the variable x to the edges connecting x to check nodes in a Tanner graph of the (γ,ρ) regular LDPC code
- Each bit copy corresponds to the message passed along an edge from variable node to check node.
- TK ⇔ Gallager B algorithm.



$x_i^{(2)}$
$x_i^{(1)}$
$x_i^{(k)}$
x_i
$x_i^{(\gamma)}$
$x_i^{(\gamma\text{-}3)}$
$x_i^{(\gamma\text{-}2)}$
$x_i^{(\gamma\text{-}2)}$



$x_i^{(j)}$
$c_i^{(j)}$
e
x_i
$x_l^{(k)}$
$c_i^{(k)}$
x_l
$n(x)\backslash\{c_i^{(k)}\}$
$n(c_i^{(k)})\backslash\{x_i\}$

Copies not used to update $x_i^{(k)}$

Copies used to update $x_i^{(k)}$

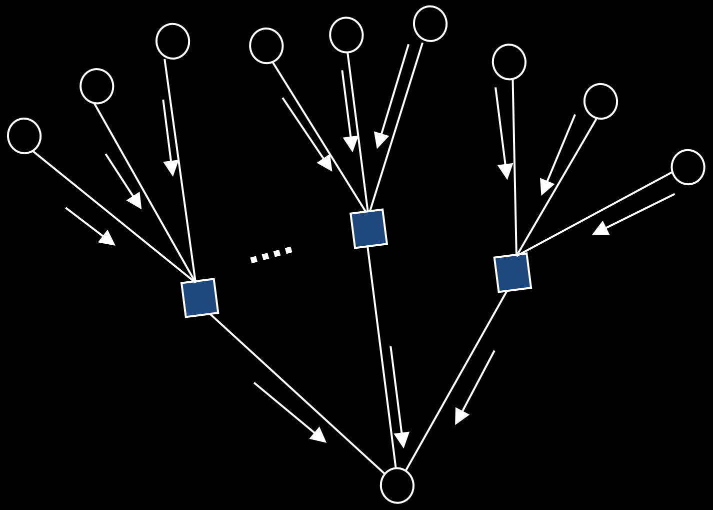
Our Results

- A decoder architecture based on expander codes.
- Lower redundancy compared to the TK scheme.
- Also based on LDPC codes but differs from the TK scheme in the decoding algorithm employed.
- Prove that this architecture achieves exponentially small probability of error for the independent failure model.
  - Only a fixed fraction of the components fail at any given time.
  - Expander arguments to show that a fraction of errors can be corrected.
  - Codes which can correct a fraction of errors achieve exponentially small probability of error on the BSC (extension to the independent failure model using Chernoff bounds).
- Determine the complexity gain over the TK scheme (i.e., bounds on memory complexity).

The Memory Architecture

- A storage circuit stores the bits in coded form in registers.
- At t=0, a codeword from (n,γ,ρ) regular LDPC code is written into the storage circuit.
- A correcting circuit updates the contents of the storage circuit every τ seconds.
- The update proceeds as follows.

1. Each variable node sends its current value to neighboring check nodes.
2. A check node sends the modulo two sum of all incoming messages to a variable node excluding the message it received from that variable node.
3. A variable node updates its value based on majority of the messages received from neighboring checks.



Complexity and Redundancy

- A check node sends an estimate of variable node by XORing the values of other neighboring variable nodes
  - This needs a (ρ-1)- input XOR gate which can be built from (ρ-2) two-input XOR gates.

- A variable node updates its value based on majority of γ estimates it receives from neighboring check nodes
  - This needs a γ-input majority logic gate.

The Failure Model

- Adversarial: at most a fraction of registers/gates can fail.

  α_{m} - fraction of registers that can fail in time τ

  $\alpha_{\oplus}$ - fraction of two-input XOR gate failures

  α_{γ} - fraction of γ –input majority logic gate failures

The Main Theorem

- Let *G* be a (γ, ρ, α, (¾+ε)γ) expander for any ε > 0. The proposed memory architecture can tolerate constant fraction of errors in all the components if

$$a_{\mathrm{m}} + g \times (r - 2) \times a_{\text{Å}} + a_{g} < \frac{1}{2} \times a \times (1 + 4e) \times \boxed{4e}$$

- This 4ε decrease is due to iterative nature of decoder.

Expander Codes

- Defined using a code and an expander graph:
  - For every check node, the neighboring variable nodes are a codeword in some code.
  - LDPC codes are special class, where the code is simply the set of all even weight patterns of length ρ.

- A Tanner graph *G* of a (n,γ,ρ) LDPC code is a $(\gamma, \rho, \alpha, \delta)$ expander if for every subset *S* of at most an αn variable nodes, at least $\delta|S|$ check nodes are incident to *S.*
- Expander codes are a class of asymptotically good error correcting codes.

Parallel Bit Flipping Decoder

- *Theorem*, Sipser and Spielman (1996):
- Let *G* be a (γ, ρ, α, (¾ +ε)γ) expander over n variable nodes, for any ε > 0. Then, parallel decoding algorithm will correct any

$$a_0 < \frac{1}{2} \times a \times (1 + \frac{1}{4} e)$$

  fraction of error after $\log_{1/(1-4e)}(a_0 n)$ decoding rounds.
- If *V* is the set of corrupt variables and |*V*|<α *n* (1 + 4ε)/2, then the parallel decoding algorithm produces a word with at most |*V*|(1-4ε) corrupt variables after <u>one</u> decoding round.

Total Fraction of Correctable Errors

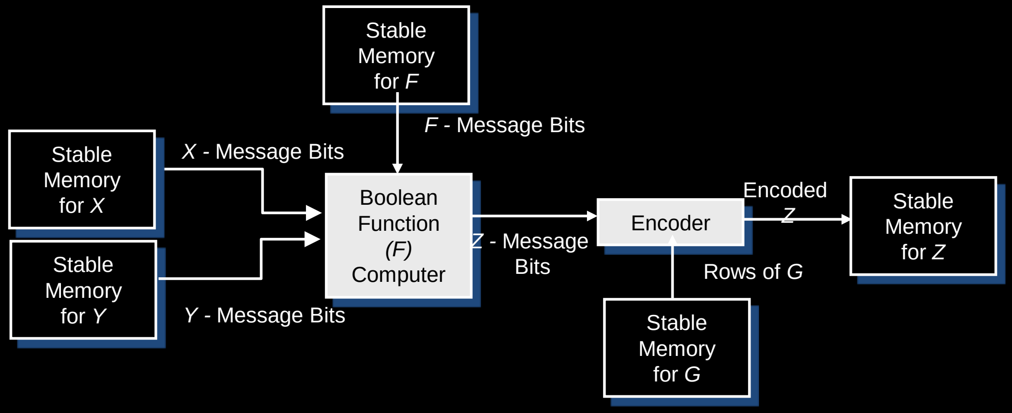
- The total fraction of error from all sources must be kept below correction capability of the code.
- The memory can correct α_{m}, $\alpha_{\oplus}$ and α_{γ} fraction of errors as long as α_{m}, $+\gamma(\rho - 2)\ \alpha_{\oplus} + \alpha_{\gamma} < \alpha_{\text{Total}}$.
- The main theorem, together with the existence of expanders of sufficient expansion proves the existence of memories. which can tolerate α_{m}, $\alpha_{\oplus}$ and α_{γ} fraction of errors.

Other Results and Recent Developments

- Fault-tolerant decoders
  - One-step majority logic decoders (Chilappagari and Vasic, 2006)
  - Bit-flipping (Vasic and Chilappagari, 2007)
  - Density evolution of faulty decoders (Varshney, 2011)
  - Density evolution of faulty Gallager B (Yazdi, 2013)
  - Noisy min-sum (Kameni Ngassa et al. 2013)
  - Gallager B under data dependent errors (Brkic et al., 2014)
  - Analysis of faulty Gallager B on QC-LDPC codes (Al Rasheed, 20013)

General Open Problems

- The situation is different if failures are permanent or a mixture of transient and permanent failures (rerouting or coding).
- Faulty encoding and wire failures not taken into account, but wires have different lengths.
- Cellular automata.
- Unreliable computers.
  - *Stable* memories can be used to construct reliable computers made of faulty components.



Хвала на пажњи!

How many errors can a fixed length code correct?

Drawbacks of expander arguments

- Bounds derived using random graph arguments on the fraction of nodes having sufficient expansion are very pessimistic
  - Richardson and Urbanke (2003): In the $(5,6)$ regular code ensemble, minimum distance is 3% of code length. But only $3.375 \text{ x } 10^{-11}$ fraction of nodes have expansion of $\geq (\square) d_v$
- Expansion arguments cannot be used for column-weight-three codes ($d_v \geq 5$)
- Determining the expansion of a given graph known to be NP hard, and spectral gap methods <u>cannot</u> guarantee an expansion factor $\geq \square$

The main idea

- The expansion arguments rely on properties of random graphs and hence do not lead to explicit construction of codes.
- Ii the expansion properties can be related to the parameters of the Tanner graph, such as g, *and* d_v , then the bounds on guaranteed error correction capability can be established as function of these parameters.

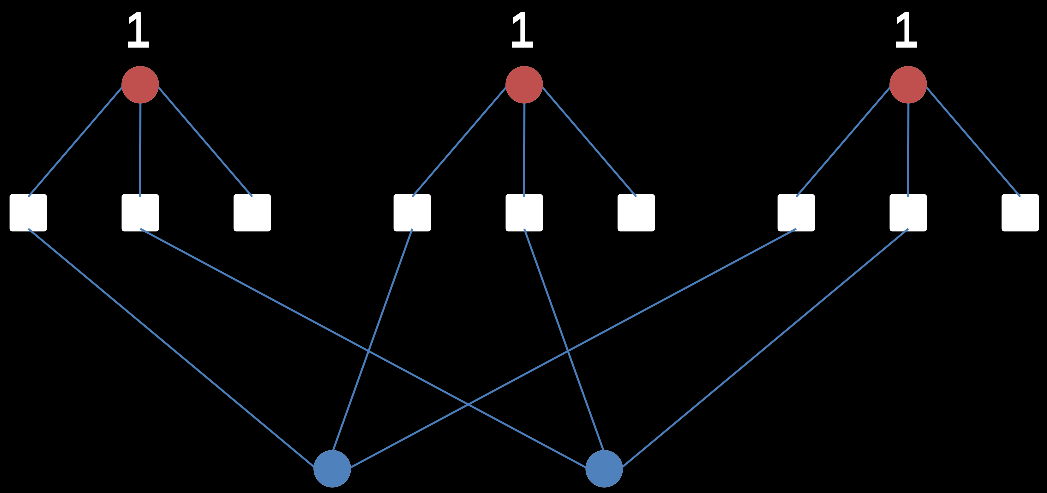
The curious case of $d_v = 3$ codes

- Gallager showed that the minimum distance of ensembles of $(d_\mathrm{v}, d_\mathrm{c})$ regular LDPC codes with $d_v \geq 3$ grows linearly with the code length
- This implies that under ML decoding, $d_v = 3$ codes <u>are capable</u> of correcting a number of errors linear in the code length
- Gallager also showed that under his algorithms A and B the bit error probability approaches zero whenever we operate below the threshold
- But, the correction of a linear fraction of errors was not shown

Other complications with $d_v = 3$ codes

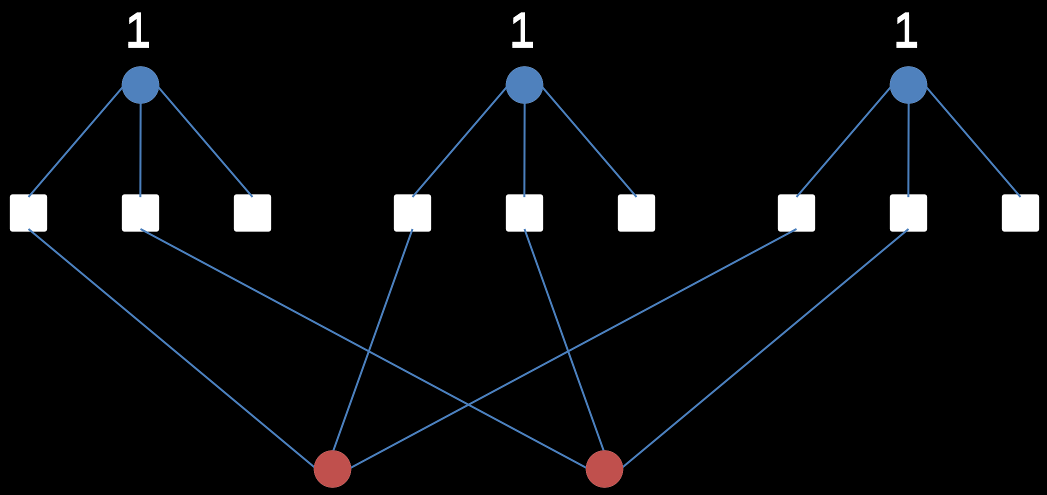
- Even for the more complex LP decoding, it has not been shown that codes with $d_\mathrm{v} = 3$ can correct a fraction of errors
- To correct linear fraction of errors the expansion factor of □ is necessary, but the best expansion factor achievable by $d_v = 3$ codes is $1\text{-}1/d_v = \frac{2}{3}$

(5,3) Trapping set: illustration



- ● Corrupt variable
- ○ Correct variable
- 🔴 Variable decoded correctly
- 🔵 Variable decoded wrongly

(5,3) Trapping set: illustration



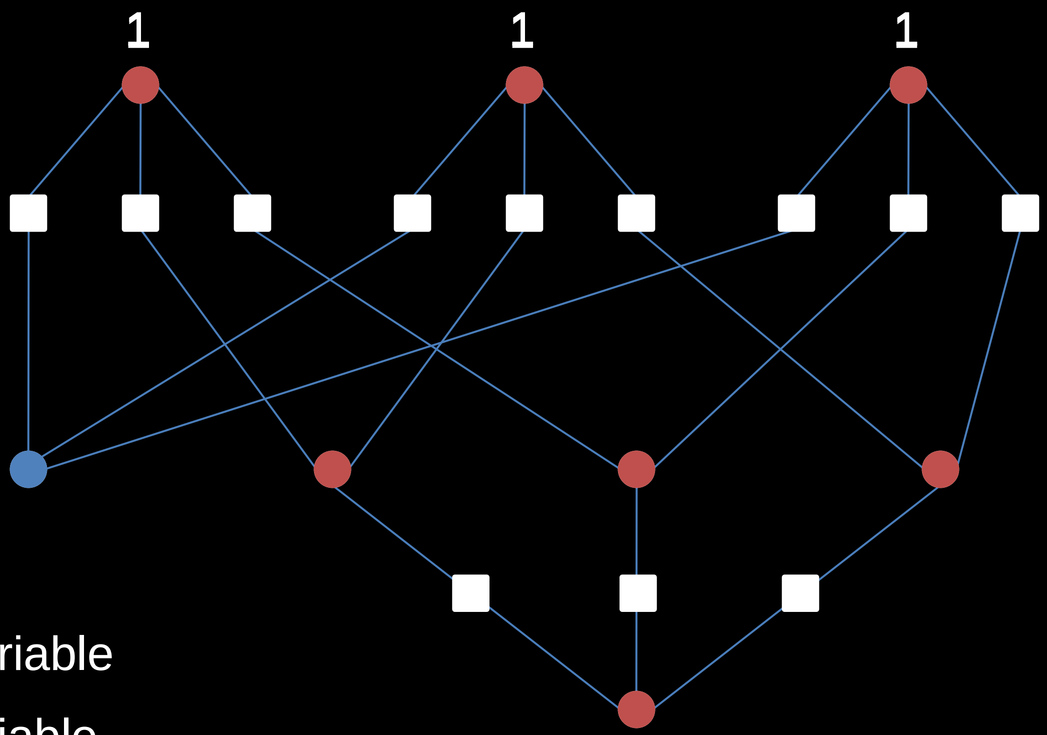
● Corrupt variable

○ Correct variable

🔴 Variable decoded correctly

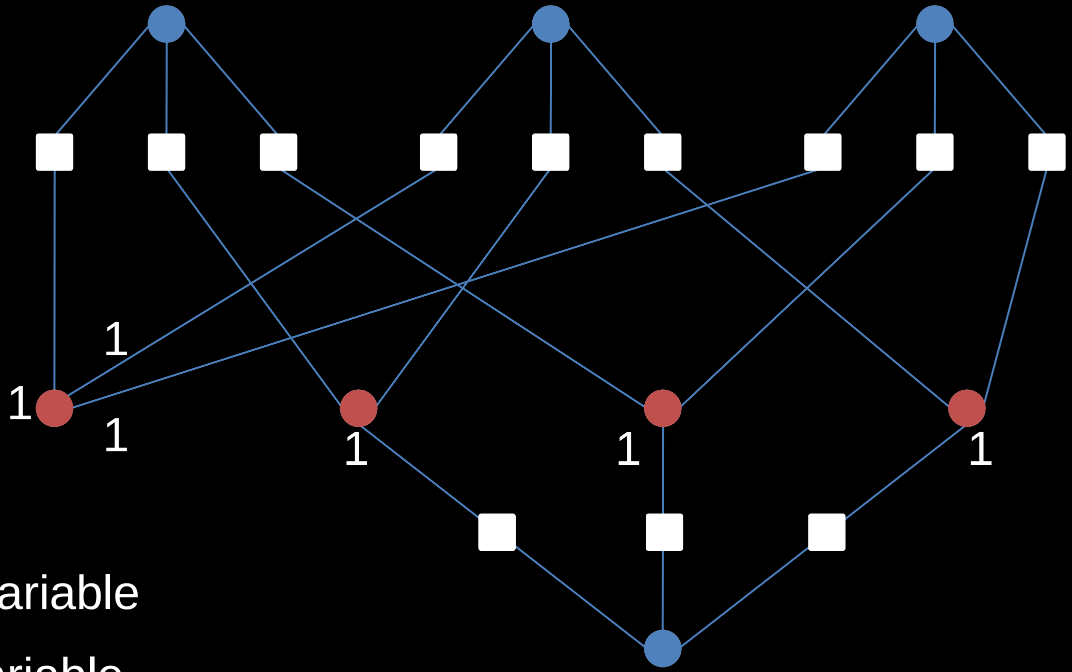
🔵 Variable decoded wrongly

Oscillations in the decoder



- ● Corrupt variable
- ○ Correct variable
- 🔴 Variable decoded correctly
- 🔵 Variable decoded wrongly

Oscillations in the decoder



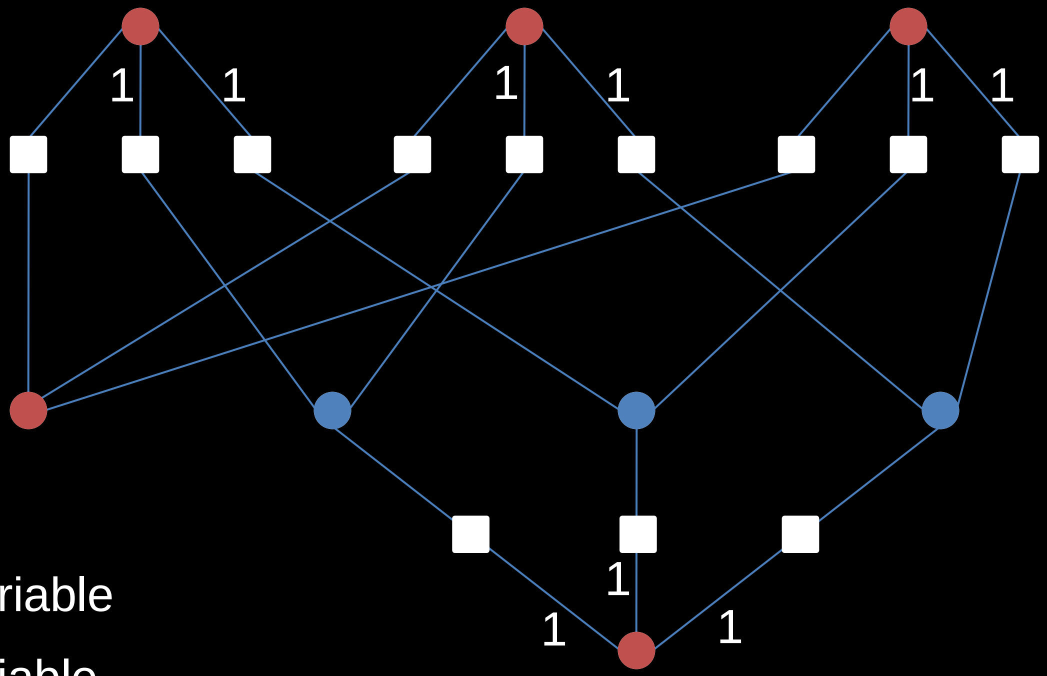
● Corrupt variable

○ Correct variable

● Variable decoded correctly

● Variable decoded wrongly

Oscillations in the decoder



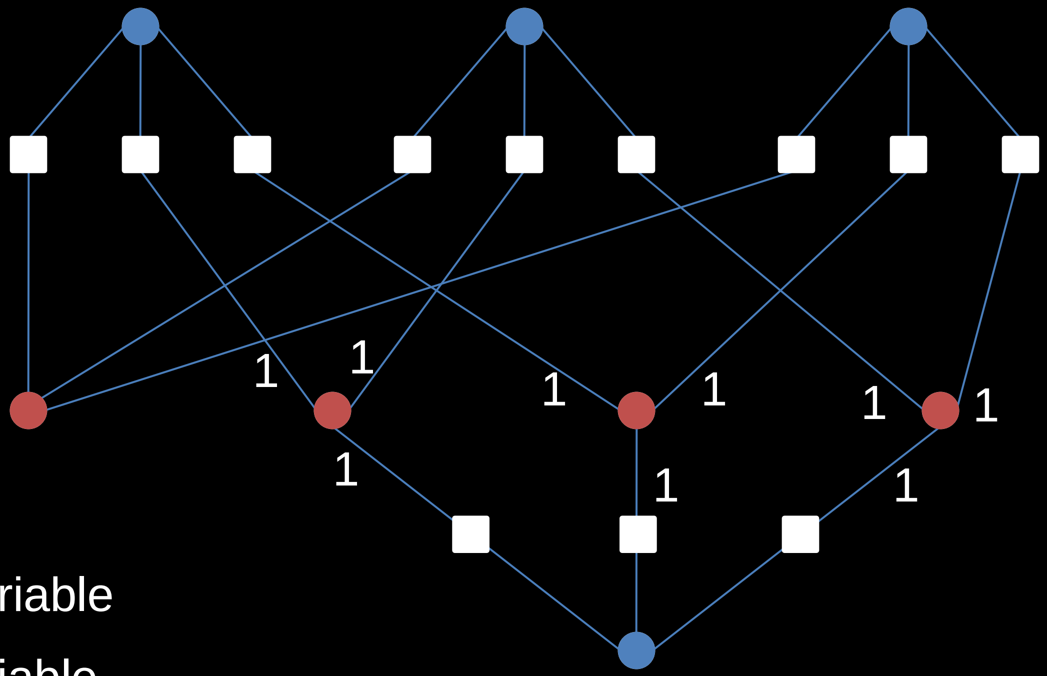
● Corrupt variable

○ Correct variable

🔴 Variable decoded correctly

🔵 Variable decoded wrongly

Oscillations in the decoder



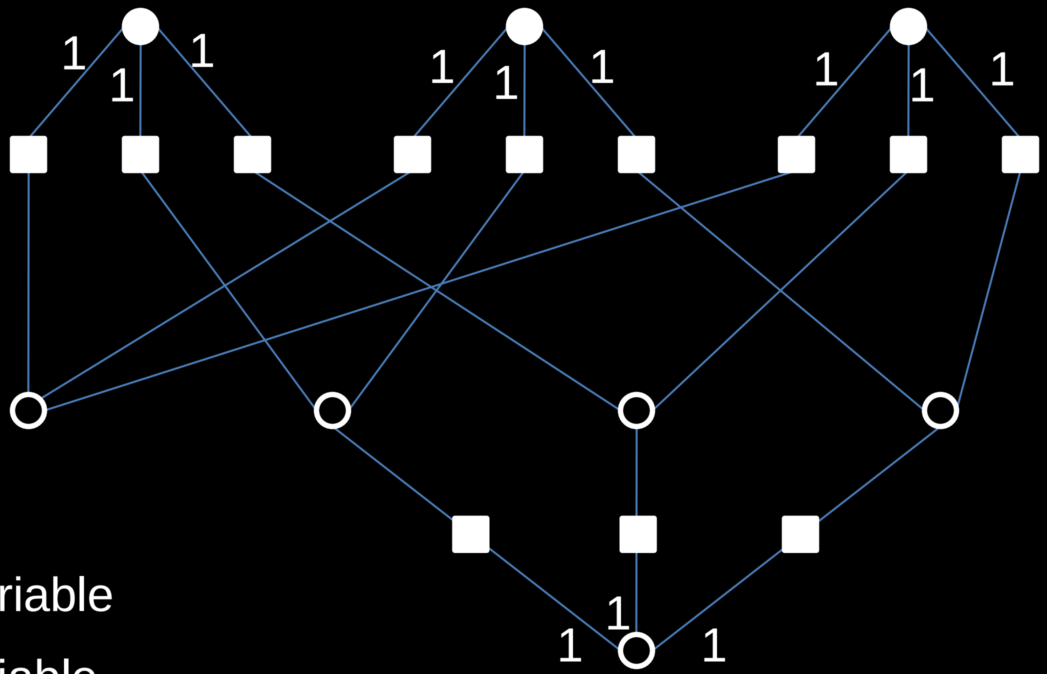
● Corrupt variable

○ Correct variable

🔴 Variable decoded correctly

🔵 Variable decoded wrongly

Oscillations in the decoder



● Corrupt variable

○ Correct variable

🔴 Variable decoded correctly

🔵 Variable decoded wrongly

Trapping sets - sufficient conditions

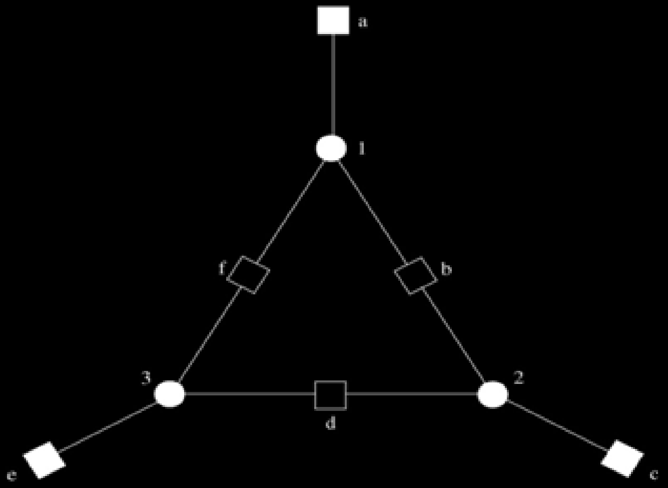
- *Theorem 1: Let C be a code in the ensemble of $(3, \rho)$* regular LDPC codes. Let Γ be a subgraph induced by the set of variable nodes T. Let the checks in Γ can be partitioned into two disjoint subsets: E consisting of checks with even degree, and O consisting of checks with odd degree. $\boldsymbol{y}$ is a fixed point if :

  (a) $\mathrm{supp}(\boldsymbol{y}) = T$,

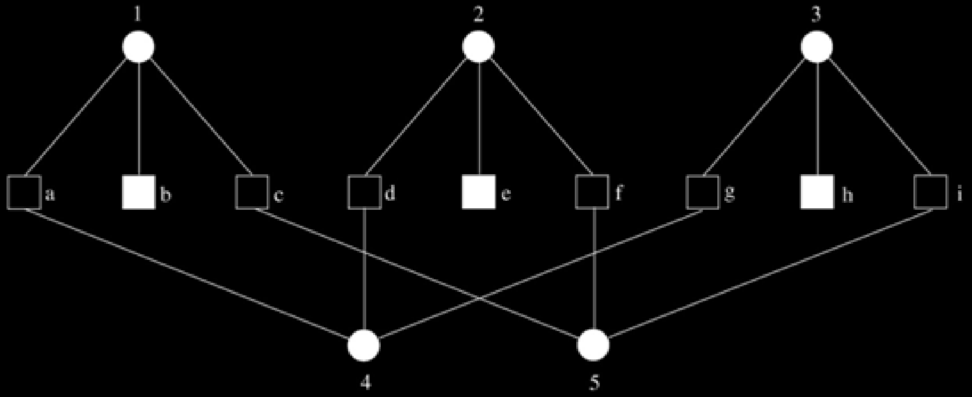
  (b) Every variable node in Γ is connected to at least two checks in E ,

  (c) No two checks of O are connected to a variable node outside Γ.

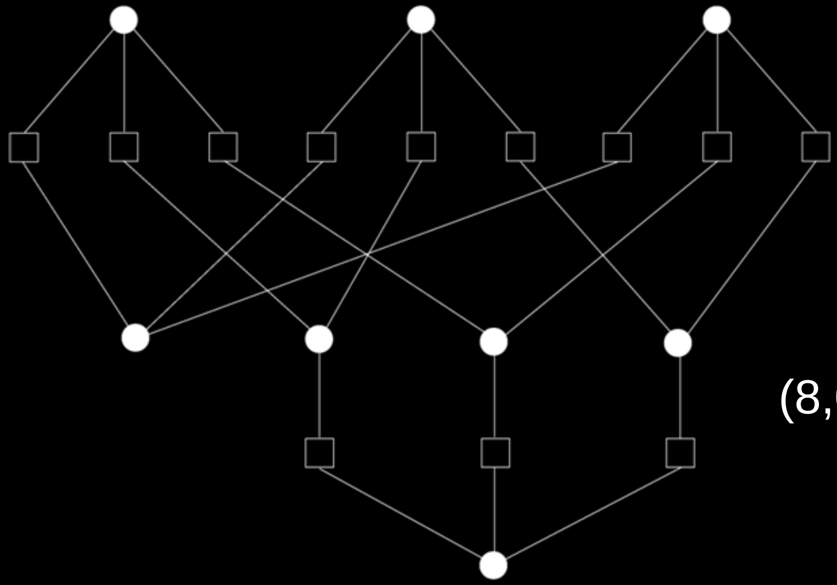
Trapping sets: examples



(3,3) trapping set



(5,3) trapping set



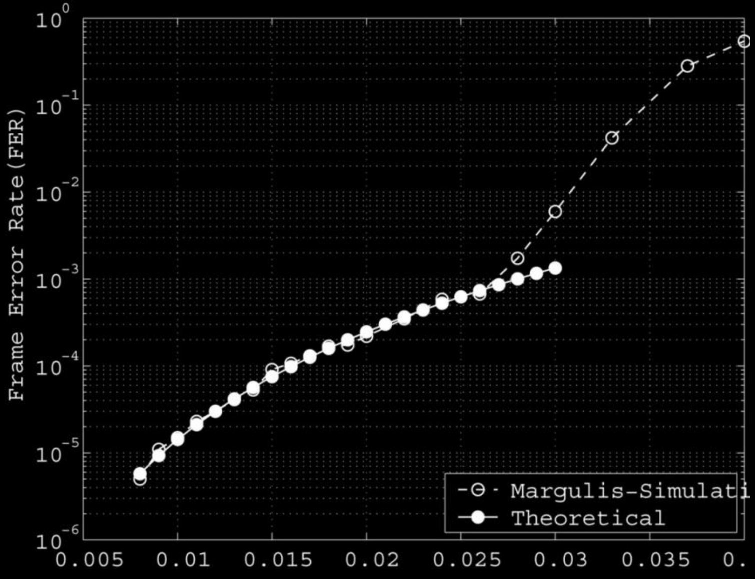
(8,0) Trapping Set

Trapping set ontology

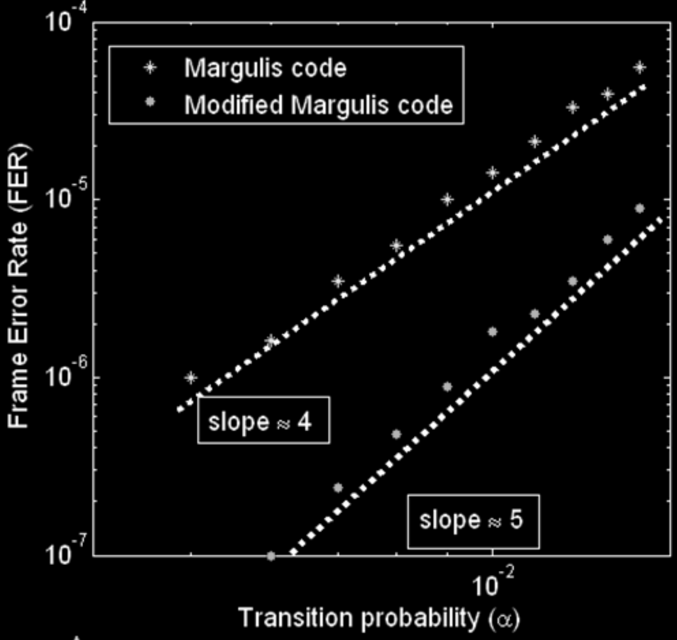
A trapping set ontology

- A database and software for systematic study of failures of iterative decoders on BSC
  http://www.ece.arizona.edu/vasiclab/Projects/CodingTheory/TrappingSetOntology.html
- Applications:

Error floor estimation



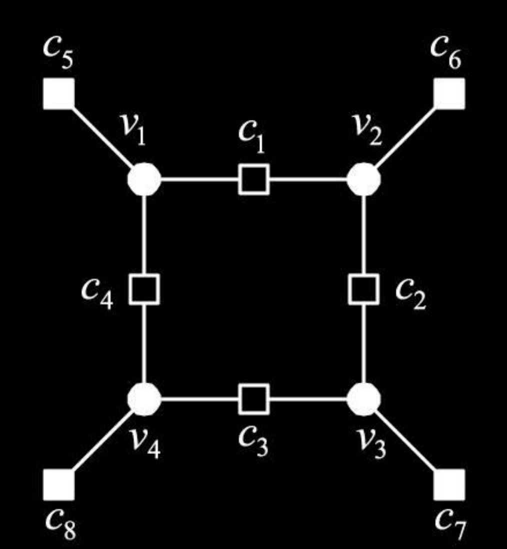
Code construction

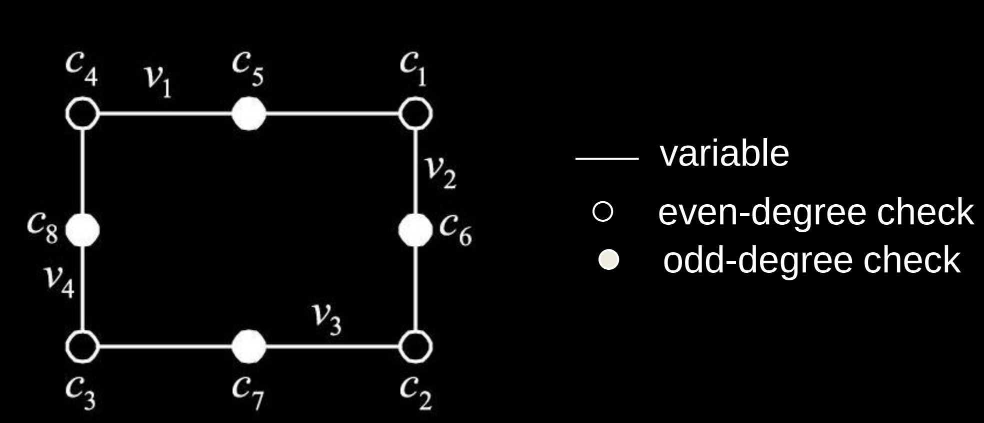


An incidence structure representation

- A trapping sets can be characterized using incidence structure of lines and points.

lines ↔ variables, checks ↔ points





—— variable

○ even-degree check

● odd-degree check

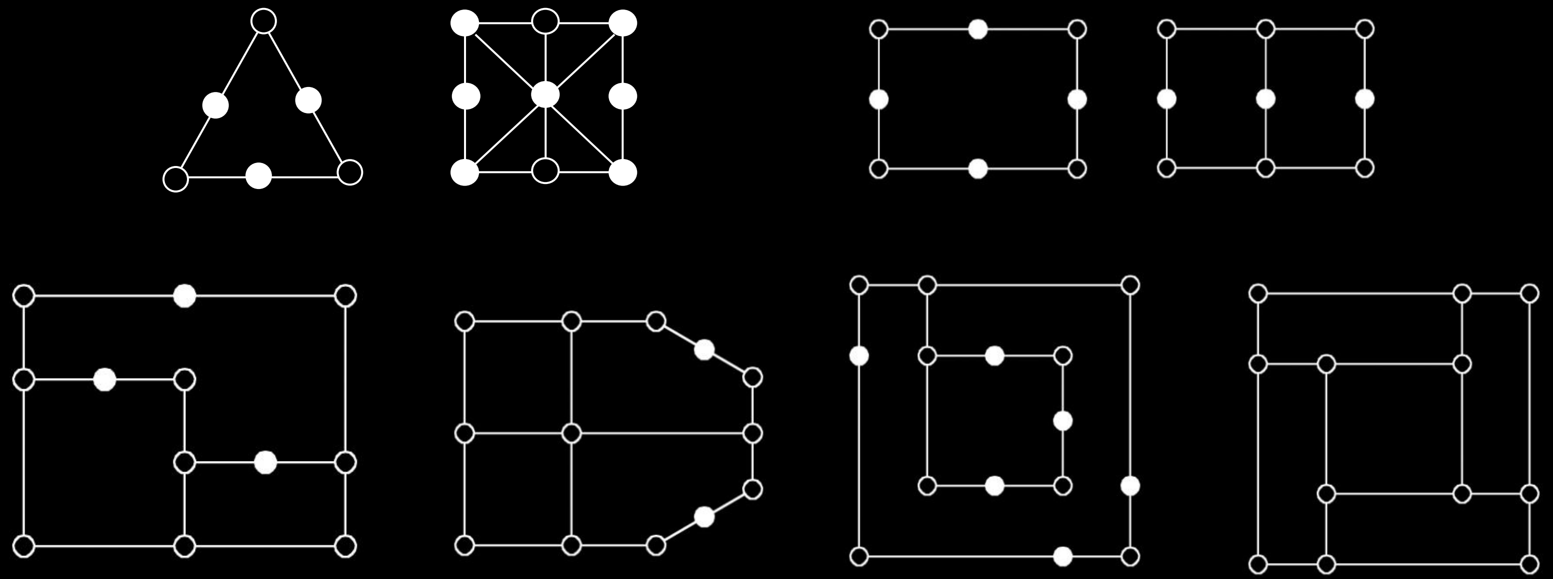
Tanner graph representation

A point-line representation

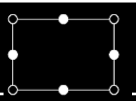
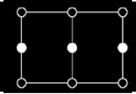
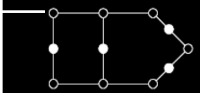
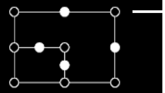
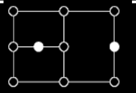
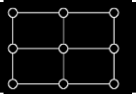
- An (a,b) trapping set is an incidence structure of a lines and b white points.

Properties of incidence structures

- A point is white if it lies on an odd number of lines and is black otherwise



Number of trapping sets

TS	#TS	*g*=6	*g*=8	*g*=10	*g*=12
(3,3)	1	1			
(4,4)	1				
(4,2)	1	1			
(4,0)	1	1			
(5,5)	1			1	
(5,3)	2	1			
(5,1)	1	1			
(6,6)	1				1
(6,4)	4	2	 		
(6,2)	4	3			
(6,0)	2	1			

Creating larger trapping sets from smaller

- The “reproduction” rule: children are obtained by adding lines to parents, changing the color of the points accordingly.

Reproduction illustration

Consequences

- For column weight three codes, the weight of correctable error patterns under Gallager A algorithm grows only linearly with girth
- For any $\alpha>0$ and sufficiently large block lengths n, no code in the $C^n(3, \rho)$ ensemble can correct all αn errors under Gallager A algorithm

The lower bound lemmas

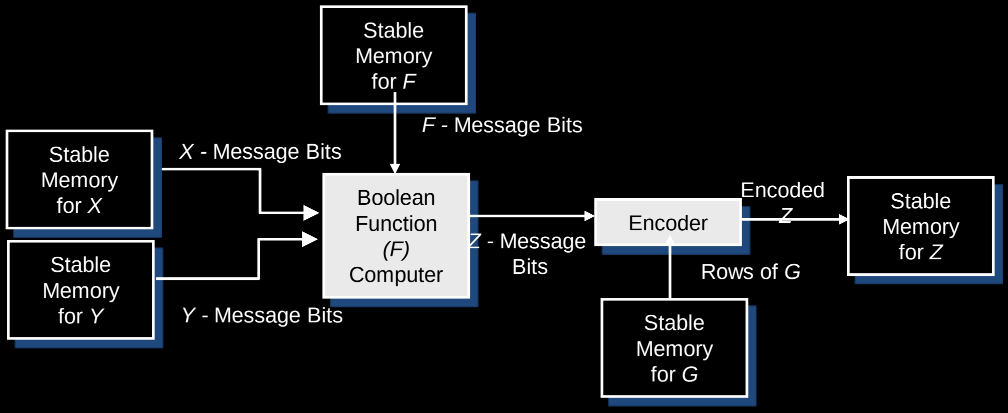
- *Theorem 3*: An $(n, 3, \rho)$ code with girth $g \geq 10$ can correct all error patterns of weight $g/2-1$ or less in $\mathrm{g}/2$ iterations of the Gallager A algorithm.
- Equivalently, there are no trapping sets with critical number less than $g/2$.
- Proof: Finding, for a particular choice of k, all configurations of $g/2-1$ or less bad variable nodes which do not converge in $k+1$ iterations and then prove that these configurations converge in subsequent iterations.

Open Coding Theory Problems

- Investigating ensemble of irregular graphs.
  - Known to have higher thresholds and approach capacity.
- Fault tolerant implementation of expander codes.
  - Work by Zemor and Barg and Burshtein and Miller, and Guruswami and Indyk let do remarkable improvement in rate vs. correction capability tradeoffs.
  - Need less expansion, but more complicated decoding.
- Explicit construction of expanders.
  - Recent work by Capalbo *et.al* (2002) gives expander construction but has very large degrees.

General Open Problems

- The situation is different if failures are permanent or a mixture of transient and permanent failures (rerouting or coding).
- Faulty encoding and wire failures not taken into account, but wires have different lengths.
- Cellular automata.
- Unreliable computers.
  - *Stable* memories can be used to construct reliable computers made of faulty components.

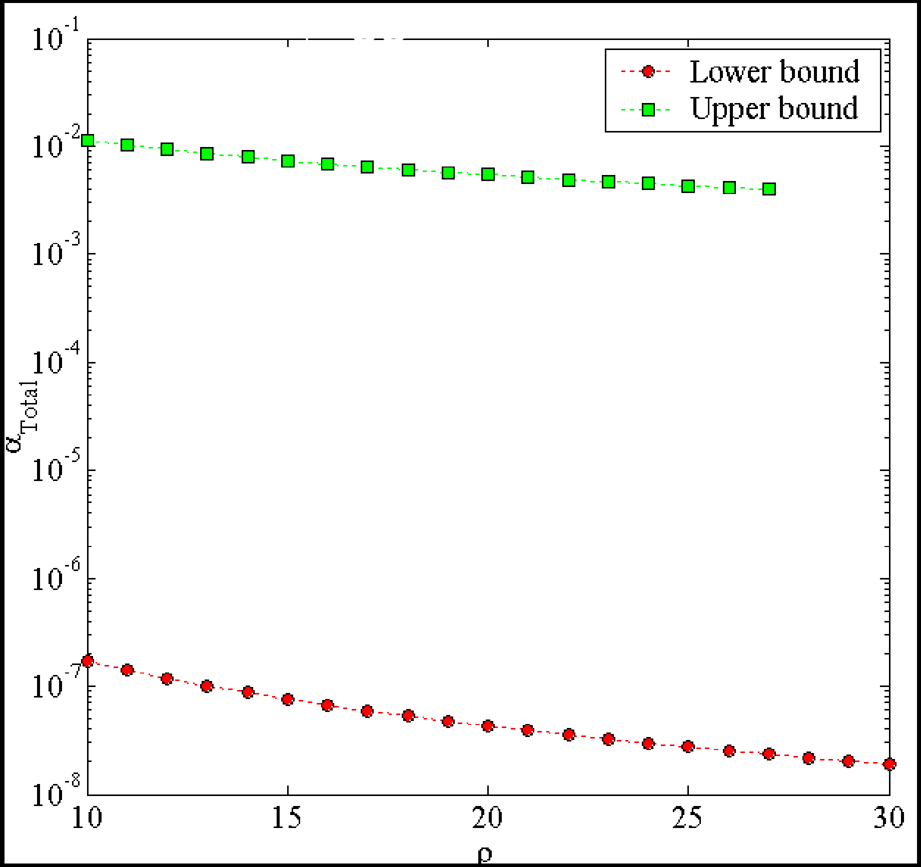


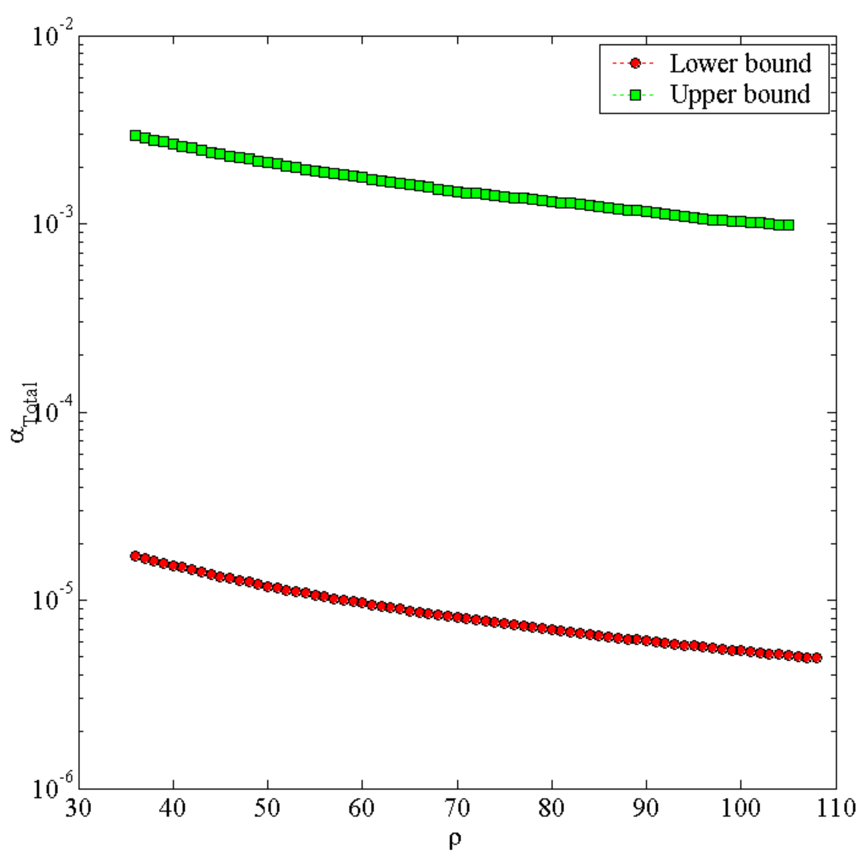
Consequences

- A decoding algorithm and corresponding sufficient condition to correct any k errors
- Possibility to correct a linear fraction of errors for column-weight-three codes!
- Decoders that do not require $(3/4)d_v$ expansion to correct linear fraction of errors for higher column-weight-codes
- Bridging the gap between sub-optimal decoders and maximum-likelihood decoders

Bounds on α_{Total}

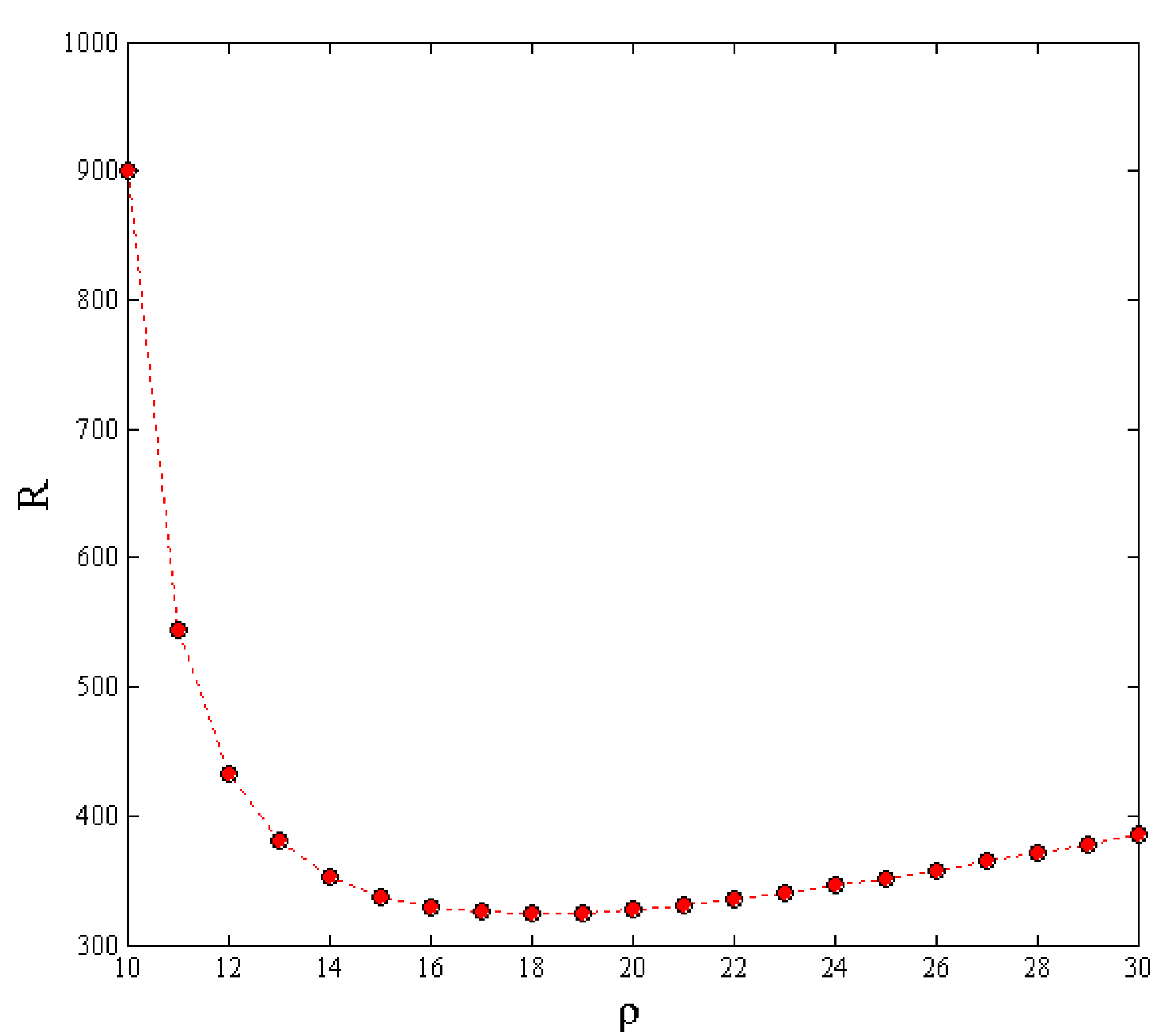
γ=9



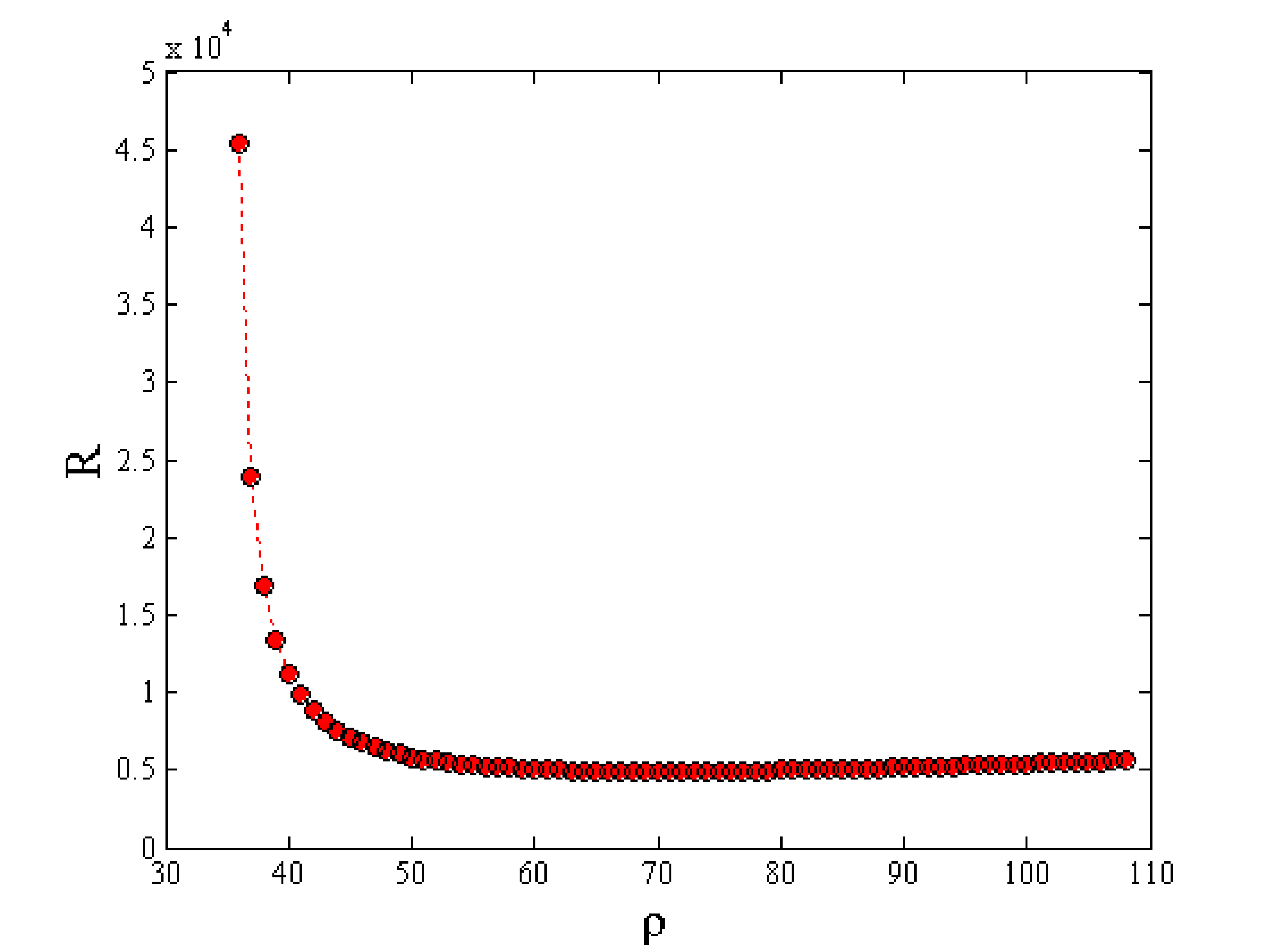


THE UNIVERSITY OF ARIZONA®
TUCSON ARIZONA

Redundancy for γ=9



Redundancy for γ=35



Bounds for the TK scheme

- There exists a solution α, $0 \le \alpha < 1$ of the equation

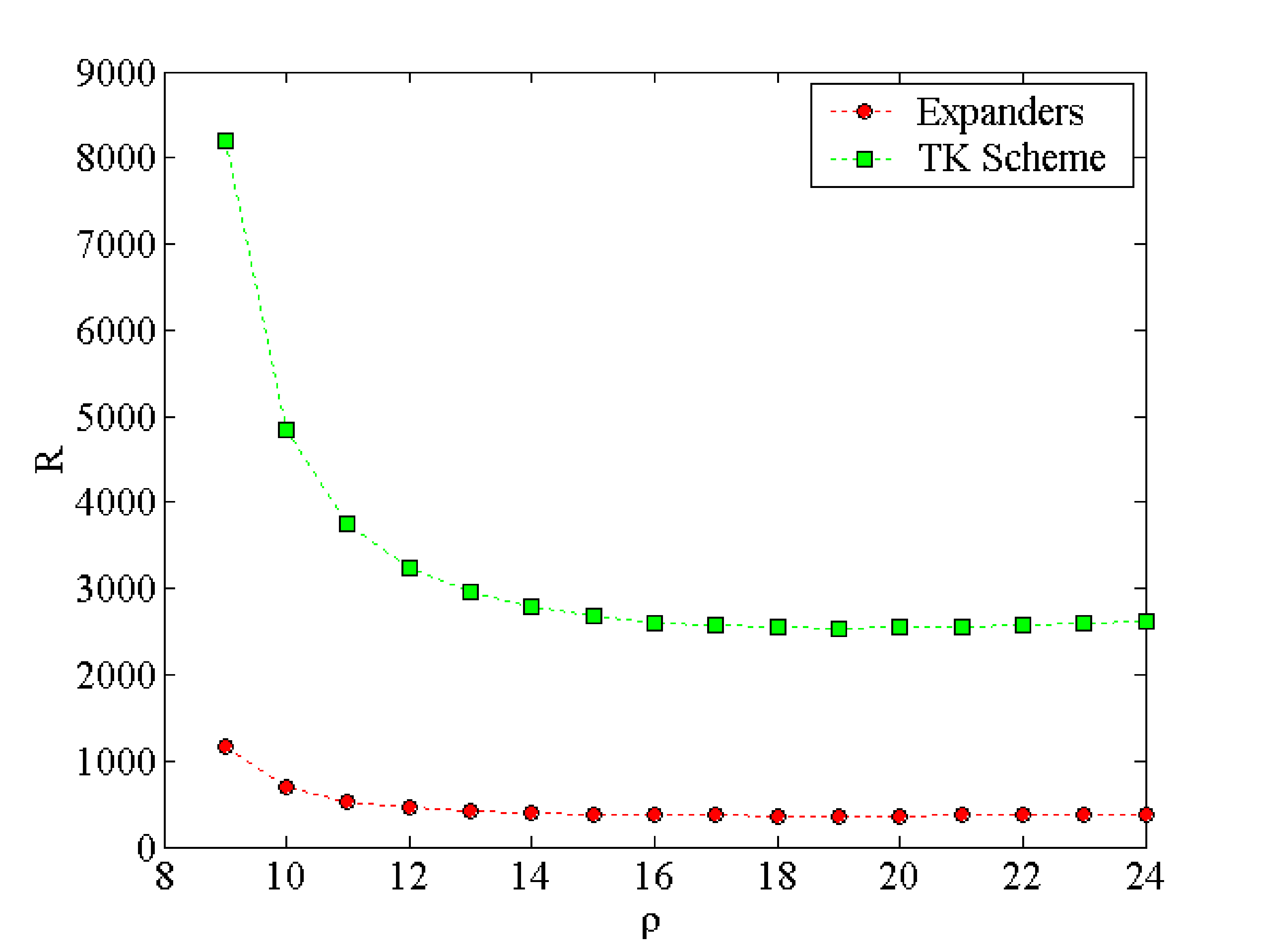
$$\binom{\gamma-1}{\gamma/2}\cdot\left[(K-1)\cdot(\alpha+\alpha_{\oplus})\right]^{\gamma/2}+\alpha_{\oplus}+\alpha_{\mu}+\alpha_{\gamma}=\alpha$$

- The minimum solution $\alpha_{\min}$, 0 · $\alpha_{\min}$ · 1 satisfies the inequality

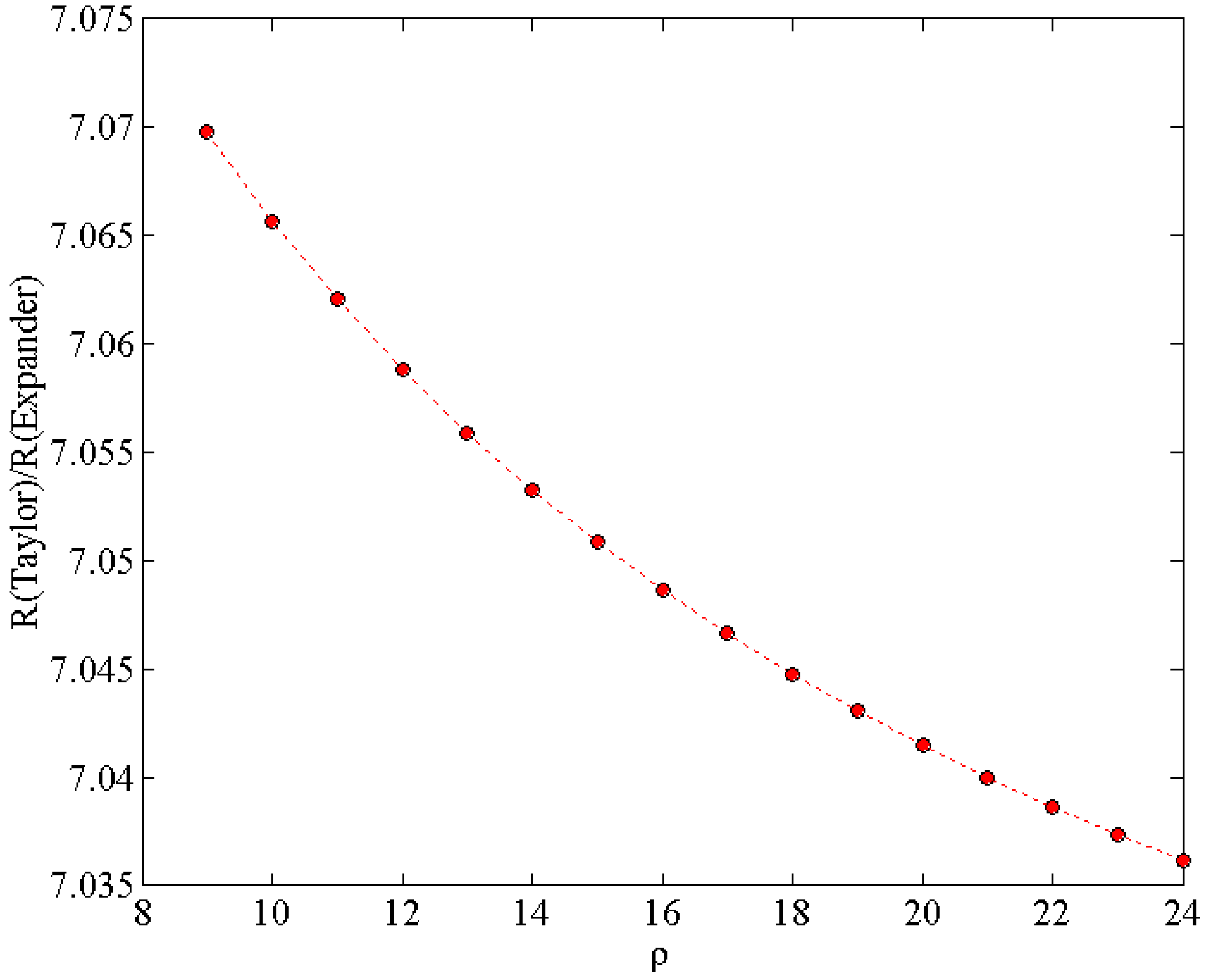
$$e^{\beta/\lambda}=\frac{2\cdot(\alpha_{\min}+\alpha_{\oplus})}{\gamma\cdot(\gamma-1)^{2}\cdot(\rho-1)^{2}\cdot\left[\alpha-(\alpha_{\oplus}+\alpha_{\mu}+\alpha_{\gamma})\right]}>1$$

$$P(T)\le A\cdot T\cdot n^{-\beta}\cdot\left(\frac{1}{2\lambda}-\ln n\right)$$

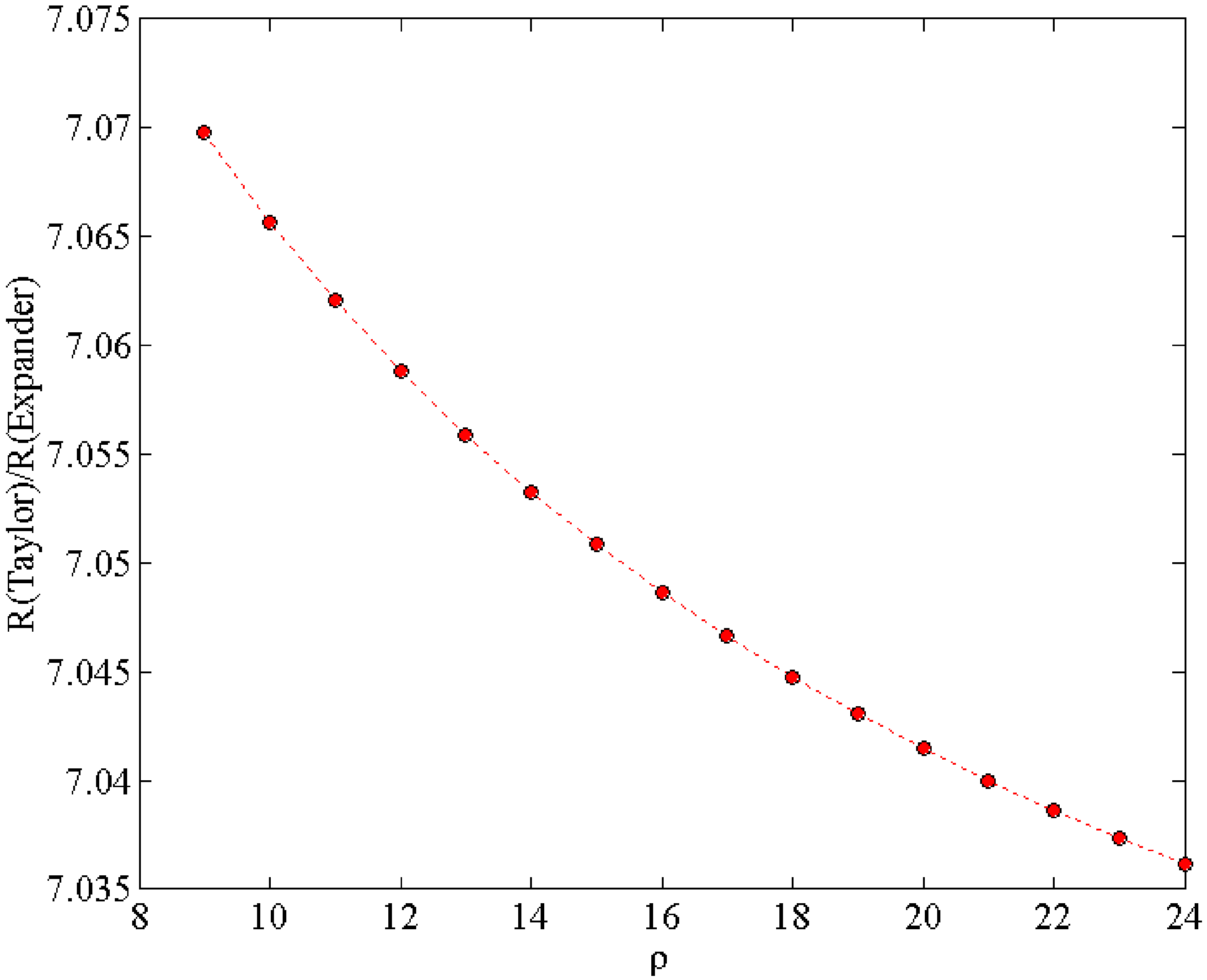
A Comparison with the TK Scheme for γ=8



Gain over the TK Scheme for γ=8



Gain over the TK Scheme for γ=8



General Comments on the TK Scheme 1/2

- The work of Taylor and Kuznetsov closely resembles Gallager's work on LDPC codes.
- Gallager methodology relies on:
  - constructing a class of codes which do not have cycles of length less or equal to $2l$,
  - applying density evolution and determining thresholds, and
  - showing that after l decoding rounds the bit-error probability decreases as $e^{-\alpha n^{\beta}}$, where α and β are constants while operating over a BSC with parameter less than the threshold.
- Contrasted with recent work by Richardson and Urbanke (RU) which is based on sequence of ensembles of codes.
- Can the arguments such as concentration, convergence to cycle-free case and density evolution be applied to the case of faulty gates?

Existence of Expanders

- It is known that a random graph is a good expander with high probability.
- However, we need it to be good in terms of both α and δ.
- Urbanke and Richardson (2007) give some bounds:
- Chose a code uniformly at random from LDPC (n, $x^{\gamma\text{-}1}$, $x^{\rho\text{-}1}$).
- Choose λ 2 [0, 1-1/*l*), and let α_{max} be the positive solution of the equation

$$P\{G \quad \text{is an}\,(g, r, a, l)\ \text{expander}\}\ {}^{3}\ 1 - O(n^{-b})$$

- Then for 0 < α < α_{max} and β=*l* (1-γ)-1

Expander arguments

- Sipser and Spielman (1996): Let G be a $(d_v, d_c, \alpha, (\square + \varepsilon)\, d_v)$ expander over n variable nodes, for any $\varepsilon > 0$. Then, the parallel bit flipping algorithm will correct any $\alpha_0 < \alpha\,(1+4\varepsilon)/2$ fraction of error after $\log_{1/(1-4\varepsilon)}(\alpha_0 n)$ decoding rounds
- Burshtein and Miller, (2001): “Expander graph arguments for message passing algorithms”
- Feldman *et al.* (2003): “ LP Decoding corrects a constant fraction of errors”

Expander Codes

- A Tanner graph *G* of a (n,γ,ρ) LDPC code is a $(\gamma, \rho, \alpha ,\delta)$ expander if for every subset *S* of at most an αn variable nodes, at least $\delta|S|$ check nodes are incident to *S.*
- Expander codes are a class of asymptotically good error correcting codes.