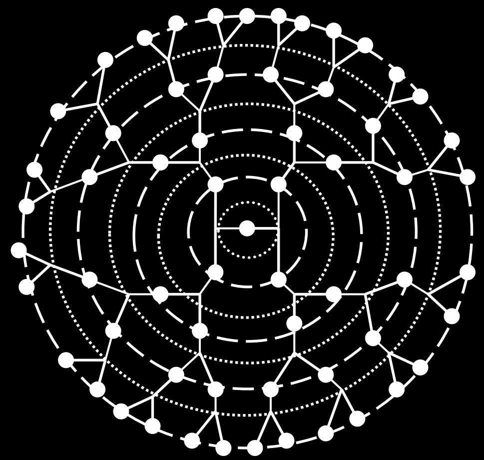
Error correction guarantees

Drawback of asymptotic analyses

- Valid only as long as the incoming messages are independent. (*independence assumption*)
- The messages are independent for l iterations only if the neighborhoods of depth l around the variable nodes are trees.



- In a Tanner graph of girth g, the number of independent iterations satisfies the relation $g/4\text{-}1 \leq l < g/4$

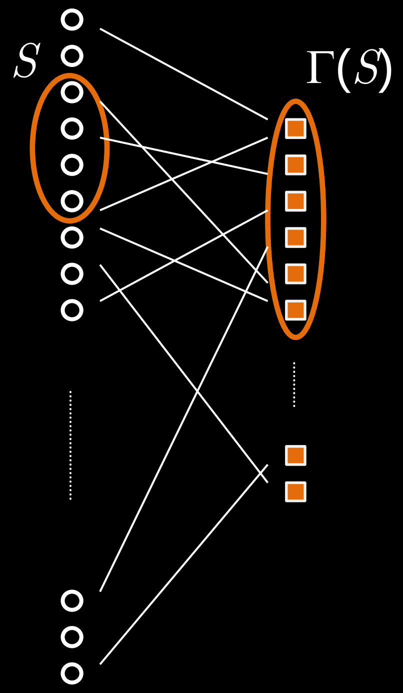
Adapted from LDPC Codes: An Introduction by Amin Shokrollahi

Independence assumption

- If n is the total number of variable nodes, this puts an upper bound on l (of the order $\log(n)$
- $l = \log(n)$ number of iterations is usually not enough to prove that the decoding process corrects all errors.
- A different analysis is needed to show that the decoder succeeds.
- A property of the graphs that guarantees successful decoding is called *expansion*.

Expanders

- Definition: A bipartite graph with n variable nodes is called an (α,β)-expander if for any subset S of the variable nodes of size at most αn 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.

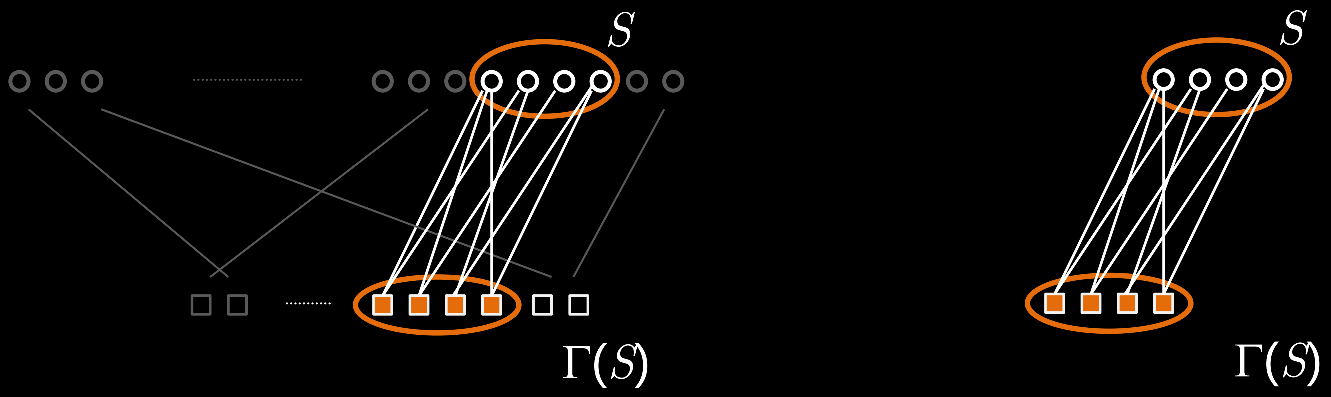


$$|S| \leq \alpha\ n \ \Rightarrow\ |\Gamma(S)| \geq \beta\, a_S\, |S|$$

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

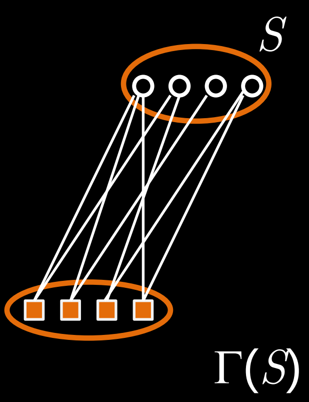
Decoding on the BEC

- Theorem: If a Tanner graph is an $(\varepsilon, 1/2)$-expander, then the erasure decoding algorithm recovers any set of εn or fewer erasures.
- Proof: Suppose that this were not the case and consider a minimal counterexample consisting of a nonempty set S of erasures. Consider the subgraph induced by S, and denote by $\Gamma(S)$ the set of neighbors of S.



Proof - continuation

- No node in $\Gamma(S)$ has degree 1, since this neighbor would recover one element in S and would contradict the minimality of S. Hence, the total number of edges emanating from these nodes is at least $2|\Gamma(S)|$.



- On the other hand, the total number of edges emanating from S is $a_S|S|$, so $a_S\,|S| \geq 2|\Gamma(S)|$,
- which implies $|\Gamma(S)| \leq a_S\,|S|/2$ and contradicts the assumption of the ½- expansion property of the graph.

Decoding on BSC

- Parallel bit-flipping algorithm:
- While there are unsatisfied check bits
  - Find a bit for which more than d/2 neighboring checks are unsatisfied
  - Flip that bit
- Properties:
  - Converges under the condition that every step reduces unsatisfied nodes by at least 1.
  - Runs in linear time.

(note: a check is *unsatisfied* if sum of its bits $\neq 0$)

Bit-flipping decoder on BSC

- Observation: The decoder progresses with correcting errors as long there are bits for which more than $d_v/2$ neighboring checks are unsatisfied.
- What property on the graph ensures that? Expansion.
- Lemma: Consider a (α,$\tfrac{3}{4}d_v$) expander with n variable nodes and let $k \leq \alpha n$ be the number of variables in error. Then, there are more than $d_v/2$ unsatisfied checks.

Expander arguments

- Sipser and Spielman (1996): Let G be a $(d_v,\ d_c,\ \alpha,\ (3/4+\varepsilon)\,d_v)$ expander over n variable nodes, for any ε > 0. Then, the parallel bit flipping algorithm will correct any $\alpha_0 < \alpha\,(1+4\varepsilon)/2$ fraction of error after $\log_{1/(1\text{-}4\varepsilon)}(\alpha_0 n)$ 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"

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 $\geq (3/4)\, d_v$
- Expansion arguments cannot be used for column-weight-three codes (they work for $d_v \geq 5$)
- Determining the expansion of a given graph known to be NP hard, and spectral gap methods <u>cannot</u> guarantee an expansion factor $\geq 1/2$

Girth and column-weight

- 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.

Finite length analysis goals

- Establish a connection between guaranteed error correction capability and graph parameters such as g, girth, and d_v , variable degree
- Column weight $d_v = 3$ is the main focus

Number of correctable errors and FER

- Consider the BSC, and let c_k - the number of configurations of received bits for which *k* channel error lead to a codeword (frame) error.
- Let i - the minimal number of channel errors that can lead to a decoding error. Then

$$FER(\alpha) = \sum_{k=i}^{n} c_k \alpha^k (1-\alpha)^{(n-k)}$$

- When α<<1

$$\log(FER(\alpha)) \approx \log(c_i) + i \log(\alpha)$$

Frame error rate (FER)

- What is usually plotted (semi-log scale):

$$\log(FER(\alpha)) = \log\Big(\sum_{k=i}^{n} c_k \alpha^k (1-\alpha)^{n-k}\Big)$$

$$= \log(c_i) + i\log(\alpha) + \log((1-\alpha)^{n-i})$$

$$+\log\left(1 + \frac{c_{i+1}}{c_i}\alpha(1-\alpha)^{-1} + \ldots + \frac{c_n}{c_i}\alpha^{n-i}(1-\alpha)^{i-n}\right)$$

- As the error probability decreases…

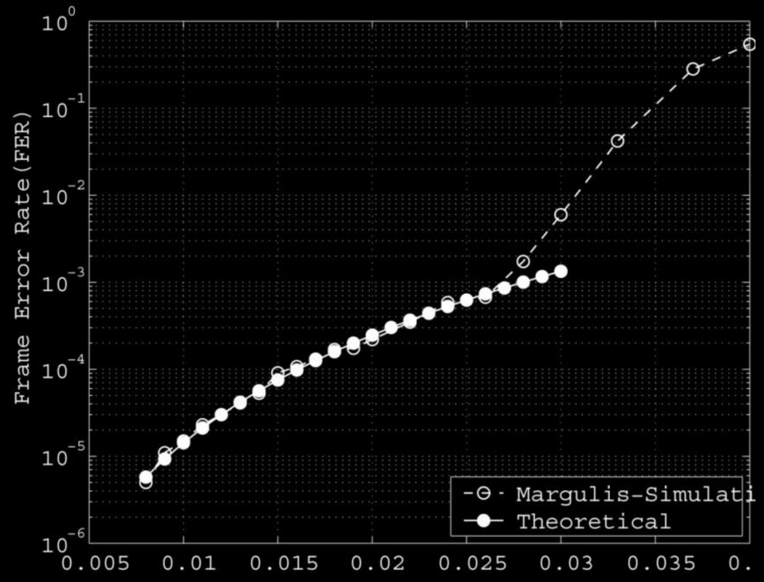
$$\lim_{\alpha\to 0}\left[\log((1-\alpha)^{n-i})\right] = 0$$

$$\lim_{\alpha\to 0}\left[\log\left(1+\frac{c_{i+1}}{c_i}\alpha(1-\alpha)^{-1}\ldots+\frac{c_n}{c_i}\alpha^{n-i}(1-\alpha)^{i-n}\right)\right] = 0$$

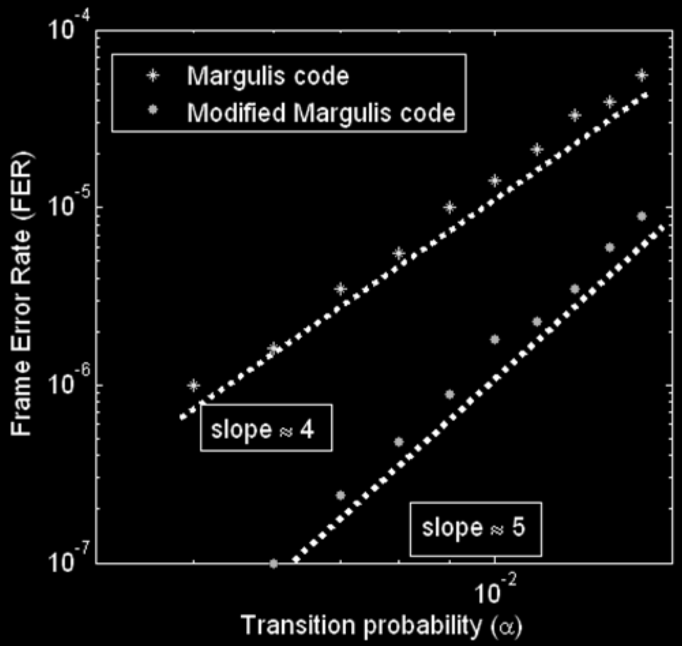
$$\log(FER(\alpha)) \approx \log(c_i) + i\log(\alpha)$$

Practical problems related to error floor

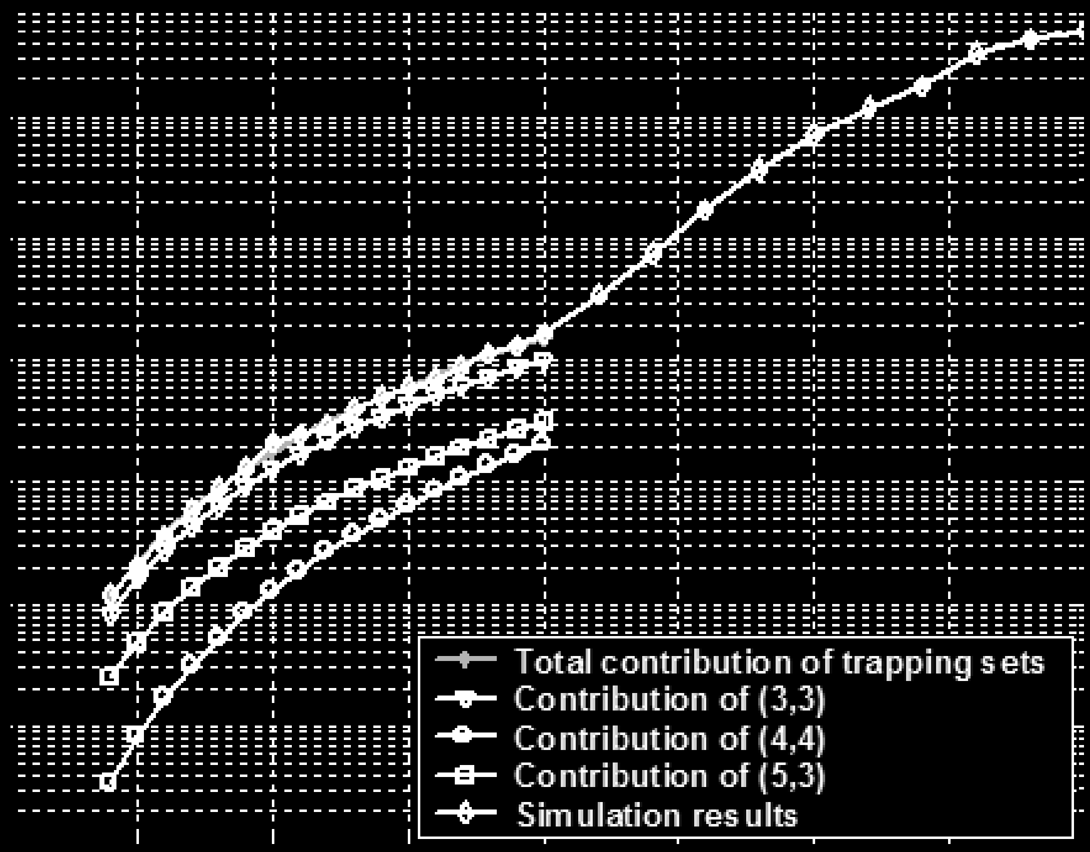
Error floor estimation



Code construction



FER contribution of different error patterns



Trapping sets

Basic concepts

- An *eventually correct* variable node
- *A fixed point* of iterative decoding
- *Inducing set*
- *Fixed set*
- The *critical number* m of a trapping set is the minimal number of variable nodes that have to be initially in error for the decoder to end up in that trapping set.

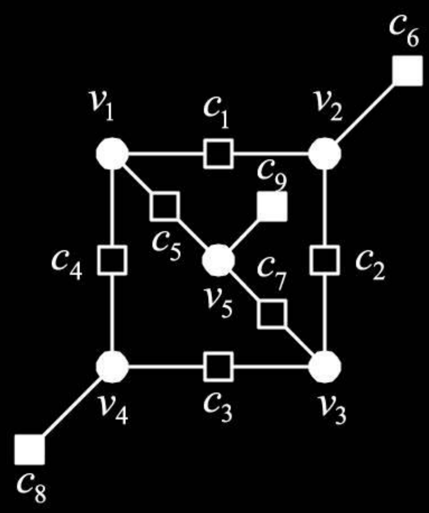
- An (a,b) trapping set: a set of not eventually correct variabe nodes of size a, and the b odd degree check nodes in the sub-graph induced by these variable nodes.

Basic terminology

- Consider an LDPC code of length n, and assume that the all-zero codeword is transmitted over the BSC, and that the word $\mathbf{y}$ is received.
- Let $\mathbf{x}^l$, $l \leq D$ be the decoder output vector at the l th iteration (D the maximum number of iterations).
- A variable node v is said to be *eventually correct* if there exists a positive integer q such that for all $l \geq q$,
$$v \notin \mathrm{supp}(\mathbf{x}^l)$$
- A *decoder failure* is said to have occurred if there does not exist $l \leq D$ such that
$$\mathrm{supp}(\mathbf{x}^l) = \emptyset.$$

Trapping sets of various decoders

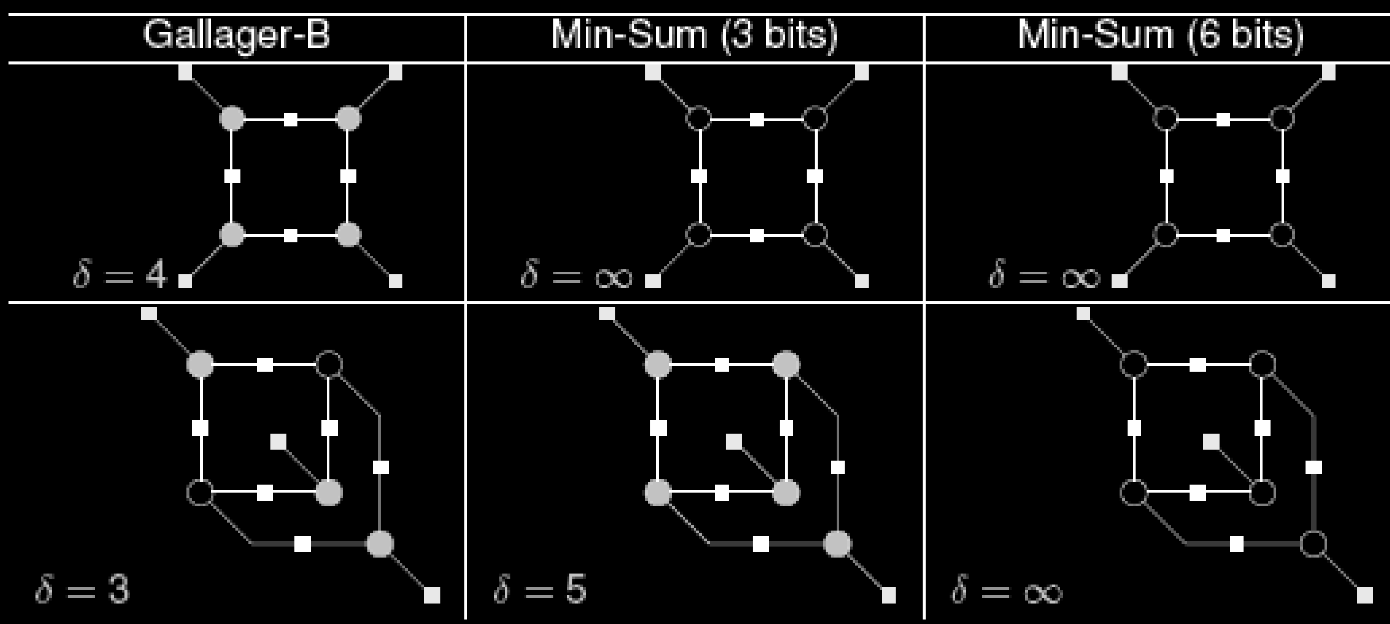
- The decoding failures for various algorithms on different channels are closely related
- Example BSC:



  - Bit flipping algorithm: $\{v_1, v_3\}, \{v_2, v_4\}, \{v_1, v_2, v_3\}\ldots$
  - Gallager A/B algorithm: $\{v_2, v_4, v_5\}$
  - LP decoder: $\{v_1, v_2, v_3, v_4, v_5\}$

Critical number

- The *critical number* m of a trapping set (for a given decoder) is the minimal number of variable nodes that have to be initially in error for the decoder to end up in that trapping set



Definitions

- *Definition 1: Let $T(\mathbf{y})$ denote the set of variable nodes that* are not eventually correct. If $T(\mathbf{y}) \neq \emptyset$, let $a = |T(\mathbf{y})|$ and b be the number of odd degree check nodes in the sub-graph induced by $T(\mathbf{y})$. We say $T(\mathbf{y})$ is an (a, b) trapping set.
- Note that for each failure of the iterative decoder, there is a corresponding set of corrupt variable nodes

$$F = \mathrm{supp}(\mathbf{x}^D)$$

- The set F is not necessarily a trapping set because it may not contain all the variable nodes that are eventually incorrect, such as variable nodes that oscillate between the right value and the wrong value.

Inducing sets and fixed sets

- *Definition 2:* Let T be a trapping set. If $\mathbf{T}(\mathbf{y}) = T$ then $\mathrm{supp}(\mathbf{y})$ is an ***inducing set*** of T.
- *Definition 3: Let* T be a trapping set and let $\mathbf{Y}(T) = \{\mathbf{y} \mid \mathbf{T}(\mathbf{y}) = T\}$. The ***critical number*** $m(T)$ of trapping set T is the minimal number of variable nodes that have to be initially in error for the decoder to end up in the trapping set T, i.e. $m(T) = \min_{\mathrm{Y}(\mathrm{T})} |\mathrm{supp}(\mathbf{y})|$
- *Definition 4:* The vector $\mathbf{y}$ is *a* ***fixed point*** of the decoding algorithm if $\mathrm{supp}(\mathbf{y}) = \mathrm{supp}(\mathbf{x}^l)$ for all l.
- *Definition 5:* If $T(\mathbf{y})$ is a trapping set and $\mathbf{y}$ is a fixed point, then $T(\mathbf{y}) = \mathrm{supp}(\mathbf{y})$ is called a ***fixed set***.

The (a,b) notation

- A *(a,b) trapping set* is a set of a variable nodes whose induced sub-graph has b odd degree checks
- The most important parameter – critical number:
  - The minimal number of variable nodes that have to be initially in error for the decoder to end up in the trapping set
- To "*end up*" in a trapping set means that (after a finite number of iterations) the decoder will be in error, on at least one variable node at every iteration

Trapping sets for column weight-three codes

- *Theorem* [Chillapagari *et al.*, (2009)]: (sufficient conditions) 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. The vector y is a fixed set if :

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

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

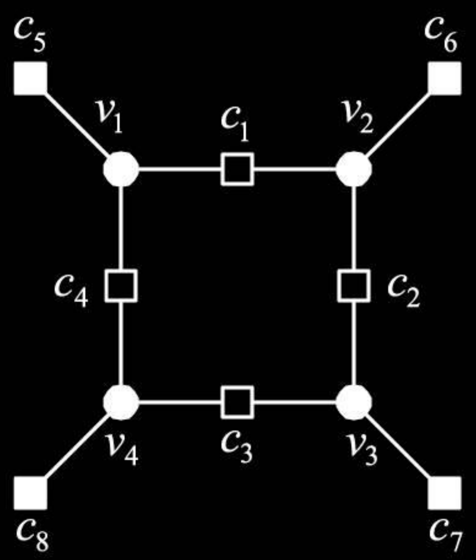
  (c) No two checks of O are connected to a variable node outside Γ.

More ambitious goal

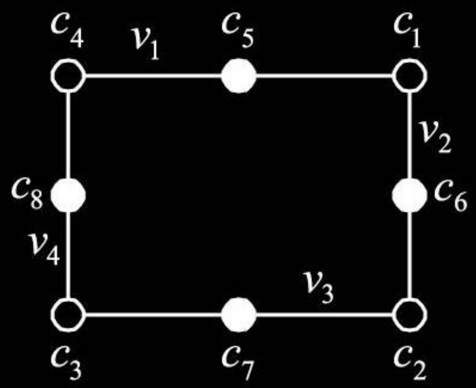
- The decoding failures for various algorithms on different channels are closely related and are dependent on only a few topological structures.
- These structures are either trapping sets for iterative decoding algorithms on the BSC or larger subgraphs containing these trapping sets.
- On the BSC, trapping sets are subgraphs formed by cycles or union of cycles.
- Ultimate goal: *Find topological interrelations among trapping sets/topological interrelations among error patterns that cause decoding failures for various algorithms on different channels.*

Graphical Representation

- Tanner graph representation

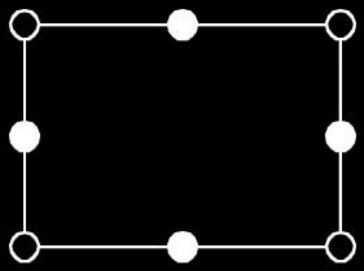


- Line and point representation

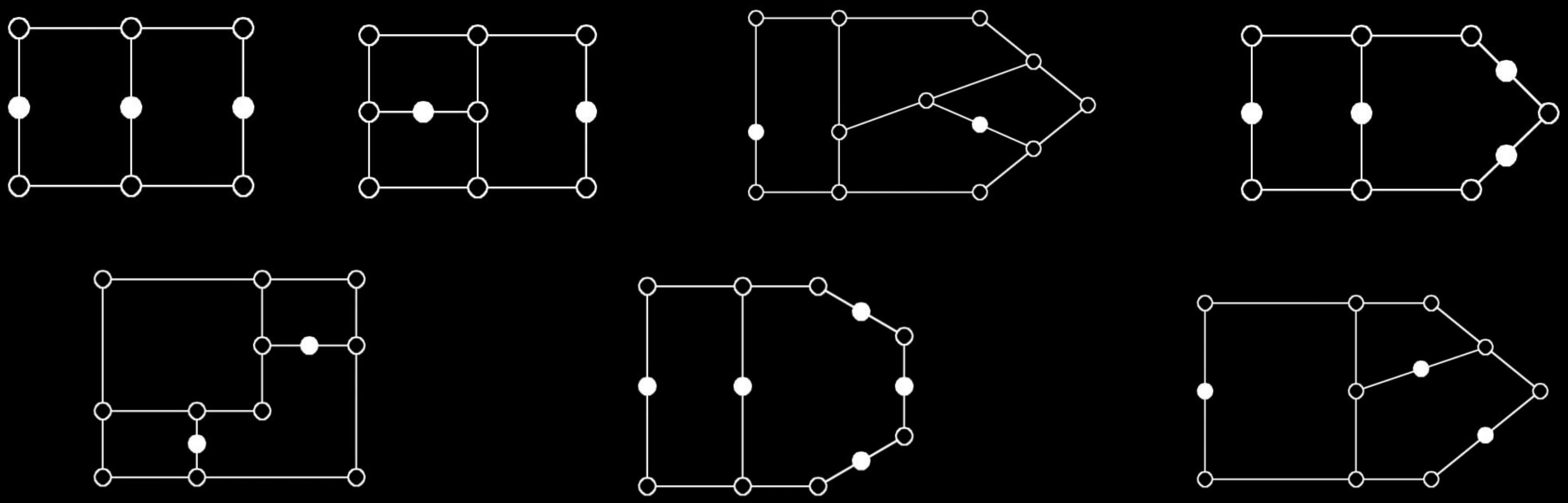


Trapping set ontology

- Parent

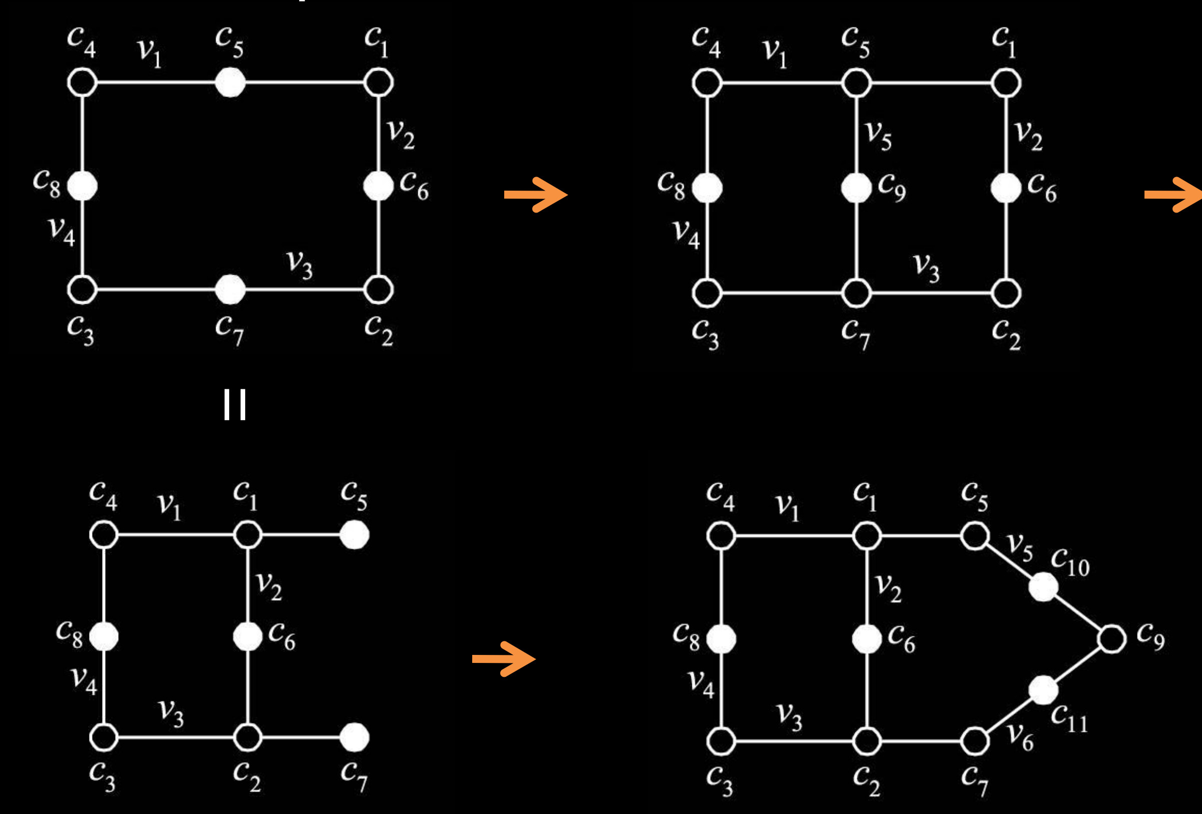


- Children



Trapping Set Ontology

- Children are obtained by adding lines to parents, changing the color of the points accordingly.
- Examples:



Evolution

On the critical number of trapping sets

- Conjecture:

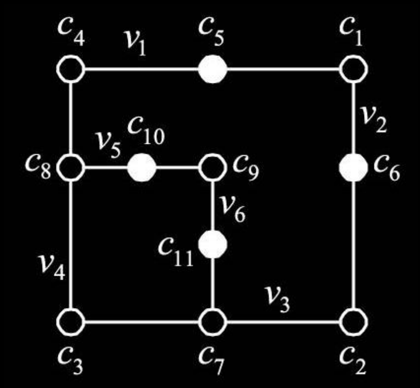
  The critical number of a trapping set T is upper bounded by the critical number of its parents.

- Relatively determine the harmfulness of a trapping set.

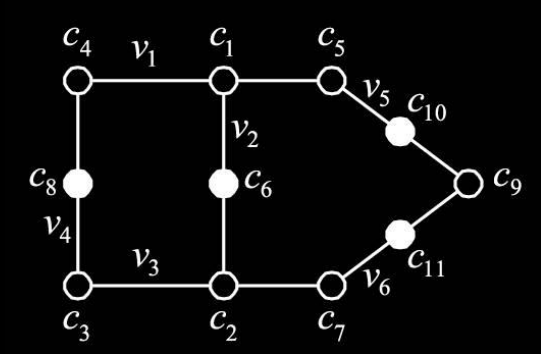
- Examples:

  - Two (6,4) trapping sets: different in number of inducing sets

Two 8-cycles
(more harmful)



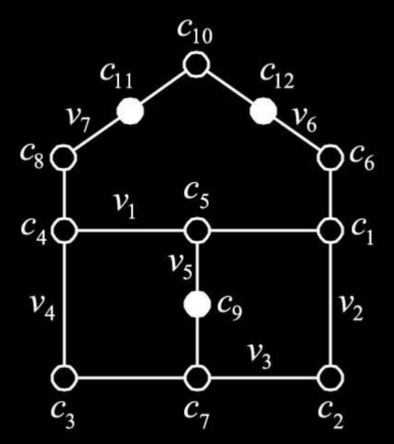
One 8-cycle
One 10-cycle
(less harmful)

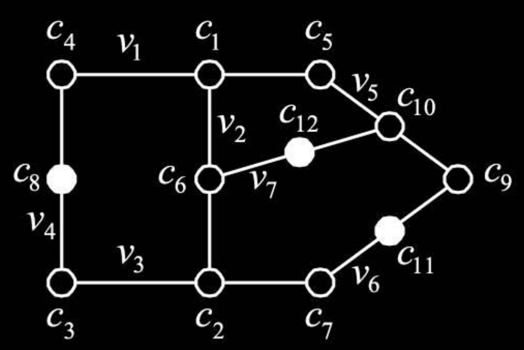


On the critical number of trapping sets

- Examples:

  - Two (7,3) trapping sets: different in critical number



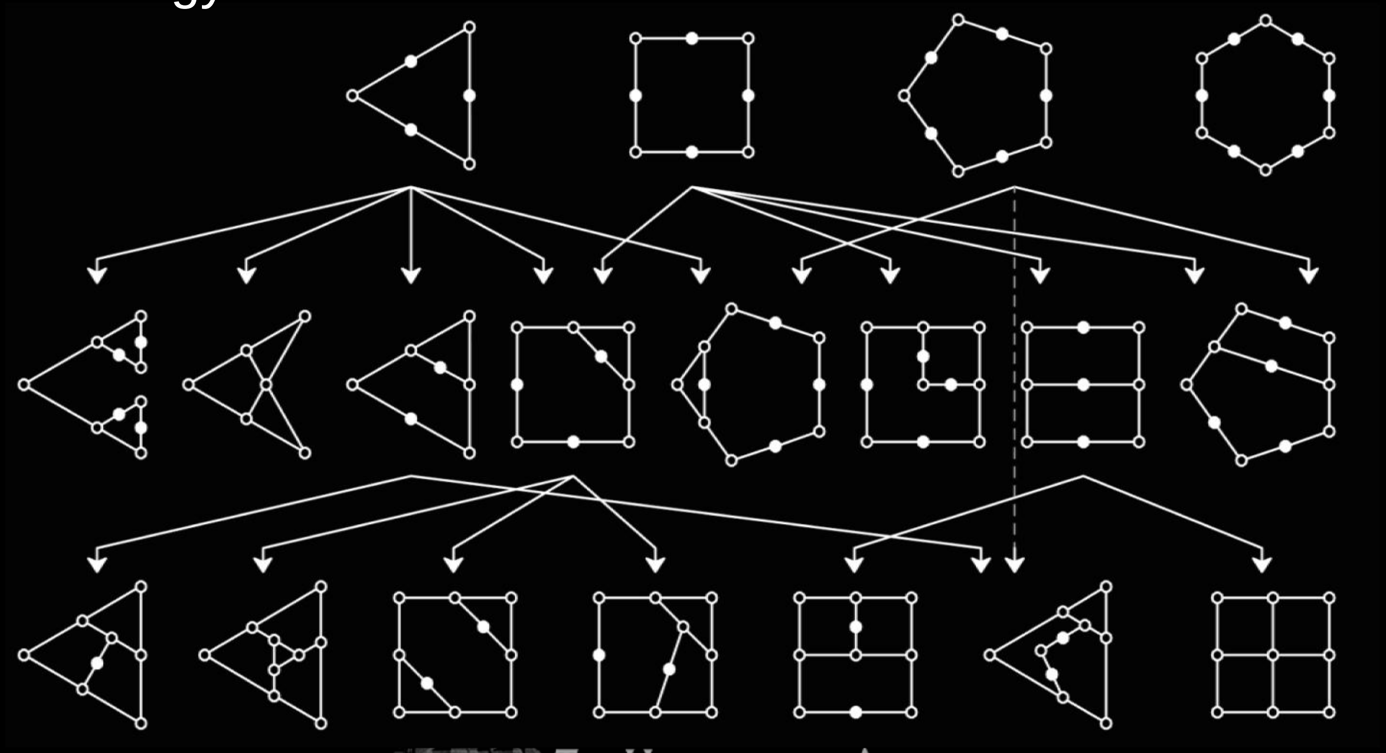


Child of (5,3)
Critical number = 3
(more harmful)

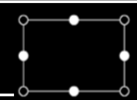
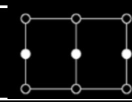
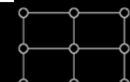
Child of (6,4)
Critical number = 4
(less harmful)

Trapping set ontology

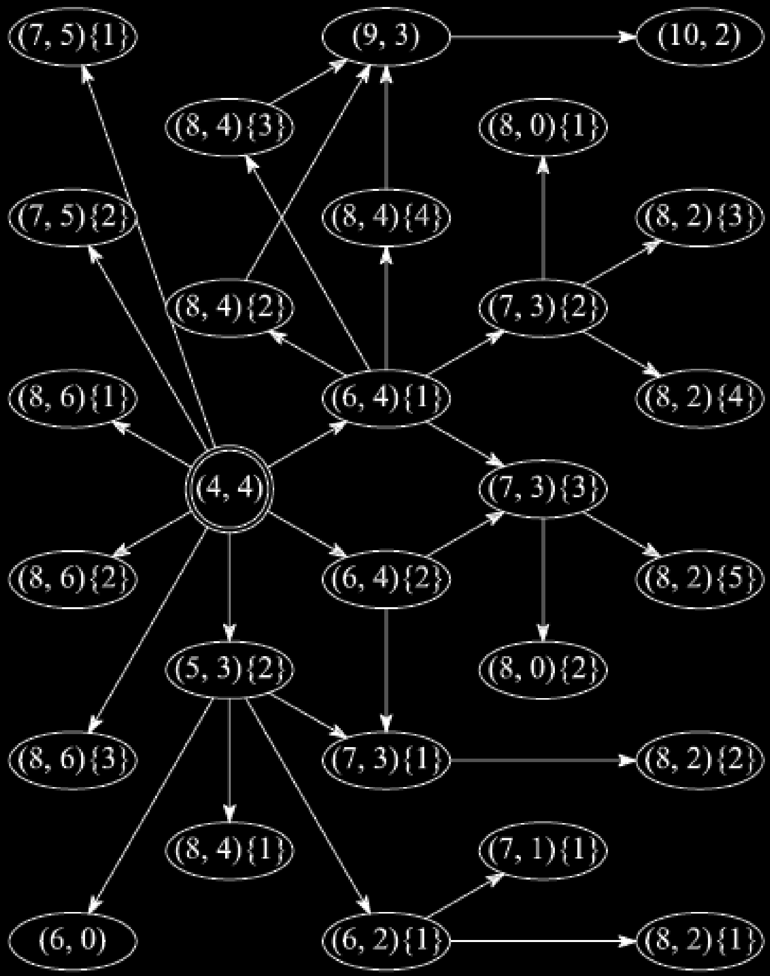
- Allerton 2009: trapping set ontology
- A database and software for systematic study of failures of iterative decoders on BSC
  http://www.ece.arizona.edu/vasiclab/Projects/CodingTheory/TrappingSetOntology.html



Number of trapping sets

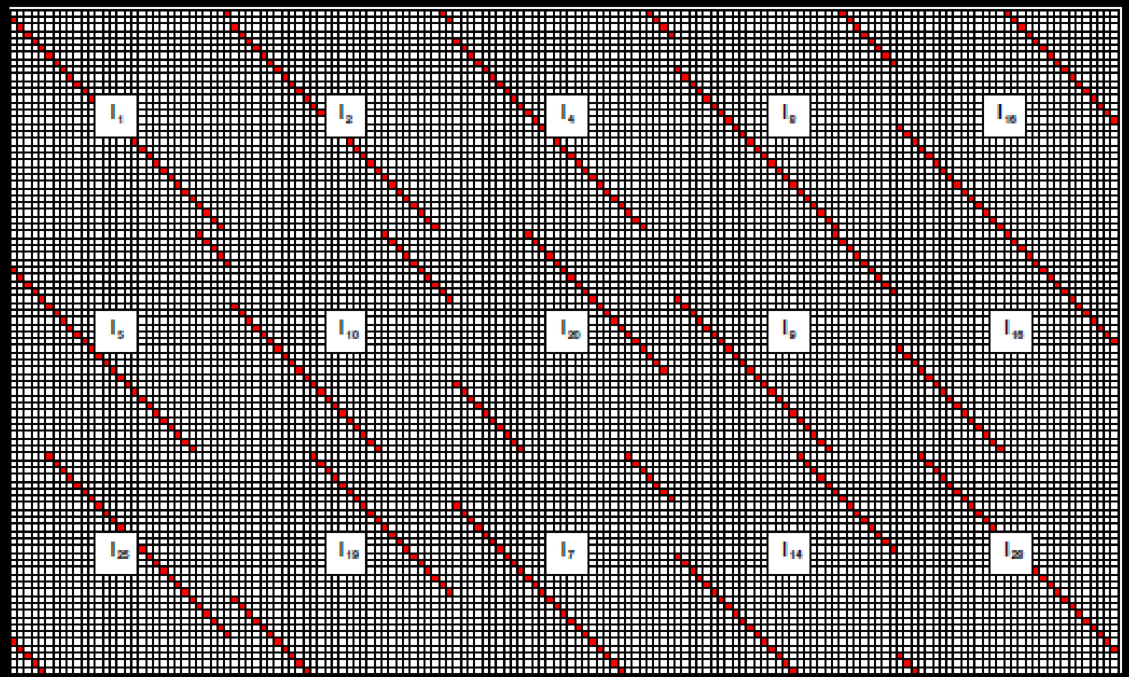
TS	#TS	g=6	g=8	g=10	g=12
(3,3)	1	1			
(4,4)	1				
(4,2)	1	1			
(4,0)	1	1			
(5,5)	1			1	
(5,3)	2	1			
(5,1)	1	1			
(6,6)	1				1
(6,4)	4	2			
(6,2)	4	3			
(6,0)	2	1			

Trapping Set Ontology

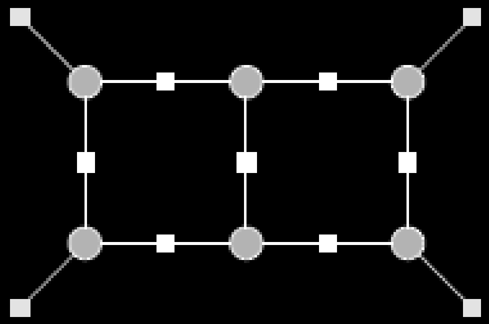


Example: Tanner code

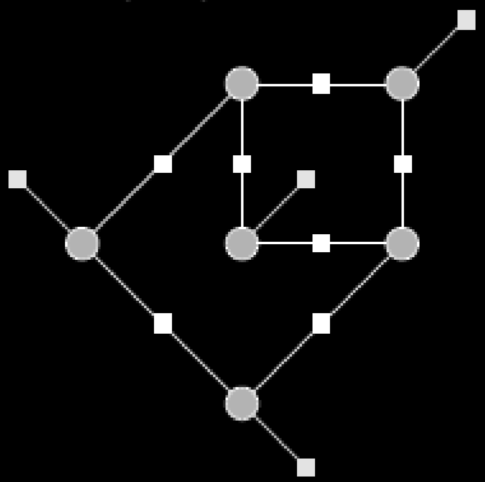
- A good test case (d_{min}=20, blocks of size 31, all codewords, trapping sets repeat 31 times)



Cycle inventory in different (*a*,*b*) topologies



TS(6,4) 2-0-1-0-0



TS(6,4) 1-2-0-0-0

Trapping set structure in Tanner code

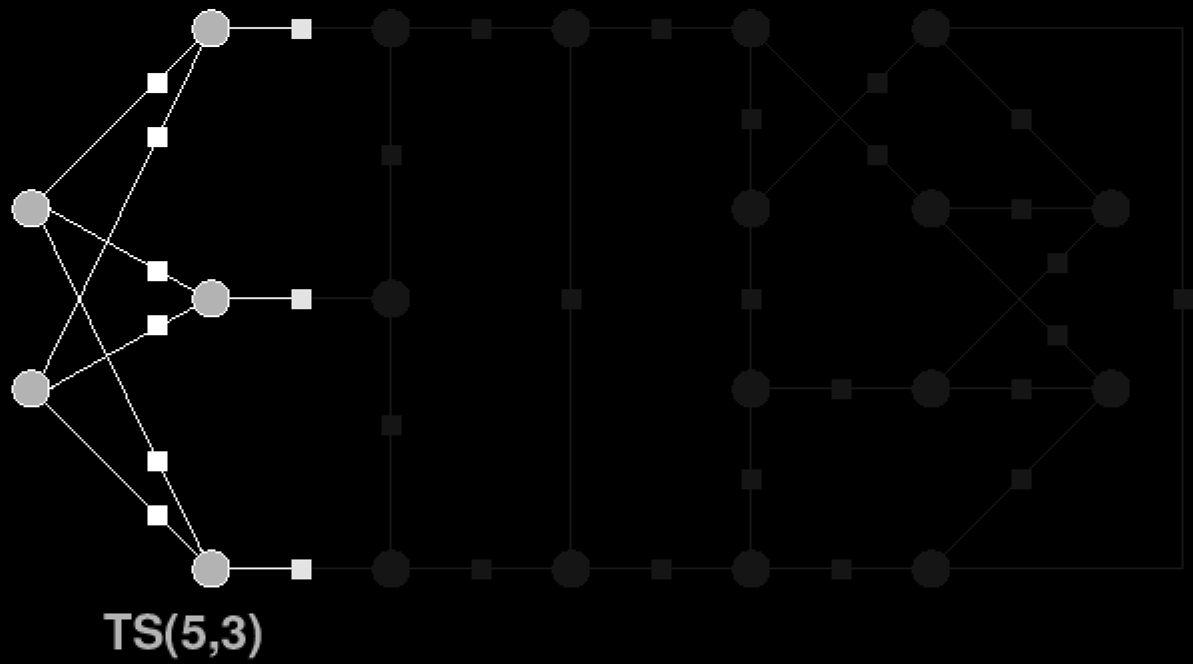
4 bits	(4,4) 1-0-0-0-0
5 bits	(5,3) 3-0-0-0-0
6 bits	(6,4) 1-2-0-0-0
7 bits	(7,3) 3-2-0-2-0
	(7,5) 1-1-0-1-0
	(7,5) 1-0-2-0-0

8 bits	(8,2) 3-4-2-4-2
	(8,4) 3-0-2-0-2
	(8,4) 1-3-1-1-1
	(8,4) 1-2-2-2-0
	(8,6) 1-0-1-0-1
	(8,6) 1-0-0-2-0

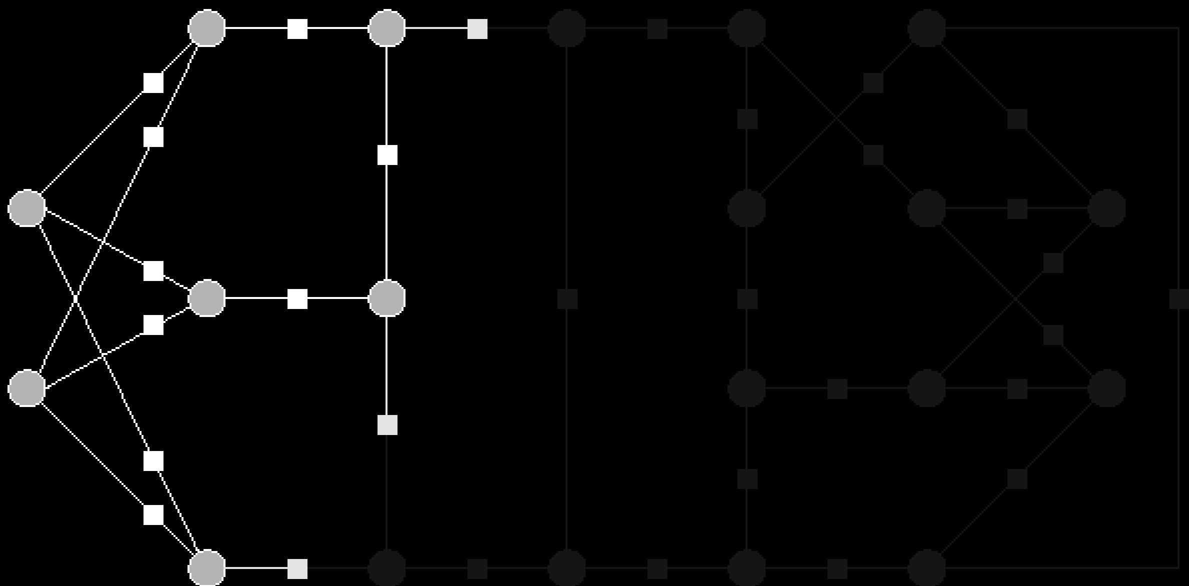
- 1023 weight-20 codewords belong in total
- Only 3 non-isomorphic graphs: (Types T1 T2 and T3).
  - Types T1 and T2 contain the minimal TS(5,3),
  - Type T3 does not contain the TS(5,3).

(155,64,20) Tanner code		
weight 20	$\rightarrow$	1023
weight 22	$\rightarrow$	6200
weight 24	$\rightarrow$	43865
weight 26	$\rightarrow$	$\simeq$ 259918

Codeword structure in Tanner code (1)

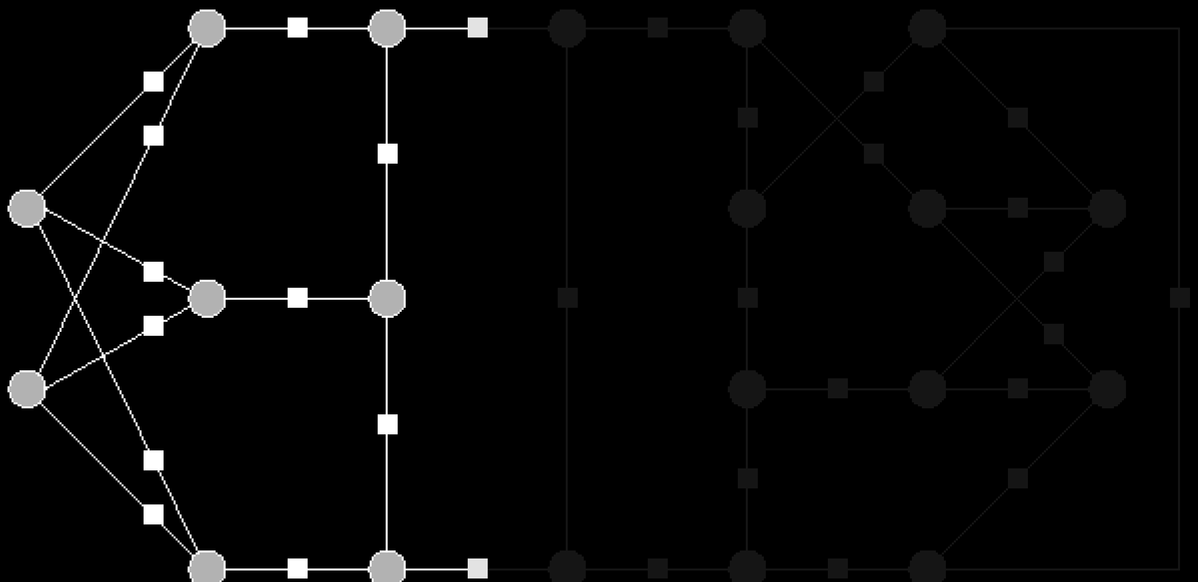


Codeword structure in Tanner code (2)



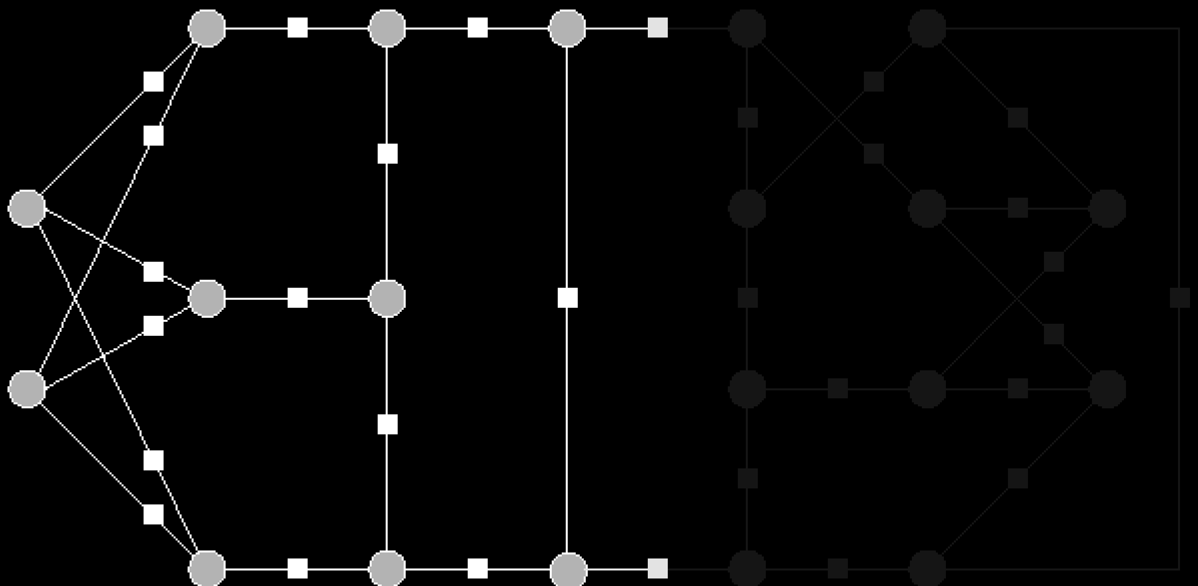
TS(7,3)

Codeword structure in Tanner code (3)



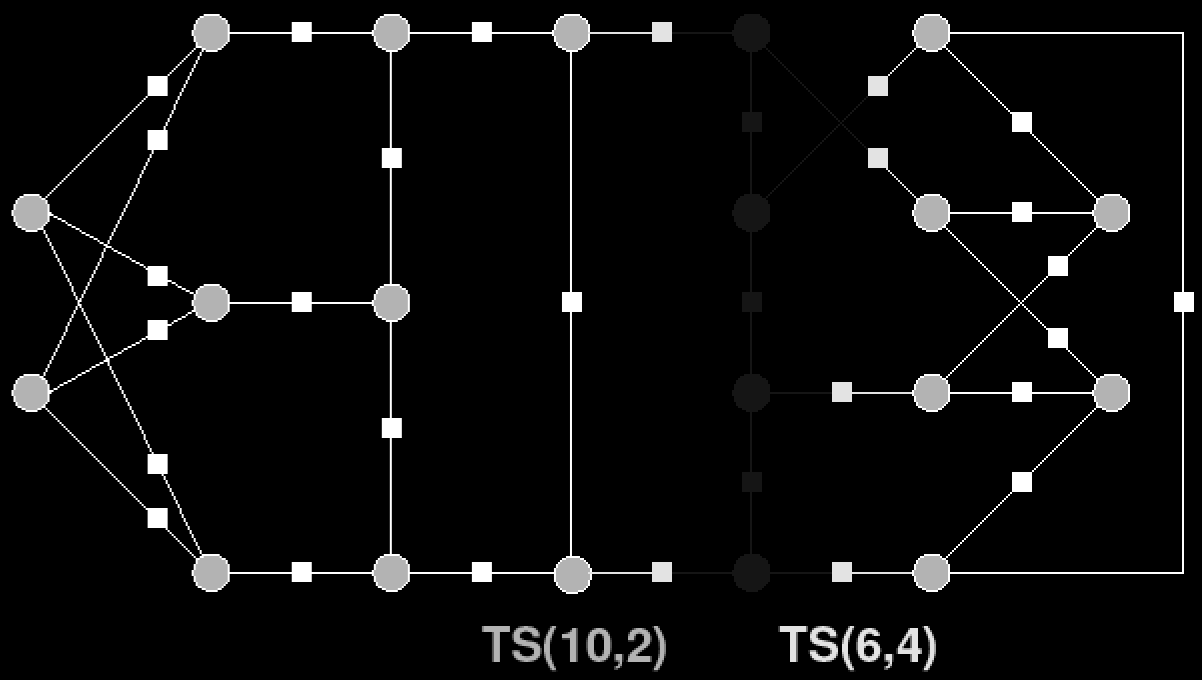
TS(8,2)

Codeword structure in Tanner code (4)

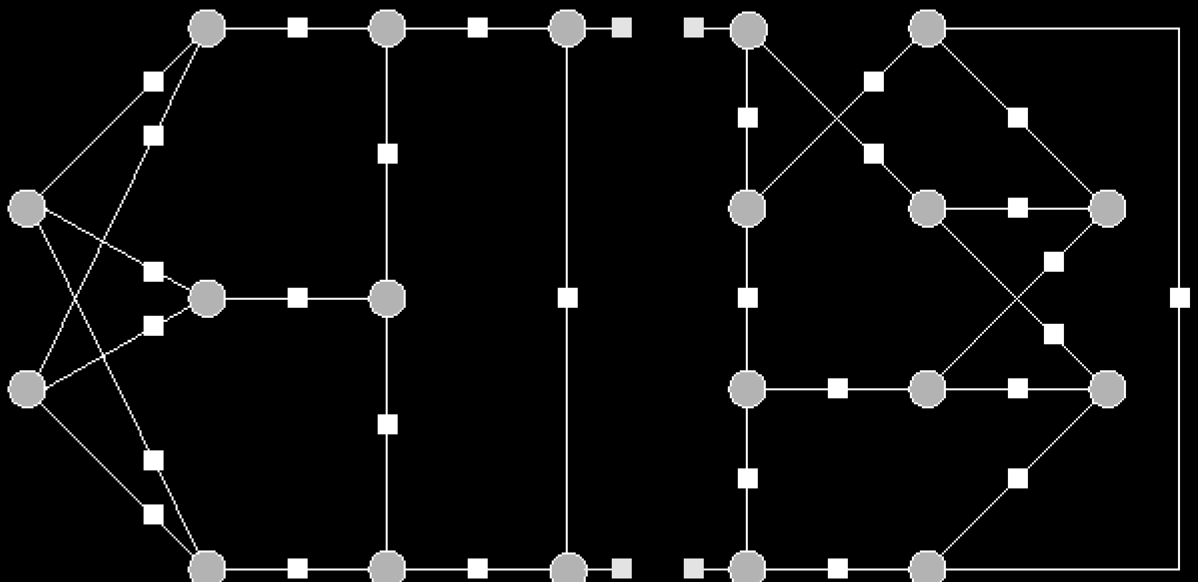


TS(10,2)

Codeword structure in Tanner code (5)



Codeword structure in Tanner code (6)



TS(10,2) TS(10,2)

Searching for trapping sets

- Trapping sets are searched for in a way similar to how they have evolved in the Trapping Set Ontology.
- Since the induced subgraph of every trapping set contains at least a cycle, the search for trapping sets begins with enumerating cycles.
- After cycles are enumerated, they will be used in the search for bigger trapping sets.
- A bigger trapping set can be found in a Tanner graph by expanding a smaller trapping set.

Searching for trapping sets

- For example: Suppose that $\mathcal{T}_2$ is a direct successor of $\mathcal{T}_1$, and that all $\mathcal{T}_1$ trapping sets have been enumerated. In order to enumerate $\mathcal{T}_2$ trapping sets, we search for sets of variable nodes such that the union of such a set with a trapping set $\mathcal{T}_1$ form a $\mathcal{T}_2$ trapping set.

- The complexity of the search for trapping sets in the Tanner graph of a structured code can be greatly reduced by utilizing the structural property of its parity-check matrix.

Searching for trapping sets

NUMBER OF CYCLES AND TRAPPING SETS OF THE TANNER CODE AND RUN-TIME OF THE SEARCHING ALGORITHMS ON A 2.3 GHZ COMPUTER

Trapping Sets	Total	Run-time (Seconds)	
		(i)	(ii)
6-cycles	0	0.024	0.004
8-cycles	465		
10-cycles	3720		
$(5,3)\{2\}$	155	0.004	0.001
$(6,4)\{2\}$	930	0.023	0.001
$(7,3)\{1\}$	930	0.008	0.002
$(8,2)\{1\}$ and $\{2\}$	465	0.008	0.001

Searching for Trapping Sets

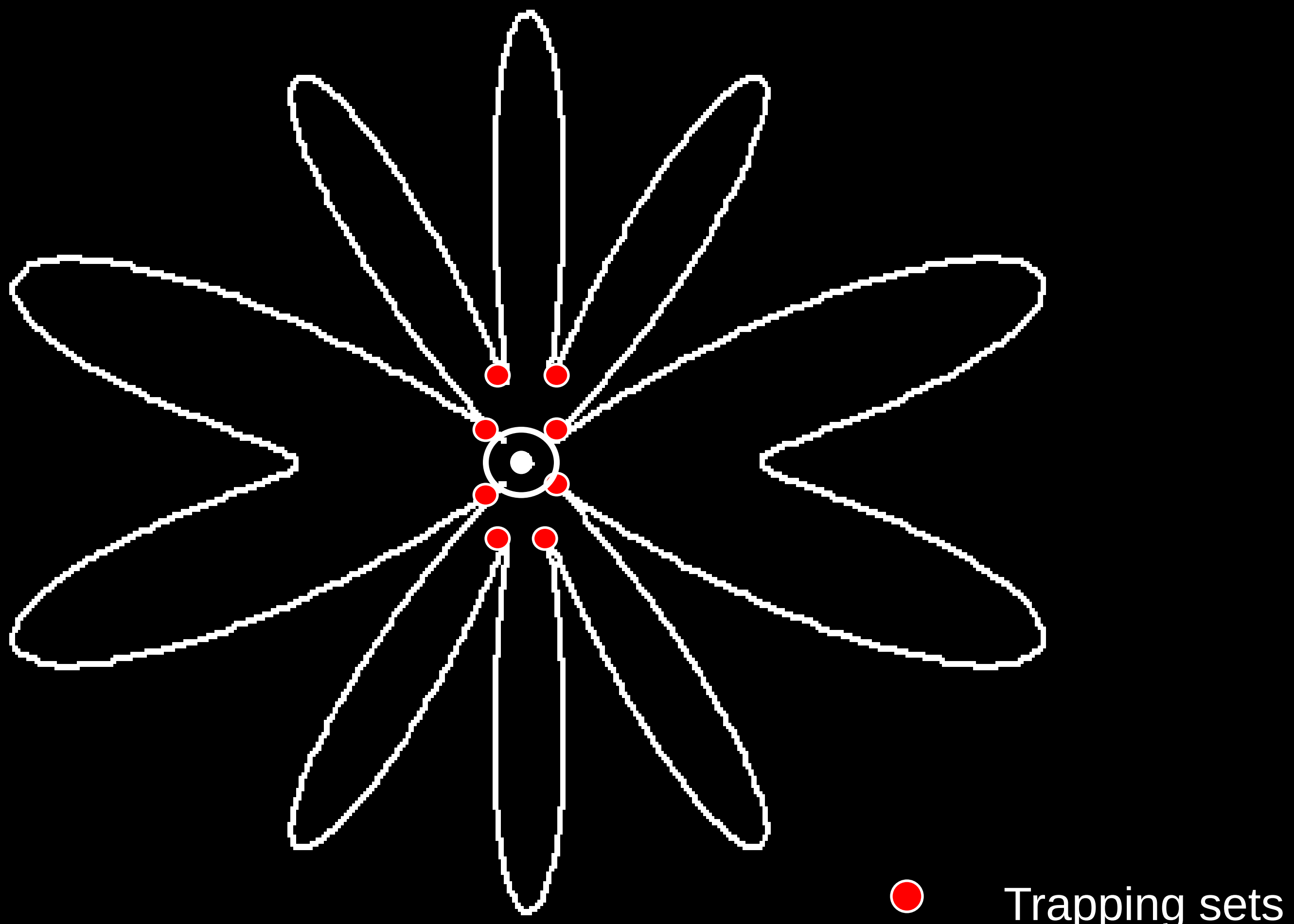
NUMBER OF CYCLES AND TRAPPING SETS OF THE CODE $\mathcal{C}_2$ AND RUN-TIME OF THE SEARCHING ALGORITHMS ON A 2.3 GHZ COMPUTER

Trapping Sets	Total	Run-time (Seconds)	
		(i)	(ii)
6-cycles	0	1.362	0.109
8-cycles	17066		
10-cycles	183433		
$(5,3)\{2\}$	1590	0.130	0.004
$(6,2)\{1\}$	424	0.009	$< 10^{-8}$
$(7,3)\{1\}$	6254	0.260	0.007
$(8,2)\{1\}$	1166	0.160	0.002
$(8,2)\{2\}$	901	0.033	0.002
$(6,4)\{1\}$	85065	85.437	1.037
$(6,4)\{1\}$ and $\{2\}$	148983	273.854	0.232
$(7,3)\{2\}$ and $\{3\}$	23850	5.750	0.045
$(8,2)\{3\}$, $\{4\}$ and $\{5\}$	5936	0.409	0.015

$\mathcal{C}_2$: Quasi-cyclic code, $n = 530$, $R = 0.7$.

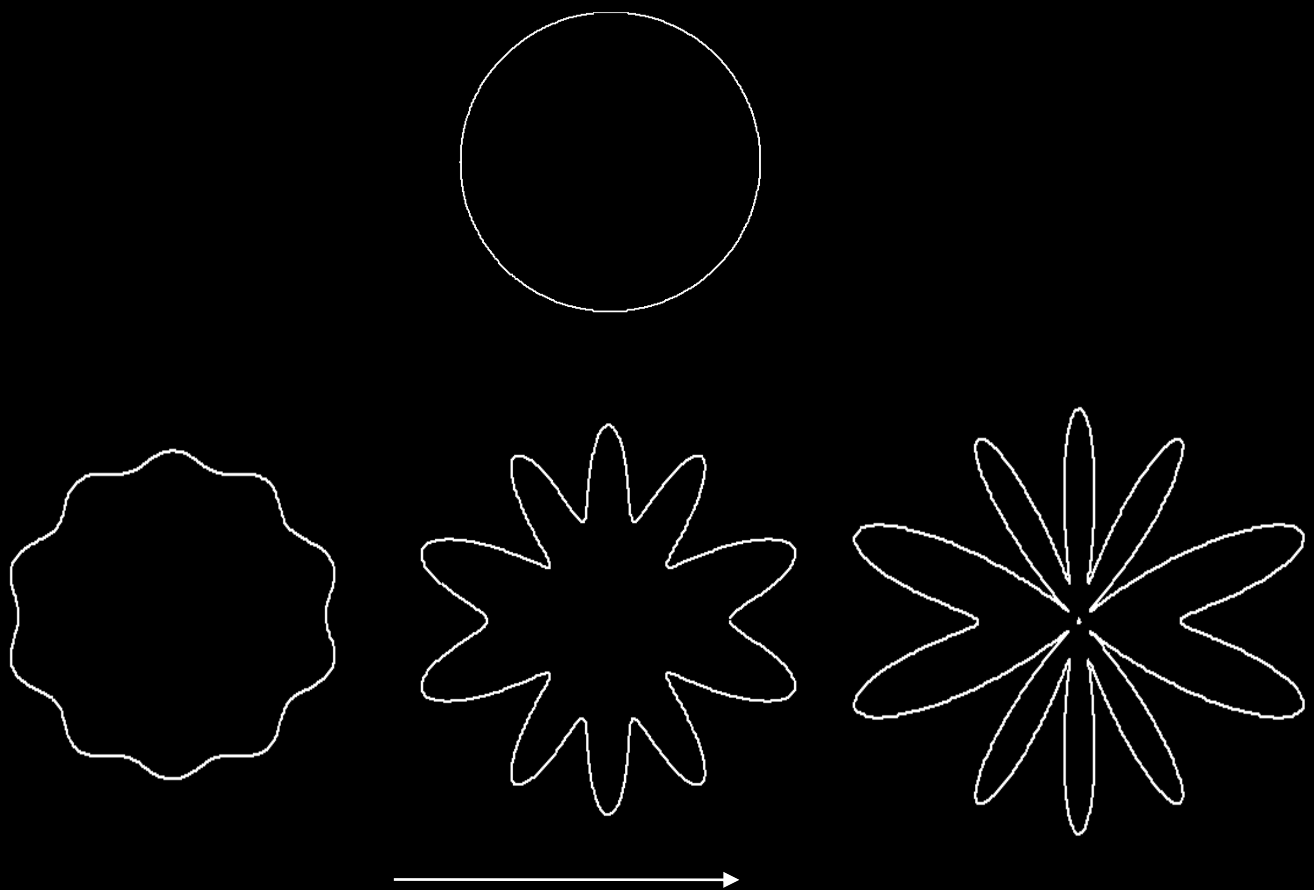
How many errors can a column weight three code correct under iterative decoding?

Instantons and trapping sets



THE UNIVERSITY OF ARIZONA
TUCSON ARIZONA

Failures of Iterative Decoders



Variable degree decrease

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 are capable 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 $d_v = 3$ can correct a fraction of errors
- To correct linear fraction of errors the expansion factor of ¾ is necessary, but the best expansion factor achievable by $d_v = 3$ codes is 1-1/d_v $= 2/3$

Correcting fixed number of errors

- Bounded distance decoders (trivial)
  - A code with minimum distance $2t+1$ can correct t errors
- Iterative decoding on BEC (solved)
  - Can recover from t erasures if the size of minimum *stopping set* is at least $t+1$
- Iterative message passing decoding on BSC (unknown)
  - Error floor

$$\log(FER(\alpha)) \approx \log(c_i) + i\log(\alpha)$$

c_k - the number of configurations of received bits for which k channel error lead to a codeword (frame) error

Trapping sets - sufficient conditions

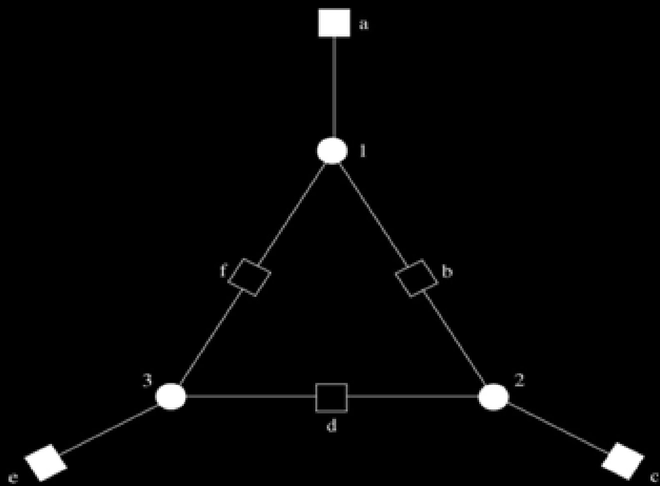
- *Theorem 1: Let C be a code in the ensemble of $(3, \rho)$* regular LDPC codes. Let Γ be a subgraph induced by the set of variable nodes T. Let the checks in Γ can be partitioned into two disjoint subsets: E consisting of checks with even degree, and O consisting of checks with odd degree. y is a fixed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

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

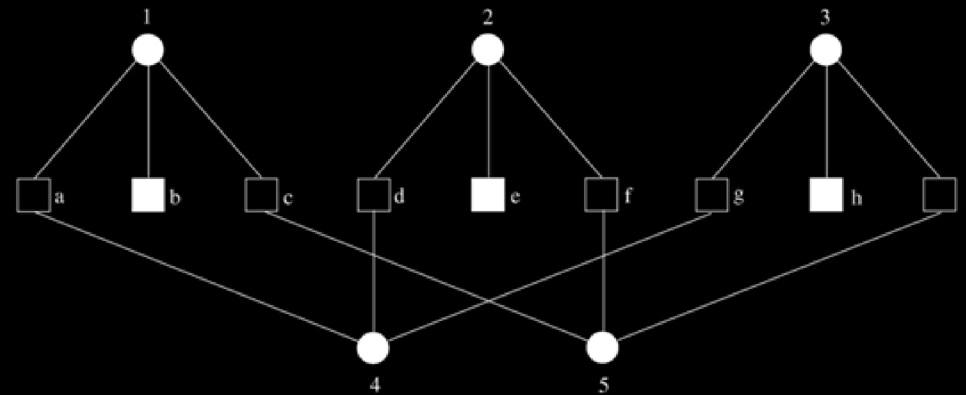
  (b) Every variable node in Γ is connected to at least two checks in E ,

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

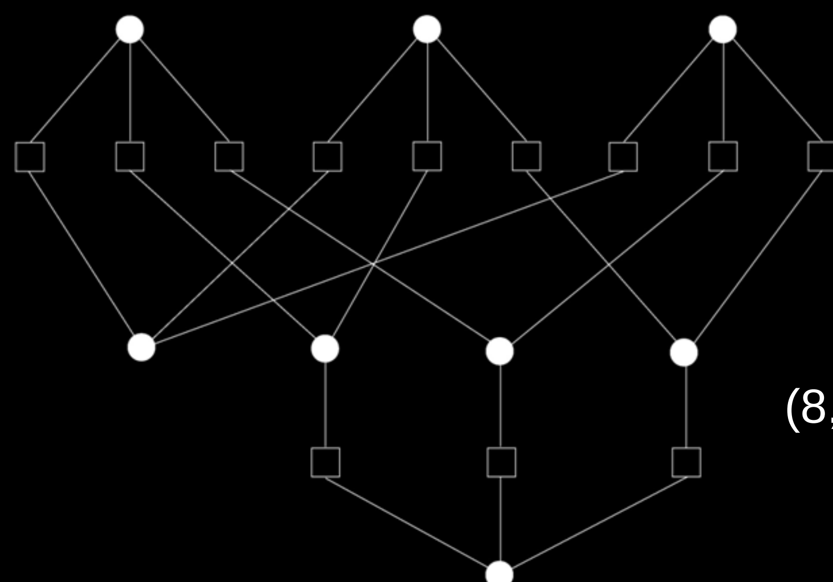
Trapping sets: examples



(3,3) trapping set



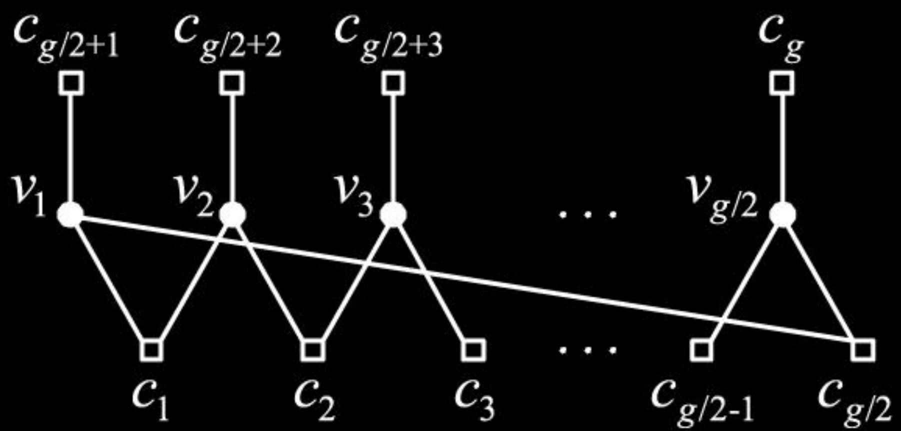
(5,3) trapping set



(8,0) Trapping Set

The upper bounds

- *Theorem 2:* Let C be an $(n,\ 3,\ \rho)$ regular LDPC code with girth g. Then:
  - If $\mathrm{g} = 4$, then C has at least one FS of size 2 or 3.
  - If $\mathrm{g} = 6$, then C has least one FS of size 3 or 4.
  - If $g = 8$, then C has at least one FS of size 4 or 5.
  - If $\mathrm{g} \geq 10$, then the set of variable nodes $\{v_1, v_2, \ldots, v_{g/2}\}$ involved in the shortest cycle is a TS of size $g/2$.
- By Theorem 1, $\{v_1, v_2, \ldots, v_{\mathrm{g}/2}\}$ is the support of a fixed point.



Consequences

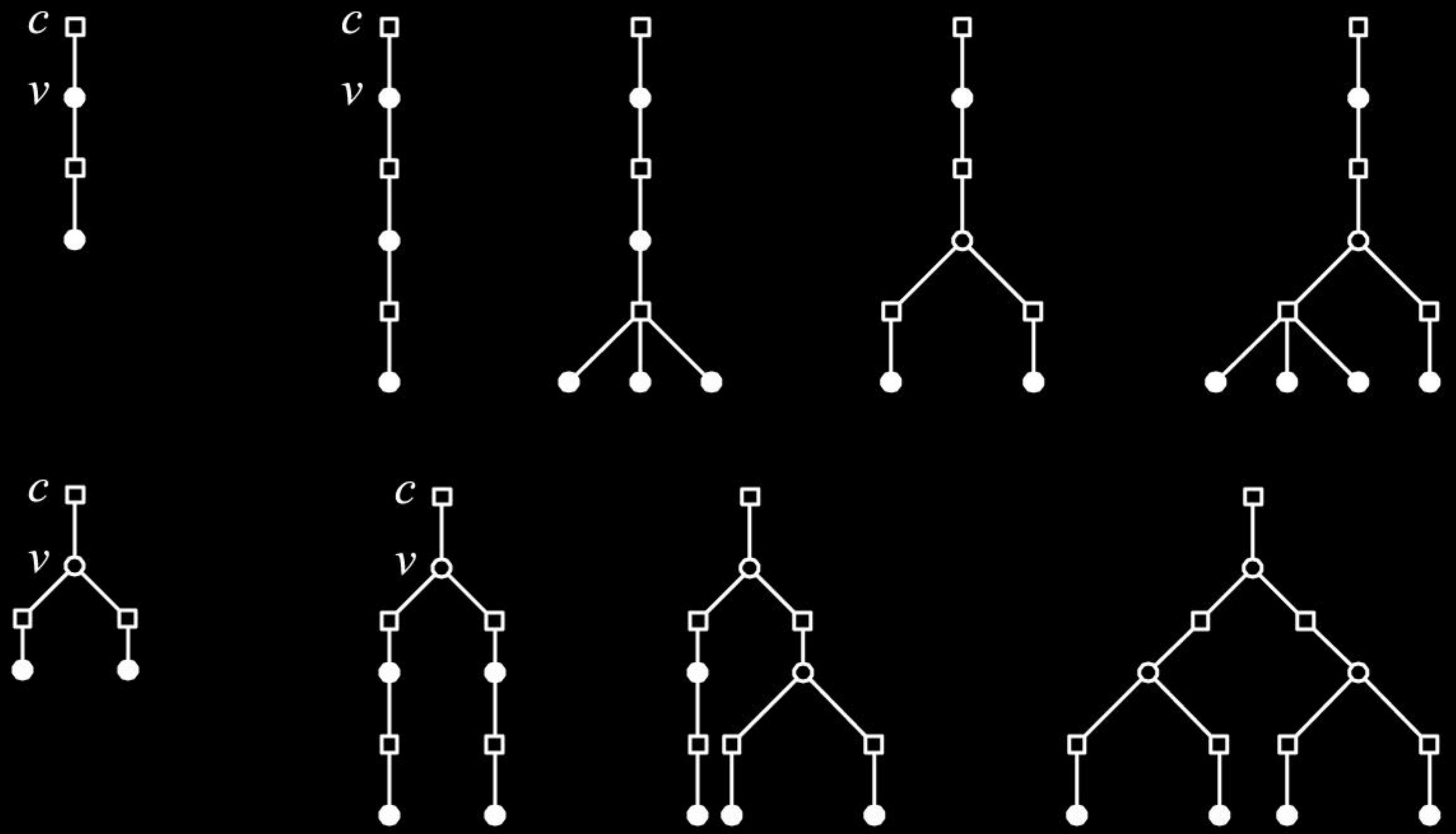
- For column weight three codes, the weight of correctable error patterns under Gallager A algorithm grows only linearly with girth

- For any $\alpha > 0$ and sufficiently large block lengths n, no code in the $\mathrm{C}^n(3, \rho)$ ensemble can correct all αn errors under Gallager A algorithm

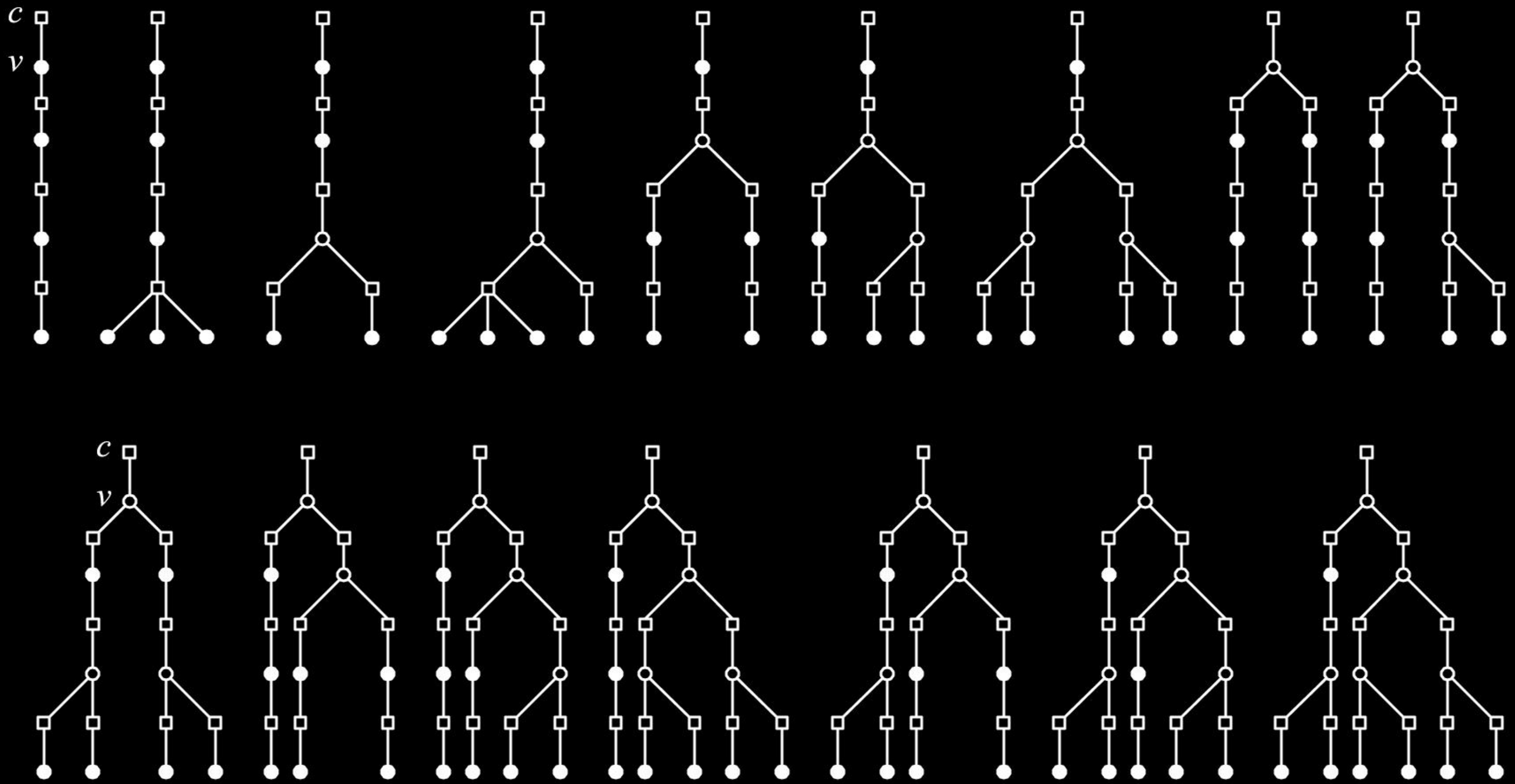
The lower bound lemmas

- *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.

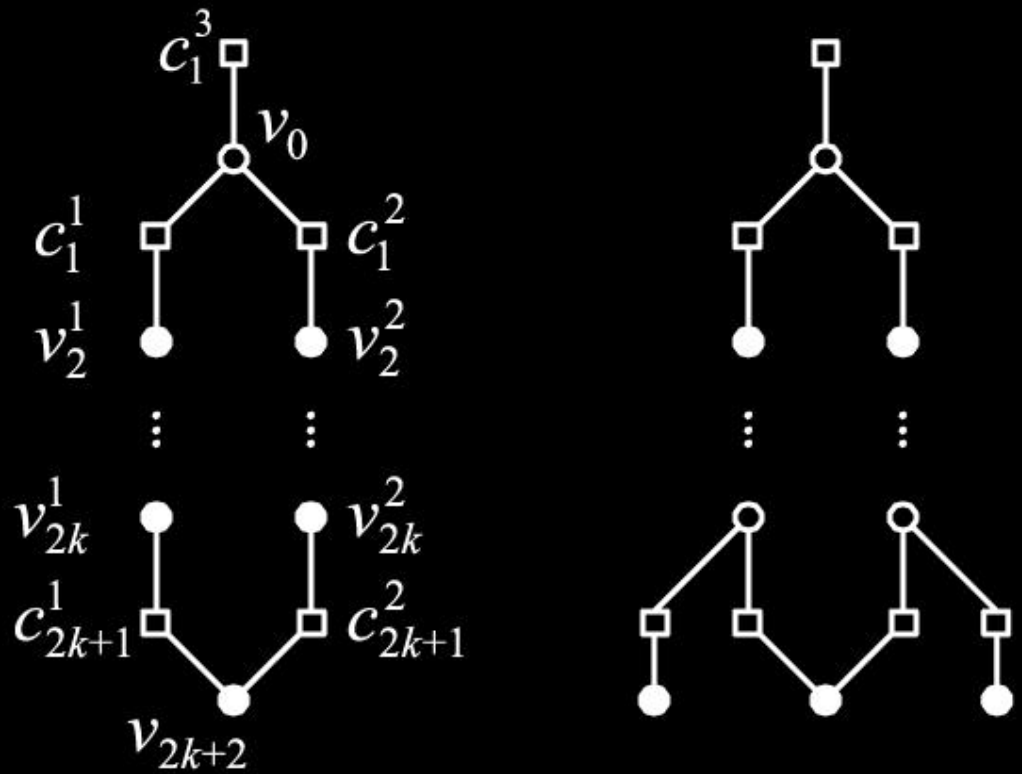
Bad configurations ($k=1$ and $k=2$)



Bad configurations ($k=3$)



Configurations not converging in $k+1$ iterations



Finding all failures of Gallager A decoder

- A fundamental question: what are all the k error patterns that the Gallager A fails to correct?
- Modified decoders can be designed to correct such error patterns
- Only partial answer form previous analysis: k variables involved in a cycle of length $2k$
  - k variables that form a fixed set
- More complicated cases possible
  - Tanner graphs with high girth also contain structures other than cycles

The Moore Bound

- *Theorem 8:* For all $k < n_0(\gamma/2, g')$, any set of k variable nodes in a $\gamma \geq$ *4-left regular* Tanner graph with girth 2g' expands by a factor of at least $3\gamma/4$.

- *Corollary 1: Let C be an LDPC code with column-weight* $\gamma \geq 4$ and girth 2g′. Then the bit flipping algorithm can correct any error pattern of weight less than $n_0(\gamma/2, g')/2$.

- $n_0(d, g)$ - the *Moore bound . A lower* bound on the least number of vertices in a d-regular graph with girth g.

Cage Graphs

- A (d, g)-*cage graph, G(d, g), is a d-regular* graph with girth *g* having the minimum possible number of nodes.

- *Theorem 10: Let C be an LDPC code with γ-left regular* Tanner graph G and girth 2g′. Let T (γ, 2g′) denote the size of smallest possible potential trapping set of C for the bit flipping algorithm. Then,

$$|T(\gamma, 2g')| = n_c(\lceil \gamma/2 \rceil, g').$$

- *Theorem 11: There exists a code C with γ-left regular* Tanner graph of girth 2g′ which fails to correct $n_c(\lceil \gamma/2 \rceil, g')$ errors.

Comments

- For γ=*3* and γ=*4,* the above bound is tight.
- Observe that for d=2, the Moore bound is $n_0(d, g)=g$ and that a cycle of length 2*g* with *g* variable nodes is always a potential trapping set.
- For a code with γ=3 or 4, and Tanner graph of girth greater than eight, a cycle of the smallest length is always a trapping set.

Refined Expansion

- *Theorem : An LDPC code with column-weight four and girth six can correct three errors in four* iterations of message-passing decoding if and only if the conditions, $4 \rightarrow 11$, $5 \rightarrow 12$, $6 \rightarrow 14$, $7 \rightarrow 16$ and $8 \rightarrow 18$ are satisfied.

- $y \rightarrow z$ means that any set of y variable nodes has at least z neighbors

Summary

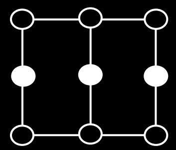
- Introduced LDPC codes, Tanner graphs, iterative decoders
- For BEC showed how to analyze failures using the concept of stopping sets
- For BSC introduced trapping sets and showed how to enumerate them.

Extra slides

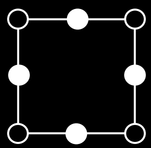
Error floor

Critical number

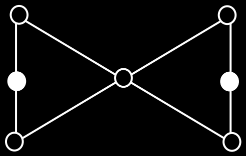
- With every trapping set T is associated a *critical number* m (or $m(\mathrm{T})$) defined as the minimum number of nodes in $\mathcal{T}$ that have to be initially in error for the decoder to end in that trapping set.
- Smaller values of m mean that fewer number of errors can result in decoding failure by ending in that trapping set.



m=3



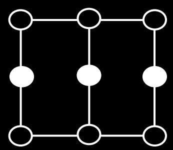
m=4



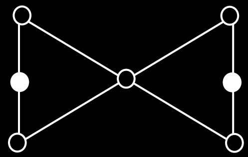
m=3

Strength of a trapping set

- Not all configurations of m errors in a trapping set result in a decoding failure.
  - (5, 3) TS: m=3, only one configuration of three errors leads to a decoding failure.
  - (4, 2) TS: m=3m all the four combinations of three errors lead to decoding failure.



m=3



m=3

- A set of m erroneous variable nodes which leads to a decoding failure by ending in a trapping set $\mathcal{T}$ of class X is called a *failure set* of X .
- The number of failure sets of T is called the *strength of* T and is denoted by s. A class X has $s|X|$ failure sets.

Approximation

- The contribution of each class of trapping set:

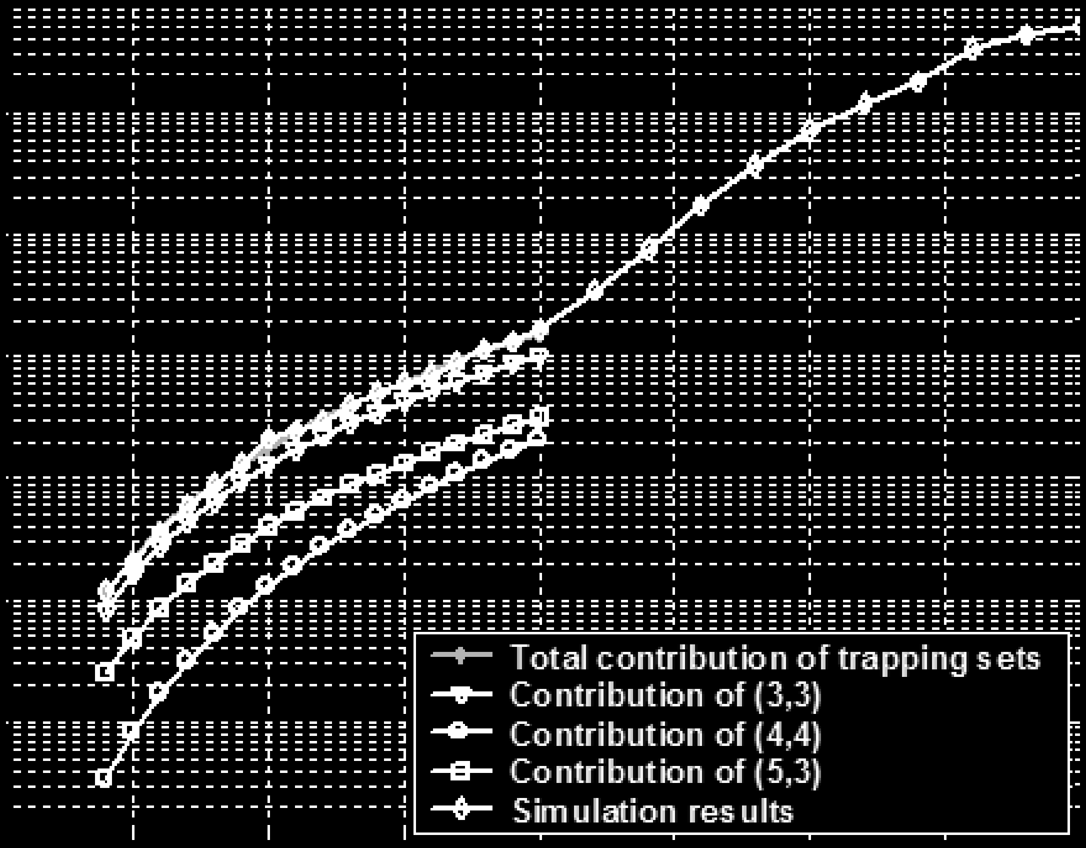
$$\Pr(\chi) = \sum_{r=m}^{M} \Pr(\chi \mid r-\text{errors}) \cdot \Pr(r-\text{errors})$$

$$\Pr(\chi \mid r-\text{errors}) = \frac{s\,|\chi|}{\binom{n}{m}} \cdot \binom{r}{m}$$

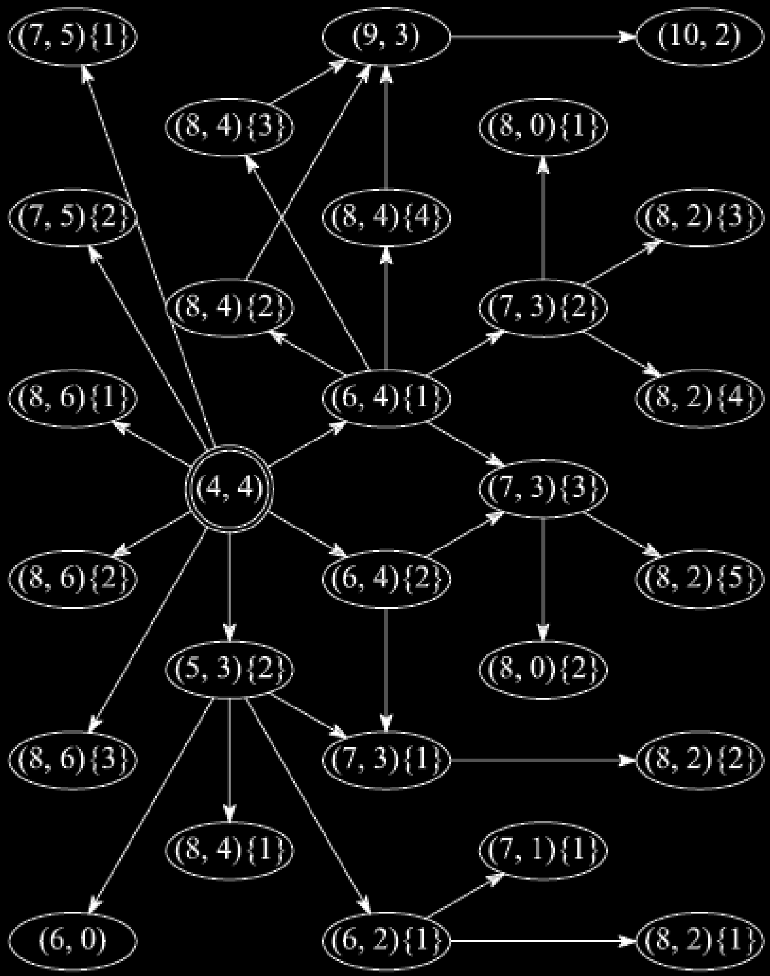
$$\Pr(r-\text{errors}) = \binom{n}{r} \cdot \alpha^{r} \cdot (1-\alpha)^{n-r}$$

- $s \cdot |\chi| / \binom{n}{m}$ is the probability that a given set of *m* variable nodes is a failure set of class χ.
- There are $\binom{r}{m}$ such subsets with cardinality m for a set with r elements (this probability is computed using the structure of Tanner graph).

FER contribution of different error patterns



Designing better codes using trapping sets



Quasi-cyclic codes

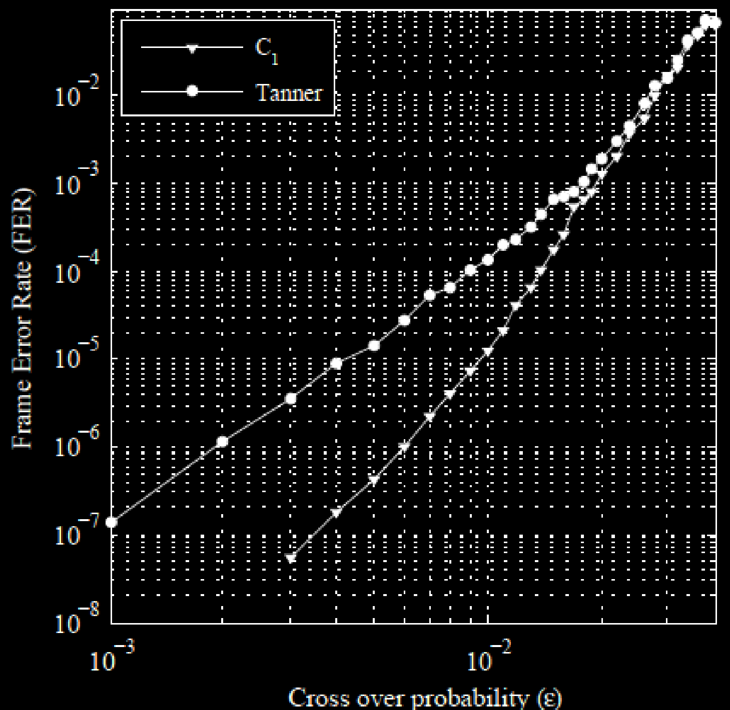


Fig. 7. Frame error rate performance of the Tanner code and code $\mathcal{C}_1$ under the Gallager A algorithm on the BSC.

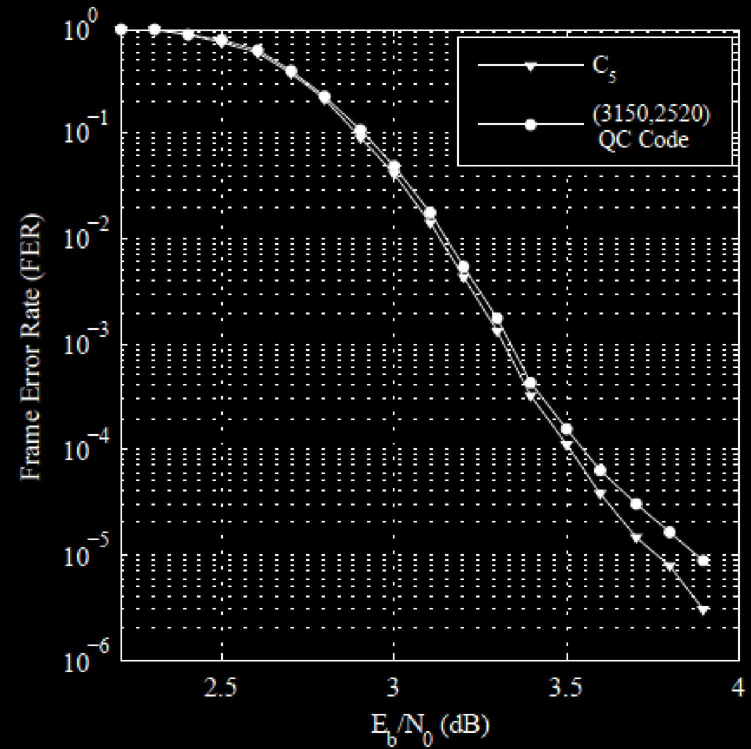
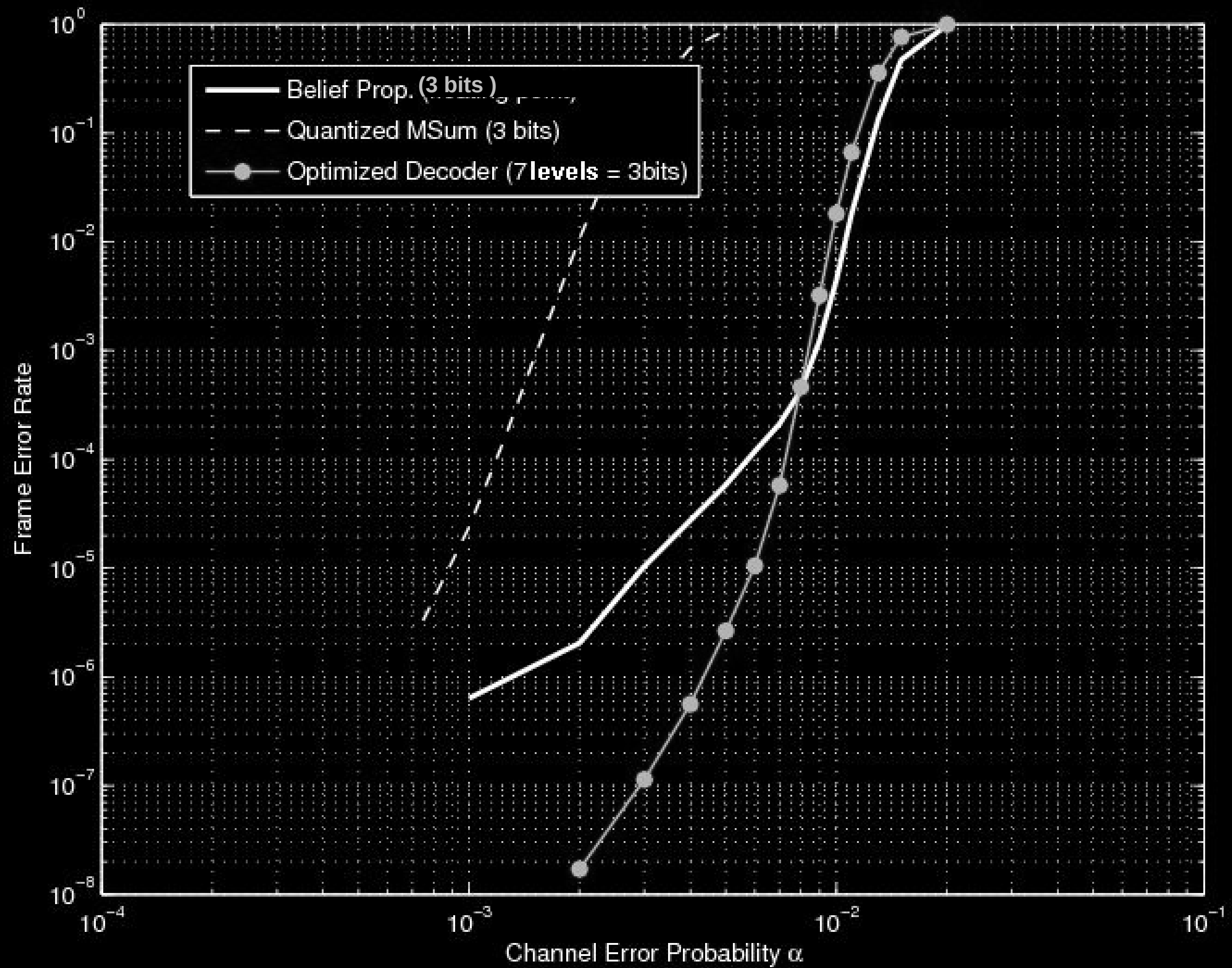


Fig. 18. Frame error rate performance of codes in Example 5 under the SPA on the AWGNC.

Designing better decoders



Multi-bit iterative decoders

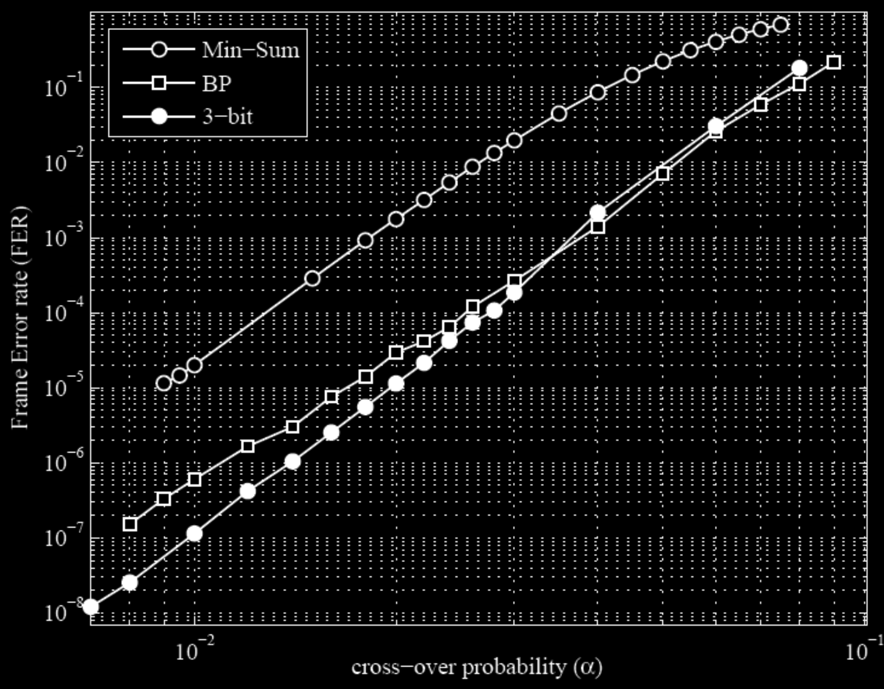
- Gallager-like algorithms, but the messages are binary vectors of length m, $m>1$.
- Variable and check node update functions – Boolean
  - no infinite number of bits for intermediate computations
- Given m bit-messages, one wants to chose the Boolean functions to guarantee correction of k errors in I *iterations*.
- We present 2-bit and 3-bit decoders
- On BSC, our decoders outperform the belief propagation (BP) decoder in the error floor region.
- More importantly, they achieve this at only a fraction of the complexity of the BP decoder.

3-bit decoder that surpasses BP

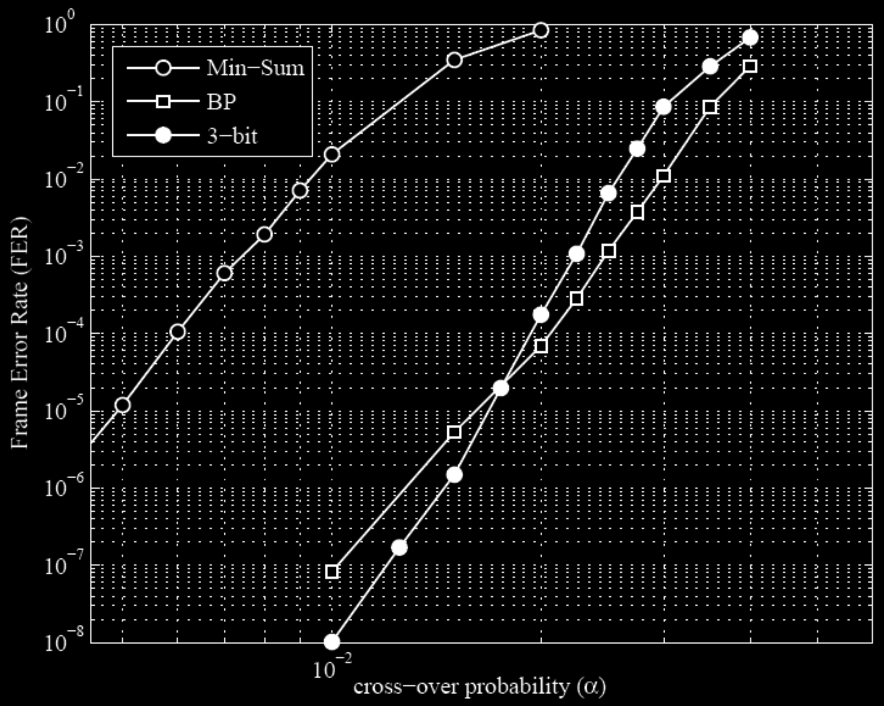
m_1	m_2	r	m_o
010	010	0	100
010	010	1	000
010	100	0	100
010	100	1	010
010	110	0	110
010	110	1	100
010	000	0	010
010	000	1	000
010	011	0	010
010	011	1	011
010	101	0	011
010	101	1	101
010	111	0	101
010	111	1	111
100	100	0	110
100	100	1	100
100	110	0	110
100	110	1	110
100	000	0	100
100	000	1	010
100	011	0	100
100	011	1	010
100	101	0	010
100	101	1	011
100	111	0	101
100	111	1	101
110	110	0	110
110	110	1	110

m_1	m_2	r	m_o
110	000	0	110
110	000	1	100
110	011	0	110
110	011	1	100
110	101	0	100
110	101	1	100
110	111	0	010
110	111	1	011
000	000	0	010
000	000	1	011
000	011	0	000
000	011	1	011
000	101	0	011
000	101	1	101
000	111	0	101
000	111	1	111
011	011	0	000
011	011	1	101
011	101	0	101
011	101	1	101
011	111	0	101
011	111	1	111
101	101	0	101
101	101	1	111
101	111	0	111
101	111	1	111
111	111	0	111
111	111	1	111

Numerical results

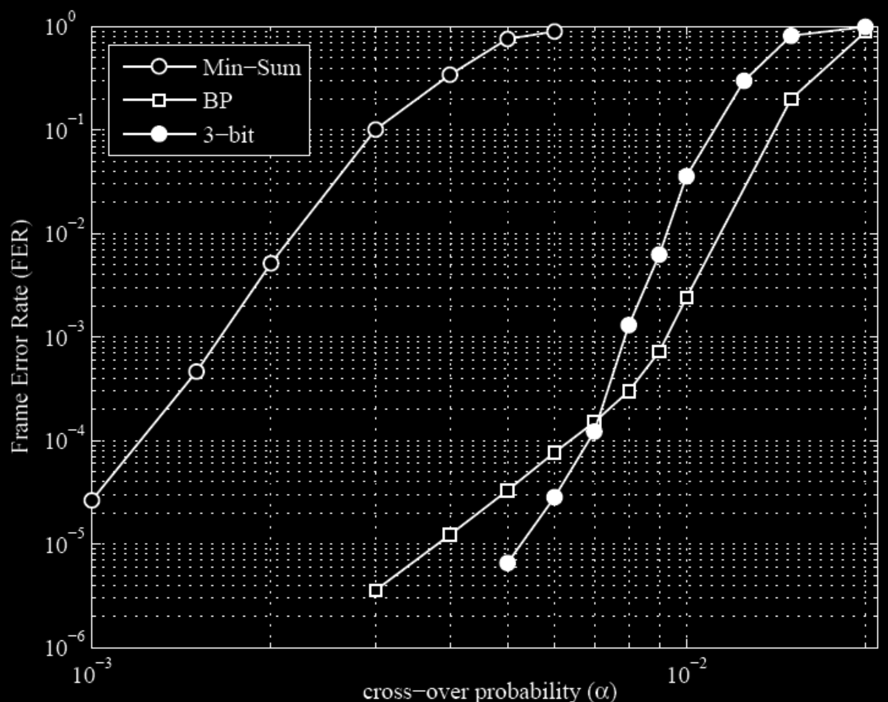


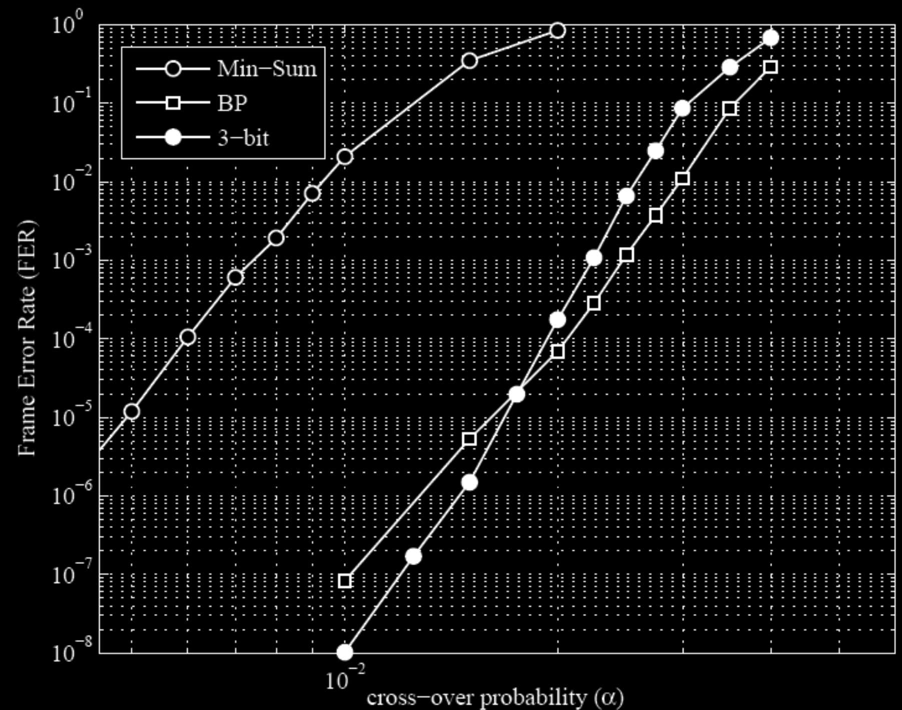
*N=155, R=0.4,*Tanner code



N=768, R=0.75, Quasicyclic code

Numerical results





*N=4085, R=0.82,*MacKay code

N=1503, R=0.668, Quasicyclic code

Note: Notice the diffference in slope of FER

Extra slides

Trapping set as decoding failures

- The all zero codeword is transmitted.
- The decoder performs *D* iterations.
  - $\boldsymbol{y} = (y_1\ y_2\ .\ .\ .\ y_n)$ - decoder input
  - $\boldsymbol{x}^l$, $l \leq D$ -the decoder output vector at the *l*-th iteration
- A variable node *v* is *eventually correct if there exists a positive integer* d *such that for all* $l > d$, $v \notin \mathrm{supp}(\boldsymbol{x}^l)$.
- *A decoder failure is said to occur if there* does not exits $l \leq D$ such that $\mathrm{supp}(x^l) = \emptyset$.
  - $T(\boldsymbol{y})$ – a nonempty set of variable nodes that are not eventually correct
  - G - subgraph induced by $T(\boldsymbol{y})$, $C(G) = E \cup O$ (even and odd degree check nodes in)
  - $T(\boldsymbol{y})$ is an (a,b) trapping set, where $a = |T(\boldsymbol{y})|$, $b = |O|$