Design of LDPC codes

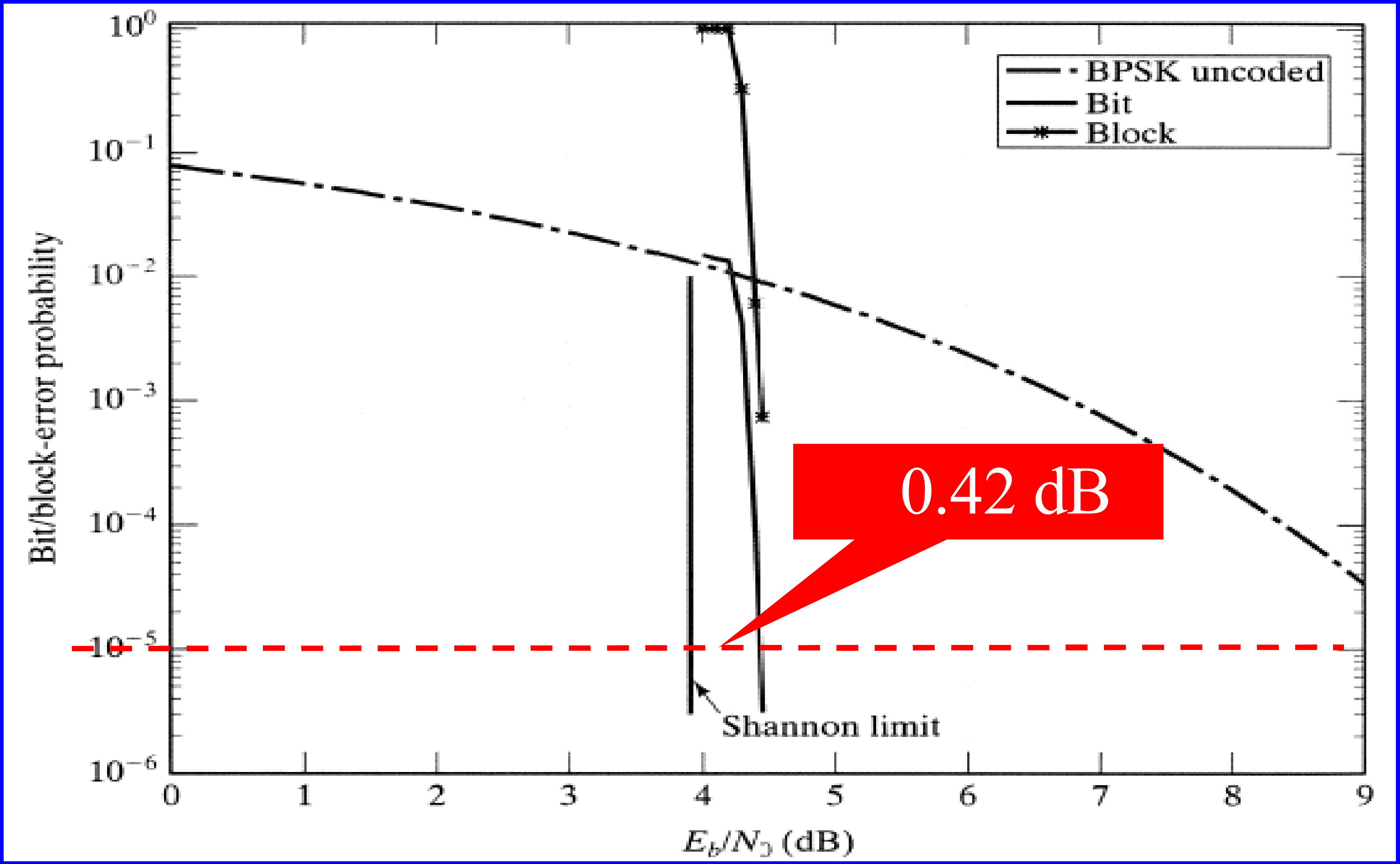
- Codes from finite geometries
- Random codes: Determine the connections of the bipartite Tanner graph by using a (pseudo)random algorithm observing the degree distribution of the code bit vertices and the parity check vertices
  - Regular
  - Irregular
- Graph theoretic codes
- Combinatorial codes
- Other algebraic constructions

Column splitting

- Can be applied to any code; also those designed by use of finite geometries
- Effects:
  - The variable nodes in the Tanner graph are split into several nodes
  - The new *extended* code C_{ext} will have the following properties:
    - More code symbols (higher n)
    - Higher code rate (J is constant; rank of $\mathbf{H}$ may increase, but usually not by much)
    - Row weight is unchanged. Any two columns will still have at most one 1 in common
    - The column weight is reduced from its original value γ to γ_{ext}
    - The minimum distance is reduced
    - Cycles in the Tanner graph are broken

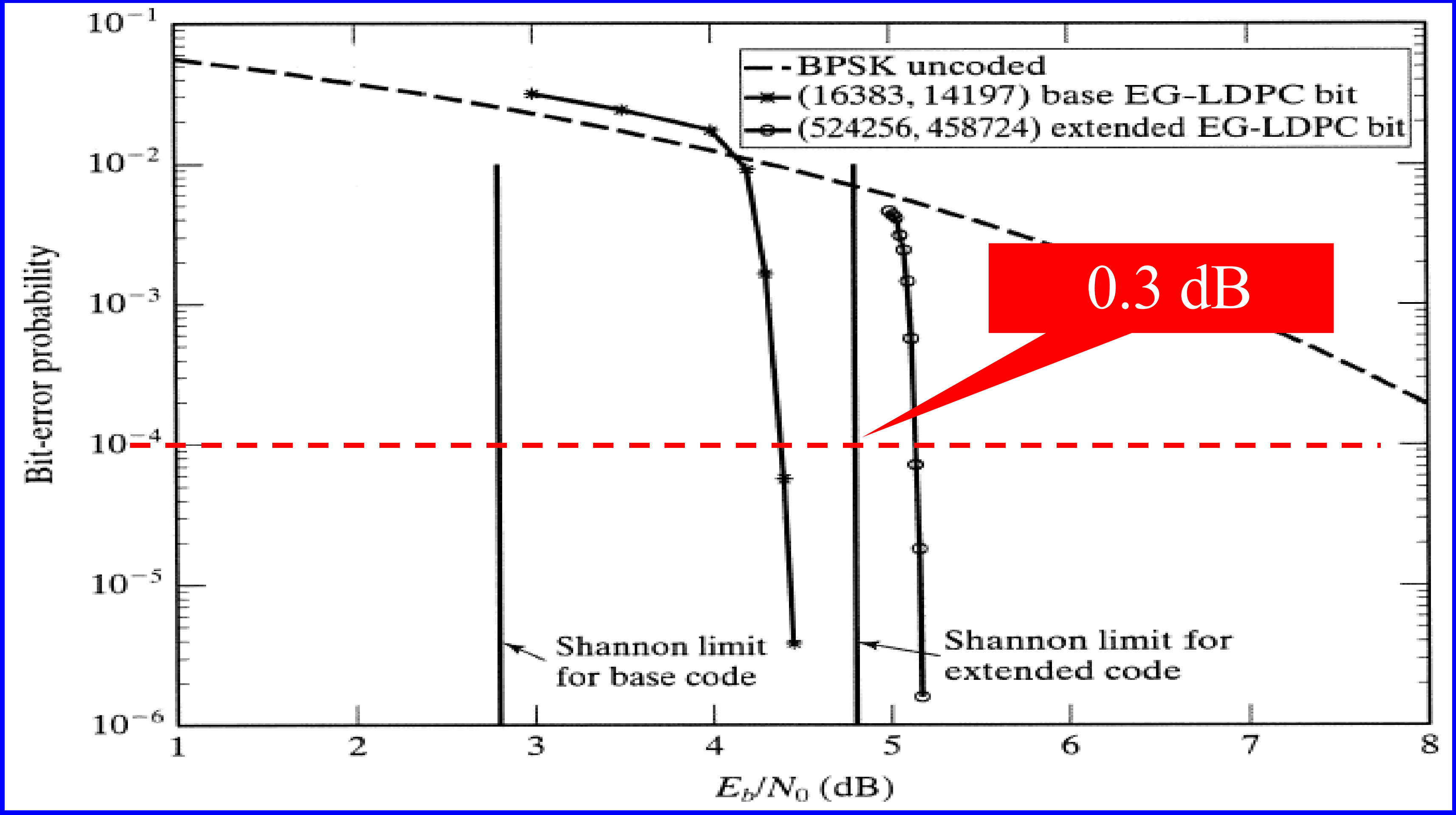
Column splitting: Example

- (4095,3367) type-I cyclic 2-dimensional (0,6)th order EG code
- Split each column of **H** into 16 new columns
- C_{ext} is a (65520,61425) code with $\rho = 64$, $\gamma_{ext} = 4$, $R = 0.9375$, $r = 0.00098$



Column splitting: Example

- (16383,14197) type-I cyclic 2-dimensional (0,7)th order EG code
- Split each column of **H** into 32 new columns
- C_{ext} is a code with n = 524256, ρ = 128, γ_{ext} = 4, R = 0.97, r = 0.00024



Column splitting

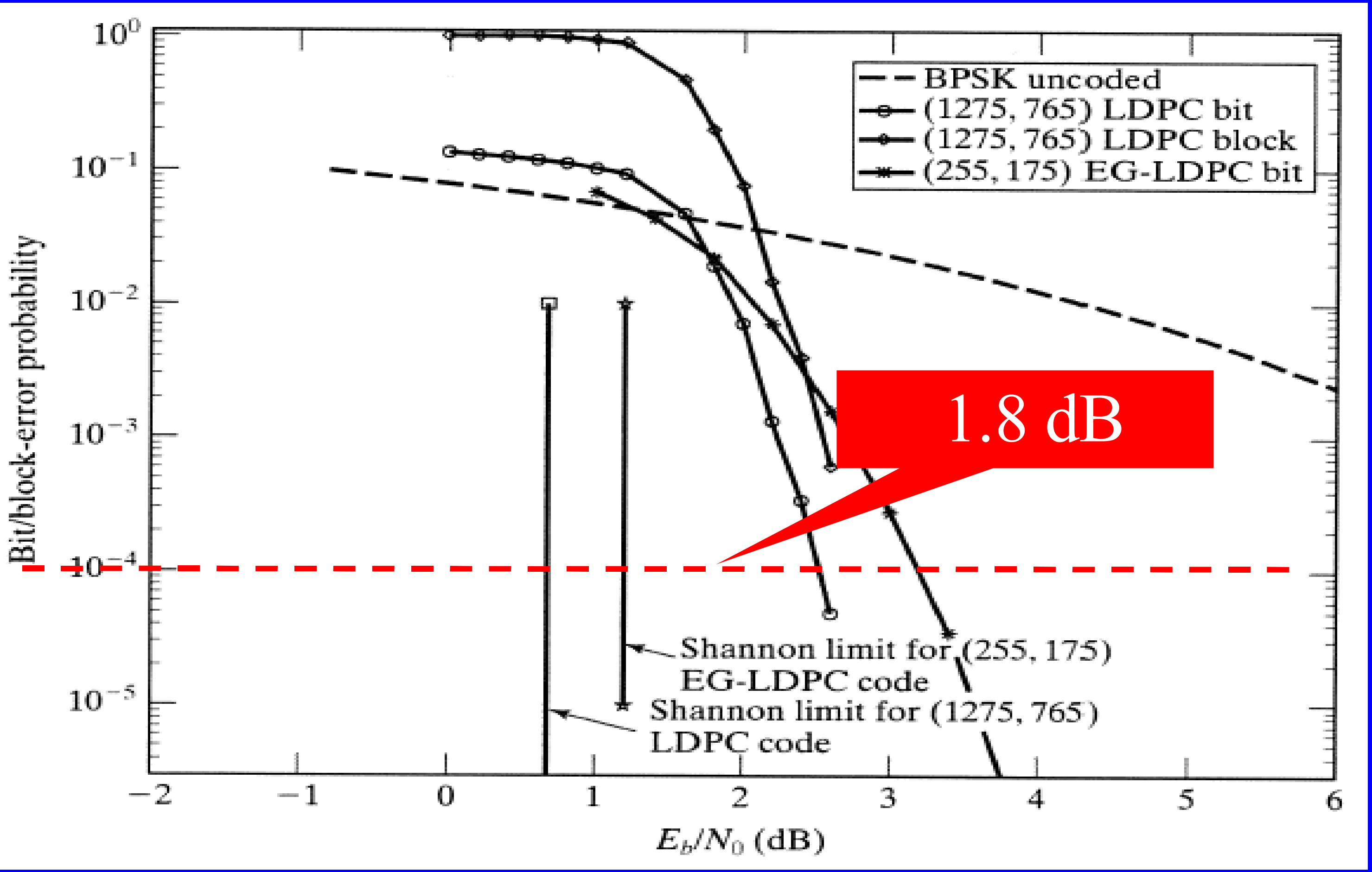
- Column splitting of cyclic code -> extended code is usually not cyclic
- By starting with a parity check matrix consisting of K $n \times n$ circulant submatrices, and splitting each column in a ”rotating and circular” fashion into a fixed number of new columns, the extended code will be quasi-cyclic
- PG codes: J may not be a multiple of n, so a modification of the above procedure is necessary

Row splitting

- Can be applied to any code
- Effects:
  - The parity check nodes in the Tanner graph are split into several nodes
  - The new code will have the following properties:
    - Same length as original code
    - More parity checks (higher J) and lower code rate
    - Column weight is unchanged. Any two columns will still have at most one 1 in common. The minimum distance ought to increase
    - The row weight is reduced
    - Cycles in the Tanner graph are broken
- Can be combined with column splitting

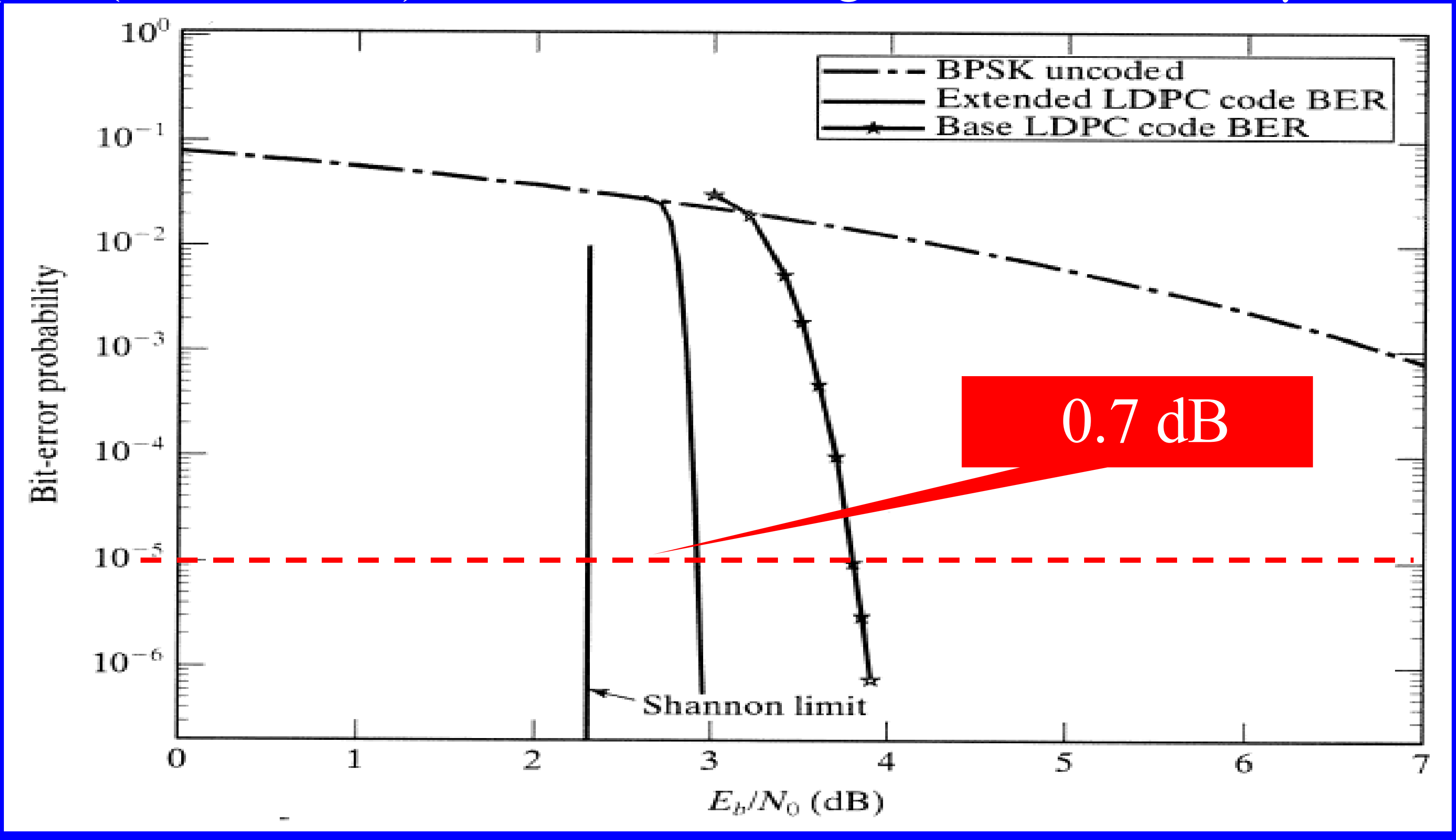
Column + row splitting: Example

- (255,175) type-I cyclic 2-dimensional (0,4)th order EG code
- Split each column of **H** into 5 new columns and each row into 2 rows
- C_{ext} is a (1275,765) code with $\rho = 8$, column weights 3 and 4, $R = 0.6$, $r = 0.00627$



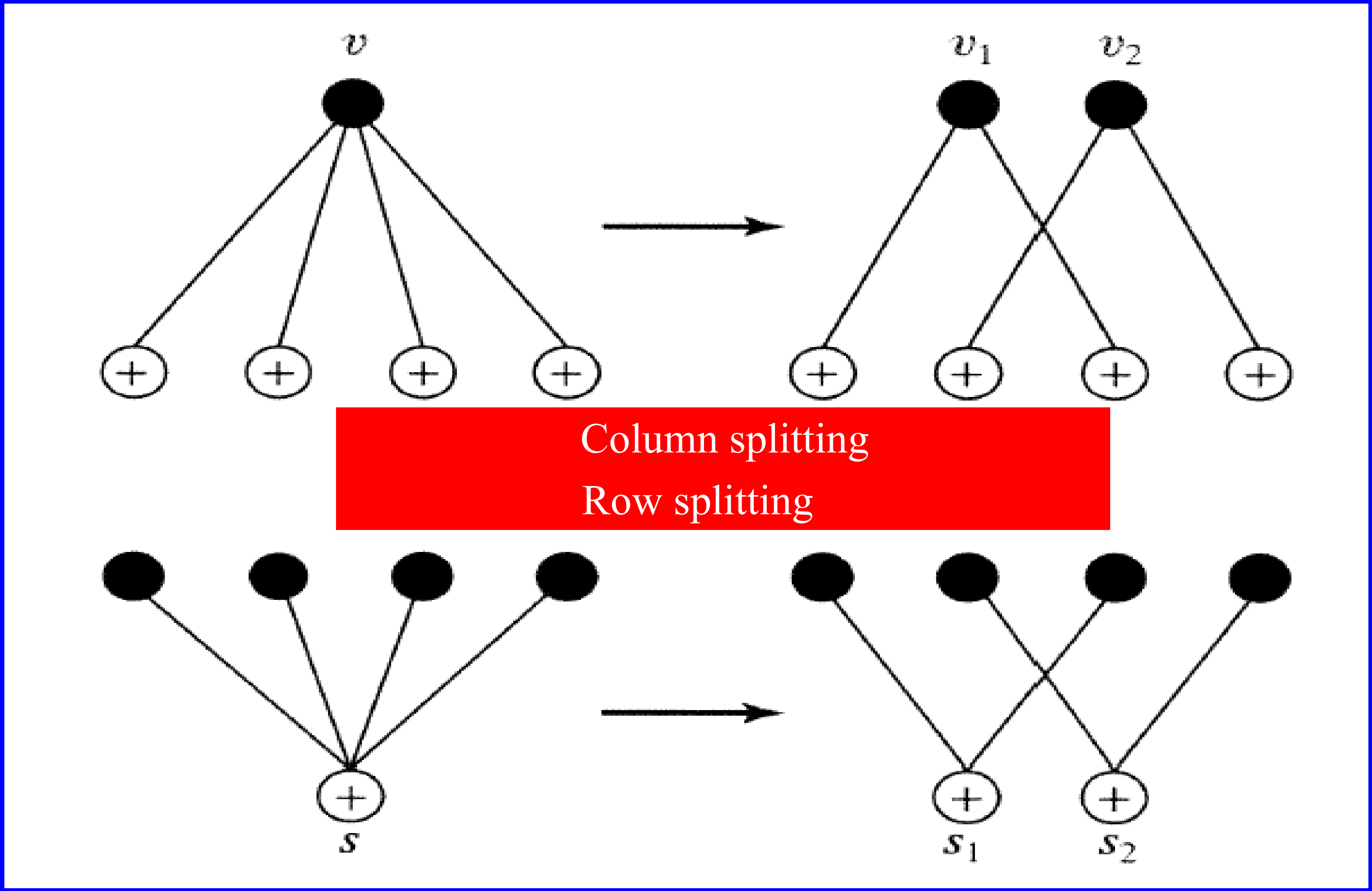
Column + row splitting: Example

- (4095,3367) type-I cyclic 2-dimensional (0,6)th order EG code
- Split each column of **H** into 16 new columns and each row into 3 rows
- C_{ext} is a (65520,53235) code with row weights 21 and 22, and $\gamma = 4$



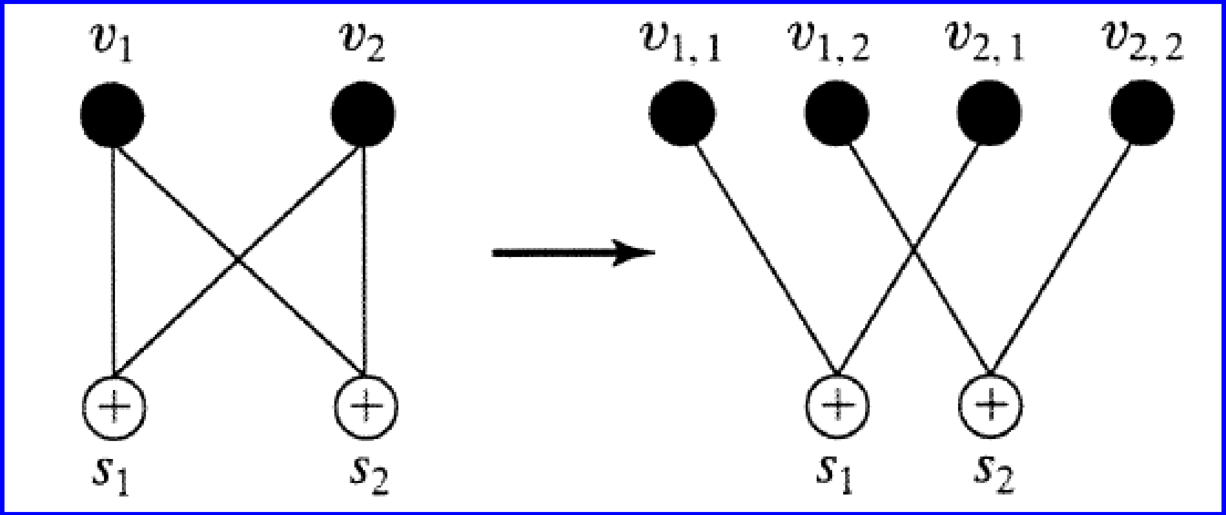
Cycle breaking

- Splitting rows and columns changes the Tanner graph

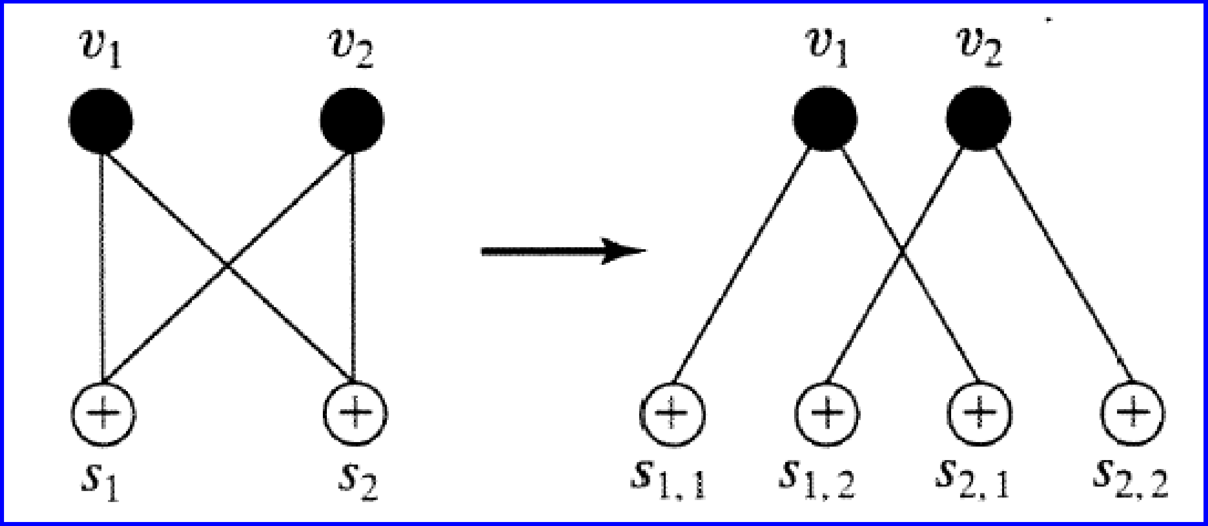


Cycle breaking

- Splitting columns to break a cycle of length 4

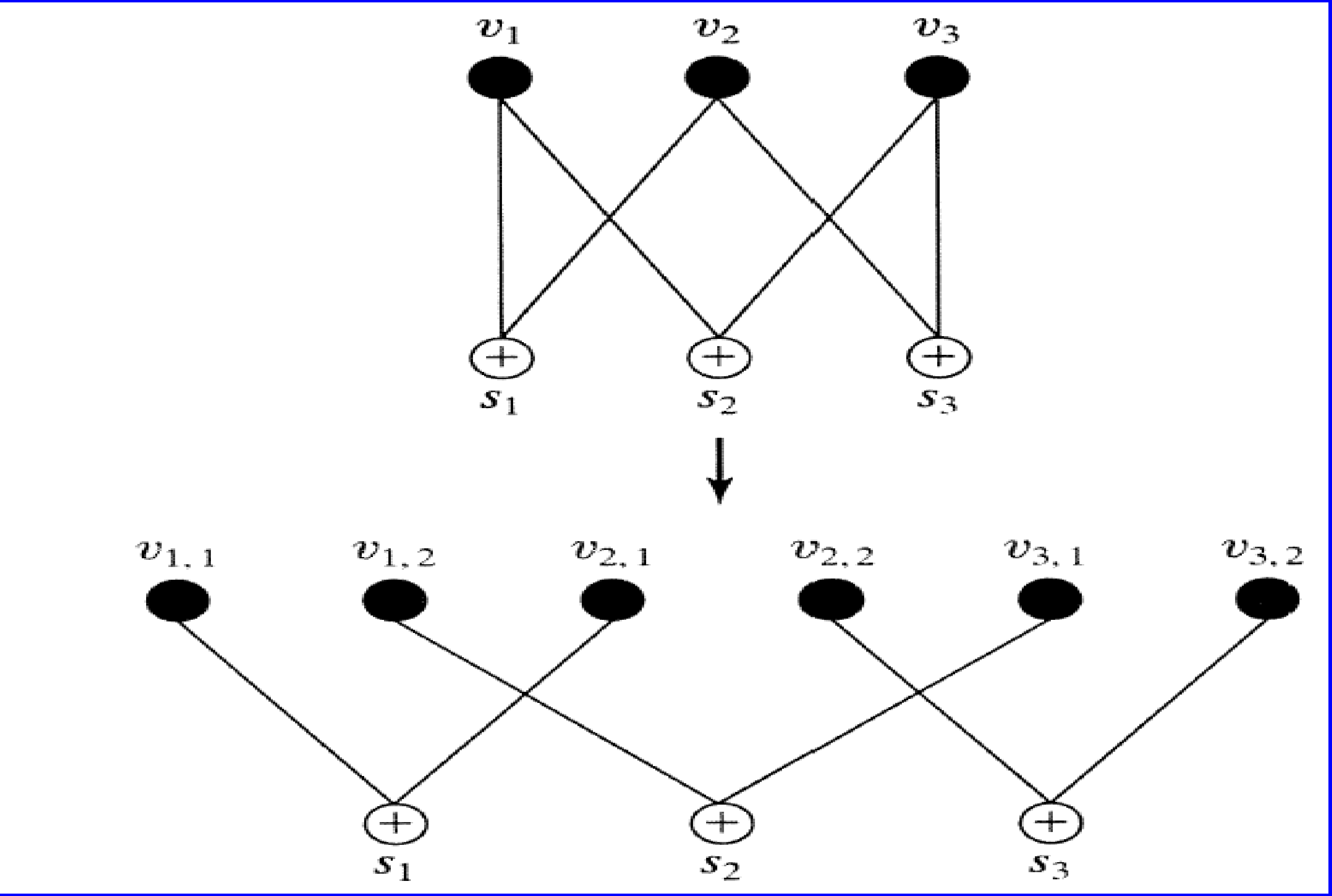


- Splitting rows to break a cycle of length 4



Cycle breaking

- Splitting columns to break a cycle of length 6



Effects of cycle breaking

- Increases the number of nodes and hence the complexity of the message passing algorithms (SPA/BF algorithm)
- Reduces the number of cycles and so improves decoding performance

Example of cycle breaking

- (7,4) Hamming code. Here: Cyclic version

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

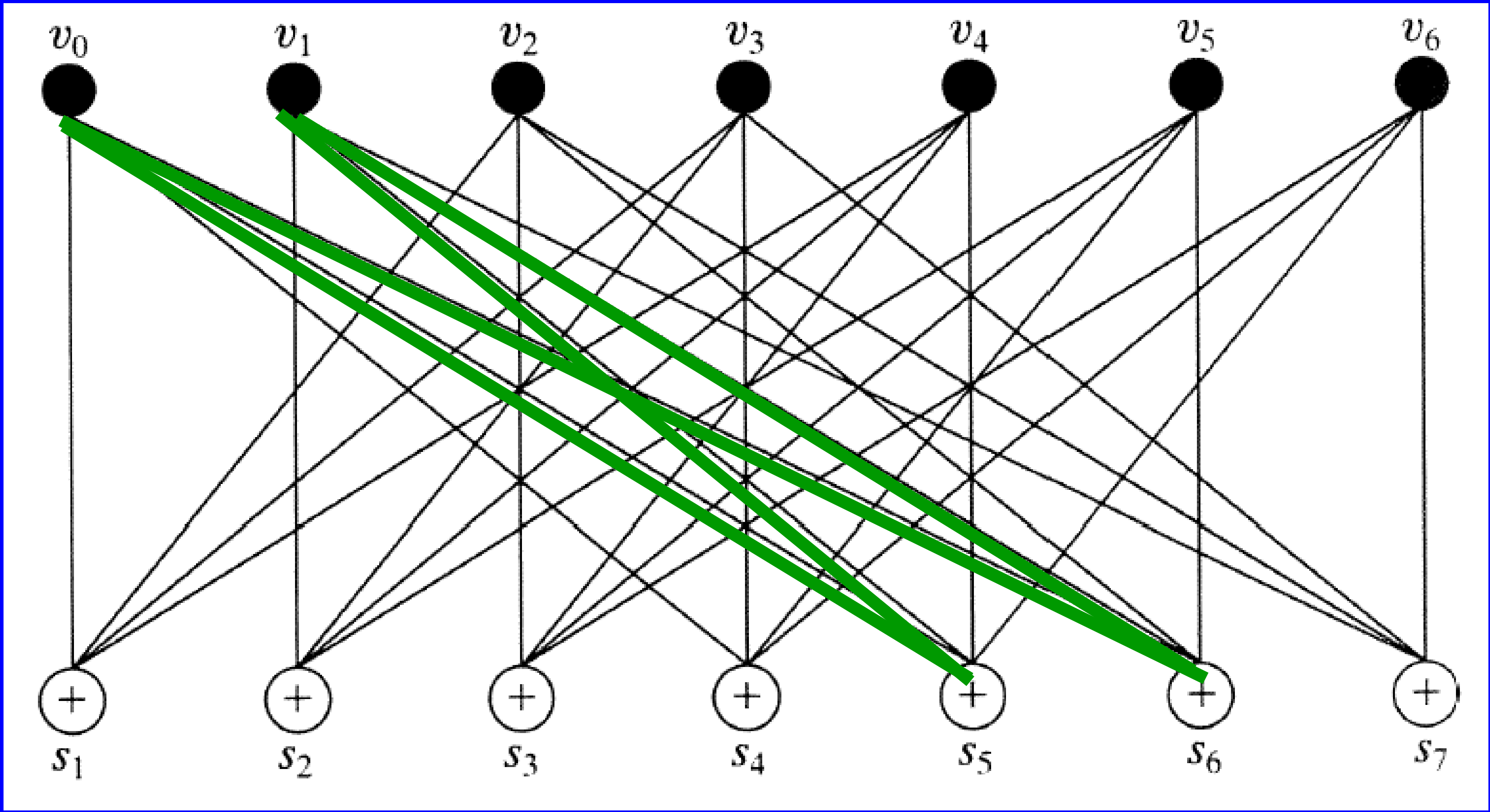
21 4-cycles

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

No 4-cycles

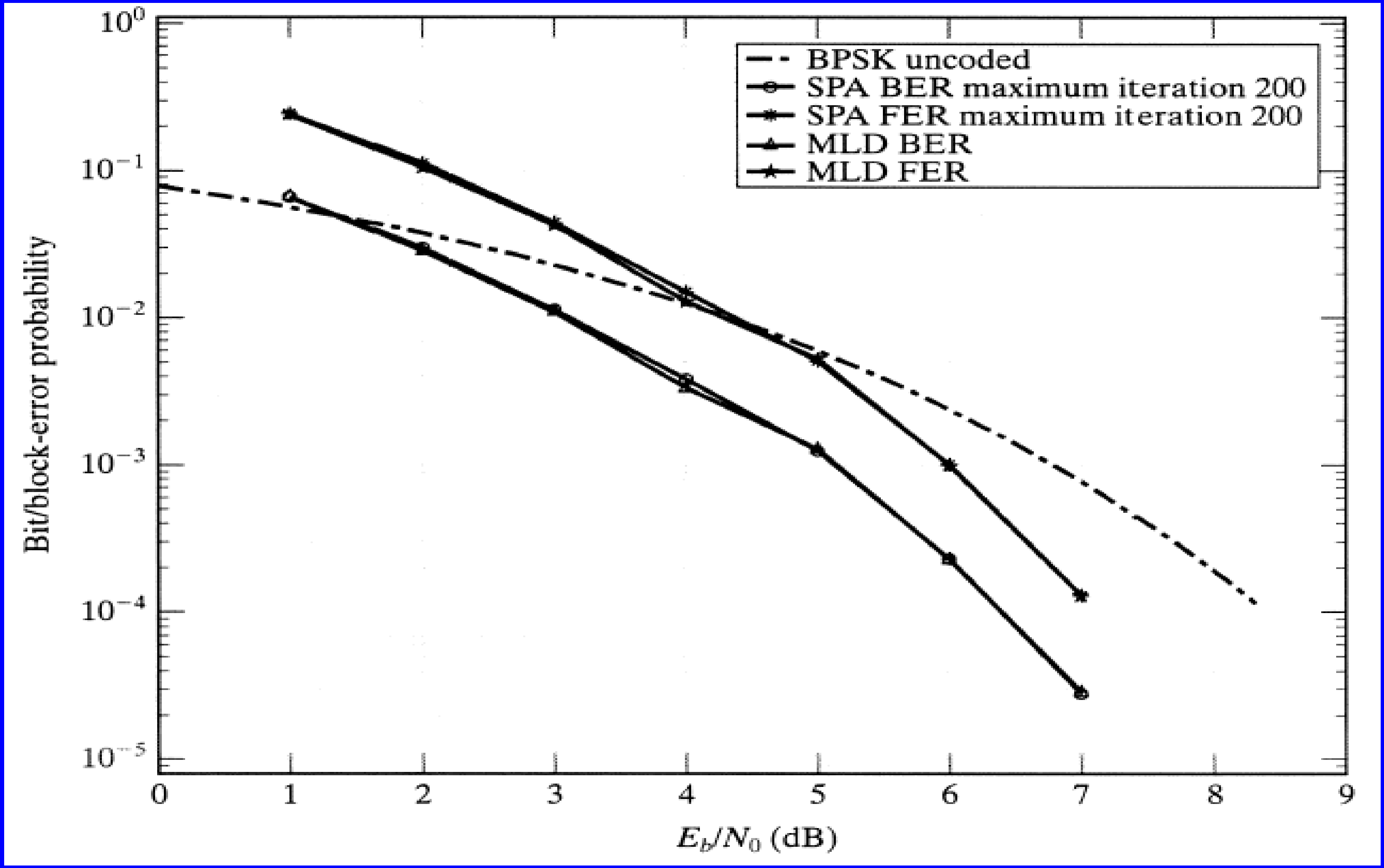
Example of cycle breaking

- (7,4) Hamming code. Tanner graph



Example of cycle breaking

- (14,8) extended Hamming code. Performance



Example 2 of cycle breaking

- (23,12) Golay code. Here: Cyclic version

$$\mathbf{H} = \left[\begin{array}{ccccccccccccccccccccccc}
1&0&1&0&0&1&0&0&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0\\
0&1&0&1&0&0&1&0&0&1&1&1&1&1&0&0&0&0&0&0&0&0&0\\
0&0&1&0&1&0&0&1&0&0&1&1&1&1&1&0&0&0&0&0&0&0&0\\
0&0&0&1&0&1&0&0&1&0&0&1&1&1&1&1&0&0&0&0&0&0&0\\
0&0&0&0&1&0&1&0&0&1&0&0&1&1&1&1&1&0&0&0&0&0&0\\
0&0&0&0&0&1&0&1&0&0&1&0&0&1&1&1&1&1&0&0&0&0&0\\
0&0&0&0&0&0&1&0&1&0&0&1&0&0&1&1&1&1&1&0&0&0&0\\
0&0&0&0&0&0&0&1&0&1&0&0&1&0&0&1&1&1&1&1&0&0&0\\
0&0&0&0&0&0&0&0&1&0&1&0&0&1&0&0&1&1&1&1&1&0&0\\
0&0&0&0&0&0&0&0&0&1&0&1&0&0&1&0&0&1&1&1&1&1&0\\
0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&1&0&0&1&1&1&1&1\\
1&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&1&0&0&1&1&1&1\\
1&1&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&1&0&0&1&1&1\\
1&1&1&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&1&0&0&1&1\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&1&0&0&1\\
1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&1&0&0\\
0&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&1&0\\
0&0&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&1\\
1&0&0&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0\\
0&1&0&0&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&1&0&1&0\\
0&0&1&0&0&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&1&0&1\\
1&0&0&1&0&0&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&1&0\\
0&1&0&0&1&0&0&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&1
\end{array}\right]$$

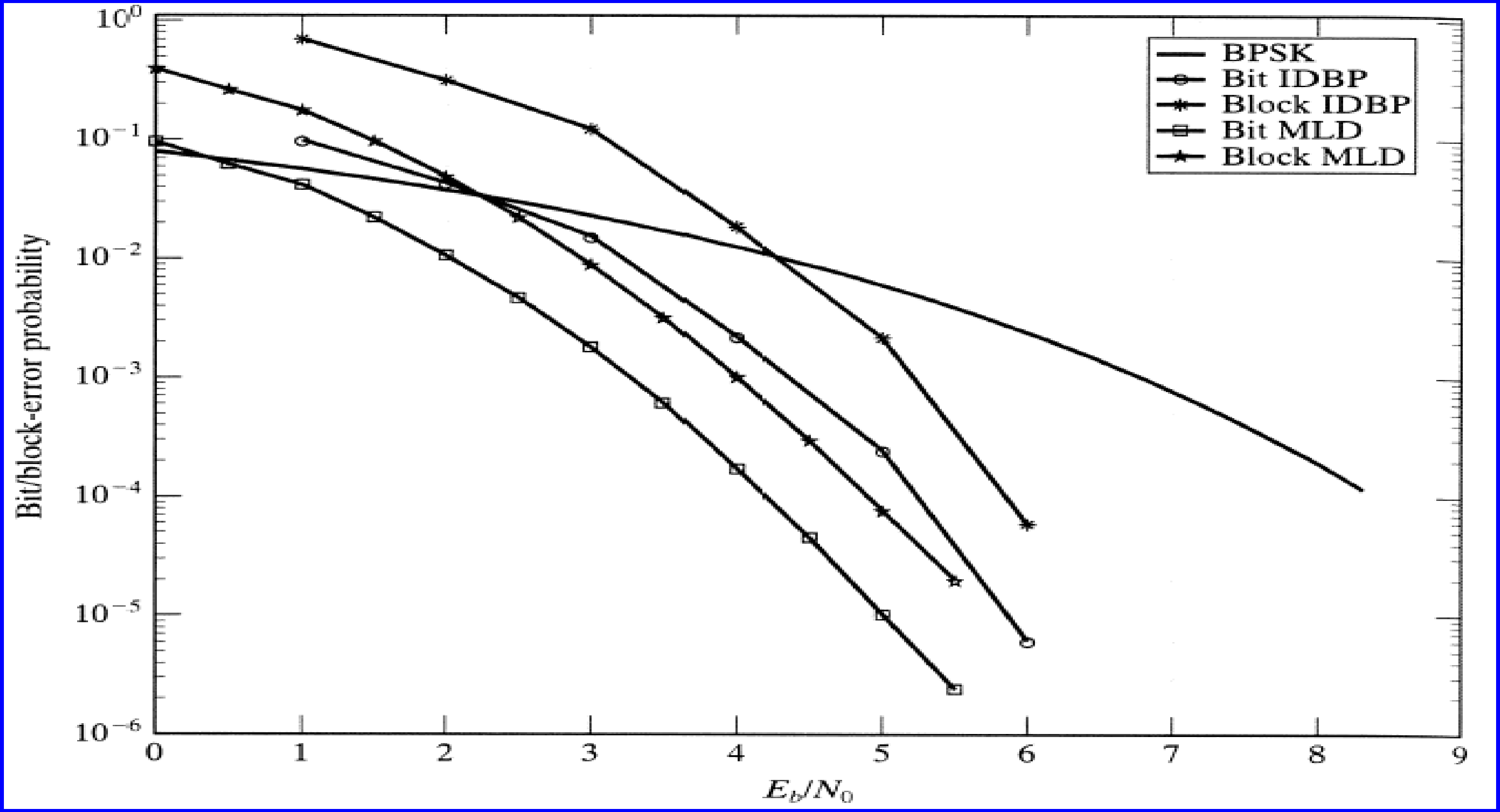
1748 4-cycles

Example 2 of cycle breaking

- (46,24) extended Golay code obtained by random splitting of columns. 106 cycles of length 4. The weight distribution is

Weight	Number of codewords	Weight	Number of codewords
0	1	23	1963152
1	0	24	1882241
2	0	25	1656075
3	0	26	1336148
4	0	27	989121
5	1	28	671322
6	5	29	417751
7	18	30	237105
8	58	31	122262
9	280	32	57390
10	1000	33	24069
11	3151	34	9034
12	9034	35	3151
13	24069	36	1000
14	57390	37	280
15	122262	38	58
16	237105	39	18
17	417751	40	5
18	671322	41	1
19	989121	42	0
20	1336148	43	0
21	1656075	44	0
22	1882241	45	0
		46	1

Performance of (46,24) extended Golay code



Random LDPC codes

- LDPC codes reinvented by Mackay (1995) using random codes
- Create a matrix of J rows and n columns, with column weight γ and row weight ρ
- J is usually chosen equal to $n - k$
- In general, $\gamma n = \rho(n - k) + b$, where b is the remainder
- Thus, we can choose $(n - k - b)$ rows of weight ρ and b rows of weight $\rho+1$

Construction of random LDPC codes

- $\mathbf{H}_i = [\mathbf{h}_1, \ldots, \mathbf{h}_i]$ is a partial parity check matrix consisting of the first i columns
- *Initialization*: $\mathbf{H}_0$ = empty matrix; $i = 1$; *Cand* = set of all nonzero $(n - k)$–dimensional column vectors of weight γ
- Let $\mathbf{h}$ be a random column from *Cand*. Delete $\mathbf{h}$ from *Cand*
- Check whether $\mathbf{h}$ has more than one 1 in common with any column in $\mathbf{H}_{i-1}$ and if any (partial) row weight exceeds its maximum weight. If so, choose another $\mathbf{h}$ from *Cand* and repeat. Otherwise, proceed to the next step
- Set $\mathbf{h}_i = \mathbf{h}$ and $i = i+1$. If $i \leq n$, then repeat
- Can use backtracking. Also, select parameters such that the number of weight-γ $(n - k)$–tuples is $\gg n$. Also, it is possible to relax the requirements on row weights
- The actual rank of $\mathbf{H}$ may end up to be $(n - k') < (n - k)$
- Efficient for small values of ρ and γ, but the lower bound of $\gamma+1$ on the minimum distance can be very poor

Irregular LDPC codes

- Use variable nodes of varying degrees and parity check nodes of varying degrees
- Degree distributions $\gamma(x) = \sum_i \gamma_i x^{i-1}$ and $\rho(x) = \sum_i \rho_i x^{i-1}$
- Approach: Density evolution / EXIT charts
- Create random LDPC codes according to distributions
- Optimum degree distributions often contain a large γ_2, which can lead to a poor minimum distance and high error floor
- But holds the world record: 0.0045 dB from the Shannon limit
- The approach assumes infinite block length and cycle free graphs
- These optimum degree distributions are not optimum for short codes in general

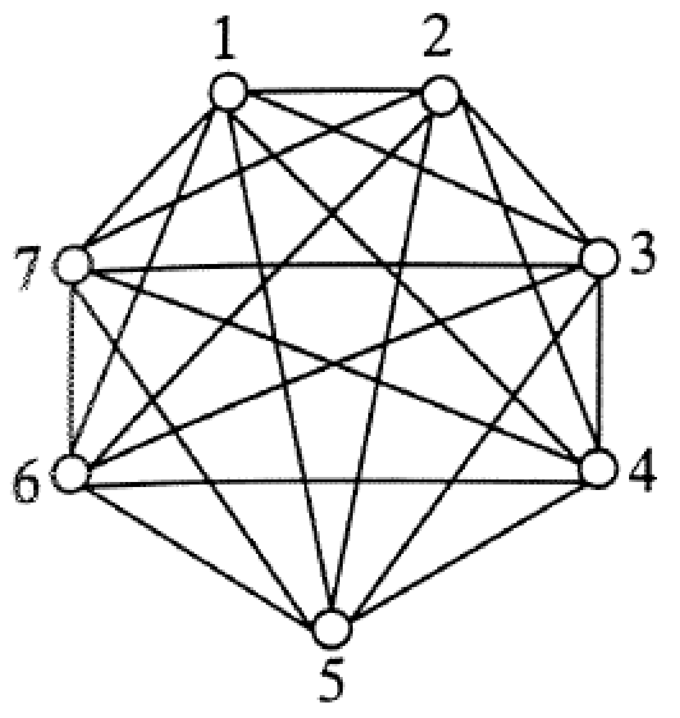
Improved irregular LDPC codes

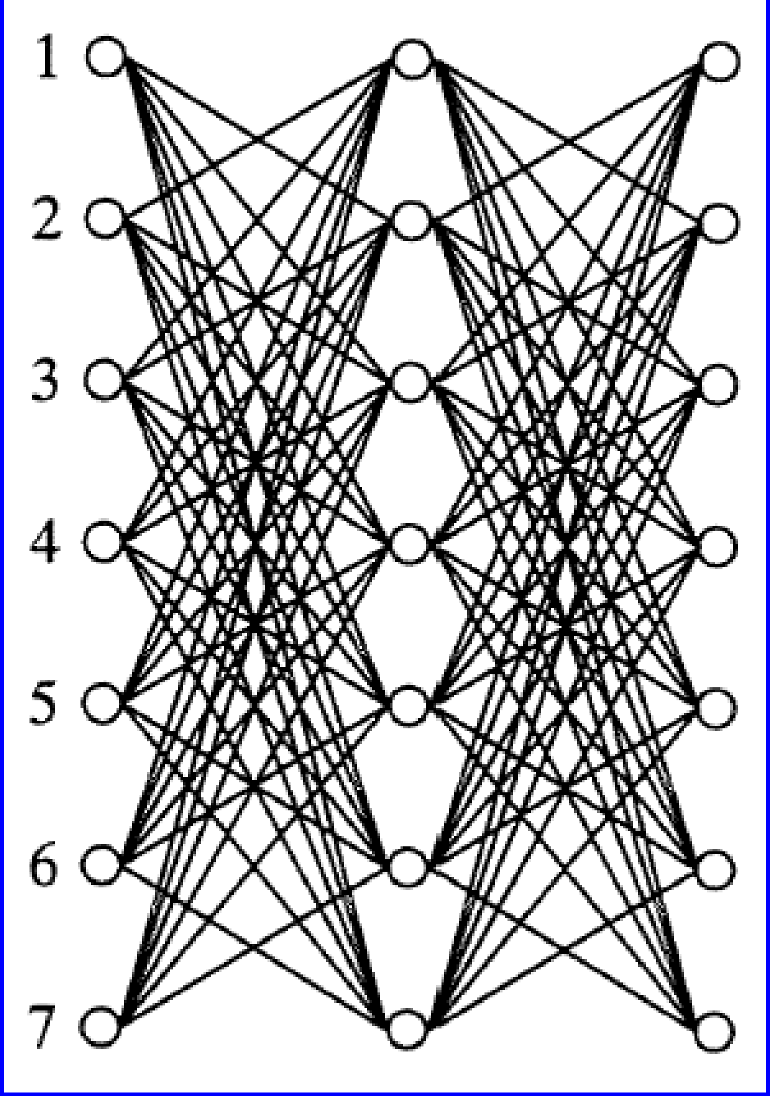
- Extra design rules:
  - All degree 2 variable nodes are associated with parity symbols (if possible)
  - No length 4 cycles
  - No short cycles involving degree 2 variable nodes
  - Limit the number of degree 2 variable nodes
  - Degree redistribution

Graph theoretic LDPC codes

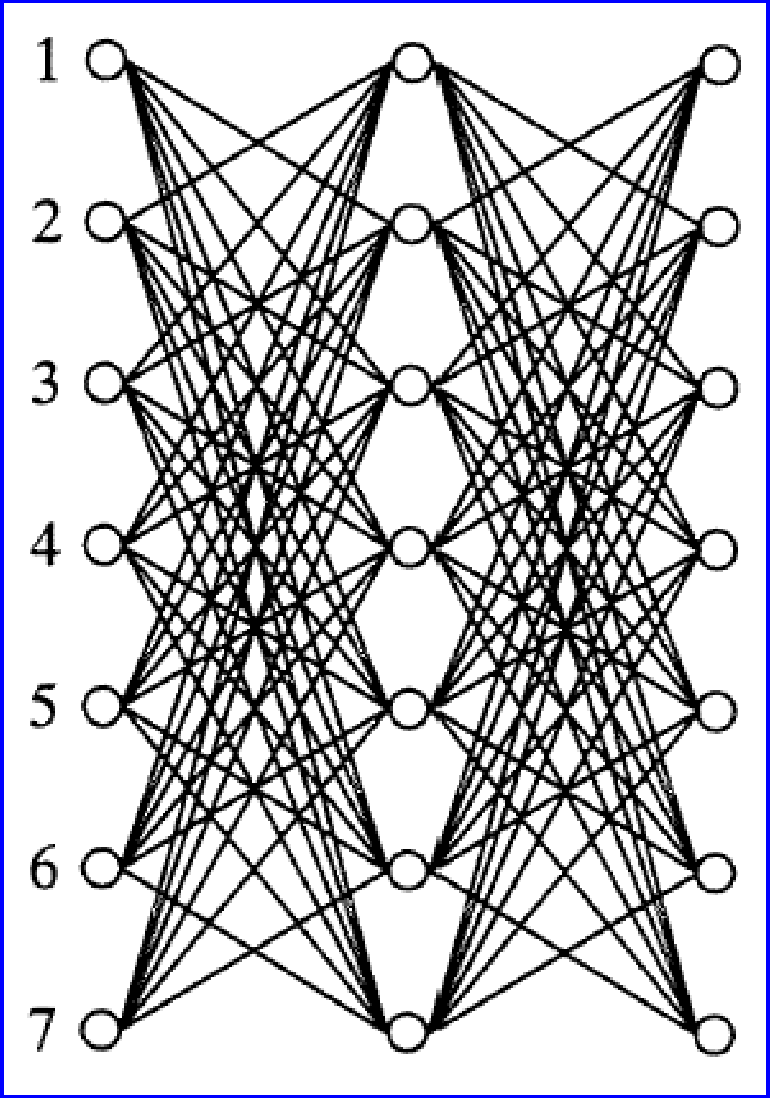
- Graph G
  - No self-loops
  - No multiple edges between a pair of vertices
- Let P be a set of n paths of length γ that are pairwise disjoint or singularly crossing, and such that the union of nodes involved in the paths contains J nodes
- Let **H** be an incidence matrix of P, i.e., a $J \times n$ matrix such that $h_{i,j} = 1$ iff. node i is on path j
- If each row of **H** has constant weight ρ, then **H** defines a (γ, ρ)-regular LDPC code

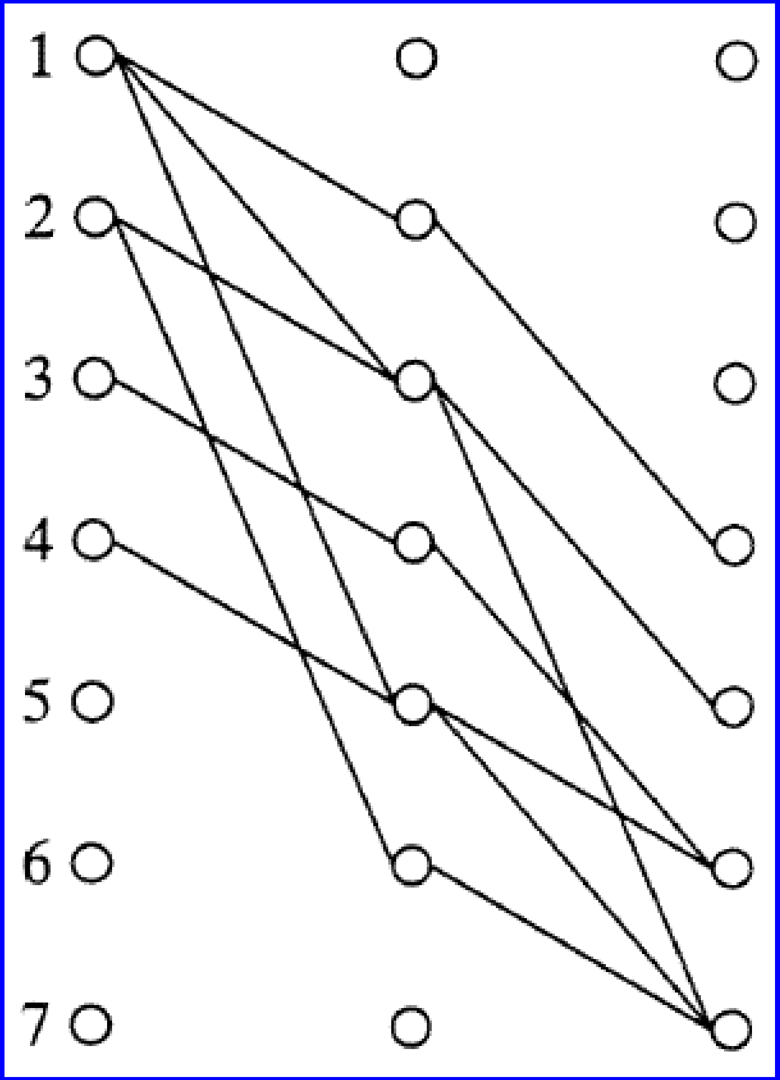
Graph theoretic LDPC codes: Example



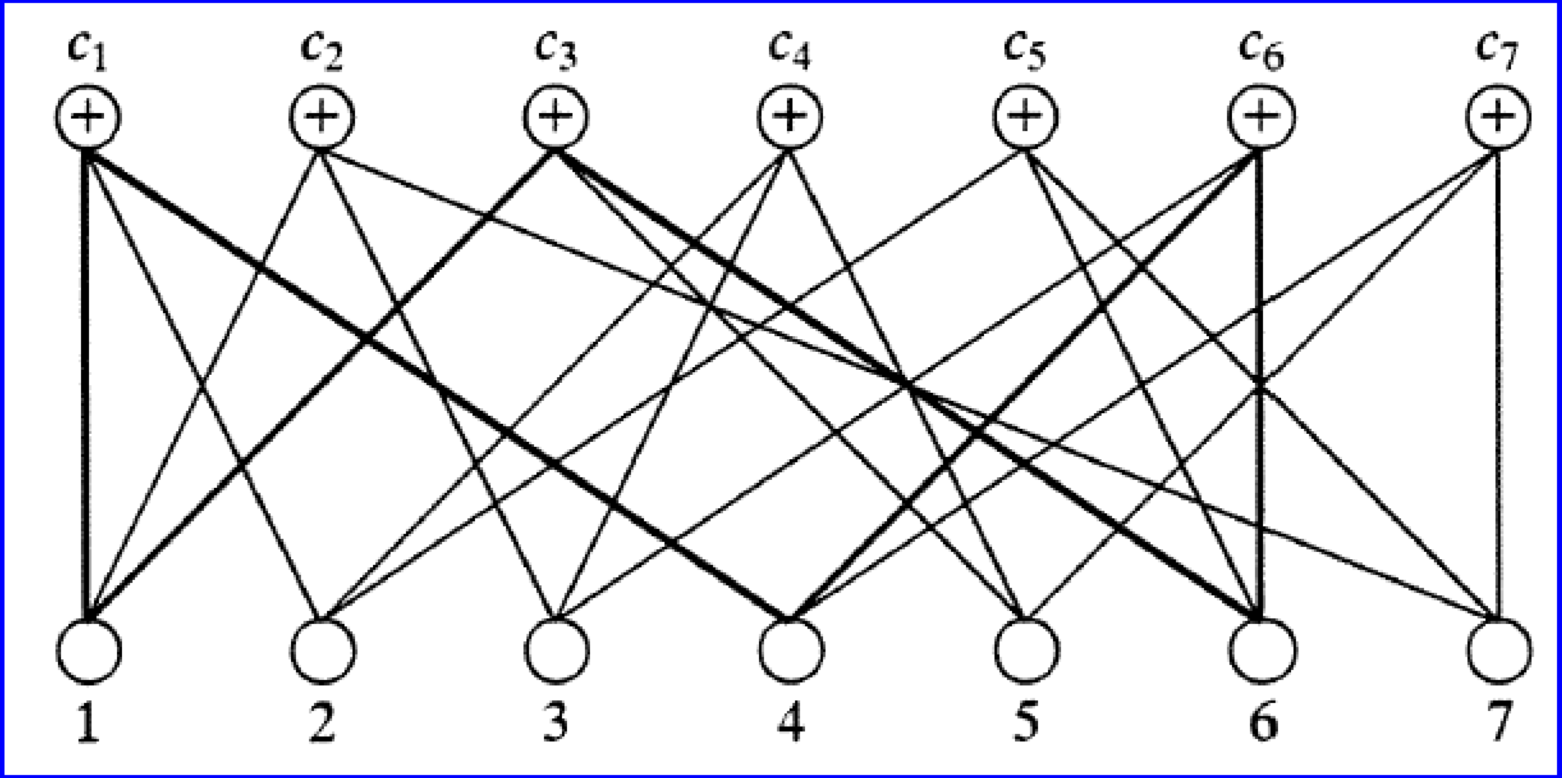


Graph theoretic LDPC codes: Example





Graph theoretic LDPC codes: Example



We have skipped

- 17.5 PG LDPC codes
- 17.9 Shortened FG LDPC codes
- 17.10 Construction of Gallager LDPC codes
- 17.11 Masked EG-Gallager codes
- 17.12 Construction of QC LDPC codes
- 17.13 LDPC codes from FGs over GF(p^s)
- 17.17 LDPC codes from BIBDs
- 17.18 LDPC codes from RS codes
- 17.19 Concatenations of LDPC and Turbo codes