ELEC 6131: Error Detecting and Correcting Codes

Instructor:
Dr. M. R. Soleymani, Office: EV-5.125, Telephone: 848-2424 ext: 4103.
Time and Place: Tuesday, 17:45 – 20:15.
Office Hours: Tuesday, 15:00 – 17:00

LECTURE 11: LDPC Codes

Low Density Parity Check (LDPC) Codes

References:

- [1] Thomas J. Richardson, M. Amin Shokrollahi, and Rüdiger L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," IEEE Trans. Inform. Theory, VOL. 47, NO. 2, FEBRUARY 2001.
- [2] Thomas J. Richardson and Rüdiger L. Urbanke, "Efficient Encoding of Low-Density Parity-Check Codes," IEEE Trans. Inform. Theory, VOL. 47, NO. 2, FEBRUARY 2001.
- [3] Amin Shokrollahi, "LDPC Codes: An Introduction," Digital Fountain, Inc. 39141 Civic Center Drive, Fremont, CA 94538, April 2, 2003.

Low Density Parity Check (LDPC) Codes

- Low Density Parity Check Codes were invented in 1963 by R.G. Gallager. In addition to suggesting the use of codes with sparse parity check matrices, Gallager suggested an iterative decoding algorithm (message –passing decoders) and showed that using this type of decoder, one can come close to Shannon's bounds.

Low Density Parity Check (LDPC) Codes

- In general, an LDPC coed is the null space of a sparse (low-density) matrix H , i.e.,

$$\underline{v}H^T = \underline{0}$$

Where H^T is a low-density matrix in the following sense:

- Assume that H has M rows and N columns, and there are (on the average) i ones in the columns and (on the average) j 1's on rows. If $i \ll M \; and \; j \ll N$, we call the matrix H low-density or sparse.
- In the regular or Gallager LDPC codes, the number of 1's in each column or on each row are the same.

Low Density Parity Check (LDPC) Codes

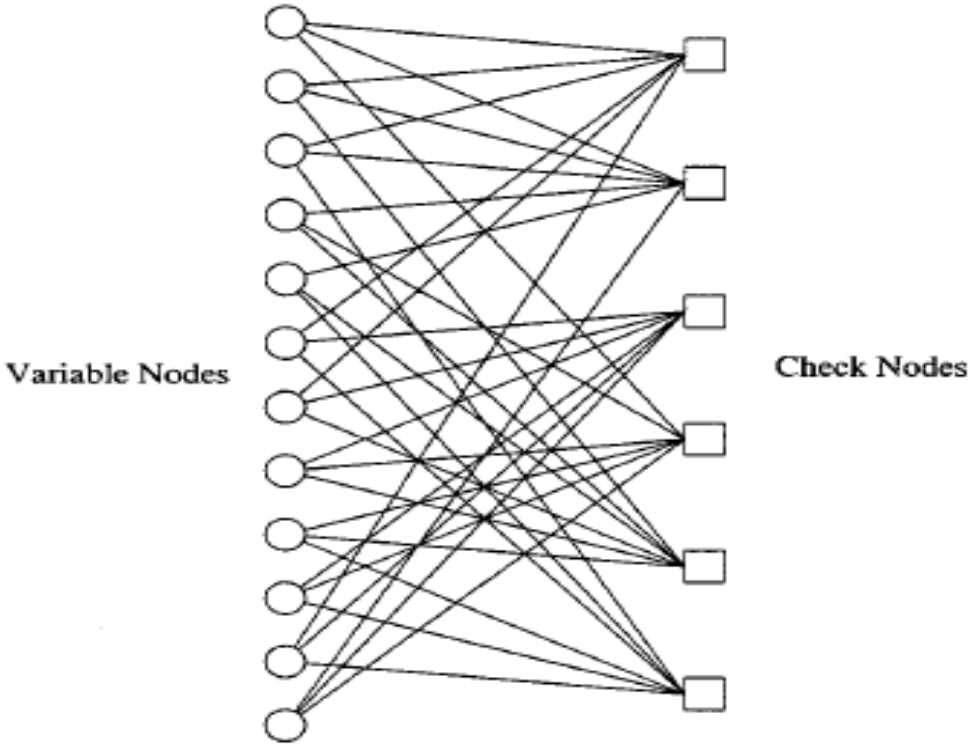
- **Example:** (3,6) Regular LDPC code.

$$H = \begin{bmatrix} 1110011100010 \\ 1111100000001 \\ 0000011101111 \\ 1001000111011 \\ 0101101110000 \\ 0010110011101 \end{bmatrix}$$

- This matrix has 3 one's in each column and 6 1's on each row.
- Graphically, LDPC code can be represented using bi-partite graphs as suggested by Tanner.

Low Density Parity Check (LDPC) Codes

Graphically, LDPC code can be represented using bi-partite graphs as suggested by Tanner.



Graphical representation of a $(3, 6)$-regular LDPC code of length 12. The left nodes represent the variable nodes whereas the right nodes represent the check nodes.

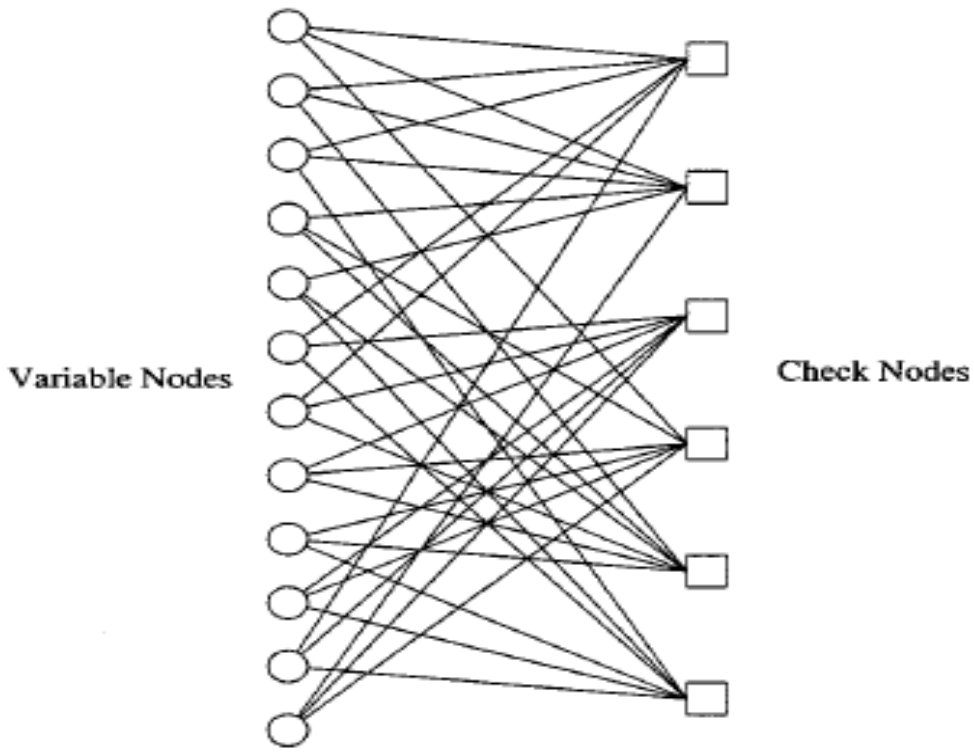
Low Density Parity Check (LDPC) Codes

- On one side are the message nodes also called variable nodes; on the other side of the graph are constraint nodes or check nodes.
- A legitimate pattern,

i.e., a code word is a bit stream that when fed to variable nodes, the check nodes will all have value zero.

- Note that this example, gives a

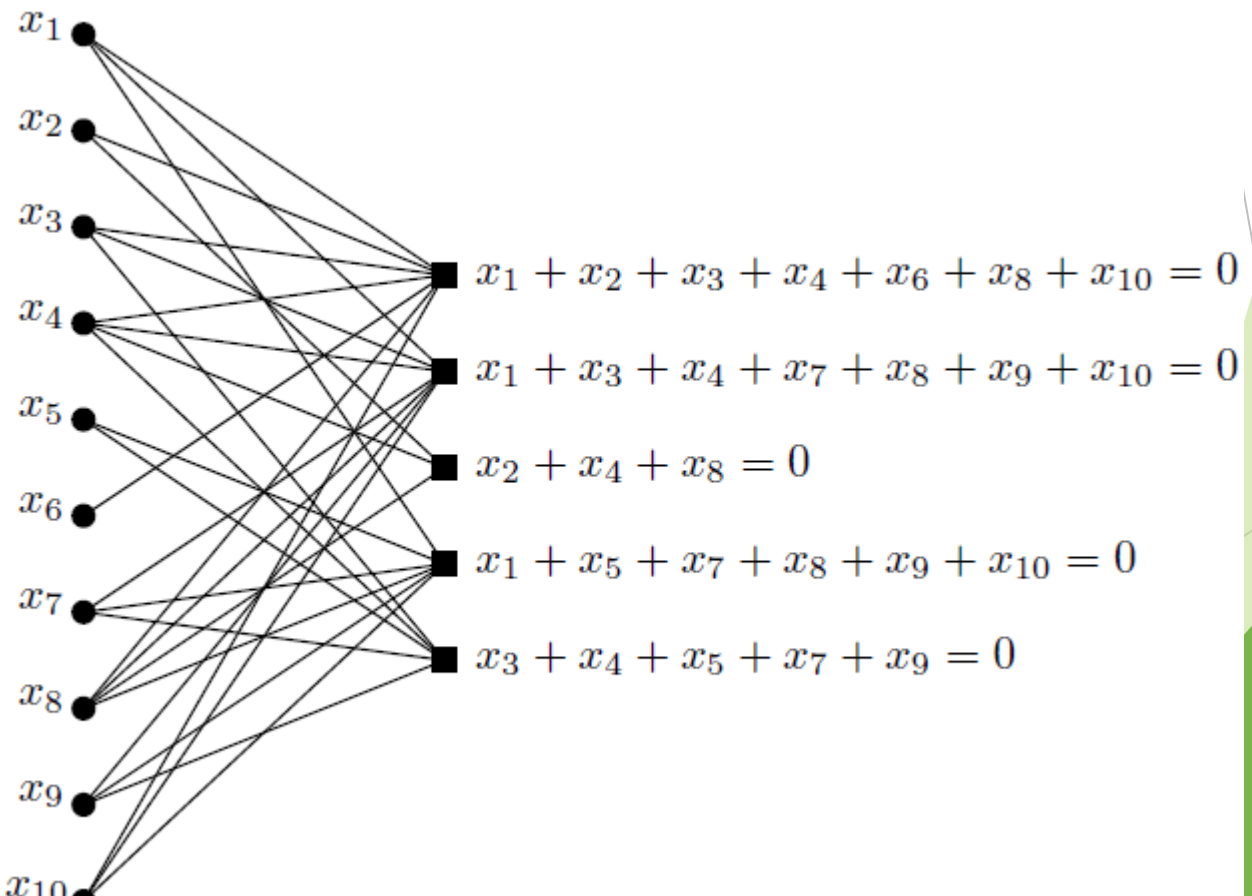
rate $\frac{1}{2}$ regular LDPC code



Graphical representation of a $(3, 6)$-regular LDPC code of length 12. The left nodes represent the variable nodes whereas the right nodes represent the check nodes.

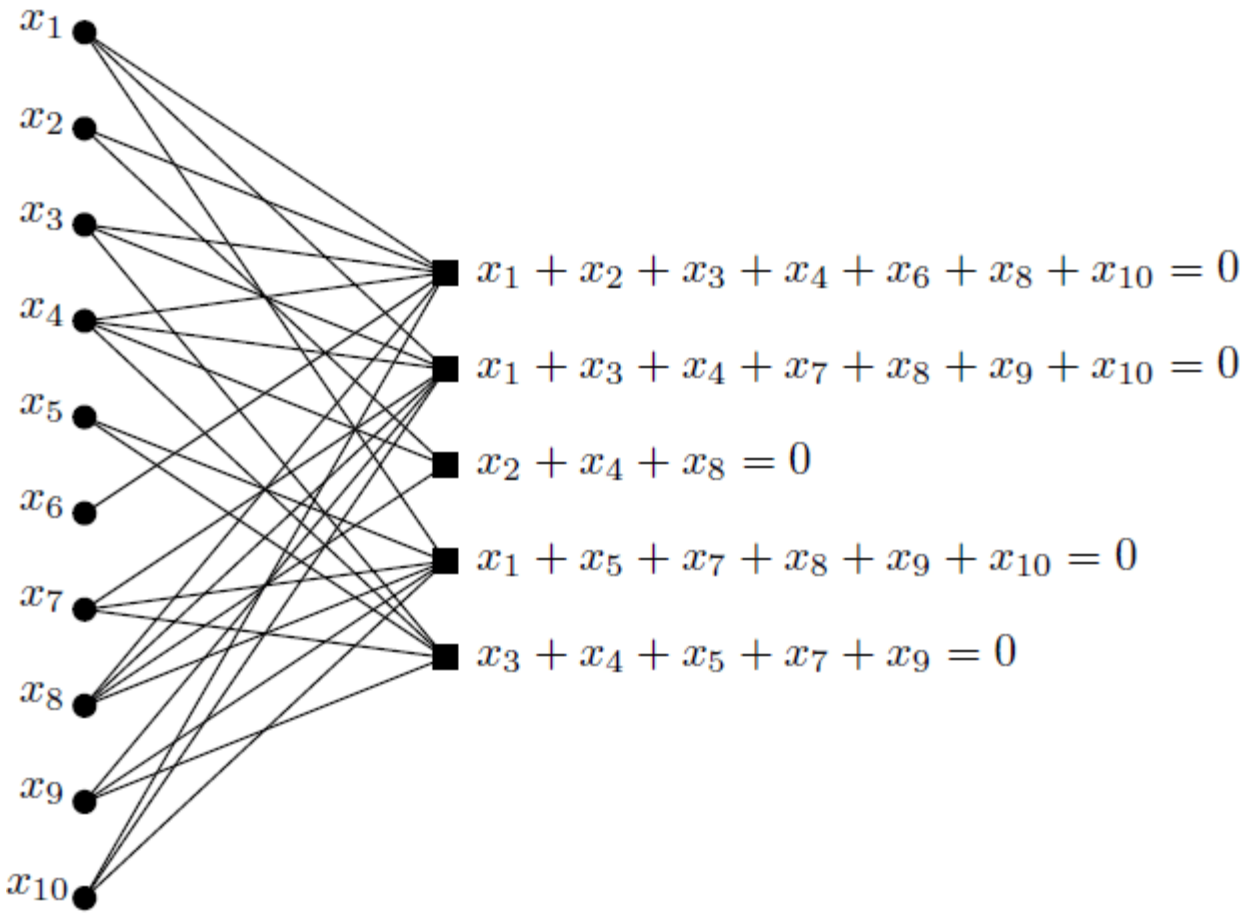
Low Density Parity Check (LDPC) Codes

- The above code is a regular LDPC code since each node on the left is incident by 3 edges and each node on right receives 6 edges. We say that message nodes have degree 3 and check nodes have degree 6.
- As another example of an irregular LDPC code consider:



Low Density Parity Check (LDPC) Codes

- Note that the check nodes receive 7, 7, 3, 6 and 5 ones.



Low Density Parity Check (LDPC) Codes

- An LDPC code is specifies in terms of an edge degree distribution for variable nodes and another degree distribution for check nodes.
- Let λ_i be the fraction of edges that exit variable nodes of degree i. Define degree distribution polynomial for variable nodes:

$$\lambda(x) = \sum_{i \geq 1} \lambda_i x^{i-1}$$

- It is clear that

$$\lambda(1) = \sum_{i \geq 1} \lambda_i = 1$$

Low Density Parity Check (LDPC) Codes

- For the above example:
- $\lambda_1 = \frac{1}{28}$ since only 1 of the edges is incident on an edge of degree 1.
- $\lambda_2 = \frac{4}{28} = \frac{1}{7}$ since 2×2 edges fall upon two nodes of degree 2. Similarly, λ_3 and λ_4 are found to be $\frac{15}{28}$ and $\frac{2}{7}$, respectively. So:

$$\lambda(x) = \frac{1}{28} + \frac{1}{7}x + \frac{15}{28}x^2 + \frac{2}{7}x^3$$

and:

$$\lambda(1) = \frac{1}{28} + \frac{1}{7} + \frac{15}{28} + \frac{2}{7} = 1$$

Low Density Parity Check (LDPC) Codes

- We can also write:

$$\int_0^1 \lambda(x)dx = \sum_{i\geq 1} \frac{\lambda_i}{i} x^i \,(\text{at } x = 1) = \sum_{i\geq 1} \frac{\lambda_i}{i}$$

- In a similar way, a degree distribution ρ(x) can be defined for the check nodes:

$$\rho(x) = \sum_{i\geq 1} \rho_i \, x^{i-1}$$

Where ρ_i is the fraction of edges incident on a check node of degree i.

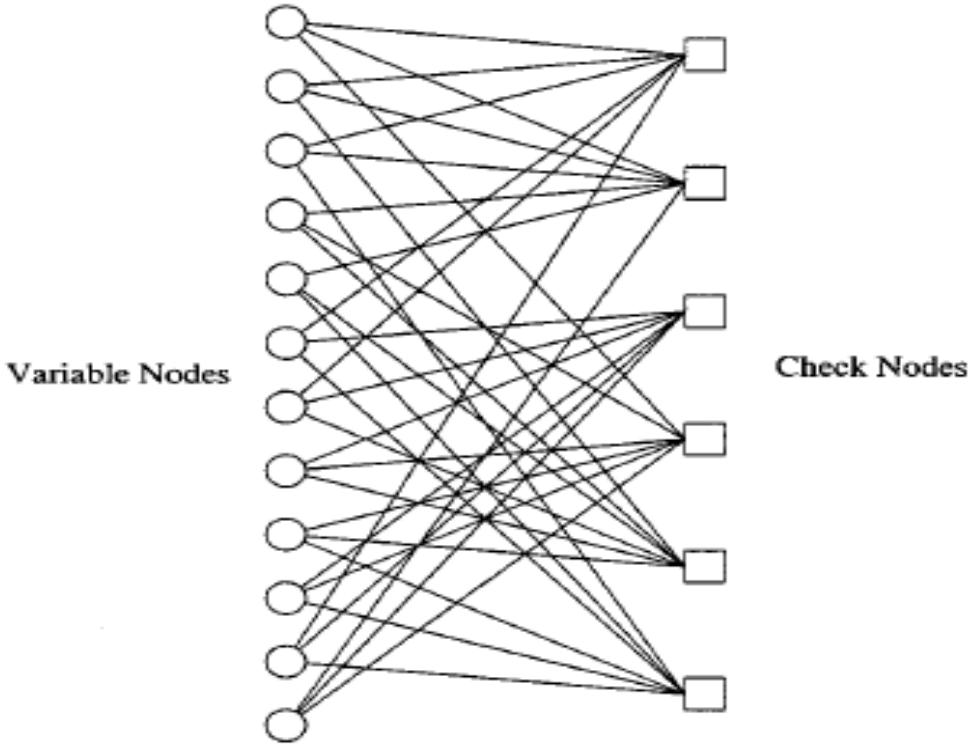
- The rate of a (λ, ρ) code is given by

$$P(\lambda, \rho) = 1 - \frac{\int \rho}{\int \lambda}$$

Where integrals are taken from 0 to 1.

Rate of LDPC Codes

- As the first example consider the (3,6) regular code we saw before·



Graphical representation of a $(3, 6)$-regular LDPC code of length 12. The left nodes represent the variable nodes whereas the right nodes represent the check nodes.

Rate of LDPC Codes

- Note that all the variable nodes have degree 3. That is, all 36 edges leave variable nodes are of degree 3. This means that:

$$\lambda_3 = 1 \; and \; \lambda_i = 0, \qquad \forall i \neq 3.$$

So,

$$\lambda(x) = x^2.$$

- Also, all check nodes (nodes on the right) have degree 6. So, 100% (36 out of 36) edges enter nodes of degree 6.

So, $\rho_6 = 1 \; and \; \rho_i = 0, \; \forall i \neq 6.$

And:

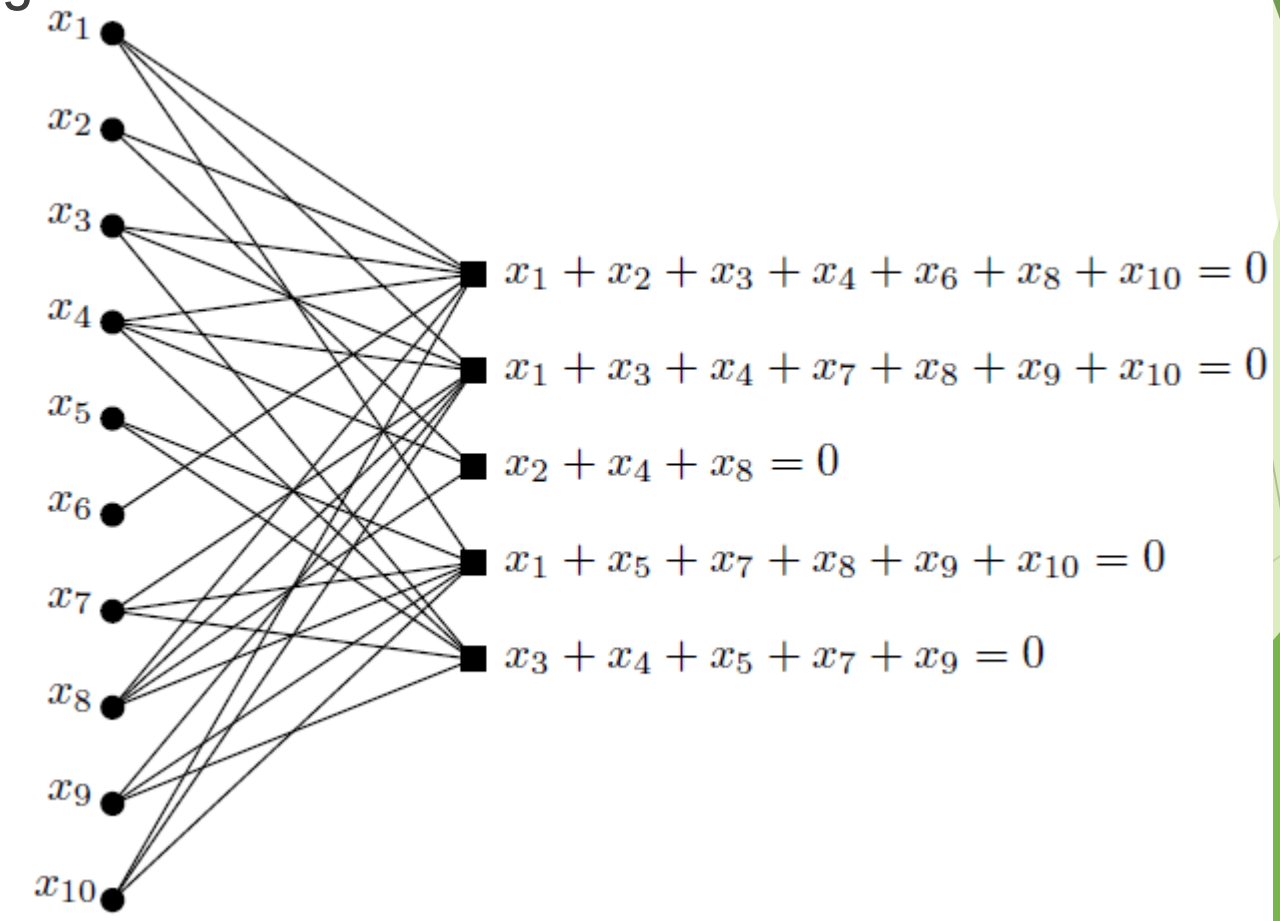
$$\rho(x) = x^5.$$

The rate is:

$$R = 1 - \frac{\int_0^1 x^5}{\int_0^1 x^2} = 1 - \frac{\frac{1}{6}x^6}{\frac{1}{3}x^3}|(x = 1) = 1 - \frac{1}{2} = \frac{1}{2}$$

Rate of LDPC Codes

- This result could already been deducted from the structure and the dimension of the H matrix or the corresponding Tanner graph.
- **Example:** Consider the Irregular Code we saw before:



Rate of LDPC Codes

- We observe that there is one variable node of degree 1 (x_6), two variable nodes of degree 2 $(x_2 \text{ and } x_5)$, 3 nodes of degree 3 and 2 nodes of degree 4. There are 28 edges connecting variable nodes to the check nodes.
- Out of 28 nodes 1 leaves a degree 1 node so, so the fraction of edges leaving a degree 1 variable node is $\frac{1}{28}$ so $\lambda_1 = \frac{1}{28}$. The fraction of edges leaving degree 2 nodes is $\lambda_2 = \frac{4}{28} = \frac{1}{7}$. Fifteen edges leave 5 degree 3 nodes. So, $\lambda_3 = \frac{15}{28}$. Finally, the 8 edges leaving nodes x_4 and x_8 gives $\lambda_4 = \frac{8}{28} = \frac{2}{7}$. So,

$$\lambda(x) = \frac{1}{28} + \frac{1}{7}x + \frac{15}{28}x^2 + \frac{2}{7}x^3$$

Rate of LDPC Codes

- Now, in order to find $\rho(x)$ let's look at the check nodes (the ones on the right).
- We have two check nodes with degree 7, one with degree 6, one with degree 5 and one with degree 3.
- So, out of 28 edges 3 are connected to a check node of degree 3, five are connected to a node of degree 5, six to a degree 6 node and 14 are connected to the two nodes of degree 7. As a result: $\rho_3 = \frac{3}{28}$, $\rho_5 = \frac{5}{28}$, $\rho_6 = \frac{6}{28} = \frac{3}{14}$ and $\rho_7 = \frac{14}{28} = \frac{1}{2}$. Therefore,

$$\rho(x) = \frac{3}{28}x^2 + \frac{5}{28}x^4 + \frac{3}{14}x^5 + \frac{1}{2}x^6.$$

Rate of LDPC Codes

- Let's use $\lambda(x)$ and $\rho(x)$ to find the rate of this code:

$$\int \lambda(x)dx = \int \left(\frac{1}{28} + \frac{1}{7}x + \frac{15}{28}x^2 + \frac{2}{7}x^3\right)dx$$

$$= \frac{1}{28}x + \frac{1}{14}x^2 + \frac{5}{28}x^3 + \frac{1}{14}x^4$$

And:

$$\int_0^1 \lambda(x)dx = \frac{10}{28} = \frac{5}{14}.$$

For the check nodes:

$$\int \rho(x)dx = \int \left(\frac{3}{28}x^2 + \frac{5}{28}x^4 + \frac{3}{14}x^5 + \frac{1}{2}x^6\right)dx$$

$$= \frac{1}{28}x^3 + \frac{1}{28}x^5 + \frac{1}{28}x^6 + \frac{1}{14}x^7$$

Rate of LDPC Codes

- So,

$$\int_0^1 \rho(x)dx = \frac{5}{28}.$$

The rate will be:

$$R = 1 - \frac{\int_0^1 \rho(x)}{\int_0^1 \lambda(x)} = 1 - \frac{\frac{5}{28}}{\frac{10}{28}} = \frac{1}{2}$$

Encoding of LDPC Codes

- While sparsity of the check matrix facilitates the decoding of LDPC codes, the fact that they are defined in terms of parity check matrix makes their encoding complex.
- Now it is a good time to reflect on the question of why we prefer cyclic codes and systematic codes. If a linear code is not cyclic, we need to find codewords by multiplying the information vector U by G. It means n vector multiplications (as the number of columns of G is n). It also is evident that for each vector multiplication, we need on the average $\frac{n}{2}$ operations (say XOR and add). So, the complexity is $O(n^2)$. For a cyclic code the complexity is $O(n)$, i.e. , it is linear in n. For a non-cyclic but systematic code, we need to find $(n-k)$ parities each requiring (on the average $\frac{k}{2}$) operation. So, the order of encoding is $O(nk)$.

Encoding of LDPC Codes

- Encoding of LDPC codes is difficult since the graph can only show whether a bit stream is a codeword or not. It cannot be used for relating the messages to code words. To ease encoding there are several different approaches:

1) To use cascaded rather than bi-partite graphs. This means doing encoding in several stages. By choosing the number and size of the stages, one can design codes that are encodeable and decodable in linear time. The disadvantage of this technique is that each stage adds parity to the message and parity form previous stage. The length of data to the total code word length is small (low rate). This results in performance loss compared to a standard LDPC code.

Encoding of LDPC Codes

2) The other approach is to use codes that have lower triangular form. This is similar to solving system of linear equations using Gauss elimination.This approach while guarantees linear time encoding complexity, results in some loss of performance due to being restricted to a class of LDPC codes.

3) Starting from a standard LDPC code, we try to make its parity check matrix lower triangular and stop when we cannot go further (Richardson and Urbanke).

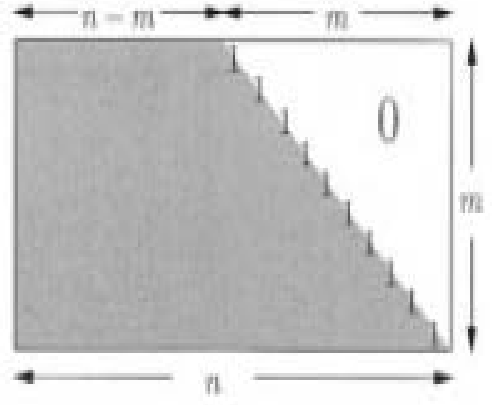
- This results in an approximate lower triangular matrix.

Encoding of LDPC Codes

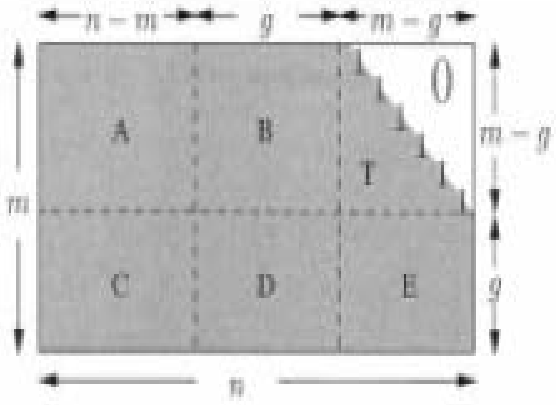
- The matrix H is written as:

$$H = \begin{pmatrix} \boldsymbol{A} & \boldsymbol{B} & \boldsymbol{T} \\ \boldsymbol{C} & \boldsymbol{D} & \boldsymbol{E} \end{pmatrix}$$

Where $\boldsymbol{A}$ is an $(m-g) \times (n-m)$ matrix, $\boldsymbol{B}$ is $(m-g) \times g$, $\boldsymbol{T}$ is $(m-g) \times (m-g)$, $\boldsymbol{C}$ is $g \times (n-m)$, $\boldsymbol{D}$ is $g \times g$, and $\boldsymbol{E}$ is $\mathrm{g} \times (m-g)$,



An equivalent parity-check matrix in lower triangular form.



The parity-check matrix in approximate lower triangular form.

Encoding of LDPC Codes

- We form the matrix:

$$S = \begin{pmatrix} I & 0 \\ -ET^{-1} & I \end{pmatrix}$$

and multiply H from the left by S to get:

$$H = \begin{pmatrix} A & B & T \\ -ET^{-1}A + C & -ET^{-1}A + D & 0 \end{pmatrix}$$

- Let the codeword be:

$$\boldsymbol{v} = (\boldsymbol{s}, \boldsymbol{p_1}, \boldsymbol{p_2})$$

Where s is the systematic part and $(\boldsymbol{p_1}, \boldsymbol{p_2})$ is the parity part with $\boldsymbol{p_1}$ of length g and $\boldsymbol{p_2}$ of length $(\mathrm{m} - g)$.

- Considering the fact that: $\boldsymbol{Hv^T} = \boldsymbol{0^T}$ we have:

$$\boldsymbol{A}.\boldsymbol{s^T} + \boldsymbol{B}.\boldsymbol{p_1^T} + \boldsymbol{T}.\boldsymbol{p_2^T} = 0,$$

$$\left(-\boldsymbol{ET^{-1}A} + \boldsymbol{C}\right).\boldsymbol{s^T} + \left(-\boldsymbol{ET^{-1}A} + \boldsymbol{D}\right).\boldsymbol{p_1^T} = 0$$

Encoding of LDPC Codes

- Assume that $\left(-\boldsymbol{ET^{-1}A}+\boldsymbol{D}\right)$ is not singular. Then the second equation gives:

$$\boldsymbol{p_1^T} = -\left(-\boldsymbol{ET^{-1}A}+\boldsymbol{D}\right)^{-1}\left(-\boldsymbol{ET^{-1}A}+\boldsymbol{C}\right).\boldsymbol{s^T}$$

Note that the $g \times (n-m)$ matrix:

$$\boldsymbol{M} = -\left(-\boldsymbol{ET^{-1}A}+\boldsymbol{D}\right)^{-1}\left(-\boldsymbol{ET^{-1}A}+\boldsymbol{C}\right)$$

- Can be precomputed and used to find $\boldsymbol{p_1^T} = \boldsymbol{Ms^T}$ with complexity of order $O\left(g \times (n-m)\right)$. We see in the next slide that this can be done with complexity $O(n) + O(g^2)$.
- Next, we compute $\boldsymbol{p_1}$:

$$\boldsymbol{p_2^T} = \boldsymbol{T^{-1}}\left(\boldsymbol{A}.\boldsymbol{s^T} + \boldsymbol{B}.\boldsymbol{p_1^T}\right).$$

Encoding of LDPC Codes

- Table 1 presents the complexity of computing $\boldsymbol{p_1}$:

TABLE I
EFFICIENT COMPUTATION OF $p_1^T = -\phi^{-1}(-ET^{-1}A + C)s^T$

Operation	Comment	Complexity
As^T	Multiplication by sparse matrix	$O(n)$
$T^{-1}[As^T]$	$T^{-1}[As^T] = y^T \Leftrightarrow [As^T] = Ty^T$	$O(n)$
$-E[T^{-1}As^T]$	Multiplication by sparse matrix	$O(n)$
Cs^T	Multiplication by sparse matrix	$O(n)$
$[-ET^{-1}As^T] + [Cs^T]$	Addition	$O(n)$
$-\phi^{-1}[-ET^{-1}As^T + Cs^T]$	Multiplication by dense $g \times g$ matrix	$O(g^2)$

- So, the complexity of computing $\boldsymbol{p_1}$ is $O(n) + O(g^2)$.

Encoding of LDPC Codes

- Table 2 presents the complexity of computing $\boldsymbol{p_2}$:

TABLE II
EFFICIENT COMPUTATION OF $p_2^T = -T^{-1}(As^T + Bp_1^T)$

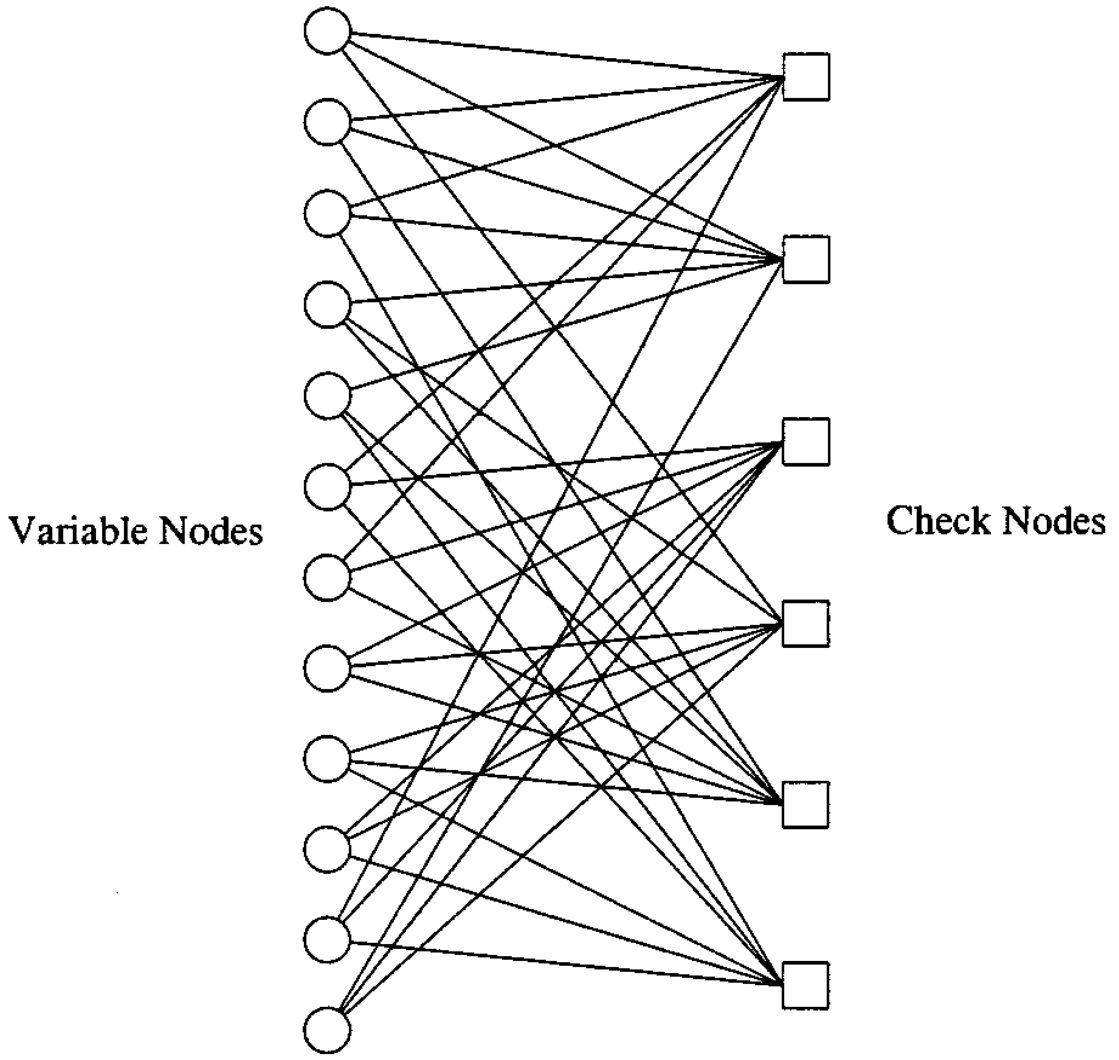
Operation	Comment	Complexity
As^T	Multiplication by sparse matrix	$O(n)$
Bp_1^T	Multiplication by sparse matrix	$O(n)$
$[As^T] + [Bp_1^T]$	Addition	$O(n)$
$-T^{-1}[As^T + Bp_1^T]$	$-T^{-1}[As^T + Bp_1^T] = y^T \Leftrightarrow -[As^T + Bp_1^T] = Ty^T$	$O(n)$

- It shows that the complexity of computing $\boldsymbol{p_2}$ is $O(n)$. So, the overall complexity is $O(n) + O(g^2)$.
- So far, we have assumed that $\left(-\boldsymbol{ET^{-1}A} + \boldsymbol{D}\right)$ is not singular. If it was singular, instead of pre-multiplication, we need to use Gaussian elimination.

Encoding of LDPC Codes: Exampl

- As an example, let's take the (3, 6) regular code we saw before.
- The parity check matrix for this code is:

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



Encoding of LDPC Codes: Example

- By reordering the columns as: 1, 2, 3, 4, 5, 6, 7, 10, 11, 12, 8, 9 , we get,

$$\left(\begin{array}{c|c|c} A & B & T \\ \hline C & D & E \end{array}\right)$$

$$= \left(\begin{array}{cccccc|cc|cccc} 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & \mathbf{1} & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & \mathbf{1} & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & \mathbf{1} \\ \hline 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{array}\right)$$

- We use Gaussian elimination to clear $\boldsymbol{E}$,

$$\left(\begin{array}{cccccc|cc|cccc} 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & \mathbf{1} & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & \mathbf{1} & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & \mathbf{1} \\ \hline 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \end{array}\right)$$

Encoding of LDPC Codes: Example

- We observe that:

$$\left(-\boldsymbol{ET^{-1}A}+\boldsymbol{D}\right)=\begin{pmatrix}1 & 1\\ 1 & 1\end{pmatrix}$$

and is singular. To remove this singularity, we can exchange columns 5 and 8. This means that the column permutation (referenced to the original matrix) is: 1, 2, 3, 4, 10, 6, 7, 5, 11, 12, 8, 9.

- The resulting equivalent H matrix is:

$$\left(\begin{array}{c|c|c} A & B & T \\ \hline C & D & E \end{array}\right)$$

$$= \left(\begin{array}{cccccc|cc|cccc} 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & \mathbf{1} & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & \mathbf{1} & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & \mathbf{1} \\ \hline 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{array}\right)$$

Encoding of LDPC Codes: Example

- Now, we have,

$$(-\boldsymbol{ET^{-1}A} + \boldsymbol{D}) = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$$

that is not singular anymore.

Writing the codeword as $\boldsymbol{v} = (\boldsymbol{s}, \boldsymbol{p_1}, \boldsymbol{p_2})$ we have:

$$\boldsymbol{A}.\boldsymbol{s}^{\boldsymbol{T}} + \boldsymbol{B}.\boldsymbol{p}_{\boldsymbol{1}}^{\boldsymbol{T}} + \boldsymbol{T}.\boldsymbol{p}_{\boldsymbol{2}}^{\boldsymbol{T}} = 0,$$

$$\boldsymbol{\mu}.\boldsymbol{s}^{\boldsymbol{T}} + \boldsymbol{\phi}.\boldsymbol{p}_{\boldsymbol{1}}^{\boldsymbol{T}} = 0$$

Where $A = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$, $B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{bmatrix}$, $T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}$,

$$\boldsymbol{\mu} = \begin{bmatrix} 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} \text{ and } \boldsymbol{\phi} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

Encoding of LDPC Codes: Example

- Let's encode $\boldsymbol{s} = (1, 0, 0, 0, 0, 0)$

$$\boldsymbol{As^T} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

$$\boldsymbol{T^{-1}As^T} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$-\boldsymbol{E}\left[\boldsymbol{T^{-1}As^T}\right] = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Encoding of LDPC Codes: Example

$$\boldsymbol{Cs^T} = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$-\boldsymbol{E}\left[\boldsymbol{T^{-1}As^T}\right] + \boldsymbol{Cs^T} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

So,

$$\boldsymbol{p}_1^T = \phi^{-1}\left[-\boldsymbol{ET^{-1}A} + \boldsymbol{C}\right]\boldsymbol{s}^T = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

- And $\boldsymbol{p}_1 = [0 \quad 1]$.

Encoding of LDPC Codes: Example

- To find $\boldsymbol{p}_2$, we compute:

$$\boldsymbol{Bp}_1^{\boldsymbol{T}} = \begin{bmatrix} \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}$$

$$\boldsymbol{As}^{\boldsymbol{T}} + \boldsymbol{Bp}_1^{\boldsymbol{T}} = \begin{bmatrix} \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \end{bmatrix}$$

$$\boldsymbol{p}_2^{\boldsymbol{T}} = \boldsymbol{T}^{-\mathbf{1}}\left[\boldsymbol{As}^{\boldsymbol{T}} + \boldsymbol{Bp}_1^{\boldsymbol{T}}\right] = \begin{bmatrix} \mathbf{1} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \end{bmatrix}$$

Therefore, the codeword is:

$$\boldsymbol{v} = (\boldsymbol{s}, \boldsymbol{p}_1, \boldsymbol{p}_2) = (1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0).$$

Decoding of LDPC Codes

- Decoding of LDPC Codes is performed using message passing or belief propagation (BP) algorithm.
- BP is an iterative algorithm where in each iteration message nodes send the likelihood of their value to all check nodes to which they are connected and check nodes send message to variable (message) nodes based on what they have received from other message nodes.
- Message nodes and check nodes exclude what they have received from one another when they send a message.
- In BP the message sent from a message node is based on that node's received information (from the channel) and what it gets from check nodes connected to it (except the one it wants to send the message to). These are in the form of probability or likelihood ratio.

Decoding of LDPC Codes

- A message node v sends to a check node c (the probability or likelihood) of v having a certain value given its observation and what it has received in the previous iteration from its neighboring check nodes other than c.
- In the same way, a check node c sends to v the probability that c has a certain value given all the messages passed to c in the previous iteration from message nodes other than v.
- The messages passed from message nodes to check nodes and vice versa are in the form of likelihood or log likelihood ratios that we encountered before when discussing Turbo Codes. Let's revisit them here.

Decoding of LDPC Codes

- Likelihood ratio of a binary random variable x is:

$$L(x) = \frac{P(x=0)}{P(x=1)}$$

- The conditional Likelihood ratio of x given $\boldsymbol{y}$ is:

$$L(x|\boldsymbol{y}) = \frac{P(x=0|\boldsymbol{y})}{P(x=1|\boldsymbol{y})}$$

- If x is uniformly distributed, we have:

$$L(x|\boldsymbol{y}) = L(\boldsymbol{y}|x)$$

- If $y_1, y_2, \ldots, y_d$ are independent random variables:

$$L(x|\boldsymbol{y}) = \prod_{i=1}^{d} L(x|y_i)$$

Or: $\log L(x|\boldsymbol{y}) = \sum_{i=1}^{d} \log L(x|y_i) = \sum_{i=1}^{d} ln\, L(x|y_i)$

Decoding of LDPC Codes

- Now assume $x_1, x_2, \dots, x_l$ are binary random variables and $y_1, y_2, \dots, y_l$ are real or integer random variables. We would like to find:

$$\ln l\,(x_1 \oplus x_2 \oplus \cdots \oplus x_l \,|y_1, \dots, y_l)$$

- For two bits ($l = 2$) if we let:

$$2P[x_1 = 0|y_1] - 1 = p \Rightarrow P[x_1 = 0|y_1] = \frac{1+p}{2}$$

And

$$2P[x_2 = 0|y_2] - 1 = q \Rightarrow P[x_2 = 0|y_2] = \frac{1+q}{2}$$

Then

$$P[x_1 \oplus x_2 = 0|y_1, y_2]$$

$$= P[x_1 = 0, x_2 = 0|y_1, y_2] + P[x_1 = 1, x_2 = 1|y_1, y_2]$$

$$= P[x_1 = 0|y_1]\, P[x_2 = 0|y_2] + P[x_1 = 1|y_1] P[x_2 = 1|y_2]$$

Decoding of LDPC Codes

- Substituting $P[x_1 = 0|y_1] = \frac{1+p}{2}$ and $P[x_2 = 0|y_2] = \frac{1+q}{2}$:

$$P[x_1 \oplus x_2 = 0|y_1, y_2] = \frac{1+p}{2}.\frac{1+q}{2} + \frac{1-p}{2}.\frac{1-q}{2}$$

$$= \frac{2+2pq}{2} = 1 + pq$$

- So: $2P[x_1 \oplus x_2 = 0|y_1, y_2] - 1 = pq.$
- Therefore,

$$2P[x_1 \oplus x_2 \oplus \cdots \oplus x_l|y_1, y_2, \dots, y_l] - 1 = \prod_{i=1}^{l} [2P(x_i = 0|y_i) - 1]$$

- Let $\lambda_i = log_l \frac{P(x_i=0|y_i)}{P(x_i=1|y_i)}$ be the log-likelihood ratio of x_i given y_i.
- So, $P(x_i = 0|y_i) = \frac{e^{\lambda_i}}{e^{\lambda_i+1}}$

Decoding of LDPC Codes

- Finally,

$$2\,P(x_i = 0|y_i) - 1 = \frac{e^{\lambda_i} - 1}{e^{\lambda_i} + 1} = \frac{e^{\lambda_i/2} - e^{-\lambda_i/2}}{e^{\lambda_i/2} + e^{-\lambda_i/2}}$$

Or:

$$2P(x_i = 0|y_i) - 1 = \tanh(\frac{\lambda_i}{2})$$

Decoding of LDPC Codes

- Let $m_{vc}^{(l)}$ be the message passed from message node v to check node c in iteration L. Similarly, denote by $m_{cv}^{(l)}$ the message from c to v. Then the update equations in BP are:

$$\mathrm{m}_{\mathrm{vc}}^{(\ell)} = \begin{cases} \mathrm{m}_{\mathrm{v}}, & \text{if } \ell = 0, \\ \mathrm{m}_{\mathrm{v}} + \sum_{\mathrm{c}' \in C_{\mathrm{v}} \setminus \{\mathrm{c}\}} \mathrm{m}_{\mathrm{c}'\mathrm{v}}^{(\ell-1)}, & \text{if } \ell \geq 1, \end{cases}$$

$$\mathrm{m}_{\mathrm{cv}}^{(\ell)} = \ln \frac{1 + \prod_{\mathrm{v}' \in V_{\mathrm{c}} \setminus \{\mathrm{v}\}} \tanh\left(\mathrm{m}_{\mathrm{v}'\mathrm{c}}^{(\ell)}/2\right)}{1 - \prod_{\mathrm{v}' \in V_{\mathrm{c}} \setminus \{\mathrm{v}\}} \tanh\left(\mathrm{m}_{\mathrm{v}'\mathrm{c}}^{(\ell)}/2\right)},$$

- Where C_v is the set of check nodes connected to v. Similarly, V_c are variable nodes connected to c.

Bit Flip Decoding of LDPC Codes

- This method was devised by Gallager:
- When we compute syndromes, i.e., the value of check nodes, if they are all zero, we assume there is no error and stop.
- Then, we find for each variable node, the number of failed (1). Check nodes and flip the one with maximum number of failed check nodes connected to it.
- We then re-calculate the syndromes and flip the bit that is most connected to those with value 1.
- We continue iteration above until either all check nodes have zero value or until a certain number of pre-determined iterations have been done with no success (failure in this case).

Bit Flip Decoding of LDPC Codes

- The above, simple BP algorithm is given below:

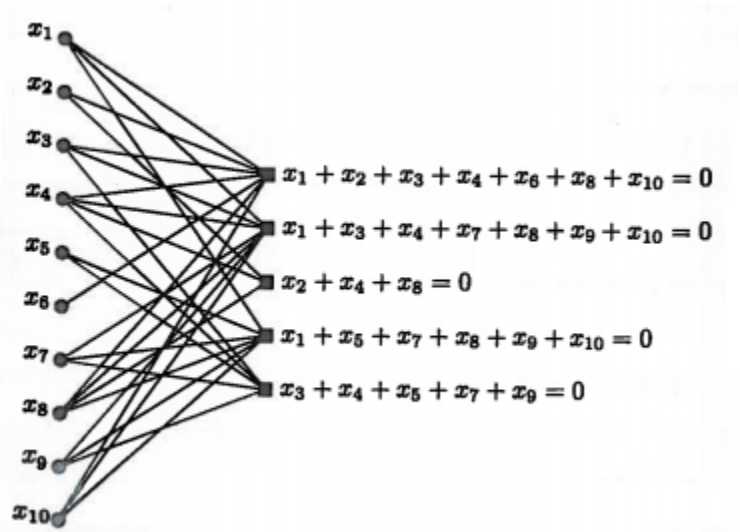
1. Compute syndromes by $\underline{r}.\underline{H}^T = \underline{S}$. If all check-sums are 0 stop.
2. Find the number of failed parity check equations for each node.

Decode the number of failed check node for each message node by f_i , $i = 1, 2, .., n$.

1. Identify the set of bits S for which f_i is the largest.
2. Flip bits in S.
3. Repeat steps 1 to 4 until the parity-check sums are zero (success), or a preset maximum number of iterations is reached (decoding failure)

Bit Flip Decoding: Example

- **Example:** Assume that we have used this code and have received 0000000100 that is $x_8 = 1$ and $x_i = 0 \;\; i \neq 8$



- Step 1: Compute syndromes:

$$X = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}\} = \{0,0,0,0,0,0,0,1,0,0\}$$

This results in syndromes as:

$$C = \{c_1, c_2, c_3, c_4, c_5\} = \{1,1,1,1,0\}$$

Bit Flip Decoding: Example

- Obviously, this indicates an error.
- Step 2: Find the number of failed parity check equations for each node:
- The table below shows the frequency of occurrence of each node in the failed parity check equations:

x_i	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
f_i	3	2	2	3	1	1	2	4	2	3

- Step 1: Compute syndromes:

Step 3: Identify the bits for which the frequency of occurrence is the largest. From step 2, this is clearly x_8, which has occurred in all four failed parity check equations.

Step 4: Flip bits from step 3: By flipping x_8, the code word will be $X = (0,0,0,0,0,0,0,0,0,0)$. This results in all zero syndromes that means successful LDPC decoding

Comparison of LDPC and Turbo Codes

This figure compares the performance of LDPC codes and Turbo codes for different codeword lengths. All codes are of rate ½ .