

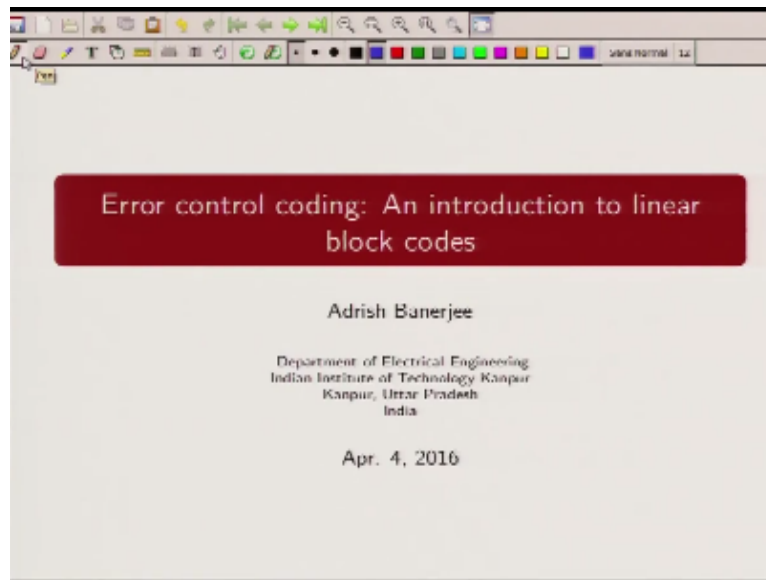
Indian Institute of Technology Kanpur
National Programme on Technology Enhanced Learning (NPTEL)
Course Title
Error Control Coding: An Introduction to Linear Block Codes

Lecture-8
Low density parity check codes

by
Prof. Adrish Banerjee
Department of Electrical Engineering, IIT Kanpur

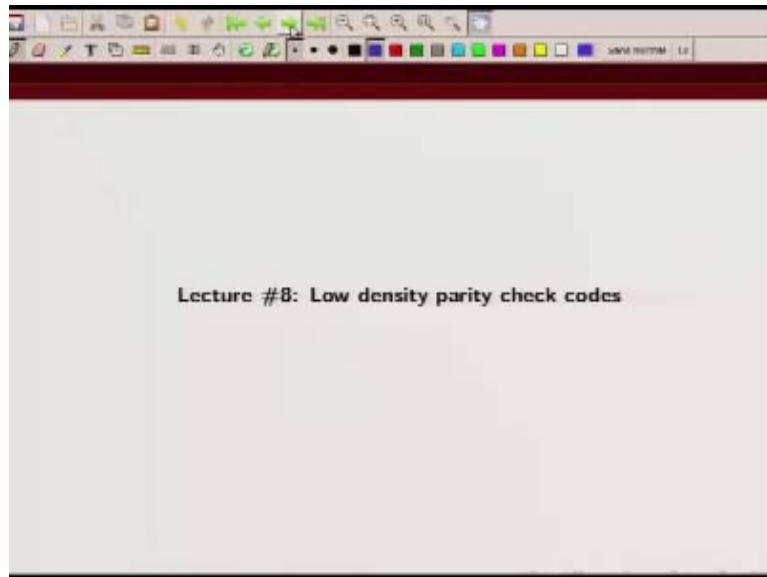
Welcome to the course on error control coding, an introduction to linear block codes.

(Refer Slide Time: 00:19)

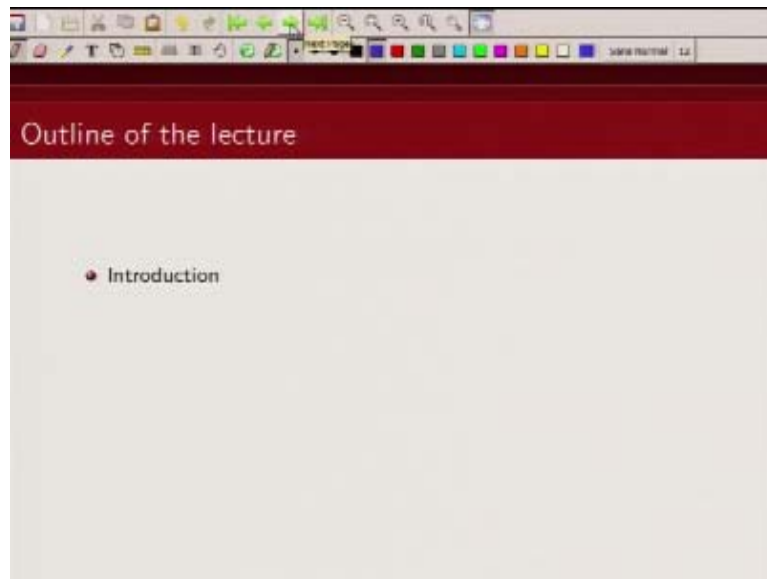


Today we are going to give a brief introduction to low density parity check codes.

(Refer Slide Time: 00:27)

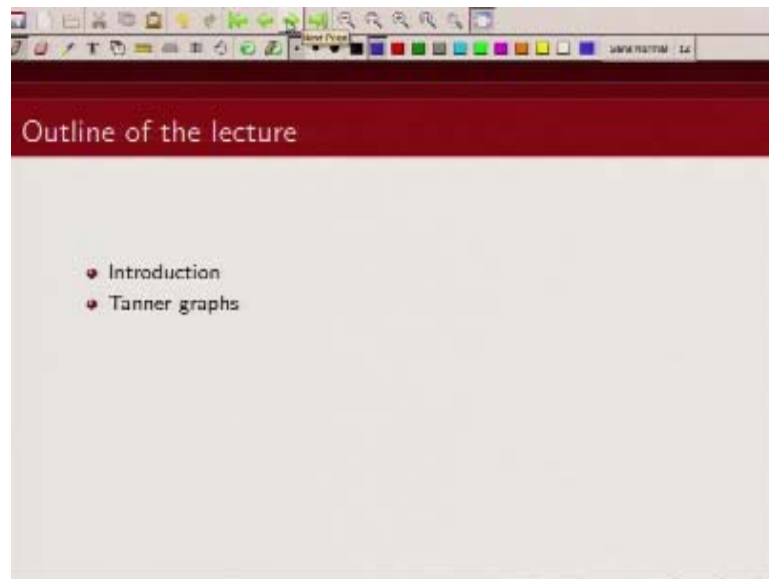


(Refer Slide Time: 00:28)



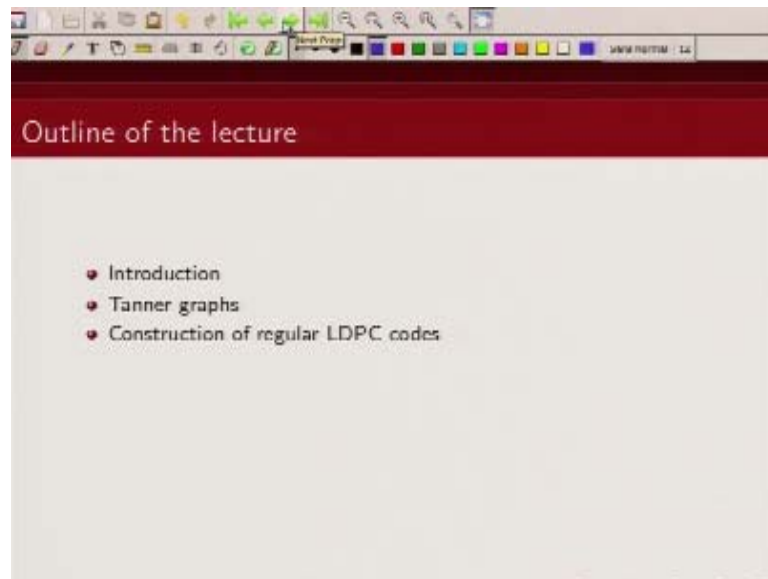
So we will start off with very basic definition of what do we mean by a low density parity check matrix. What do we mean by low density?

(Refer Slide Time: 00:38)



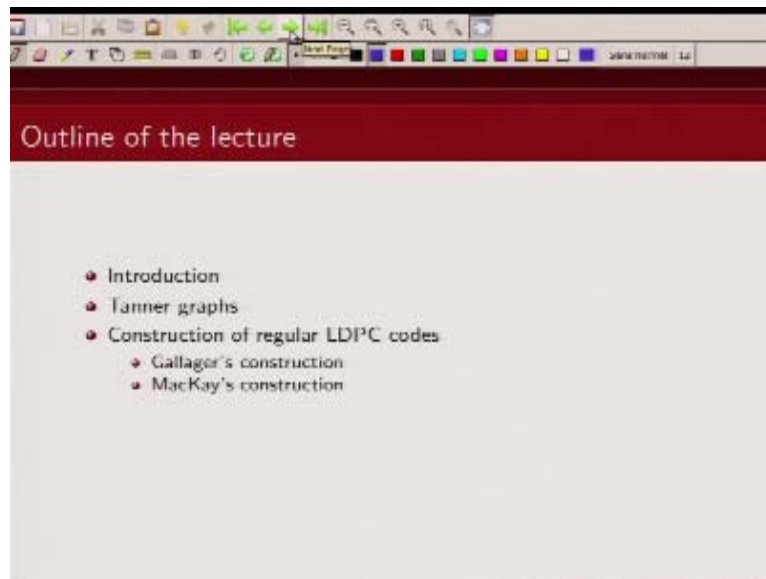
And then we will show how we can write the parity check matrix using a bipartite graph which is known as tanner graph.

(Refer Slide Time: 00:49)



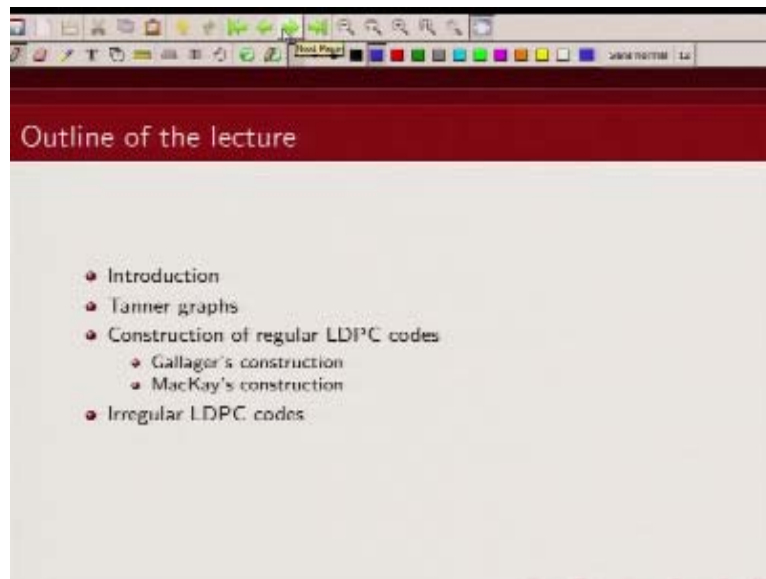
Then we will talk about what is a regular LDPC code.

(Refer Slide Time: 00:54)



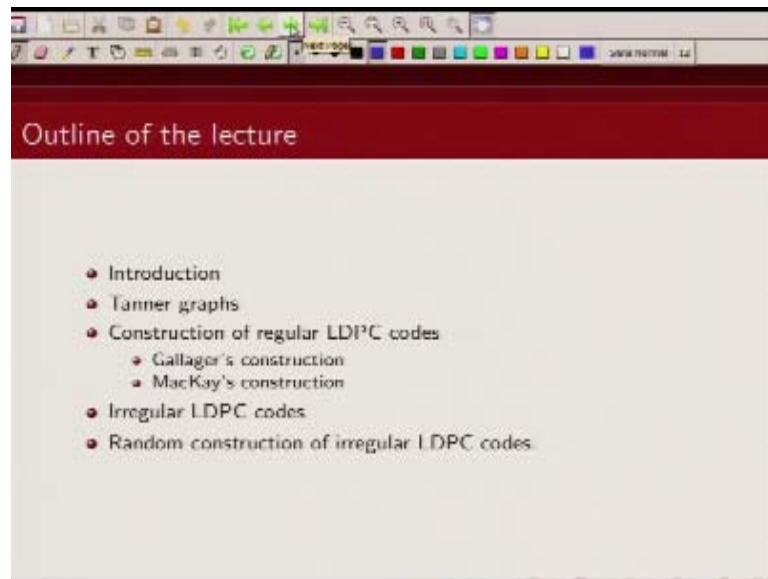
And we will give some few simple constructions of regular LDPC code.

(Refer Slide Time: 01:01)



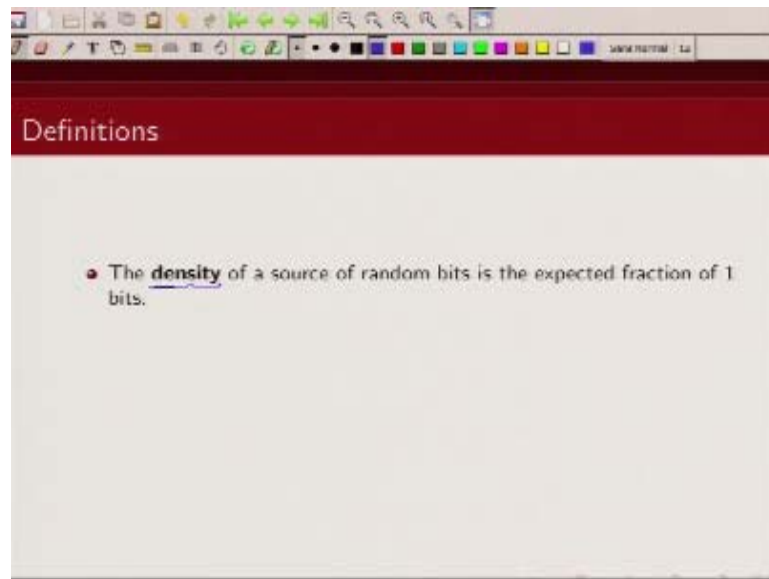
Then we will talk about irregular LDPC code.

(Refer Slide Time: 01:05)



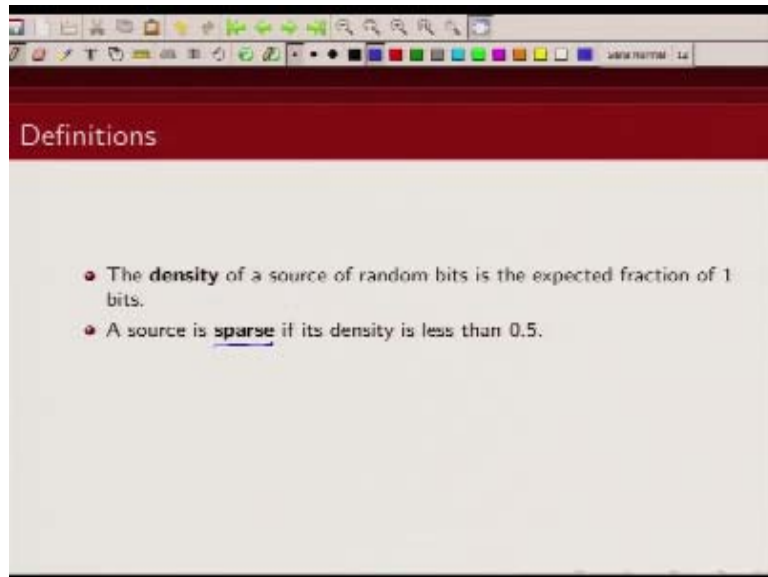
Then again we will give some very simple construction of irregular LDPC codes.

(Refer Slide Time: 01:11)



So what do we mean by low density? So we will first define what do we mean by density. So a density of a source is basically the expected number of ones in the source.

(Refer Slide Time: 01:26)

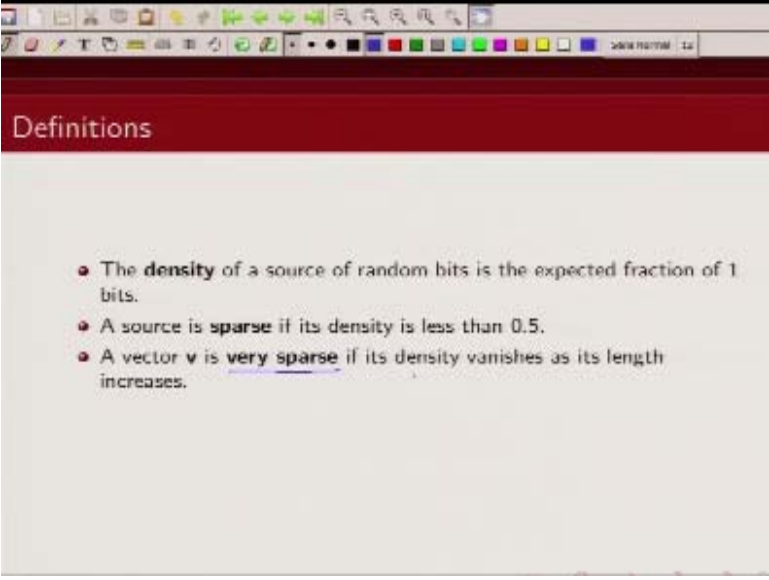


The image shows a presentation slide with a red header bar containing the word "Definitions" in white. Below the header, on a light gray background, are two bullet points. The first bullet point defines "density" as the expected fraction of 1 bits. The second bullet point states that a source is "sparse" if its density is less than 0.5. The word "sparse" is underlined in the original image.

- The **density** of a source of random bits is the expected fraction of 1 bits.
- A source is sparse if its density is less than 0.5.

Now when is it a low density, now a source is low density or sparse if the density of 1 is less than 0.5.

(Refer Slide Time: 01:37)

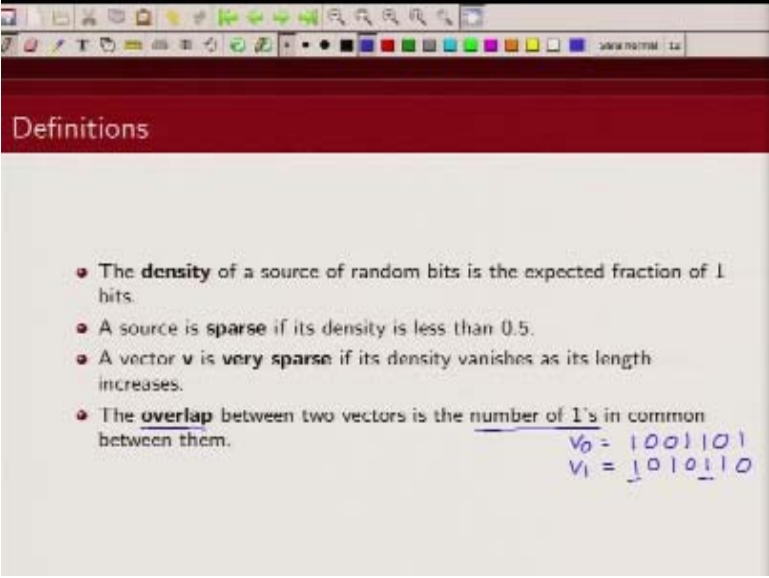


Definitions

- The **density** of a source of random bits is the expected fraction of 1 bits.
- A source is **sparse** if its density is less than 0.5.
- A vector v is **very sparse** if its density vanishes as its length increases.

And we say the vector is very low density or it is low density if the density vanishes as the length of the vector increases. In other words number of ones are fixed even if we increase the length of the vector, in that case the density will vanish as length increases.

(Refer Slide Time: 02:02)



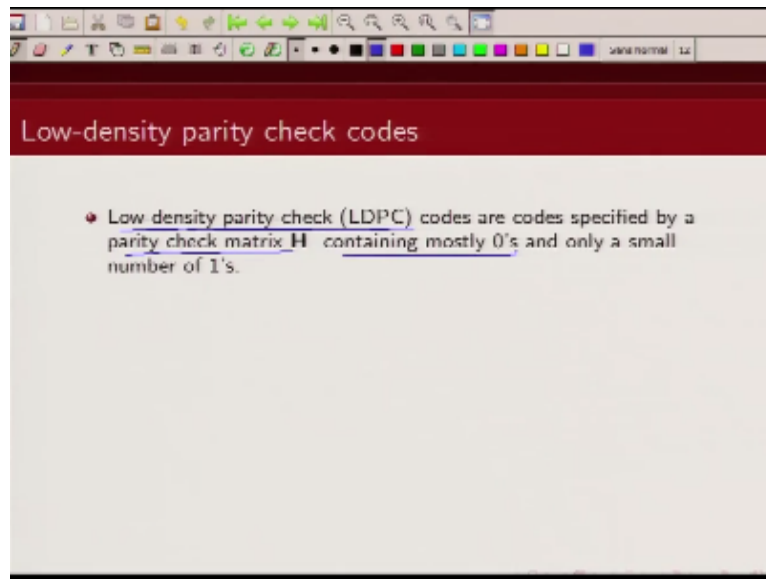
The slide is titled "Definitions" in a red header. It contains a bulleted list of definitions. The fourth definition, "The **overlap** between two vectors is the number of 1's in common between them," is underlined. To its right, handwritten in blue ink, are two binary vectors: $v_0 = 1001101$ and $v_1 = 1010110$. The overlap of two 1s at the first and second positions is visually apparent.

- The **density** of a source of random bits is the expected fraction of 1 bits.
- A source is **sparse** if its density is less than 0.5.
- A vector v is **very sparse** if its density vanishes as its length increases.
- The **overlap** between two vectors is the number of 1's in common between them.

$v_0 = 1001101$
 $v_1 = 1010110$

We will also define a term which is called an overlap, so if you have two n tuples we call an overlap between two vectors as the number of positions in which the ones are common. So for example if you have a vector let us call it v_0 which is 1001101 and you have a vector v_1 which is 1010110, then we can see there is an overlap here in one location, two location, so there is an overlap of two.

(Refer Slide Time: 02:45)



So what is a low density parity check code, as the name suggest a low density parity check code are specified by a parity check matrix which is of low density. And what do we mean by low density, so the number of ones in this parity check matrix is very small, it is less than $\frac{1}{2}$. So an LDPC codes are specified by a parity check matrix which consist of mostly zeros and very few ones.

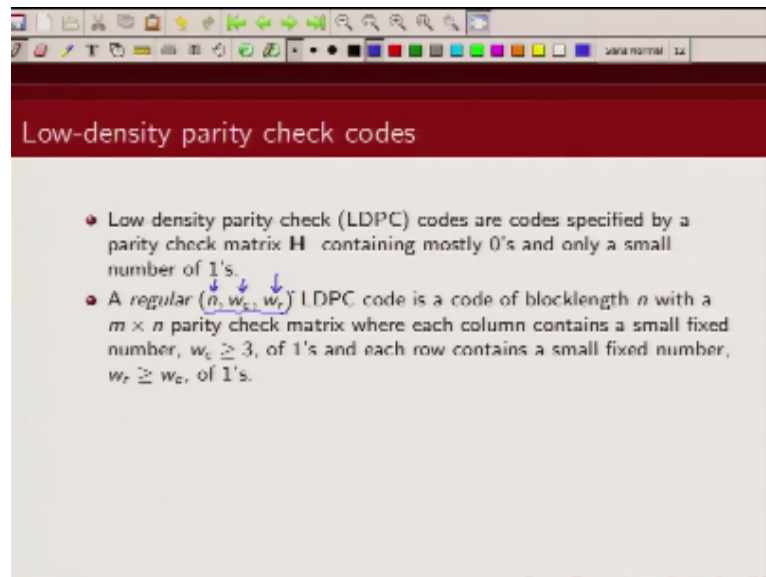
(Refer Slide Time: 03:23)

Low-density parity check codes

- Low density parity check (LDPC) codes are codes specified by a parity check matrix \mathbf{H} containing mostly 0's and only a small number of 1's.
- A *regular* (n, w_c, w_r) LDPC code is a code of blocklength n with a $m \times n$ parity check matrix where each column contains a small fixed number, $w_c \geq 3$, of 1's and each row contains a small fixed number, $w_r \geq w_c$, of 1's.

Now what is a regular LDPC code? A regular LDPC code is defined by these three parameters, this is the code length, this is number of ones in the columns of the parity check matrix. So a regular LDPC code has same number of ones in each of the columns of the parity check matrix.

(Refer Slide Time: 03:51)

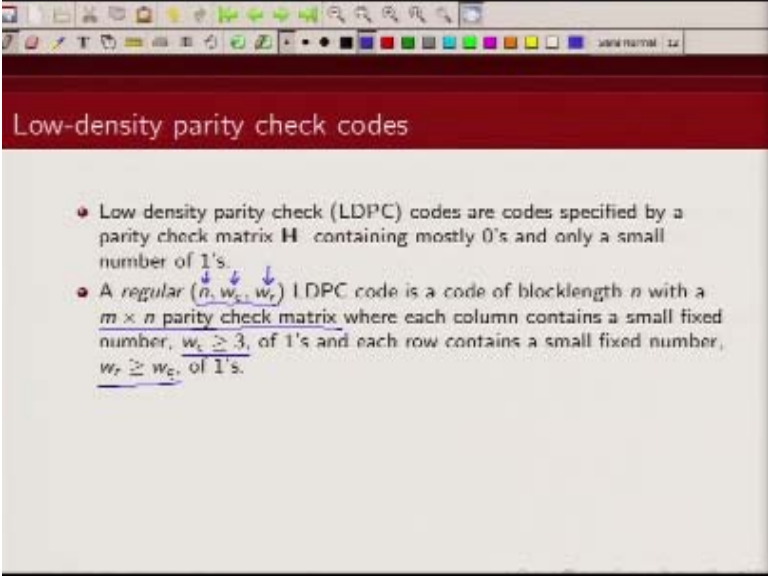


Low-density parity check codes

- Low density parity check (LDPC) codes are codes specified by a parity check matrix \mathbf{H} containing mostly 0's and only a small number of 1's.
- A *regular* (n, w_c, w_r) LDPC code is a code of blocklength n with a $m \times n$ parity check matrix where each column contains a small fixed number, $w_c \geq 3$, of 1's and each row contains a small fixed number, $w_r \geq w_c$, of 1's.

And that number is given by w subscript C. Similarly w subscript R gives us the number of ones in each of the row of this parity check matrix.

(Refer Slide Time: 04:08)



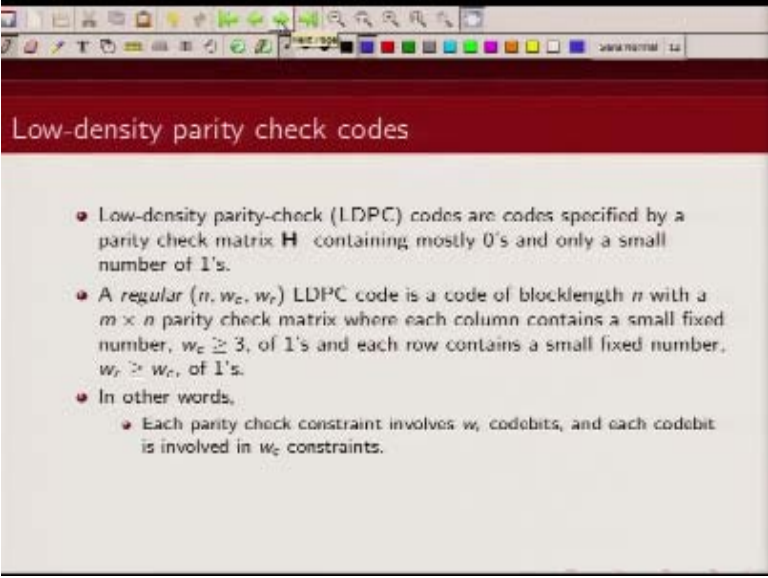
The image shows a presentation slide with a red header bar containing the title "Low-density parity check codes". Below the header, there is a list of two bullet points. The first bullet point states that LDPC codes are specified by a parity check matrix H with mostly 0s and a few 1s. The second bullet point defines a regular LDPC code with parameters (n, w_c, w_r) , where n is the blocklength, w_c is the column weight, and w_r is the row weight. The slide also features a standard Windows taskbar at the top with various application icons.

Low-density parity check codes

- Low density parity check (LDPC) codes are codes specified by a parity check matrix H containing mostly 0's and only a small number of 1's.
- A *regular* (n, w_c, w_r) LDPC code is a code of blocklength n with a $m \times n$ parity check matrix where each column contains a small fixed number, $w_c \geq 3$, of 1's and each row contains a small fixed number, $w_r \geq w_c$, of 1's.

Again for a regular LDPC code the number of ones in each row is same. So a regular LDPC code is specified by this block length n and number of ones in each of the columns and number of ones in each of the rows. So we can describe it by a low density parity check matrix of $m \times n$ where each column has a fixed number of ones and that is $w_c \geq 3$ and that as we greater than 3, this has to do with distance properties of LDPC codes. And each row has w_r odd number of ones where w_r is greater than equal to w_c .

(Refer Slide Time: 04:51)

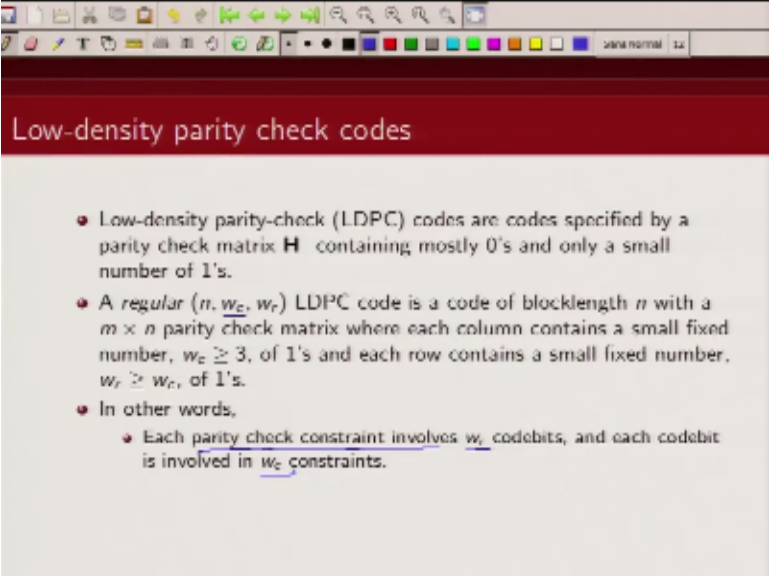


Low-density parity check codes

- Low-density parity-check (LDPC) codes are codes specified by a parity check matrix \mathbf{H} containing mostly 0's and only a small number of 1's.
- A regular (n, w_c, w_r) LDPC code is a code of blocklength n with a $m \times n$ parity check matrix where each column contains a small fixed number, $w_c \geq 3$, of 1's and each row contains a small fixed number, $w_r \geq w_c$, of 1's.
- In other words,
 - Each parity check constraint involves w_c codebits, and each codebit is involved in w_r constraints.

In other words, now what do the rows in the parity check matrix specify? Now if there are w_r ones in rows of the parity check matrix it specifies that w_r bits are participating in a parity check equation. And in all the parity check equations the same number of bits are participating.

(Refer Slide Time: 05:20)

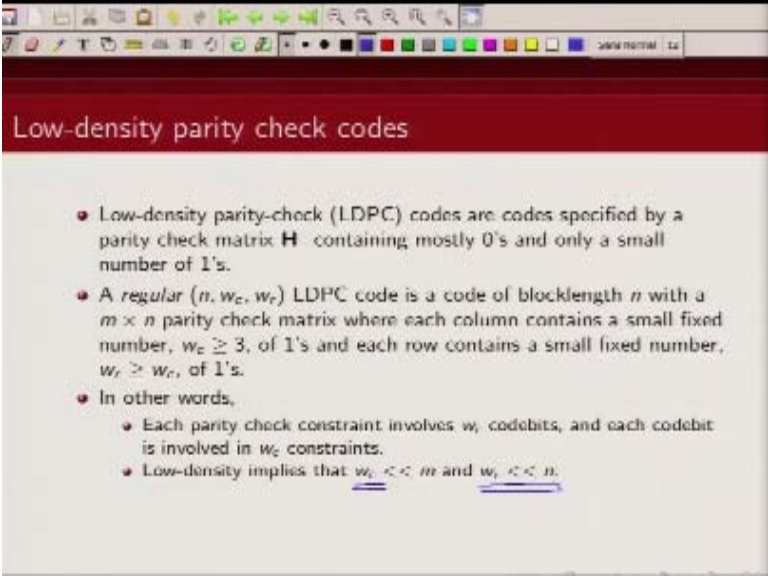


Low-density parity check codes

- Low-density parity-check (LDPC) codes are codes specified by a parity check matrix \mathbf{H} containing mostly 0's and only a small number of 1's.
- A regular (n, w_c, w_r) LDPC code is a code of blocklength n with a $m \times n$ parity check matrix where each column contains a small fixed number, $w_c \geq 3$, of 1's and each row contains a small fixed number, $w_r \geq w_c$, of 1's.
- In other words,
 - Each parity check constraint involves w_c codebits, and each codebit is involved in w_r constraints.

And what is the implication of w_c ones in each column, it means that each bit appears in w_c parity check equations. So each bit participates in w_c parity check equations. So that is what I am saying here, so each parity check constraint in an regular LDPC code will have w_r code bits. And each code bit appears in w_c parity check constraints.

(Refer Slide Time: 05:57)

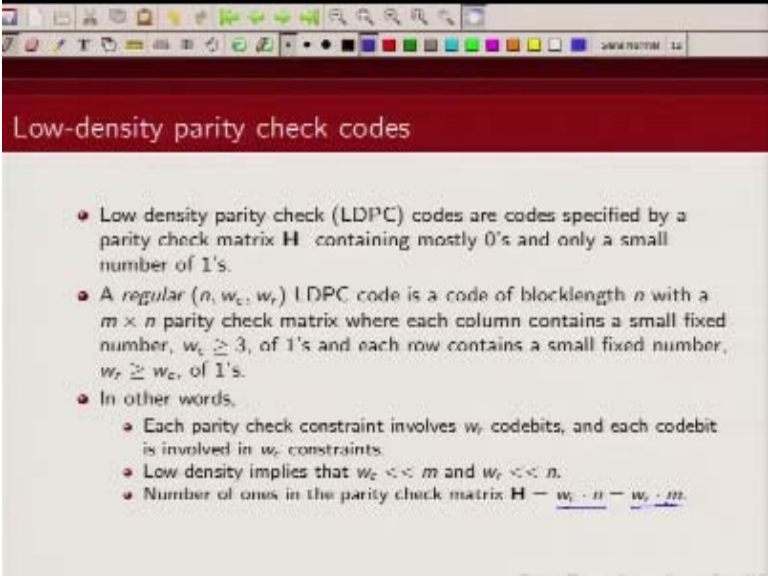


Low-density parity check codes

- Low-density parity-check (LDPC) codes are codes specified by a parity check matrix \mathbf{H} containing mostly 0's and only a small number of 1's.
- A *regular* (n, w_c, w_r) LDPC code is a code of blocklength n with a $m \times n$ parity check matrix where each column contains a small fixed number, $w_c \geq 3$, of 1's and each row contains a small fixed number, $w_r \geq w_c$, of 1's.
- In other words,
 - Each parity check constraint involves w_c codebits, and each codebit is involved in w_r constraints.
 - Low-density implies that $w_c \ll m$ and $w_r \ll n$.

Now typically the number of ones, because it is a low density parity check matrix, so numbers of ones are much less than the dimension of these matrix and w_r is also much less than n .

(Refer Slide Time: 06:13)

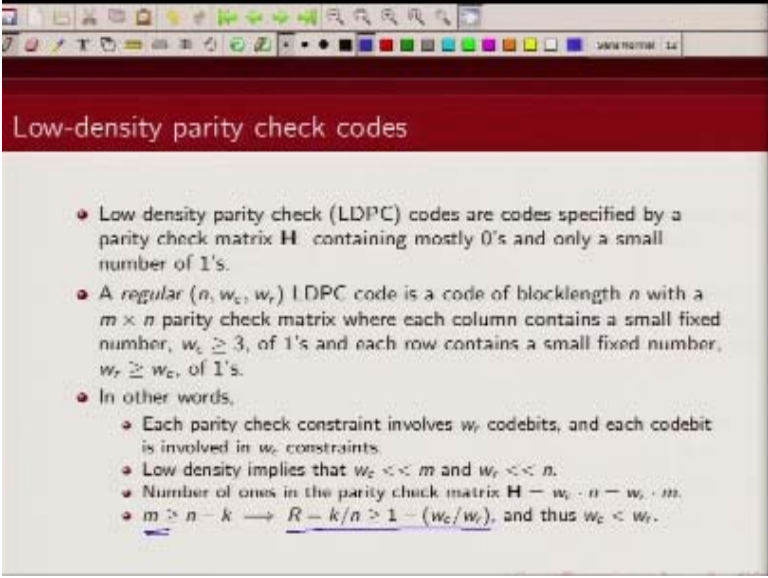


Low-density parity check codes

- Low density parity check (LDPC) codes are codes specified by a parity check matrix \mathbf{H} containing mostly 0's and only a small number of 1's.
- A *regular* (n, w_c, w_r) LDPC code is a code of blocklength n with a $m \times n$ parity check matrix where each column contains a small fixed number, $w_c \geq 3$, of 1's and each row contains a small fixed number, $w_r \geq w_c$, of 1's.
- In other words,
 - Each parity check constraint involves w_r codebits, and each codebit is involved in w_c constraints.
 - Low density implies that $w_c \ll m$ and $w_r \ll n$.
 - Number of ones in the parity check matrix $\mathbf{H} = \underline{w_c \cdot n} = \underline{w_r \cdot m}$.

Number of ones we can count it column wise or n columns and there are w_c ones in each column, that number should be equal to number of rows multiplied by number of ones in each row.

(Refer Slide Time: 06:31)



Low-density parity check codes

- Low density parity check (LDPC) codes are codes specified by a parity check matrix \mathbf{H} containing mostly 0's and only a small number of 1's.
- A *regular* (n, w_c, w_r) LDPC code is a code of blocklength n with a $m \times n$ parity check matrix where each column contains a small fixed number, $w_c \geq 3$, of 1's and each row contains a small fixed number, $w_r \geq w_c$, of 1's.
- In other words,
 - Each parity check constraint involves w_r codebits, and each codebit is involved in w_c constraints.
 - Low density implies that $w_c \ll m$ and $w_r \ll n$.
 - Number of ones in the parity check matrix $\mathbf{H} = w_c \cdot n = w_r \cdot m$.
 - $m \geq n - k \implies R = k/n \geq 1 - (w_c/w_r)$, and thus $w_c < w_r$.

And number of parity check equations is atleast equal to $n-k$ so the rate is atleast $1-w_c/w_r$.
Sometimes we do have some redundant parity check equations in the LDPC parity check matrix.

(Refer Slide Time: 06:55)

Regular low-density parity check code

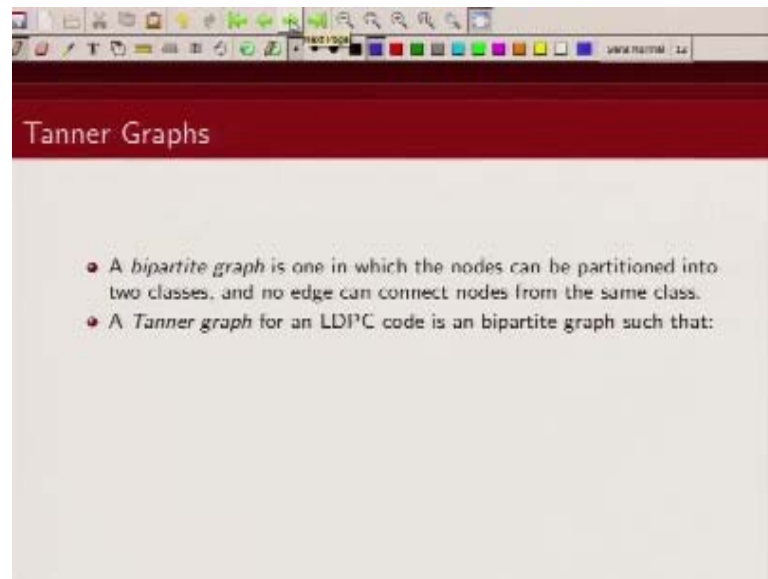
Example of a regular low density code matrix; $n = 20$, $w_c = 3$,
 $w_r = 4$
 $R = 1 - \frac{3}{4} = \frac{1}{4}$

This is one example of a low density parity check code, you can see in this matrix most of the entries are zeros, these are all zeros, these are zeros, these are zeros. You can see most of the entries in this matrix are zero, very few are ones. And you can see that each row, let us look at row number 1, row number 1 has four ones, row number 2 has four ones, you can check any row, you can check let us say this row.

This has 1, 2, 3, 4, there are four, so each row of this low density parity check matrix has four number of ones. So w_r in this case is four, and each column let us take column 1, there is a 1 here, there is a 1 here, and there is a 1 here. So column weight is 3, you can check any column. Look at this column 1 here, 1 here, and 1 here. So column weight is 3 you can take this column there is a 1 here, there is a 1 here, and there is a 1 here, so the column weight is 3.

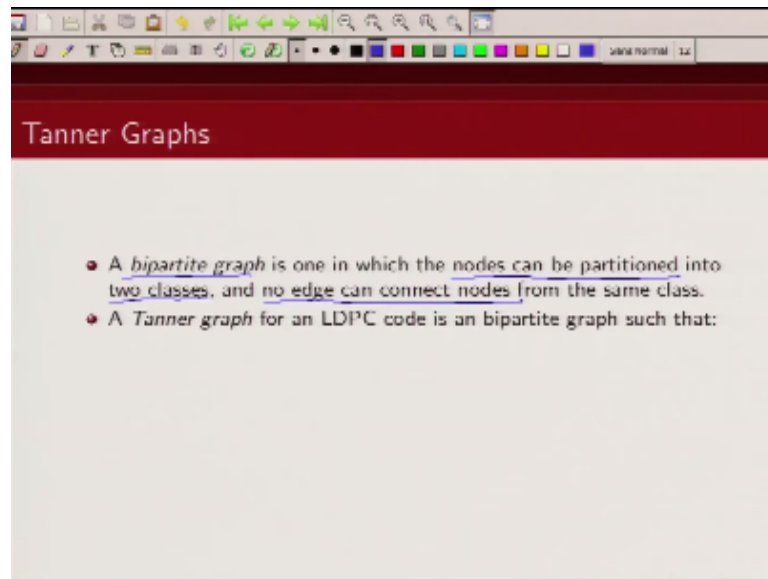
So w_c is 3, n is 20 you can see there is 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, so the block size is 20 and this is full rate, so the rate here is $1 - 3/4$ which is a rate $1/4$ code. So this is an example of a low density parity check code. You can see the fraction of ones is much smaller than number of zeros.

(Refer Slide Time: 09:00)



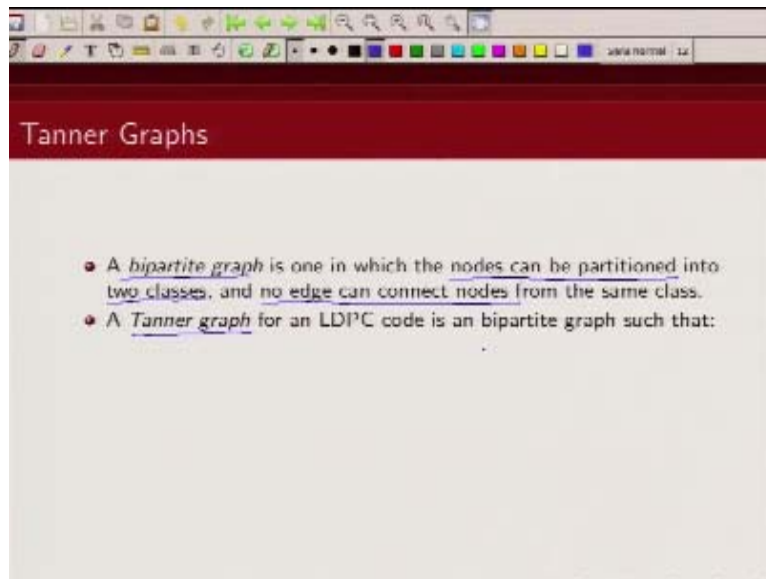
Now we can represent these parity check matrix using a bipartite graph and these – this bipartite graph representation of parity check matrix of a linear block code is known as tanner graph named after Michael Tanner.

(Refer Slide Time: 09:20)



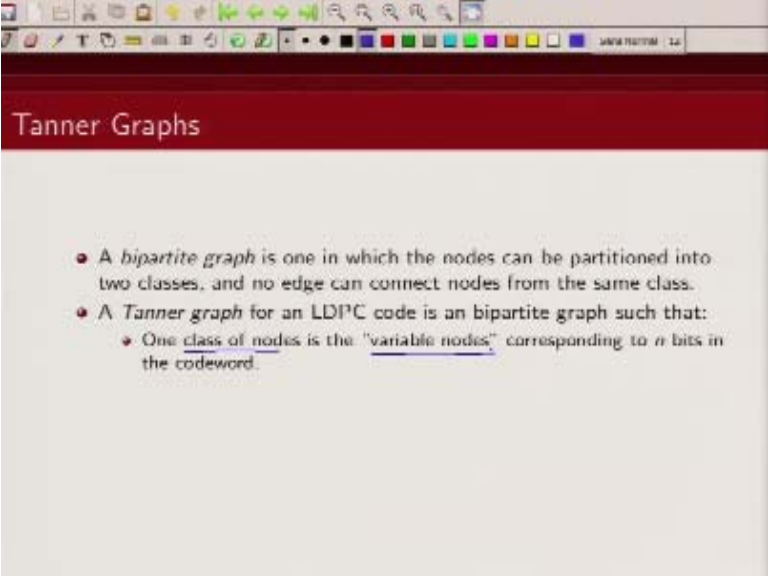
So what is a bipartite graph, in a bipartite graph the nodes can be partitioned into two classes? Now what are those two classes, what is the property that no edge can connect nodes from the same class, so when we partition the nodes of this graph into two classes, there is no connection between nodes within a class. So if you want to reach another node within a class you at least have to have a travel twice okay.

(Refer Slide Time: 10:05)



So a bipartite graph is one where I can separate out the nodes into two classes such that there is no edge connecting nodes in the same class. Now we can draw a bipartite graph for an LDPC code parity check matrix, so a tanner graph and that is basically known as tanner graph.

(Refer Slide Time: 10:30)



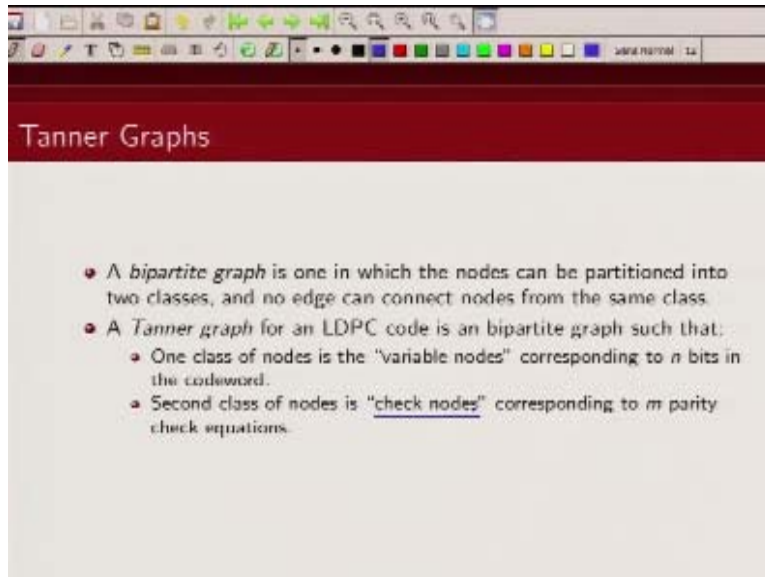
The image shows a presentation slide with a red header bar containing the title "Tanner Graphs". Below the header, the slide content is on a light gray background. It features two bullet points, each preceded by a red circular icon. The first bullet point defines a bipartite graph. The second bullet point defines a Tanner graph for an LDPC code, with a sub-bullet point defining one of its classes as "variable nodes".

Tanner Graphs

- A *bipartite graph* is one in which the nodes can be partitioned into two classes, and no edge can connect nodes from the same class.
- A *Tanner graph* for an LDPC code is an bipartite graph such that:
 - One class of nodes is the "variable nodes," corresponding to n bits in the codeword.

So tanner graph for an LDPC code is a bipartite graph which has the property that there are two sets of class of nodes, one class of nodes which we call variable nodes, they represent the n bits of the code word.

(Refer Slide Time: 10:44)



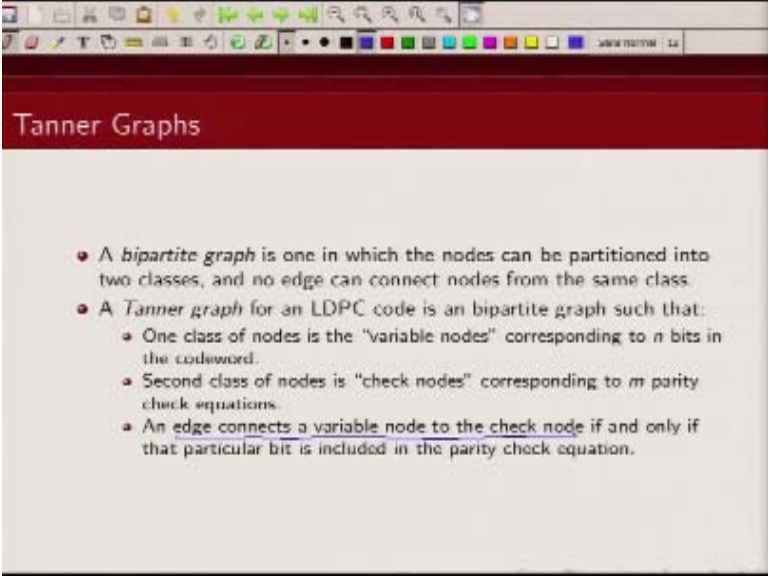
The image is a screenshot of a presentation slide. At the top, there is a dark red header bar with the title "Tanner Graphs" in white text. Below the header, the slide has a light gray background. The content consists of two main bullet points, each preceded by a red diamond symbol. The first bullet point defines a bipartite graph. The second bullet point defines a Tanner graph for an LDPC code, which is a bipartite graph with two specific classes of nodes: "variable nodes" and "check nodes". The text "check nodes" is underlined in the original image. The slide is framed by a standard presentation software toolbar at the top.

Tanner Graphs

- A *bipartite graph* is one in which the nodes can be partitioned into two classes, and no edge can connect nodes from the same class.
- A *Tanner graph* for an LDPC code is an bipartite graph such that:
 - One class of nodes is the "variable nodes" corresponding to n bits in the codeword.
 - Second class of nodes is "check nodes" corresponding to m parity check equations.

And the other class is what is known as check nodes, they represent these m parity check equations.

(Refer Slide Time: 10:56)

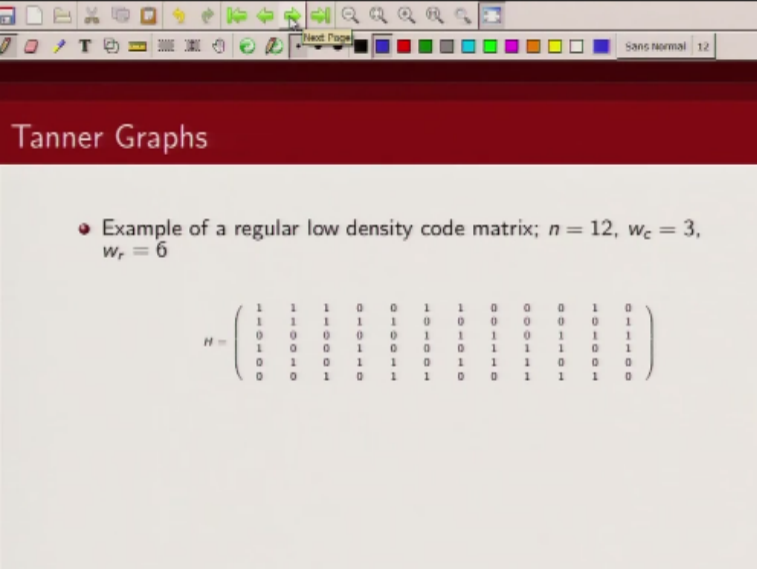


Tanner Graphs

- A *bipartite graph* is one in which the nodes can be partitioned into two classes, and no edge can connect nodes from the same class.
- A *Tanner graph* for an LDPC code is a bipartite graph such that:
 - One class of nodes is the "variable nodes" corresponding to n bits in the codeword.
 - Second class of nodes is "check nodes" corresponding to m parity check equations.
 - An edge connects a variable node to the check node if and only if that particular bit is included in the parity check equation.

And how do we connect an edge, an edge connects a variable node to the check node if and only if that particular bit participates in that parity check equation.

(Refer Slide Time: 11:14)



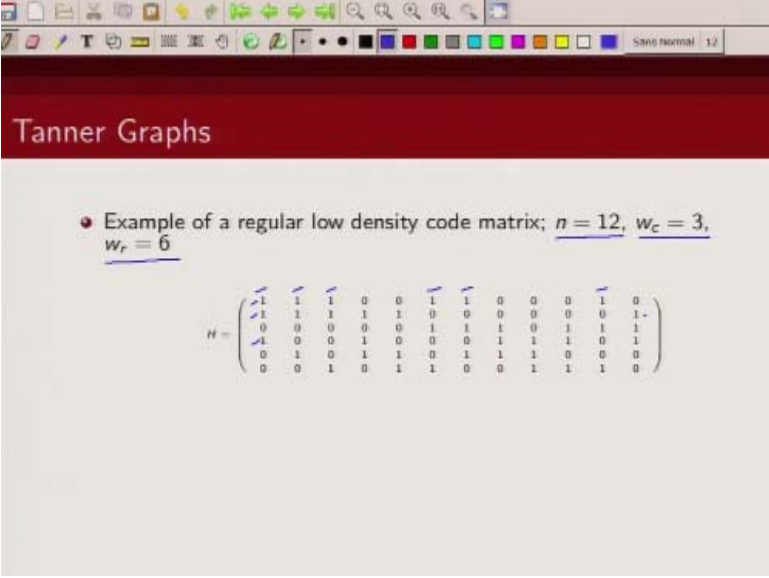
Tanner Graphs

- Example of a regular low density code matrix; $n = 12$, $w_c = 3$, $w_r = 6$

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

So let us take an example to illustrate how we can draw the tanner graph of an LDPC code, so this is an LDPC code block

(Refer Slide Time: 11:25)



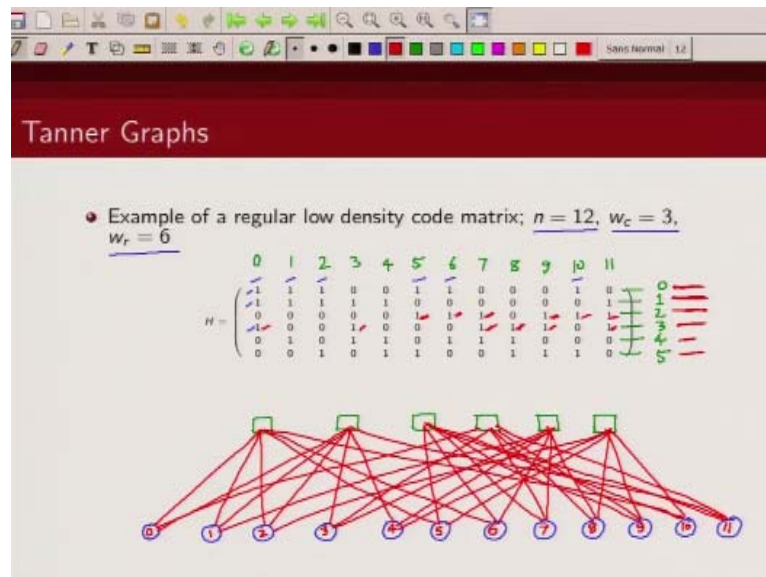
Tanner Graphs

- Example of a regular low density code matrix; $n = 12$, $w_c = 3$, $w_r = 6$

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

n is 12 number of ones in each column you can see 1, 2, 3, that w_c is 3 and number of ones in each row is 6 you can check 1,2,3,4,5,6, each row has 6 ones each column has 3 ones, now how do we draw the tanner graph of this so I as I said there are two class of nodes, one class of nodes for the variable nodes

(Refer Slide Time: 11:58)



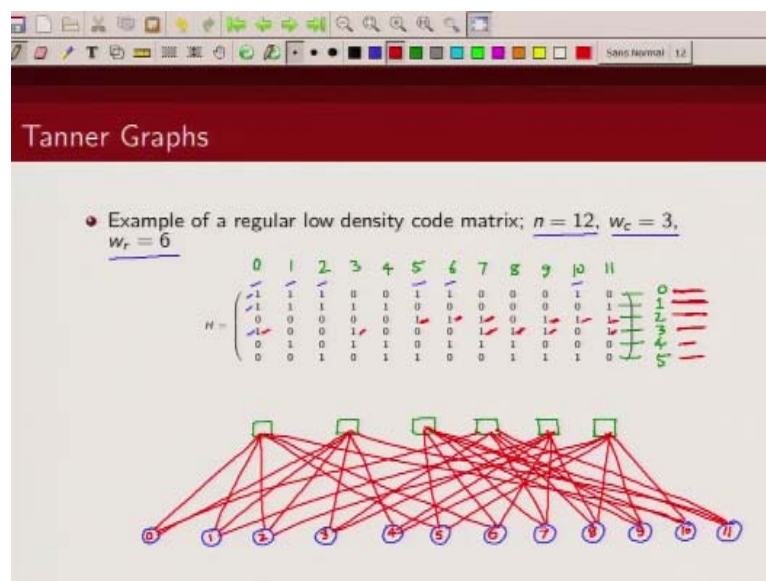
And how many variable nodes we have we have 12 variable nodes, so let us just draw 12 variables node 1, this 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, let us just label them, let us just label them as 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, and 11 and then you have how many parity check equations 1, 2, 3, 4, 5, 6, so we will have the next set of nodes will be for parity check equations and they are 6 of them 4, 5, 6, okay let us just label these bits that will be easier for a, so let us this 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, okay now let us similarly label these parity check equations this let us say a zero parity check equation 1, 2, 3, 4, 5, so let us look at this one. This 0 at parity check equation. Now which are the bits that are participating in the parity check constrain.

Bit number 0 so we will draw and edge from bit number 0 to this parity check constrain, bit number 1 that is this, this bit number 2 that is this, bit number 5 that is this, bit number 6 that is this, bit number 10 okay. So this is my first parity check constrain, let us look at now this one, which are the bits participating bit number 0, bit number 0, bit number 1, bit number 1, bit number 2, so there is an edge from bit number 2 to this parity check constrain, bit number three that is this, bit number 4 that is this, and then you have bit number 11.

So you have this 1 okay now look at this parity check constrain, now which are the bits that are participating, this is 5 so that is here, 6 that is this one, 7 that is this one, 9 that is this one, then 10 that is this one, and then 11 that is this one, okay. Similarly for this parity check constrain bit number 0 is participating so you have edge from here to here, then bit number 3 is participating so there is an edge from here to here, bit number 7 is participating so there is an edge from here to here, bit number 8 is participating so there is an edge from here to here, bit number 9 is participating so there is an edge from here to here, and bit number 11 is participating so there is an edge from here to here.

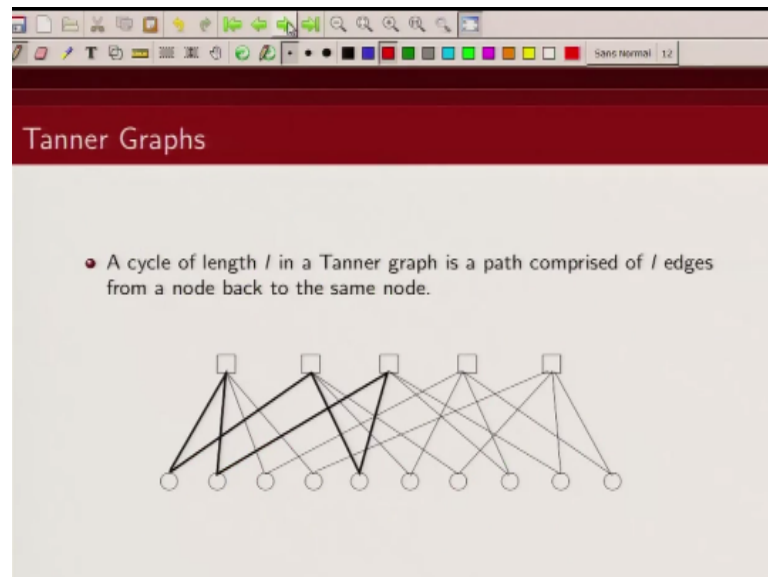
And similarly we can do for this I will just do it this is 1, 3, 4, 6, 7, 8, that is it and this finally this parity check constrain bit number 2, bit number 4, bit number 5, bit number 8, bit number 9, bit number 10 so this is the tanner graph representation of this low density parity check code.

(Refer Slide Time: 16:45)



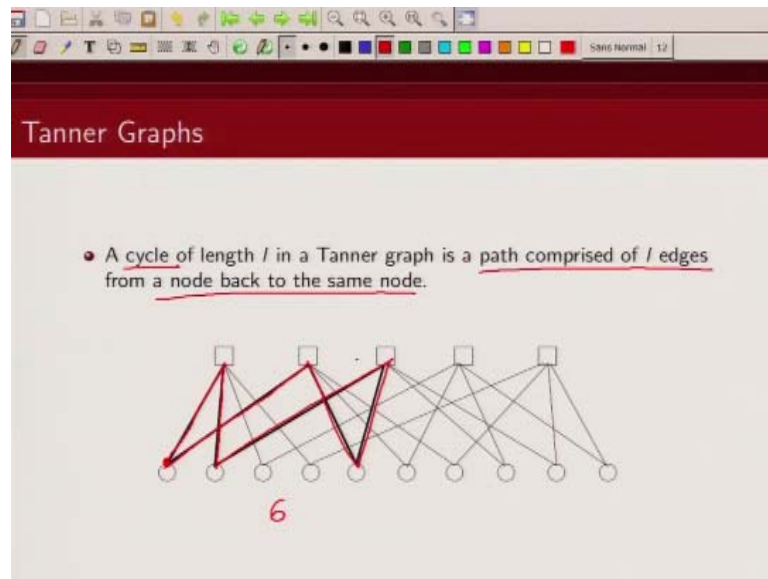
Okay

(Refer Slide Time: 16:48)



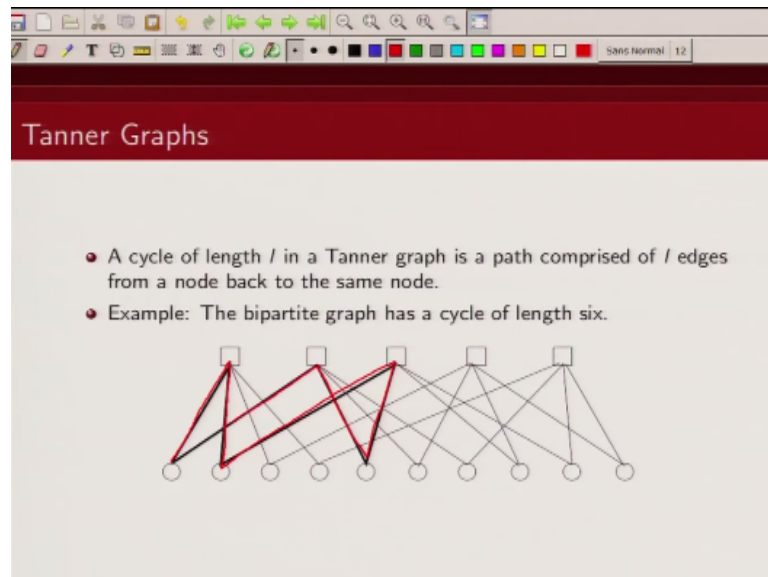
Now let us define what do we mean by cycle in this tanner graph so cycle is defined as

(Refer Slide Time: 16:58)



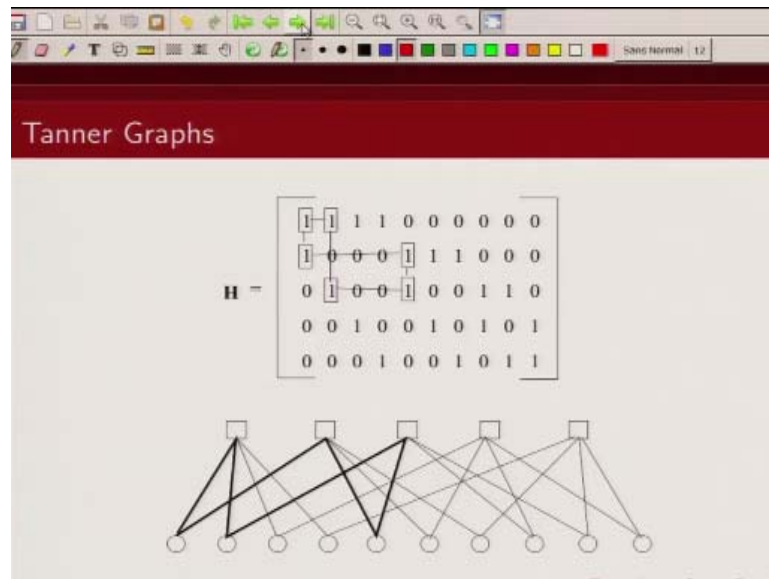
A path consisting of length l which will start from a node and come back to the same node, so what is a cycle so it is a path cycle of length l is starts from particular node and comes back to same node so let us look at this as I said this node, so if you start from this node this edge 1, 2, 3, 4, 5, 6, so this is a cycle of length 6. Now note that it is a bipartite graph so it will only have even length cycles because there is no connection between nodes of the same class.

(Refer Slide Time: 17:43)



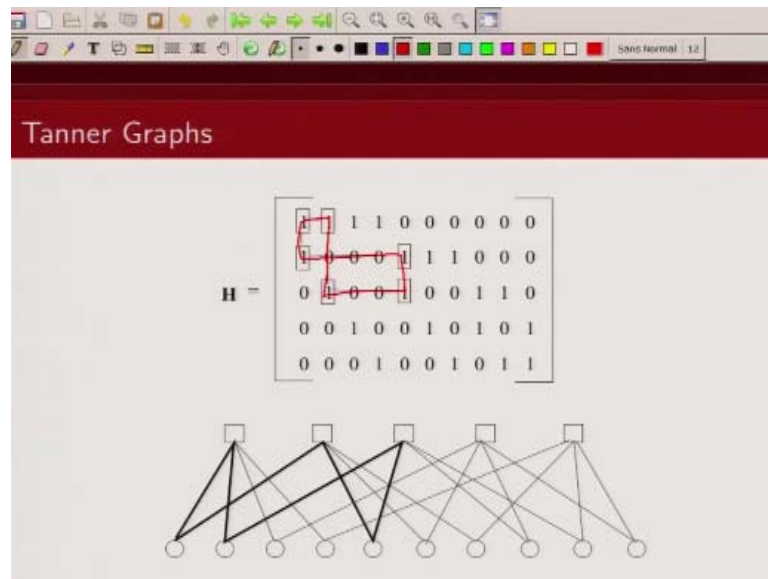
As I said in this particular example this has cycle 6. 1, 2, 3, 4, 5, 6, and

(Refer Slide Time: 17:43)



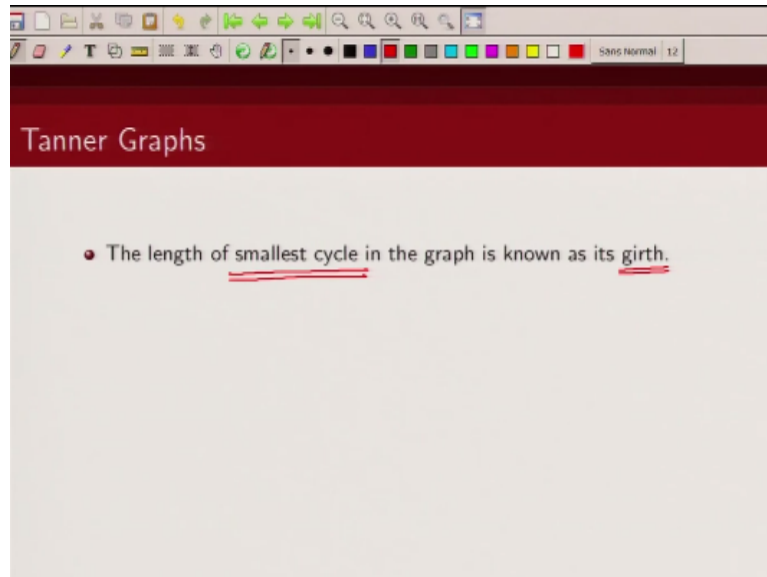
This can be viewed from the parity check matrix also

(Refer Slide Time: 17:56)



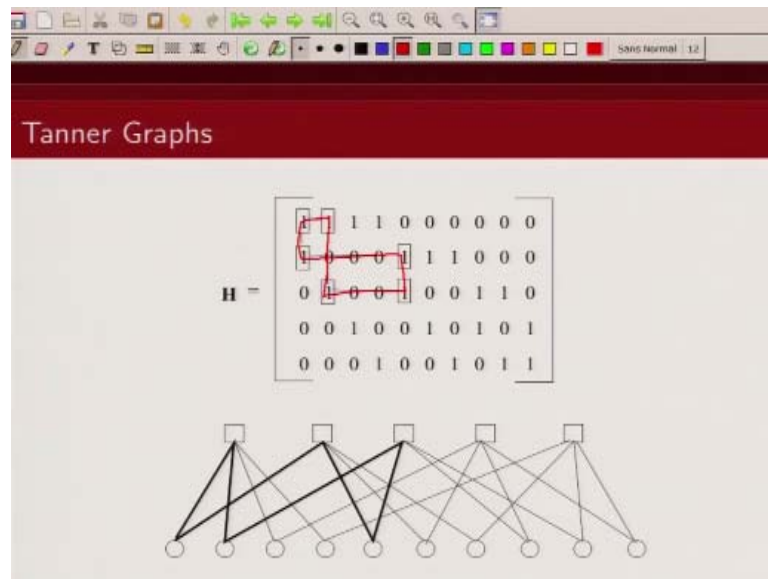
So in 1, 2, 3, 4, 5, 6, okay

(Refer Slide Time: 18:05)



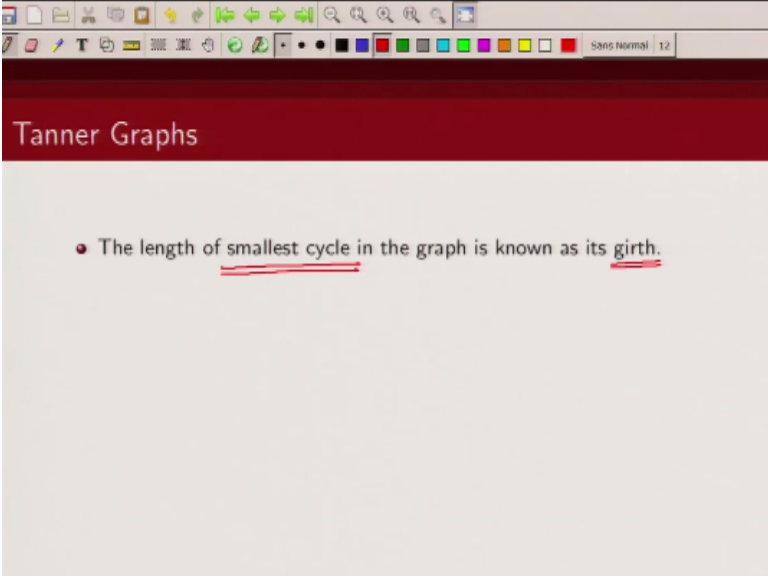
Now we will define what is known as girth. Girth is the length of the smallest cycle in this graph so girth is defined as the length of the smallest cycle in this graph

(Refer Slide Time: 18:25)



In this particular example the smallest cycle is 6 you can see there is no cycle of length 2 or 4

(Refer Slide Time: 18:33)



The image is a screenshot of a presentation slide. At the top, there is a dark red header bar with the text "Tanner Graphs" in white. Below the header, the slide content is on a light beige background. A single bullet point is centered on the slide, stating: "• The length of smallest cycle in the graph is known as its girth." The words "smallest cycle" and "girth" are underlined in red. Above the slide content, a standard presentation software toolbar is visible, containing various icons for navigation and editing. To the right of the toolbar, the text "Sans Normal 12" indicates the font style and size.

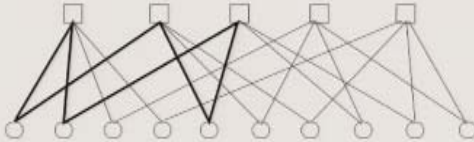
Tanner Graphs

- The length of smallest cycle in the graph is known as its girth.

(Refer Slide Time: 18:37)

Tanner Graphs

- The length of smallest cycle in the graph is known as its girth.
- When decoding LDPC codes using sum-product algorithm, the number of independent iterations of the algorithm is proportional to the girth of its associated Tanner graph.

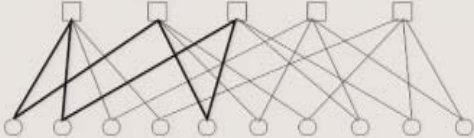


Now when we are decoding LDPC codes we would like the girth to be very large because number of independent detritions that we can get is proportional to the girth of the corresponding tanner graph

(Refer Slide Time: 18:52)

Tanner Graphs

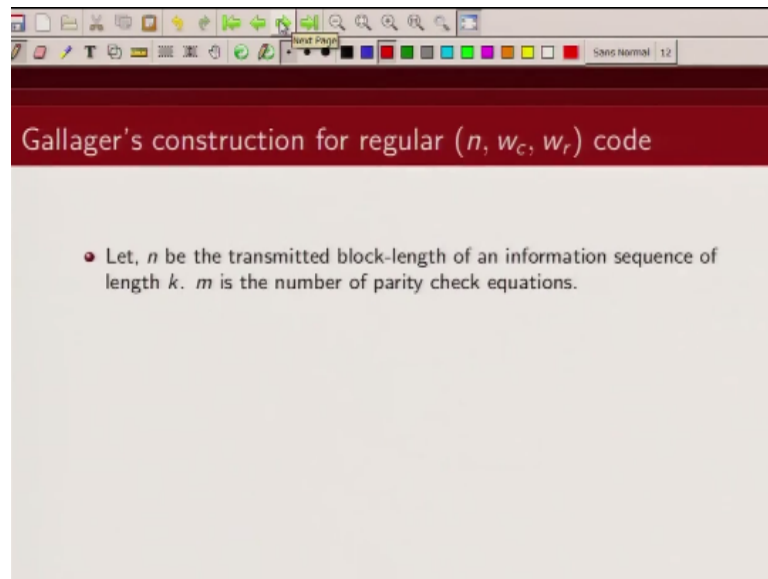
- The length of smallest cycle in the graph is known as its girth.
- When decoding LDPC codes using sum-product algorithm, the number of independent iterations of the algorithm is proportional to the girth of its associated Tanner graph.



- The girth of this Tanner graph is six.

Of the LDPC code

(Refer Slide Time: 18:55)

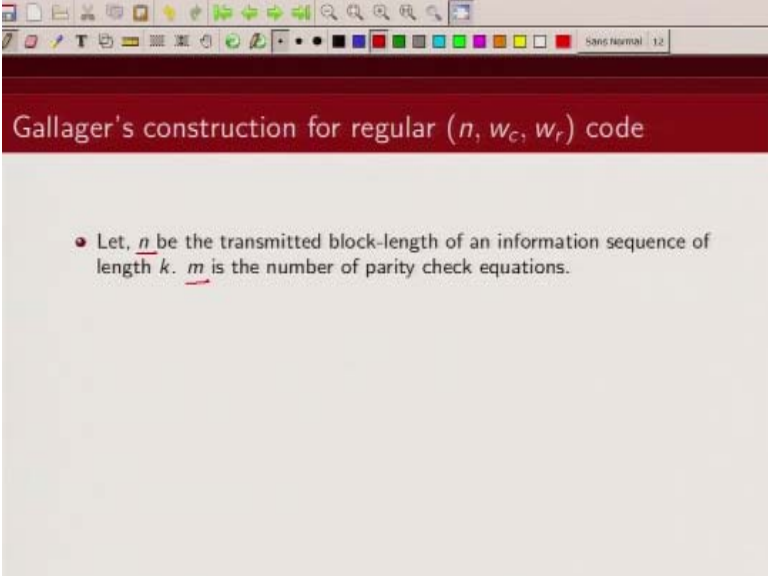


Gallager's construction for regular (n, w_c, w_r) code

- Let, n be the transmitted block-length of an information sequence of length k . m is the number of parity check equations.

Now that we have defined what is a regular LDPC code let us talk about how we can construct these LDPC codes, so we will first start with random construction of LDPC codes given by Gallager.

(Refer Slide Time: 19:12)

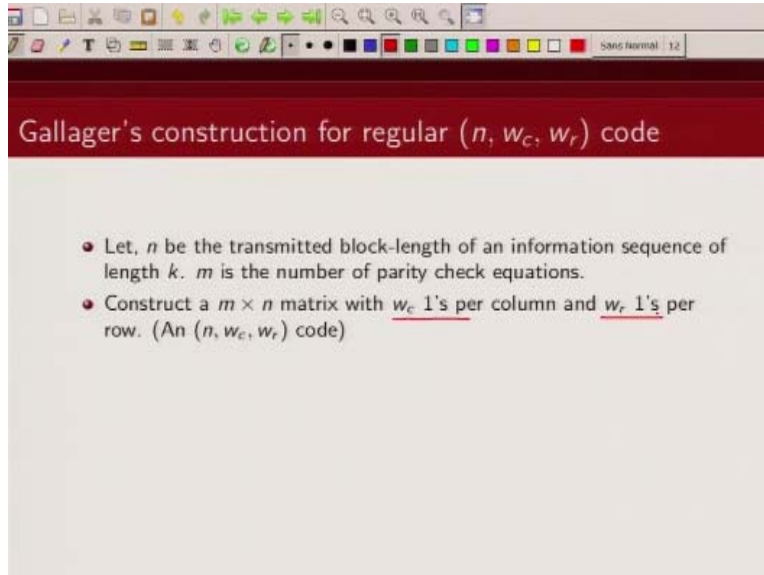


Gallager's construction for regular (n, w_c, w_r) code

- Let, n be the transmitted block-length of an information sequence of length k . m is the number of parity check equations.

So if n is the block length and m is the number of parity check equations

(Refer Slide Time: 19:20)



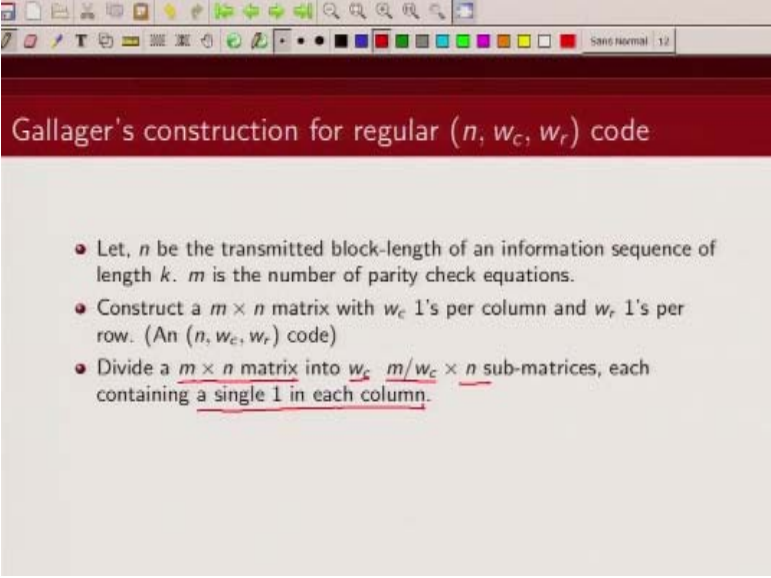
The image shows a presentation slide with a red header and a light gray body. The header contains the title "Gallager's construction for regular (n, w_c, w_r) code". The body contains two bullet points. The first bullet point states: "Let, n be the transmitted block-length of an information sequence of length k . m is the number of parity check equations." The second bullet point states: "Construct a $m \times n$ matrix with w_c 1's per column and w_r 1's per row. (An (n, w_c, w_r) code)". The text in the second bullet point has red underlines under w_c and w_r .

Gallager's construction for regular (n, w_c, w_r) code

- Let, n be the transmitted block-length of an information sequence of length k . m is the number of parity check equations.
- Construct a $m \times n$ matrix with w_c 1's per column and w_r 1's per row. (An (n, w_c, w_r) code)

And if you are asked to design a regular LDPC code with W_c 1's per column and W_r 1's per row you can follow this procedure.

(Refer Slide Time: 19:33)



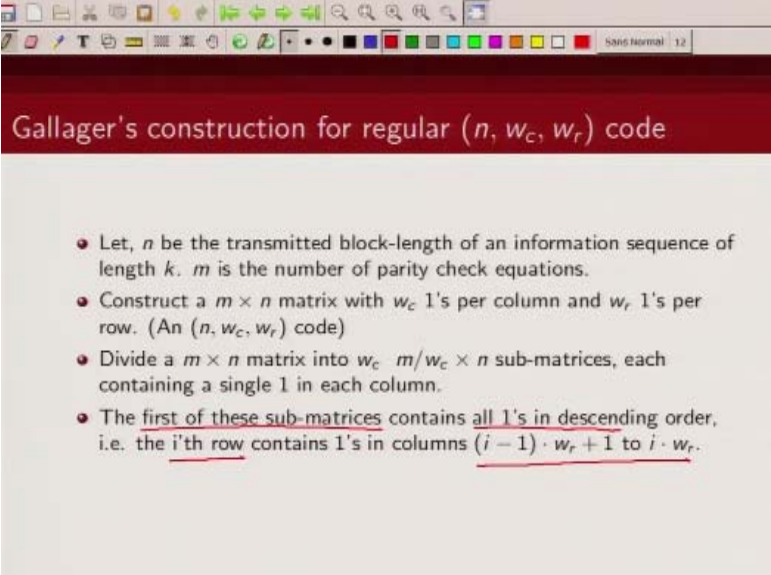
The image is a screenshot of a presentation slide. At the top, there is a dark red header bar with the title "Gallager's construction for regular (n, w_c, w_r) code" in white text. Below the header, the slide has a light gray background. It contains three bullet points, each preceded by a red circular icon. The text in the third bullet point is underlined. The slide is displayed within a window that has a standard toolbar at the top with various icons for editing and presentation control. The status bar at the bottom of the window shows "Slide Normal" and the number "12".

Gallager's construction for regular (n, w_c, w_r) code

- Let, n be the transmitted block-length of an information sequence of length k . m is the number of parity check equations.
- Construct a $m \times n$ matrix with w_c 1's per column and w_r 1's per row. (An (n, w_c, w_r) code)
- Divide a $m \times n$ matrix into w_c $\frac{m}{w_c} \times n$ sub-matrices, each containing a single 1 in each column.

So divide $m \times n$ matrix which is your parity check matrix, you divide them into w_c $\frac{m}{w_c}$ into n sub matrices, such that in each of these sub matrices your column will only have a single 1

(Refer Slide Time: 19:59)

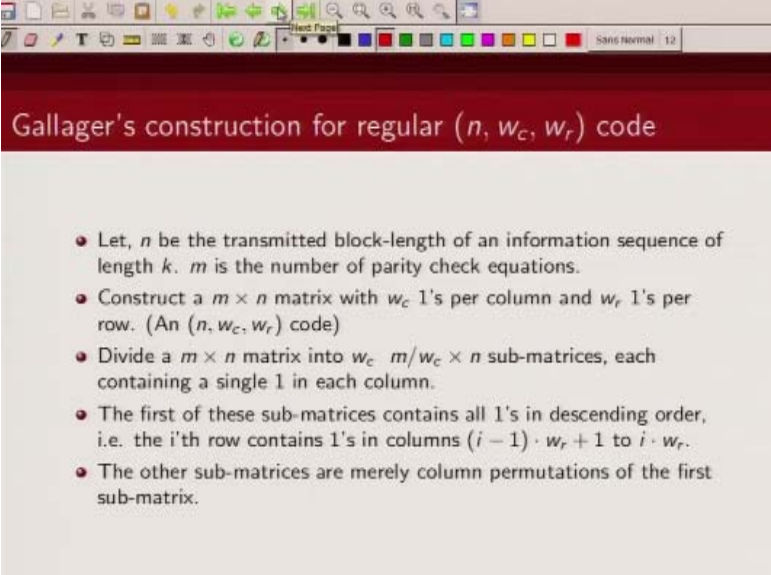


Gallager's construction for regular (n, w_c, w_r) code

- Let, n be the transmitted block-length of an information sequence of length k . m is the number of parity check equations.
- Construct a $m \times n$ matrix with w_c 1's per column and w_r 1's per row. (An (n, w_c, w_r) code)
- Divide a $m \times n$ matrix into w_c $m/w_c \times n$ sub-matrices, each containing a single 1 in each column.
- The first of these sub-matrices contains all 1's in descending order, i.e. the i^{th} row contains 1's in columns $(i-1) \cdot w_r + 1$ to $i \cdot w_r$.

Next you start writing so in each of this sub matrices what you do is you write 1's in the descending order, so the i^{th} row will have 1's from location $(i-1) \cdot w_r + 1$ to $i \cdot w_r$, so what you do is so I will just illustrate

(Refer Slide Time: 20:21)



Gallager's construction for regular (n, w_c, w_r) code

- Let, n be the transmitted block-length of an information sequence of length k . m is the number of parity check equations.
- Construct a $m \times n$ matrix with w_c 1's per column and w_r 1's per row. (An (n, w_c, w_r) code)
- Divide a $m \times n$ matrix into w_c $m/w_c \times n$ sub-matrices, each containing a single 1 in each column.
- The first of these sub-matrices contains all 1's in descending order, i.e. the i 'th row contains 1's in columns $(i - 1) \cdot w_r + 1$ to $i \cdot w_r$.
- The other sub-matrices are merely column permutations of the first sub-matrix.

Basically

(Refer Slide Time: 20:23)

Gallager's construction for regular (n, w_c, w_r) code

1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1
0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0
1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0
0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1

Example of a regular low density code matrix; $n = 20$, $w_c = 3$,
 $w_r = 4$

So what you would do is first you have an

(Refer Slide Time: 20:27)

Gallager's construction for regular (n, w_c, w_r) code

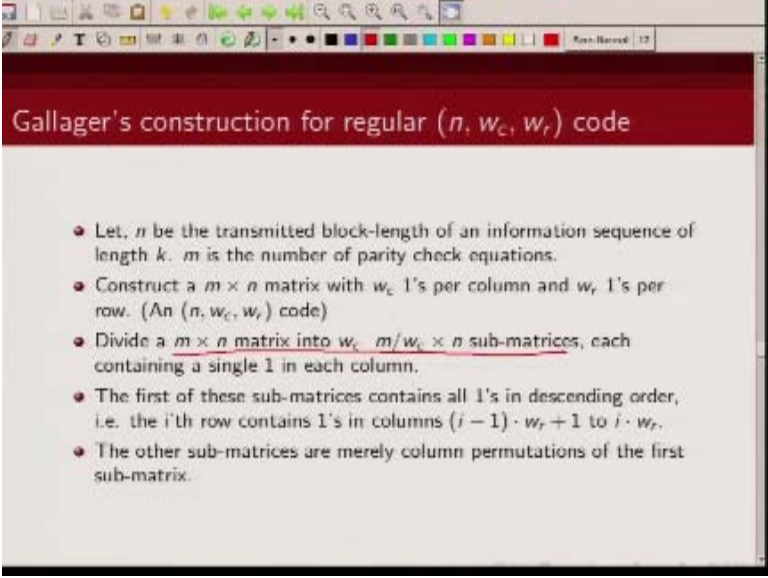
1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1
0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1
1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0
0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0
0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1
0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0

Example of a regular low density code matrix; $n = 20$, $w_c = 3$, $w_r = 4$

$m \times n$ $R = \frac{1}{4}$ 15×20 $3 \times 5 \times 20$

$m \times n$ matrix now this was a rate half code remember rate $\frac{1}{4}$ code, remember w_c is 3 w_r is 4 so this is 15×20 matrix, now what you do is you divide this 15×20 matrix into $3 \times 5 \times 20$ matrix sub matrices, right that was the first step

(Refer Slide Time: 20:58)



Gallager's construction for regular (n, w_c, w_r) code

- Let, n be the transmitted block-length of an information sequence of length k . m is the number of parity check equations.
- Construct a $m \times n$ matrix with w_c 1's per column and w_r 1's per row. (An (n, w_c, w_r) code)
- Divide a $m \times n$ matrix into w_c $m/w_c \times n$ sub-matrices, each containing a single 1 in each column.
- The first of these sub-matrices contains all 1's in descending order, i.e. the i 'th row contains 1's in columns $(i-1) \cdot w_r + 1$ to $i \cdot w_r$.
- The other sub-matrices are merely column permutations of the first sub-matrix.

Divide an $m \times n$ matrix into w_c $m/w_c \times n$ sub-matrices.

(Refer Slide Time: 21:05)

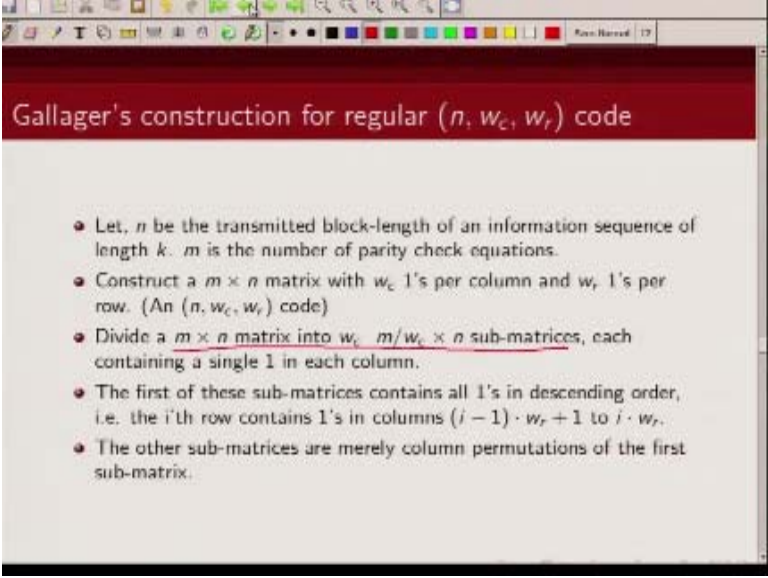
Gallager's construction for regular (n, w_c, w_r) code

Example of a regular low density code matrix; $n = 20$, $w_c = 3$.

$m \times n$ $R \approx \frac{1}{4}$ $\underline{w_r = 4}$ 15×20 5×20

So once you divide them so these so this is 15 x 20 sub-matrices, this is another 5 x 20 sub-matrix, this is another 5 x 20 sub matrix, and each of this sub matrix should have only ones and what you do you start writing ones in the descending order, so you write start writing one here from location 0 now wr in this case is 4 so in the 0 through you start writing from column 0 to 3 next row you start writing from 4 to 7 like that you start writing ones so that is what.

(Refer Slide Time: 21:51)

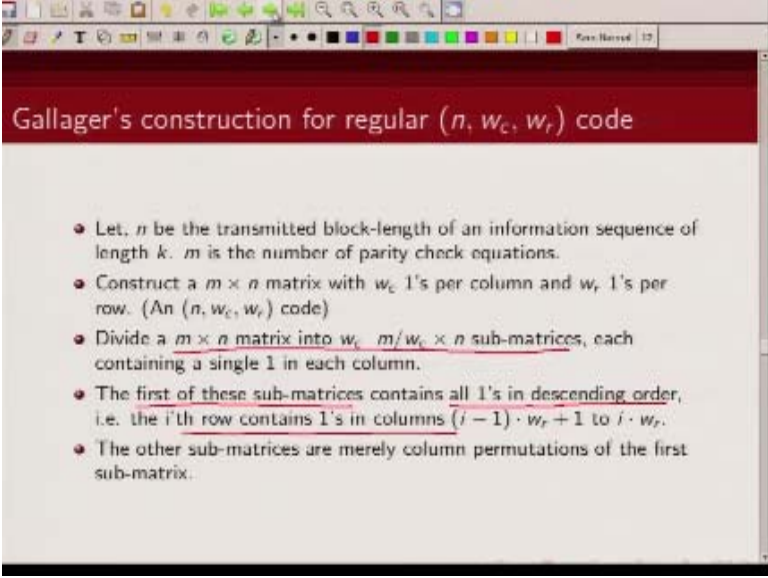


Gallager's construction for regular (n, w_c, w_r) code

- Let, n be the transmitted block-length of an information sequence of length k . m is the number of parity check equations.
- Construct a $m \times n$ matrix with w_c 1's per column and w_r 1's per row. (An (n, w_c, w_r) code)
- Divide a $m \times n$ matrix into w_r $m/w_r \times n$ sub-matrices, each containing a single 1 in each column.
- The first of these sub-matrices contains all 1's in descending order, i.e. the i 'th row contains 1's in columns $(i-1) \cdot w_r + 1$ to $i \cdot w_r$.
- The other sub-matrices are merely column permutations of the first sub-matrix.

I have written here that first of these sub-matrices contains all ones in the descending order such that i^{th} row contains ones from location $(i-1) \times w_r + 1$ to $i \times w_r$ so once you have constructed.

(Refer Slide Time: 22:15)



Gallager's construction for regular (n, w_c, w_r) code

- Let, n be the transmitted block-length of an information sequence of length k . m is the number of parity check equations.
- Construct a $m \times n$ matrix with w_c 1's per column and w_r 1's per row. (An (n, w_c, w_r) code)
- Divide a $m \times n$ matrix into w_c $m/w_c \times n$ sub-matrices, each containing a single 1 in each column.
- The first of these sub-matrices contains all 1's in descending order, i.e. the i 'th row contains 1's in columns $(i-1) \cdot w_r + 1$ to $i \cdot w_r$.
- The other sub-matrices are merely column permutations of the first sub-matrix.

(Refer Slide Time: 22:15)

Gallager's construction for regular (n, w_c, w_r) code

Example of a regular low density code matrix; $n = 20, w_c = 3,$
 $w_r = 4$
 $R = \frac{1}{4}$
 15×20
 $3 [5 \times 20] \leftarrow$

This sub-matrix this 5 x 20 sub matrix by putting one slide this in descending like this, once you have constructed this rest all are zeros, now note that each of the columns here have 1 ones.

(Refer Slide Time: 22:36)

Gallager's construction for regular (n, w_c, w_r) code

Example of a regular low density code matrix; $n = 20$, $w_c = 3$,

$m \times n$ $R \approx \frac{1}{4}$ $\underline{w_r = 4}$ 15×20
 8×20

And what about you can check any column of these sub-matrix it has only 11 and note what is the row weight, now each of the row here has w_r which is just 4 number of ones, you can check this is 4 ones, you can check here this has 4ones here, these are the 4 ones here, you can check this, this is 4 ones, so what you have created is you have created a sub-matrix which has column weight 1 and row weight w_r , now next.

(Refer Slide Time: 23:14)

Gallager's construction for regular (n, w_c, w_r) code

Example of a regular low density code matrix; $n = 20$, $w_c = 3$,

$m \times n$ $R = \frac{1}{4}$ $w_r = 4$ 15×20

8 5 x 20 ←

To get w_c column weight what do I need, I need to design a similar sub matrix here which will have 1, 1 in each of the column and w_r ones in each row. Now how do I get this sub matrix and this sub matrix, so what I can do is I can do column permutation so for example.

(Refer Slide Time: 23:45)

Gallager's construction for regular (n, w_c, w_r) code

Example of a regular low density code matrix; $n = 20$, $w_c = 3$, $w_r = 4$

$m \times n$ $R = \frac{1}{4}$ $w_c = 4$ 15×20
 $3 [5 \times 20] \leftarrow$

I can move this column here I can move this column here I can do column permutation, now when I do column permutation I do not change the weight of the column it is still 1. And I do not change the weight of the rows it is still 4.

(Refer Slide Time: 24:04)

Gallager's construction for regular (n, w_c, w_r) code

Example of a regular low density code matrix; $n = 20$, $w_c = 3$, $w_r = 4$.

Handwritten notes:

- $m \times n$
- $R = \frac{1}{4}$
- $w_r = 4$
- 15×20
- 5×20

So by doing call up and mutation I will get this sub matrix which will again have 11 in each column and 4 ones in each row.

(Refer Slide Time: 24:19)

Gallager's construction for regular (n, w_c, w_r) code

Example of a regular low density code matrix; $n = 20$, $w_c = 3$, $w_r = 4$

max $R \approx \frac{1}{4}$ $\underline{\underline{w_r = 4}}$ 15×20 5×20

The same thing I will do for this sub matrix, I will again do column permutation of this matrix and I will get this sub matrix, so the subsequent sub matrices are obtained by doing column permutation and by doing column permutation I am not disturbing the way the row weight or the column weight, the column weight of this sub matrix is still one, the row weight is still 4, now once I do this then I am able to design these 3, 5 x 20 sub matrices, each of them have column weight one.

(Refer Slide Time: 24:55)

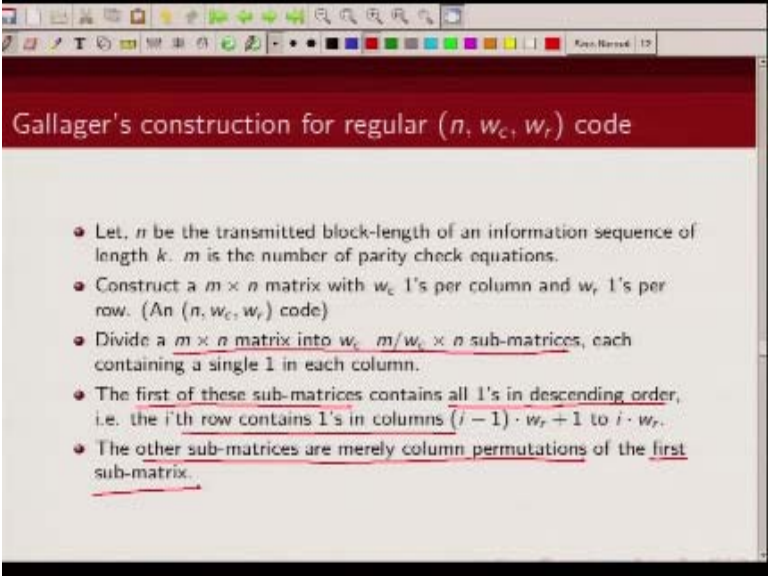
Gallager's construction for regular (n, w_c, w_r) code

Example of a regular low density code matrix; $n = 20$, $w_c = 3$, $w_r = 4$.

$m \times n$ $R = \frac{1}{4}$ $\underline{w_c = 4}$ 15×20 $8 \times 5 \times 20 \leftarrow$

So overall column rate will be 3 and overall row weight is still 4 so I am able to design a low density parity check matrix which has 3 ones in each column and 4 ones in each row.

(Refer Slide Time: 25:15)

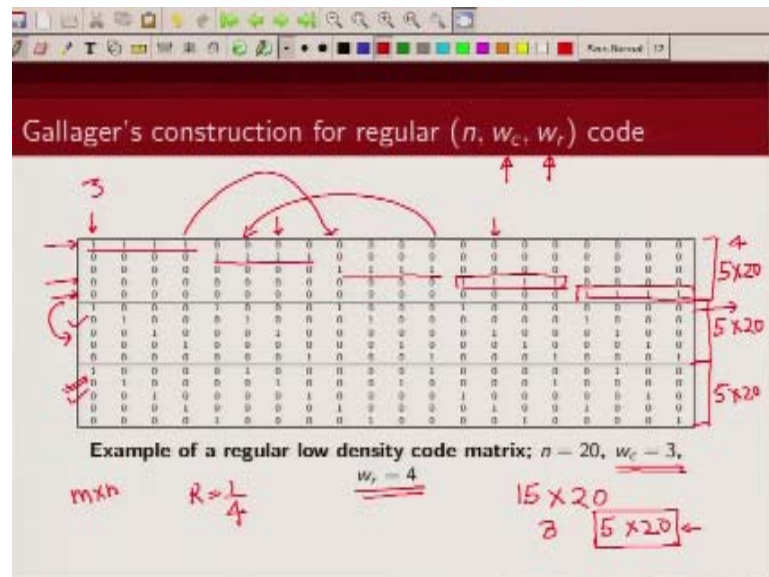


Gallager's construction for regular (n, w_c, w_r) code

- Let, n be the transmitted block-length of an information sequence of length k . m is the number of parity check equations.
- Construct a $m \times n$ matrix with w_c 1's per column and w_r 1's per row. (An (n, w_c, w_r) code)
- Divide a $m \times n$ matrix into $w_c \cdot m/w_c \times n$ sub-matrices, each containing a single 1 in each column.
- The first of these sub-matrices contains all 1's in descending order, i.e. the i 'th row contains 1's in columns $(i-1) \cdot w_r + 1$ to $i \cdot w_r$.
- The other sub-matrices are merely column permutations of the first sub-matrix.

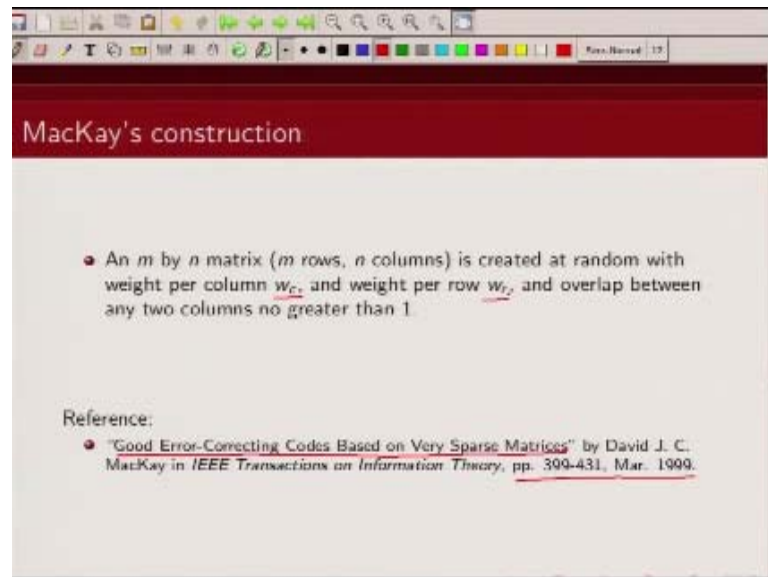
So that is what I said that other sub matrices are merely column permutation of the first sub matrix so this is.

(Refer Slide Time: 25:26)



Gallager's construction of a regular LDPC code which has w_c ones in each column and w_r ones in each row.

(Refer Slide Time: 25:41)



MacKay's construction

- An m by n matrix (m rows, n columns) is created at random with weight per column w_c and weight per row w_r and overlap between any two columns no greater than 1.

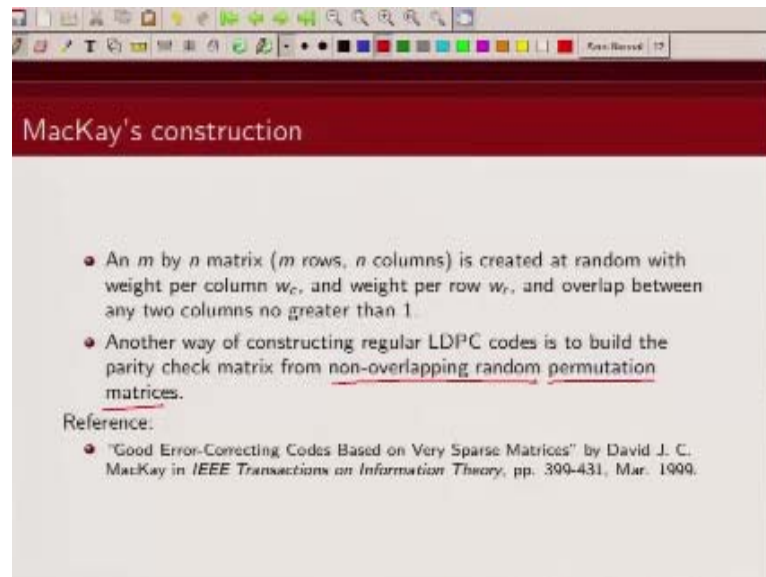
Reference:

- "Good Error-Correcting Codes Based on Very Sparse Matrices" by David J. C. MacKay in *IEEE Transactions on Information Theory*, pp. 399-431, Mar. 1999.

Now we will talk about simple constructions based on permutation matrix and these were given by MacKay we can read this paper Good Error-correcting Codes on very sparse matrices by David MacKay which appeared in IEEE Transactions on Information Theory in May 1999, so one way of designing an $m \times n$ matrix which has w_c column weight and w_r row weight is you can randomly put ones ensuring these criteria is satisfied and also you want to ensure that overlap between 2 rows.

Of this parity check matrix is not more than one, otherwise you will have cycles.

(Refer Slide Time: 26:29)



MacKay's construction

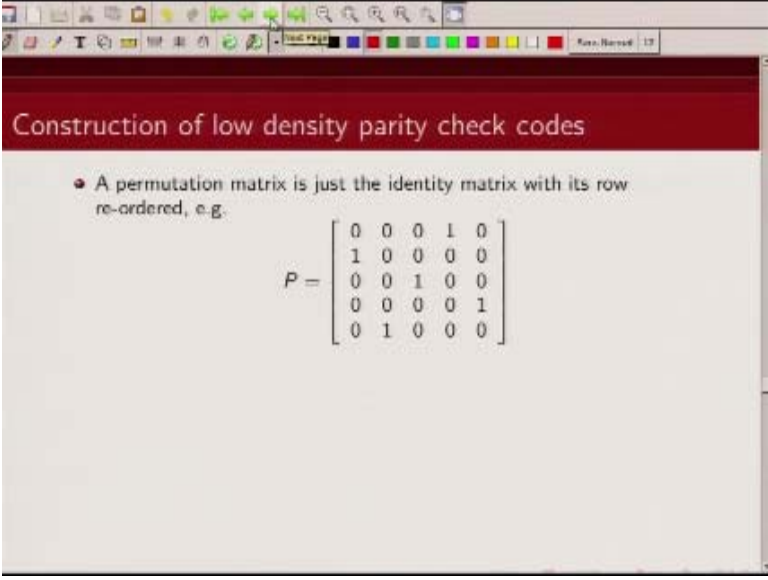
- An m by n matrix (m rows, n columns) is created at random with weight per column w_c , and weight per row w_r , and overlap between any two columns no greater than 1.
- Another way of constructing regular LDPC codes is to build the parity check matrix from non-overlapping random permutation matrices.

Reference:

- "Good Error-Correcting Codes Based on Very Sparse Matrices" by David J. C. MacKay in *IEEE Transactions on Information Theory*, pp. 399-431, Mar. 1999.

For an in your LDPC code, now next what we are going to talk about is how we can use permutation matrix to design our LDPC codes, so that is what I am going to show in the next few slides.

(Refer Slide Time: 26:52)



Construction of low density parity check codes

- A permutation matrix is just the identity matrix with its row re-ordered, e.g.

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

So let us define what a permutation matrix, so a permutation matrix is an identity matrix which is row reordered.

(Refer Slide Time: 27:01)

Construction of low density parity check codes

- A permutation matrix is just the identity matrix with its row re-ordered, e.g.

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

Red annotations on the matrix: two downward arrows point to the first and second rows; four leftward arrows point to the first, second, third, and fourth columns; and a red '5x5' label is placed to the right of the matrix.

So this is an example of a 5 x 5 permutation matrix, you can see each row has 11 and each column so each row has only 11 and each column has a single 1 and this is just an identity matrix with row reordered.

(Refer Slide Time: 27:23)

Construction of low density parity check codes

- A permutation matrix is just the identity matrix with its row re-ordered, e.g.

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

- A circulant matrix is defined by the property that each row is a cyclic shift of the previous row to the right by one position.

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

Now we could also use a circular matrix to design our LDPC codes, now what is circular matrix? So a circular matrix has a property that each row is just a circular shift of previous row, so for example if we take this example, this first row is 01, 001 now this 0 comes here, this 1 comes here, this 0 comes here, this 0 comes here, and this 1 comes here, so that is your second row. Now in third row this 0 comes here, 1 comes here, this 0 comes here, 1 comes here, this 0 comes here, and this 0 comes here.

Similarly here this 0 comes here, this 0 comes here, 1 comes here, 0 comes here, and 1 comes here, and likewise this 1 comes here, this 0 comes here, 0 comes here, 1 here, 0 here, so you can see each row has a circular shift of previous row, so we will now show how we, so we can randomly construct these permutation matrix this is just an identity matrix which is row reordered. Now we will show how we can construct our LDPC codes using these permutation matrix.

(Refer Slide Time: 28:49)

MacKay's construction

Schematic Illustration of Regular Gallager Codes

Notation: An integer represents a number of permutation matrices superposed on the surrounding square.

Column Weight	Fraction of columns	Row weight	Fraction
3	1	6	1

Handwritten notes below the table:

- $w_c = 3$
- $1 - \frac{w_c}{w_r} = 1 - \frac{3}{6} = \frac{1}{2}$
- $w_r = 6$

So the first example that we are going to consider is an example of a regular LDPC code, now this regular LDPC code has column weight 3 so w_c is 3 and row weight is 6 so w_r is 6. You can see if this a regular LDPC code because all the rows and columns have the same weight okay. Now how can we use permutation matrix to design this so let us take this example so let us say you have to design and so what is the rate here, rate here is $1 - w_c/w_r$ so this is $1 - 3/6$ that is the rate $1/2$ so m will be $n/2$.

So m here will be $n/2$ so what we did was so this is you can think of this as $n/2 \times n$, so what we did was we divided this $n/2 \times n$ matrices into sub matrices in this particular way so this is, so this one that you see this is $n/2 \times n/2$ matrix and there is another $n/2 \times n/2$ matrix so these are 2 $n/2 \times n/2$ matrices which are further divided into sub matrices $n/6 \times n/6$ so each of them are your $n/6 \times n/6$ matrix and what does this signifies, this signifies that this is the notation that we are using to denote a permutation matrix.

(Refer Slide Time: 30:50)

MacKay's construction

Schematic Illustration of Regular Gallager Codes

Notation: An integer represents a number of permutation matrices superposed on the surrounding square.

Column Weight	Fraction of columns	Row weight	Fraction
3	1	6	1

$w_c = 3$


$1 - \frac{w_c}{w_r} = 1 - \frac{3}{6} = \frac{1}{2}$

$w_r = 6$

So this one random permutation matrix, there is another random permutation matrix by the location of one is different from what was here in this particular matrix. So this is another random permutation matrix so these ones that you see here these are all random permutation matrices, okay. Now note that we need a column weight of three, now if we stack if we stack three permutation matrices like this, now each of these permutation matrix has 1,1 in its column.

(Refer Slide Time: 31:32)

MacKay's construction

$\frac{n}{2} \times \frac{n}{2}$

 $\frac{n}{2} \times n$

$\boxed{1}$

$3 \quad 3$

Schematic Illustration of Regular Gallager Codes

Notation: An integer represents a number of permutation matrices superposed on the surrounding square.

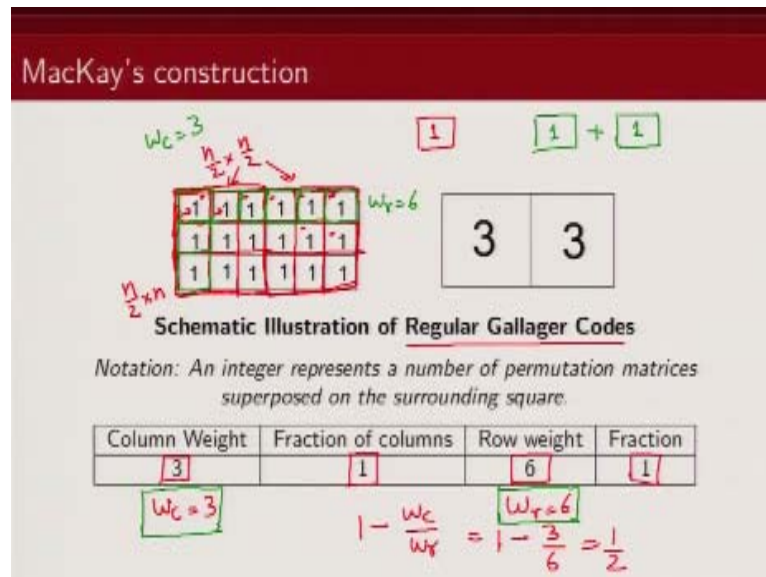
Column Weight	Fraction of columns	Row weight	Fraction
$\boxed{3}$	$\boxed{1}$	$\boxed{6}$	$\boxed{1}$

$w_c = 3$

$1 - \frac{w_c}{w_r} = 1 - \frac{3}{6} = \frac{1}{2}$

$w_r = 6$

(Refer Slide Time: 31:35)



So overall we will get $w_c = 3$ and if we stack six of them like this, then each row also has 1,1 so we will get overall six ones in each row, so this way we can generate a LDPC code with these parameters $w_c = 3$ and $w_r = 6$, now the same thing can be generated using by overlapping of random permutation matrices, now what do I mean by overlap of random permutation matrix so let us say you have a permutation matrix and you add another permutation matrix, now please note you ensure that there is no overlap between ones in this matrix and this matrix.

(Refer Slide Time: 32:43)

MacKay's construction

$w_c = 3$
 $\frac{n}{2} \times \frac{n}{2}$

$w_r = 6$

$\frac{n}{2} \times n$

Schematic Illustration of Regular Gallager Codes

Notation: An integer represents a number of permutation matrices superposed on the surrounding square.

Column Weight	Fraction of columns	Row weight	Fraction
3	1	6	1

$w_c = 3$

$1 - \frac{w_c}{w_r} = 1 - \frac{3}{6} = \frac{1}{2}$

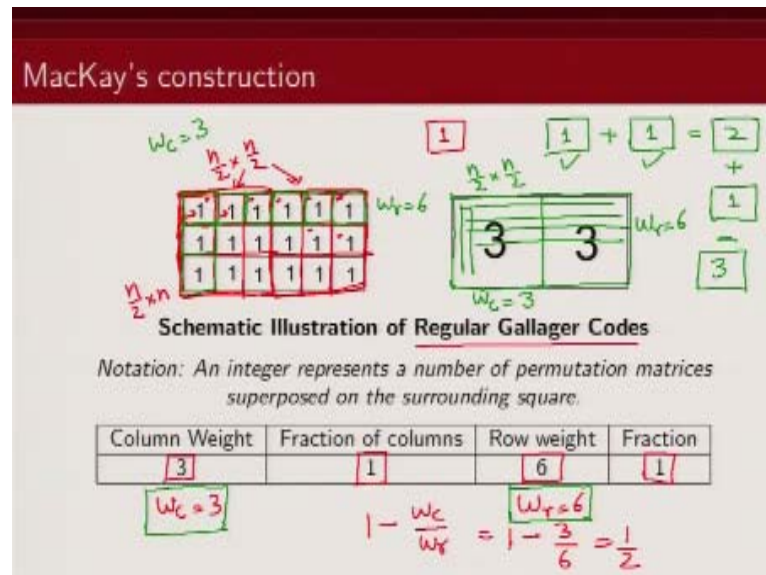
$w_r = 6$

$\frac{1}{\checkmark} + \frac{1}{\checkmark}$

3 3

If you add these two permutation matrix what will you get?

(Refer Slide Time: 32:52)



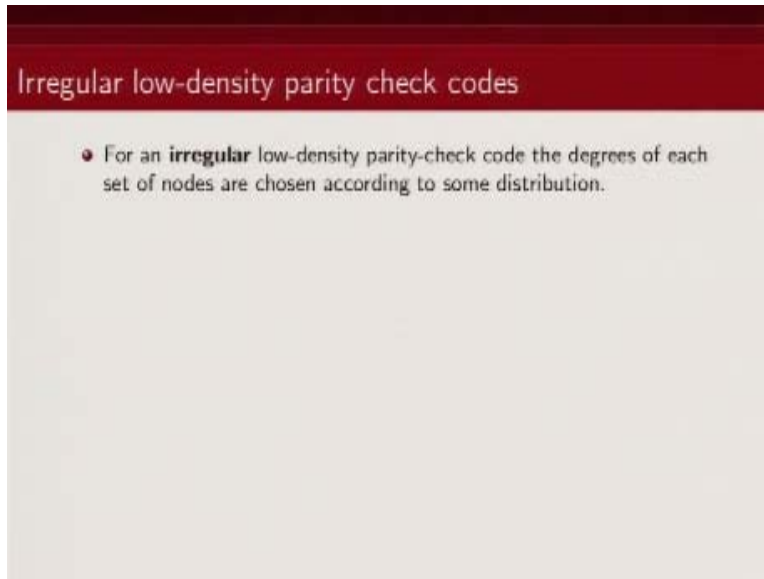
You will get a matrix which will have two ones in each row and two ones in each column, now if I add another permutation matrix and I ensure that this the one in this permutation matrix does not overlap with ones in this permute, in this matrix which I got by adding two permutation matrix then the resultant permutation matrix that I will get will have row weight three and column weight also three, so another way of designing this LDPC code is by overlapping of random permutation matrices.

Please note when I overlap them I have to ensure that there is no ones that are getting overlap then only I will be able to retain the weight, so if I overlap three such random permutation matrices and I ensure that there is no overlap between the ones in each of these three random permutation matrix then what I will get is a matrix which will have three ones in each row and three ones in each column, I can construct another $n/2 \times n/2$ matrix this one.

Which is again by overlapping of three different random permutation matrices ensuring that there is no overlap of ones I can get another matrix which will have three ones in each row and three ones in each column and this will ensure that each column of this matrix will have $w_c = 3$

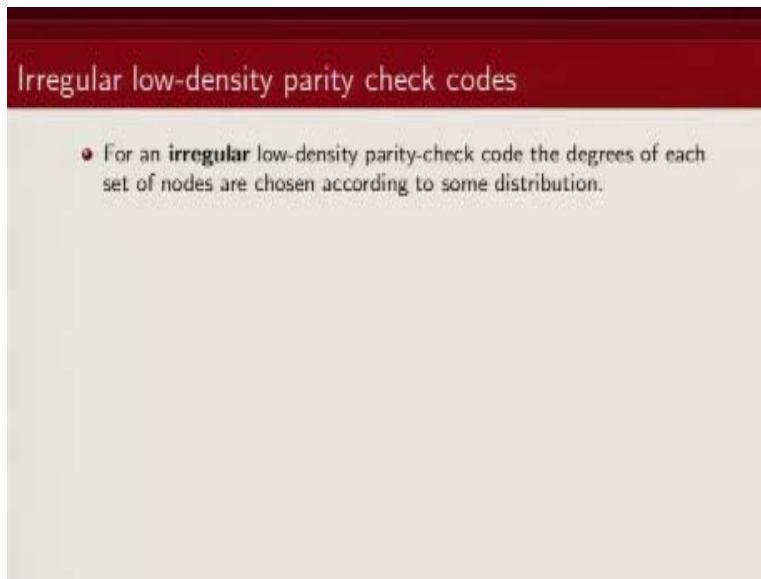
and each of the rows of this matrix will have $w_r = 6$, so this is another way I can use the permutation matrices to construct my regular LDPC code.

(Refer Slide Time: 35:06)



Now let us talk about what is an irregular LDPC code, so in irregular LDPC code as opposed to regular LDPC code the number of ones in each column and number of ones in each row are different.

(Refer Slide Time: 35:24)

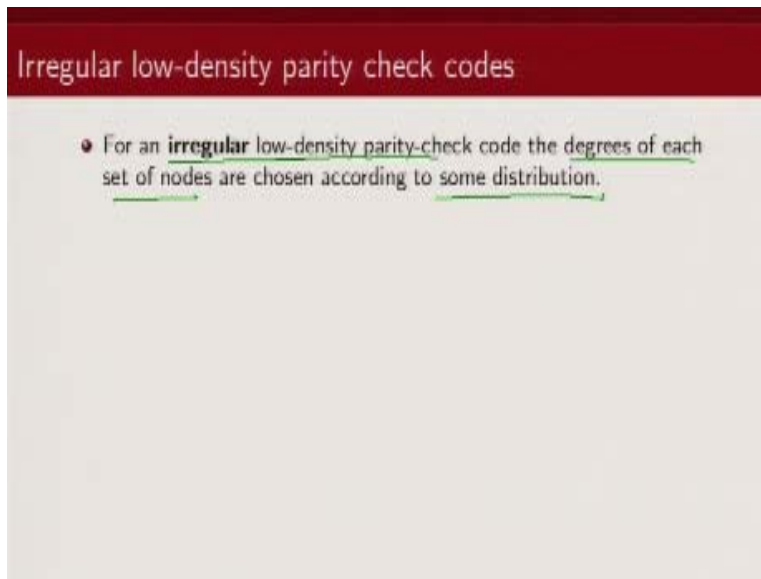


Irregular low-density parity check codes

- For an **irregular** low-density parity-check code the degrees of each set of nodes are chosen according to some distribution.

So we will have to specify the degree the node distribution the column node distribution as well as the row node distribution.

(Refer Slide Time: 35:35)



Irregular low-density parity check codes

- For an irregular low-density parity-check code the degrees of each set of nodes are chosen according to some distribution.

So for an irregular LDPC code we define the distribution of column nodes as well as row nodes according to some distribution.

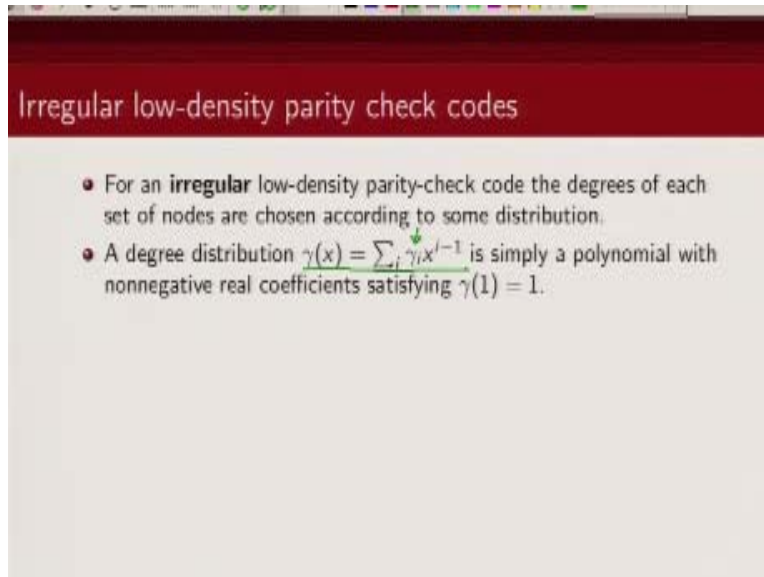
(Refer Slide Time: 35:47)

Irregular low-density parity check codes

- For an **irregular** low-density parity-check code the degrees of each set of nodes are chosen according to some distribution.
- A degree distribution $\gamma(x) = \sum_i \gamma_i x^{i-1}$ is simply a polynomial with nonnegative real coefficients satisfying $\gamma(1) = 1$.

So what is a degree distribution? We define the degree distribution by this polynomial.

(Refer Slide Time: 35:53)



Irregular low-density parity check codes

- For an **irregular** low-density parity-check code the degrees of each set of nodes are chosen according to some distribution.
- A degree distribution $\gamma(x) = \sum_i \gamma_i x^{i-1}$ is simply a polynomial with nonnegative real coefficients satisfying $\gamma(1) = 1$.

Which has a property that $\gamma^{(1)}$ is basically 1 so these are the fraction of nodes with degree i .

(Refer Slide Time: 36:05)

Irregular low-density parity check codes

- For an **irregular** low-density parity-check code the degrees of each set of nodes are chosen according to some distribution.
- A degree distribution $\gamma(x) = \sum_i \gamma_i x^{i-1}$ is simply a polynomial with nonnegative real coefficients satisfying $\gamma(1) = 1$.
- An irregular low-density code is a code of block-length N with a sparse parity check matrix where column distribution $\lambda(x)$ and row distribution $\rho(x)$ is respectively given by

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

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

where λ_i and ρ_i denote the fraction of edges incident to variable and check nodes with degree i , respectively.

(Refer Slide Time: 36:08)

Irregular low-density parity check codes

- For an **irregular** low-density parity-check code the degrees of each set of nodes are chosen according to some distribution.
- A degree distribution $\gamma(x) = \sum_i \gamma_i x^{i-1}$ is simply a polynomial with nonnegative real coefficients satisfying $\gamma(1) = 1$.
- An irregular low-density code is a code of block-length N with a sparse parity check matrix where column distribution $\lambda(x)$ and row distribution $\rho(x)$ is respectively given by

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

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

where λ_i and ρ_i denote the fraction of edges incident to variable and check nodes with degree i , respectively.

Now an irregular LDPC code we have to specify two degree distribution one is the column degree distribution, other is the row degree distribution.

(Refer Slide Time: 36:19)

Irregular low-density parity check codes

- For an **irregular** low-density parity-check code the degrees of each set of nodes are chosen according to some distribution.
- A degree distribution $\gamma(x) = \sum_i \gamma_i x^{i-1}$ is simply a polynomial with nonnegative real coefficients satisfying $\gamma(1) = 1$.
- An **irregular low-density code** is a code of block-length N with a sparse parity check matrix where column distribution $\lambda(x)$ and row distribution $\rho(x)$ is respectively given by

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

where λ_i and ρ_i denote the fraction of edges incident to variable and check nodes with degree i , respectively.

So the column degree distribution we are denoting by $\lambda(x)$ and the row degree distribution we are denoting by $\rho(x)$ so this is my column degree distribution, this is my row degree distribution where λ_i is the fraction of edges incident on the variable node which has degree i and ρ_i is fraction of edges incident to the check node with degree i . So let us take an example to illustrate this let us just see.

(Refer Slide Time: 37:02)

Irregular low-density parity check codes

- For an **irregular** low-density parity-check code the degrees of each set of nodes are chosen according to some distribution.
- A degree distribution $\gamma(x) = \sum_i \gamma_i x^{i-1}$ is simply a polynomial with nonnegative real coefficients satisfying $\gamma(1) = 1$.

If we use this same degree.

(Refer Slide Time: 37:03)

Irregular low-density parity check codes

- For an irregular low-density parity check code the degrees of each set of nodes are chosen according to some distribution.

(Refer Slide Time: 37:04)

MacKay's construction

Schematic Illustration of Regular Gallager Codes

Notation: An integer represents a number of permutation matrices superposed on the surrounding square.

Column Weight	Fraction of columns	Row weight	Fraction
3	1	6	1

Handwritten notes and equations:

- $w_c = 3$
- $w_r = 6$
- $n/2 \times n/2$
- $1 - \frac{w_c}{w_r} = 1 - \frac{3}{6} = \frac{1}{2}$

Notation to represent this, so the column row distribution is defined by $\lambda(x)$.

(Refer Slide Time: 37:11)

Irregular low-density parity check codes

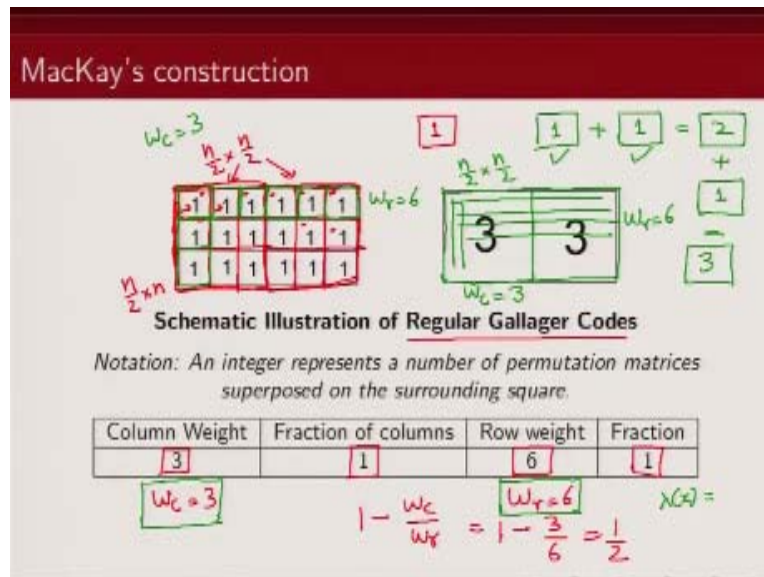
- For an **irregular** low-density parity-check code the degrees of each set of nodes are chosen according to some distribution.
- A degree distribution $\gamma(x) = \sum_i \gamma_i x^{i-1}$ is simply a polynomial with nonnegative real coefficients satisfying $\gamma(1) = 1$.
- An **irregular low-density code** is a code of block-length N with a sparse parity check matrix where column distribution $\lambda(x)$ and row distribution $\rho(x)$ is respectively given by

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

where λ_i and ρ_i denote the fraction of edges incident to variable and check nodes with degree i , respectively.

Now here all the columns for the regular LDPC code all the columns have weight 1.

(Refer Slide Time: 37:18)



So then λ here is 1 and we will define x^{i-1} and i degree here is three so degree distribution for the column for this regular LPDC code will be x^2 , similarly the row distribution here is because all the nodes have row weight six so this will be x^{6-5} , x^5 so that is how we are writing the degree distribution so again.

(Refer Slide Time: 37:47)

Irregular low-density parity check codes

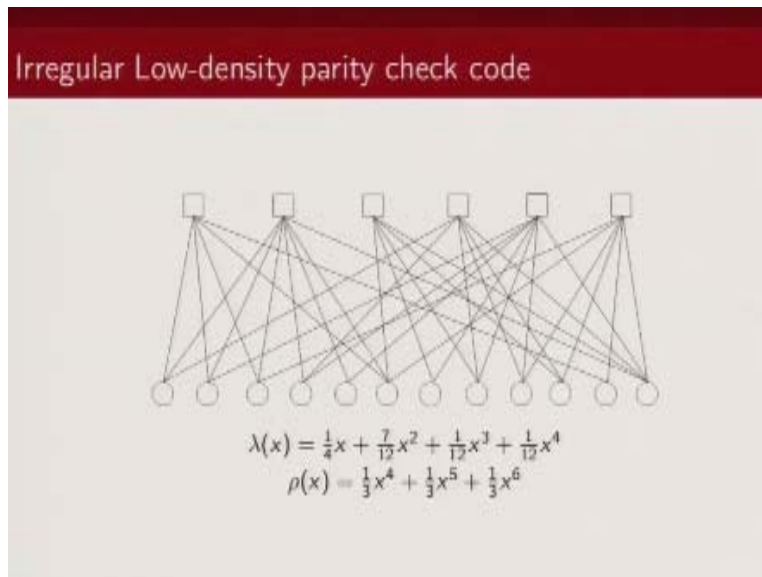
- For an **irregular** low-density parity-check code the degrees of each set of nodes are chosen according to some distribution.
- A degree distribution $\gamma(x) = \sum_i \gamma_i x^{i-1}$ is simply a polynomial with nonnegative real coefficients satisfying $\gamma(1) = 1$.
- An irregular low-density code is a code of block-length N with a sparse parity check matrix where column distribution $\lambda(x)$ and row distribution $\rho(x)$ is respectively given by

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

where λ_i and ρ_i denote the fraction of edges incident to variable and check nodes with degree i , respectively.

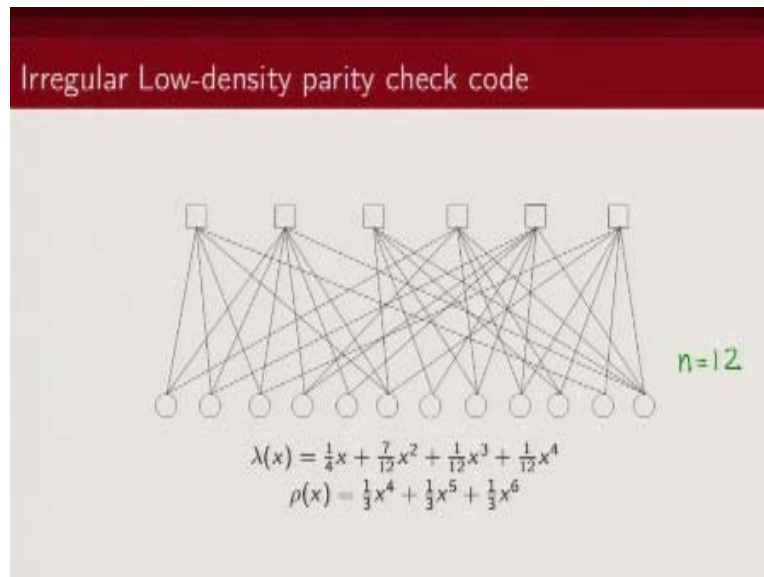
Have a look at the degree distribution this is the fraction of nodes with degree i and these are fraction of nodes with degree i , this is the row degree distribution; this is the column degree distribution.

(Refer Slide Time: 38:04)



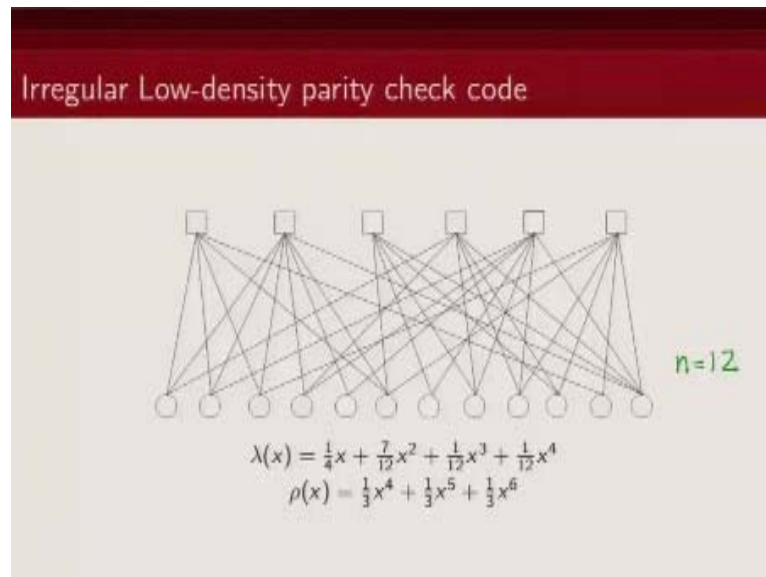
Let us take this example, so we have 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, so n is 12, okay.

(Refer Slide Time: 38:14)



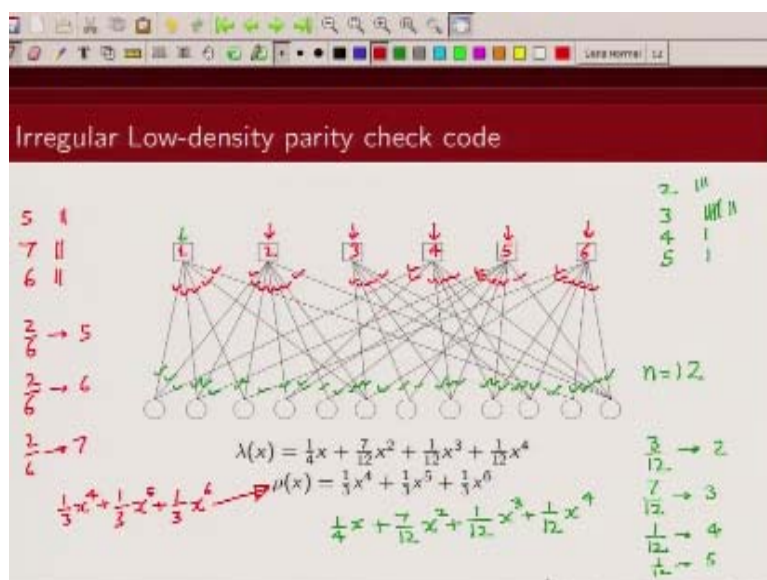
n is 12, now what is column degree distribution? So this is basically each node participating in how many parity check equations.

(Refer Slide Time: 38:32)



Let us look at this node, how many parity check equation it is participating in 1, 2, 3?

(Refer Slide Time: 38:39)



So let us just so there is, this is participating in three parity check equations, what about this? It is participating 1, 2, 3, so it is also participating in three, this is participating in 1, 2, 3, this is participating in three, this is participating in 1, 2, 3, this is participating in two 1, 2, so there is one node which is participating in two, what about this? It is participating in four 1, 2, 3, 4, so this one node which is participating in four, this is participating in two 1, 2, this is participating in three 1, 2, 3.

So this is participating in three 1, 2, 3, it is participating in three 1, 2, 3, it is participating in two 1, 2, and this is participating in 1, 2, 3, 4, 5, okay, so then how many nodes are participating in two parity check constrain that is three so what is the fraction, that is 3/12, they are participating in two parity check constrain, there are seven of them which are participating in three parity check constrain.

So fraction of them is 7/12 then this is 1/12 and this which participate in four and then 1/12 which participate in 5, so then what is the column distribution? So this fraction is $\frac{1}{4}x^{2-1}$ that is x plus $\frac{7}{12}x^{3-1}$ that is x^2 plus $\frac{1}{12}x^{4-1}$ that is three, plus $\frac{1}{12}x^4$ so this is the column degree

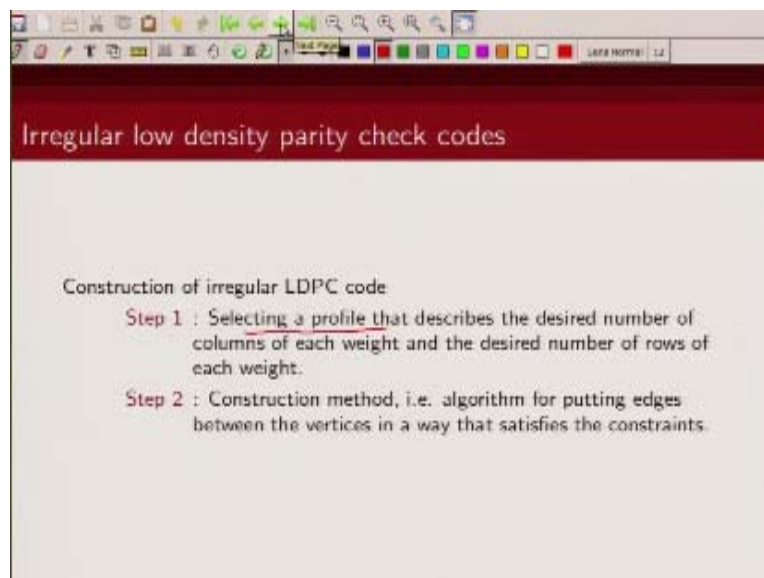
distribution for an LDPC code which is described by this tanner graph, similarly we can find the row distribution.

Let us look parity digit constraints. So let us look at this parity digit constraint. So here, 1, 2, 3, 4, 5, so there are five nodes which are participating in this. What about this one 1,2,3,4,5,6,7 so there are seven bits which are participating in this. This one 1,2,3,4,5 so there is five of them. Then here 1,2,3,4,5,6 so there is six.

This one is 1,2,3,4,5,6,7 and this one is 1,2,3,4,5,6 so you can see out of these and what is the total number of parity check equations these are six 1,2,3,4,5 and 6. So then fraction of parity check equations where five bits are participating is $\frac{2}{6}$ and same is the ratio for 6 and 7. So then we can write the row distribution as $\frac{1}{3} x^{5-1}$ that is $4 + \frac{1}{3} x^{6-1}$ that is $5 + \frac{1}{3} x^{7-1}$ that is 6 and that is precisely what I have written here okay.

So to describe an irregular LDPC code we need to describe these two degree distribution namely the column degree distribution and the row degree distribution.

(Refer Slide Time: 43:10)



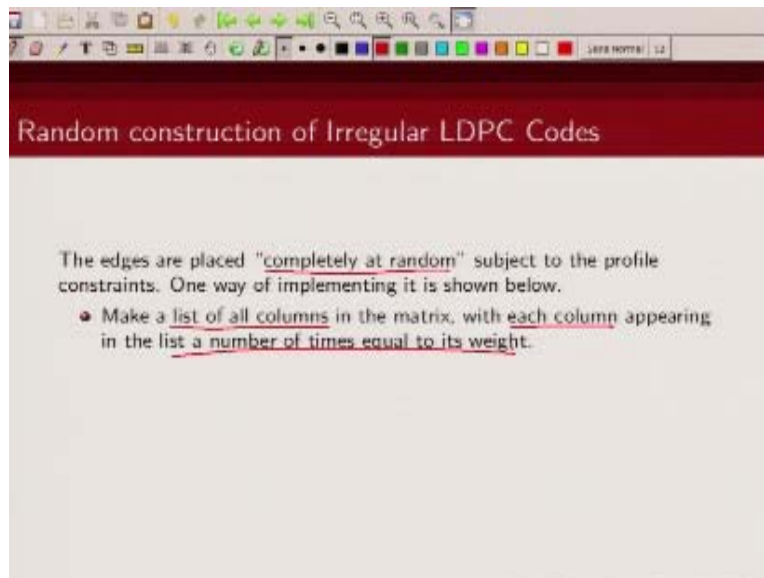
Irregular low density parity check codes

Construction of irregular LDPC code

- Step 1 : Selecting a profile that describes the desired number of columns of each weight and the desired number of rows of each weight.
- Step 2 : Construction method, i.e. algorithm for putting edges between the vertices in a way that satisfies the constraints.

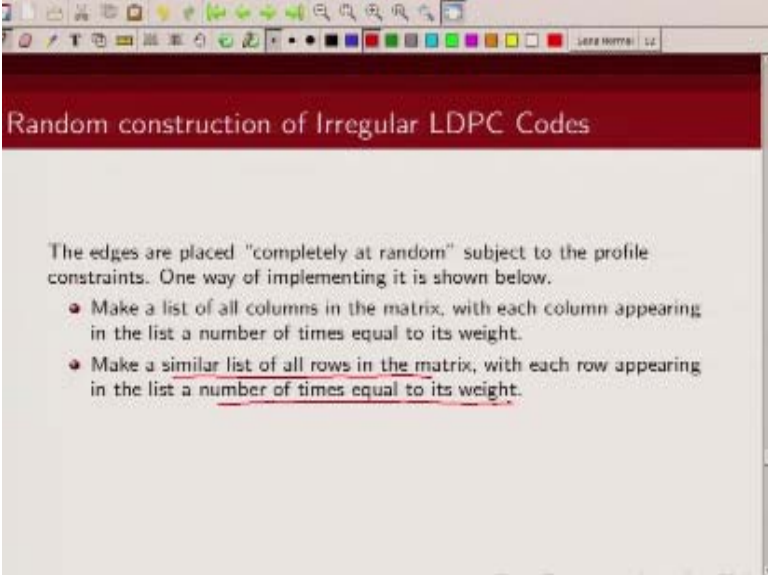
So we can use a random algorithm to construct an LDPC code so we first select a profile that describes the number of rows of each weight and desired number of columns of each weight and then we need to put out mechanism of putting edges between the vertices in such a way that this constraint is satisfy that. So many number of rows should have so many weight and so many number of columns should have so many weight.

(Refer Slide Time: 43:43)



So we can place edges at random subject to the profile constraint and one way of constructing it is as follows. So make a list of all columns in a matrix with each column appearing in the list number of times which is equal to the its weight. So if let say there are five columns which have weight three. So then you will repeat each of these five columns three times.

(Refer Slide Time: 44:14)



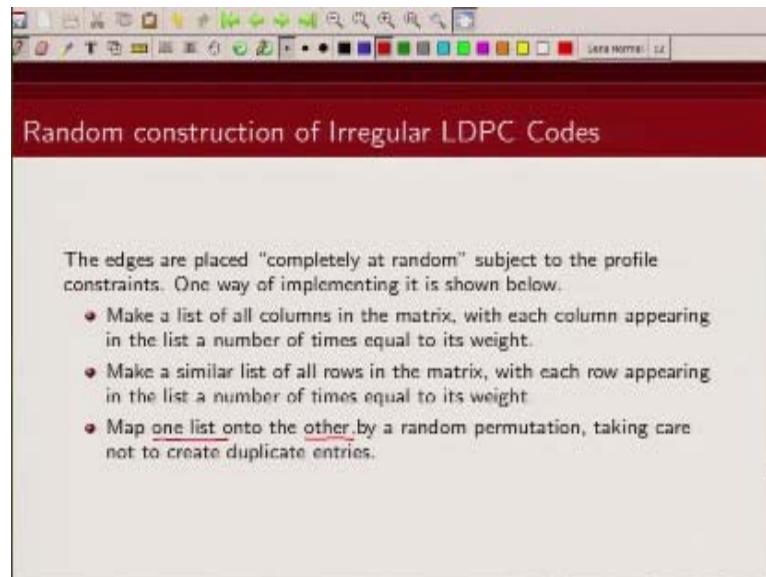
Random construction of Irregular LDPC Codes

The edges are placed "completely at random" subject to the profile constraints. One way of implementing it is shown below.

- Make a list of all columns in the matrix, with each column appearing in the list a number of times equal to its weight.
- Make a similar list of all rows in the matrix, with each row appearing in the list a number of times equal to its weight.

In this matrix. Similarly you make a list of all rows in the matrix such that in each row is repeated equal to the its weight. And then what you do is.

(Refer Slide Time: 44:28)



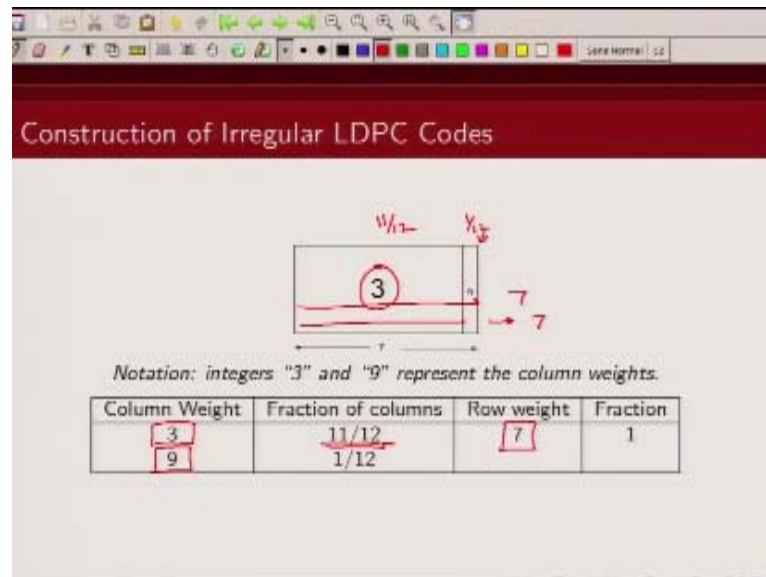
Random construction of Irregular LDPC Codes

The edges are placed “completely at random” subject to the profile constraints. One way of implementing it is shown below.

- Make a list of all columns in the matrix, with each column appearing in the list a number of times equal to its weight.
- Make a similar list of all rows in the matrix, with each row appearing in the list a number of times equal to its weight.
- Map one list onto the other by a random permutation, taking care not to create duplicate entries.

You map one list which is a list of columns into the other list which is the list of rows by random permutation making sure that there is no duplication. So you, you just create a link from the list of columns to the list of rows. And this way you can construct an irregular LDPC code which will satisfy the degree distribution profile. We can also use random permutation matrices to generate our irregular LDPC code; again we are going for very simple constructions so.


(Refer Slide Time: 45:12)



We will go with this construction so let us say we would like to design an LDPC code which has column weight such that $\frac{11}{12}$ of the columns have column weight three and $\frac{1}{12}$ of column has column weight nine. So we are interested in and we are interested in row, row weight of seven. So we want each of these row to have weight seven each of these row should have weight seven and we want that $\frac{11}{12}$ of the column should have weight three and $\frac{1}{12}$ of the column should have weight nine. So how can we construct this?

(Refer Slide Time: 46:05)

Construction of Irregular LDPC Codes

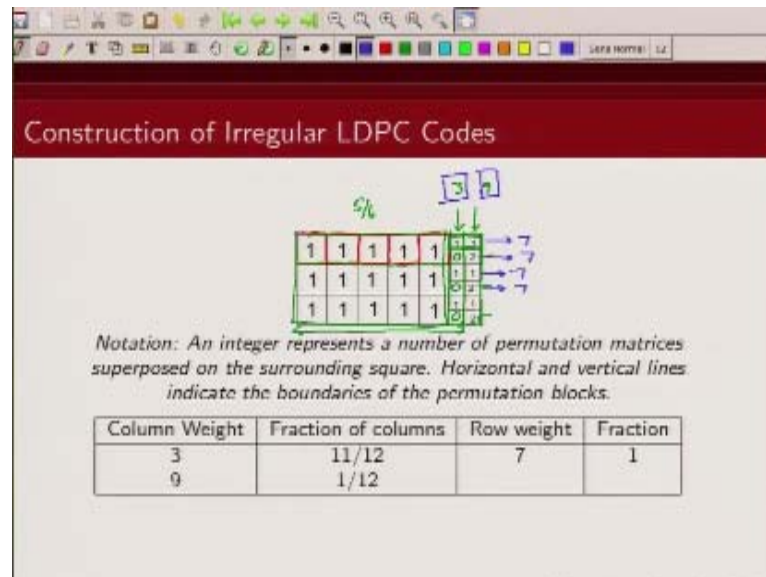


Notation: An integer represents a number of permutation matrices superposed on the surrounding square. Horizontal and vertical lines indicate the boundaries of the permutation blocks.

Column Weight	Fraction of columns	Row weight	Fraction
3	11/12	7	1
9	1/12		

Using the construction using random permutation matrices that we have studied so again I am splitting up this into so these are by so this is my rate half code so this $n/2 \times n$ each of them are $n/6 \times n/6$ matrices these are $n/6 \times n/6$ matrices. Now note that up to this I am ensuring that each column has weight three and each row has weight five. Now I need column weight of.

(Refer Slide Time: 46:53)



11/12 matrices to be three. So this is already I have got 5/6 columns I have got column weight as three. So then what I can do for the remaining 1/6 fraction of columns I split them into two and what I do is I have column weight here and then I further split them into so these are all zero matrices. So this you can see this row will have weight three so this will ensure that 11/12 fraction of columns have weight three.

And here you can see this has weight nine I have added this is weight two, weight two, weight two that is $6+3$, 9 so this column this set of columns will have weight nine. And you can check I already had each row weight up to five. And now I have this row has weight two so this will be overall weight will be seven. This has weight $0+2$ the overall weight is seven. Here the overall row weight is seven.

So I am ensuring that the way I split up this matrices I am ensuring that each row will have weight seven whereas 11/12 fraction of the columns have column weight three whereas 1/12 fraction of the column has weight nine.

(Refer Slide Time: 48:28)

Construction of Irregular LDPC Codes

Notation: An integer represents a number of permutation matrices superposed on the surrounding square. Horizontal and vertical lines indicate the boundaries of the permutation blocks.

Column Weight	Fraction of columns	Row weight	Fraction
3	11/12	<u>7</u>	1
9	1/12		

Now same thing can be done in multiple ways, this is another construction. You can see here again I am ensuring that each of the columns have weight three so up to this point you have $5/6^{\text{th}}$ of the column have weight three. Now I need another $1/12^{\text{th}}$ column which have weight three. So I can do that by placing ones like this. Now I also have to ensure that row weight is seven, so how do I ensure so I have three here so I need a weight four here.

So I do it by one and three. So this is column weight of the row weight seven. Here it is three this was zero so this has to have weight of four. Now again this what are these three, four these are again obtain by overlapping of random permutation matrices. And when we are over lapping make sure that there is no overlap of ones.

(Refer Slide Time: 49:31)

Construction of Irregular LDPC Codes

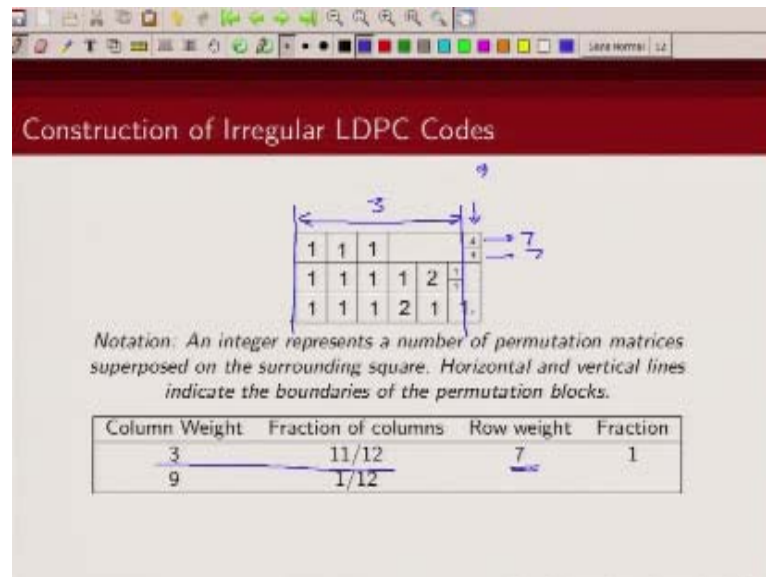
5/6

Notation: An integer represents a number of permutation matrices superposed on the surrounding square. Horizontal and vertical lines indicate the boundaries of the permutation blocks.

Column Weight	Fraction of columns	Row weight	Fraction
3	11/12	<u>7</u>	1
9	1/12		

Okay. So you can verify that each row here so 1,2,3,4,5,6 so this is seven, this is weight zero again here this is weight seven each row will have weight seven. So it is the same profile, different construction.

(Refer Slide Time: 49:52)



There is another construction, same column weight distribution that 11/12 fraction of the bits should have column weight three and 1/12 should have column weight nine. And row weight should be seven, and you can check it each of the row here has weight seven and all of these rows from here to here have column weight three and this has column weight nine four, four and this will have weight one. So we can use random permutation matrix to construct irregular LDPC codes as well. Thank you.

Acknowledgement

Ministry of Human Resource & Development

Prof. Satyaki Roy

Co-ordinator, NPTEL IIT Kanpur

NPTEL Team

Sanjay Pal

Ashish Singh

Badal Pradhan

Tapobrata Das

Ram Chandra

Dilip Tripathi

Manoj Shrivastava

Padam Shukla

**Sanjay Mishra
Shubham Rawat
Shikha Gupta
K. K. Mishra
Aradhana Singh
Sweta
Ashutosh Gairola
Dilip Katiyar
Sharwan
Hari Ram
Bhadra Rao
Puneet Kumar Bajpai
Lalty Dutta
Ajay Kanaujia
Shivendra Kumar Tiwari**

an IIT Kanpur Production

©copyright reserved