

DESIGN METHOD AND APPARATUS FOR QUASI-CYCLIC LOW-DENSITY PARITY-CHECK

CROSS-REFERENCES TO RELATED APPLICATIONS

[0001] This application claims priority to a Chinese patent application No. 201710496533.7 filed on June 26, 2017, disclosure of which is incorporated herein by reference in its entirety.

TECHNICAL FIELD

[0002] The present invention relates to the field of communications and, in particular, to a design method and apparatus for quasi-cyclic low-density parity-check encoding, and a computer storage medium.

BACKGROUND

[0003] A digital communication system in the related art generally includes three parts: a transmitting end, a channel, and a receiving end. The transmitting end can perform channel coding on an information sequence to obtain an encoded codeword, interleave the encoded codeword, and map the interleaved bits into modulation symbols, and then process and transmit the modulation symbols according to communication channel information. In the channel, a specific channel response due to factors such as multipath and movement results in distorted data transmission, and noise and interference can further deteriorate the data transmission. The receiving end receives the distorted modulation symbol data transmitted through the channel and needs to perform specific processing to restore the original information sequence.

[0004] According to an encoding method used by the transmitting end for encoding the information sequence, the receiving end can perform corresponding processing on the received data to reliably restore the original information sequence. The encoding method must be visible to both the transmitting end and the receiving end. Generally, the encoding method is based on forward error correction (FEC) encoding. The FEC encoding adds some redundant information to the information sequence. The receiving end can reliably restore the original information sequence with the redundant information.

[0005] Some common FEC codes include: a convolutional code, a Turbo code, and a low

density parity check (LDPC) code. In the FEC encoding process, the FEC encoding is performed on an information sequence with the number k of bits to obtain an FEC encoded codeword with the number n of bits (including $n-k$ redundant bits), and an FEC encoding rate is k/n . The LDPC code is a linear block code defined with a very sparse parity check matrix or a bipartite graph. The sparsity of the check matrix helps achieve low-complexity LDPC encoding and decoding, thus making the LDPC code more practical. Various practices and theories prove that the LDPC code has the best channel encoding performance which is very close to the Shannon limit under additive white Gaussian noise (AWGN). In the parity check matrix of the LDPC code, each row is a parity check code. If an element at a certain index position has a value of 1 in each row, it indicates that the bit at this position participates in the parity check code. If the element is equal to 0, it indicates that the bit at this position does not participate in the parity check code.

[0006] Due to a structured characteristic, a quasi-cyclic LDPC code has become a mainstream application, for example, the quasi-cyclic LDPC code has wide applications in fields such as IEEE802.11ac, IEEE802.11ad, IEEE802.11aj, IEEE802.16e, IEEE802.11n, DVB, microwave communications, and fiber-optic communications. The parity check matrix H of the quasi-cyclic LDPC code is a matrix of $M \times Z$ rows and $N \times Z$ columns, which is composed of $M \times N$ sub-matrices. Each sub-matrix is a different power of a $Z \times Z$ basic permutation matrix. That is, each sub-matrix is obtained through a cyclic shift of a $Z \times Z$ identity matrix. To more easily describe the cyclic shift of the identity matrix from a mathematical perspective, the parity check matrix of the quasi-cyclic LDPC code can be written as the following mathematical formula form:

$$H = \begin{bmatrix} P^{hb_{11}} & P^{hb_{12}} & P^{hb_{13}} & \dots & P^{hb_{1N}} \\ P^{hb_{21}} & P^{hb_{22}} & P^{hb_{23}} & \dots & P^{hb_{2N}} \\ \dots & \dots & \dots & \dots & \dots \\ P^{hb_{M1}} & P^{hb_{M2}} & P^{hb_{M3}} & \dots & P^{hb_{MN}} \end{bmatrix} = P^{Hb}$$

[0007] If $hb_{ij} = -1$, $P^{hb_{ij}}$ is an all-zero matrix of size $Z \times Z$; otherwise, $P^{hb_{ij}}$ is a non-negative integer power of the standard permutation matrix P . The standard permutation matrix P is written as:

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

[0008] With this definition, Z and the power hb_{ij} can uniquely identify each block matrix. If a certain block matrix is an all-zero matrix, the block matrix can be represented by -1 or a null value or in other forms. If the block matrix is obtained through a cyclic shift of value s of the identity matrix, the block matrix is equal to s . All hb_{ij} can constitute the basic matrix Hb of the quasi-cyclic LDPC code, which can be written as:

$$Hb = \begin{bmatrix} hb_{11} & hb_{12} & hb_{13} \cdots hb_{1N} \\ hb_{21} & hb_{22} & hb_{23} \cdots hb_{2N} \\ \cdots & \cdots & \cdots \cdots \\ hb_{M1} & hb_{M2} & hb_{M3} \cdots hb_{MN} \end{bmatrix}$$

[0009] The basic matrix Hb includes two types of elements: an element indicating an all-zero square matrix, and an element indicating a value of the cyclic shift of the identity matrix and generally represented by an integer within a range of 0 to $(Z-1)$. The basic matrix Hb can be referred to as a basic check matrix or a shift value matrix or a permutation value matrix. In the basic matrix Hb , if the element indicating the all-zero matrix is replaced with “0”, and other elements are replaced with “1”, a template matrix of the quasi-cyclic LDPC code can be obtained. Therefore, the basis matrix of the quasi-cyclic LDPC code can be determined according to the template matrix of the quasi-cyclic LDPC code and a group of shift values (or coefficients). The number Z of dimensions of the basic permutation matrix or the all-zero square matrix can be defined as a lifting size/lifting size or an expansion factor.

[0010] In the practical quasi-cyclic LDPC encoding process, since multiple lifting sizes are required to support a flexible design of code lengths, and each lifting size corresponds to one basic matrix, a large number of basic matrices need to be stored. Therefore, in the quasi-cyclic LDPC encoding process, a basic matrix coefficient is required to support the basic matrices corresponding to the multiple lifting sizes. For example, one basic matrix coefficient is Hb_0 and

supports lifting sizes of Z_0, Z_1, Z_2 , and Z_3 . All element values of non“-1” are divided by Z_i to obtain remainders and rounded down proportionally to obtain the basic matrix of the quasi-cyclic LDPC code corresponding to the i -th lifting size Z_i so that all the element values of non“-1” are less than the lifting size. When the basic matrix corresponding to Z_i is obtained, quasi-cyclic LDPC encoding can be performed on the information bit sequence to be encoded.

[0011] Therefore, a structured LDPC code can be uniquely determined by the basic check matrix H_b and the lifting size Z . For example, the basic matrix H_b (2 rows and 4 columns) corresponds to the lifting size z of 4 and is written as:

$$\mathbf{H_b} = \begin{bmatrix} 0 & 1 & 0 & -1 \\ 2 & 1 & 2 & 1 \end{bmatrix}$$

[0012] The template matrix corresponding to the basic matrix H_b is written as:

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

[0013] The parity check matrix H obtained according to the basic matrix H_b and the lifting size Z is written as:

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

[0014] To gain good decoding performance of the quasi-cyclic LDPC code, concepts of a girth and a short circle are to be introduced. A circle length of 4 is defined as follows: if 4 positive integer values $\{a_0, a_1, b_0, b_1\}$ correspond to the elements $H_b(a_0, b_0)$, $H_b(a_0, b_1)$, $H_b(a_1, b_0)$, and $H_b(a_1, b_1)$ in the basic matrix and these elements are all elements of non“-1”, the basic matrix has a short circle of length 4. Similarly, if 6 positive integer values $\{a_0, a_1, a_2, b_0, b_1, b_2\}$ correspond to the elements $H_b(a_0, b_0)$, $H_b(a_0, b_1)$, $H_b(a_1, b_1)$, $H_b(a_1, b_2)$, $H_b(a_2, b_2)$, and $H_b(a_2, b_0)$ in the basic matrix and these elements are all elements of non“-1”, the basic matrix has a short circle of length 6.

b_2), and $Hb(a_2, b_0)$ in the basic matrix and these elements are all the elements of non-“-1”, the basic matrix has a short circle of length 6. Similarly, if 8 positive integer values $\{a_0, a_1, a_2, a_3, b_0, b_1, b_2, b_3\}$ correspond to the elements $Hb(a_0, b_0)$, $Hb(a_0, b_1)$, $Hb(a_1, b_1)$, $Hb(a_1, b_2)$, $Hb(a_2, b_2)$, $Hb(a_2, b_3)$, $Hb(a_3, b_2)$, and $Hb(a_3, b_0)$ in the basic matrix and these elements are all the elements of non-“-1”, the basic matrix has a short circle of length 8. In the same manner, short circles of lengths 10, 12 and the like of the basic matrix can be defined. The girth is defined as follows: if at least one of all the short circles of length 4 in the basic matrix satisfies $\text{mod}(Hb(a_0, b_0) - Hb(a_0, b_1) + Hb(a_1, b_1) - Hb(a_1, b_0), Z) = 0$, it is considered that the parity check matrix determined according to the basic matrix and the lifting size Z has a girth of 4. If all the short circles of length 4 satisfy $\text{mod}(Hb(a_0, b_0) - Hb(a_0, b_1) + Hb(a_1, b_1) - Hb(a_1, b_0), Z) \neq 0$, and at least one short circle of length 6 satisfies $\text{mod}(Hb(a_0, b_0) - Hb(a_0, b_1) + Hb(a_1, b_1) - Hb(a_1, b_2) + Hb(a_2, b_2) - Hb(a_2, b_0), Z) = 0$, it is defined that the parity check matrix determined according to the basic matrix and the lifting size Z has a girth of 6. Similarity, if short circles with the girth of 4 and the girth of 6 do not exist, and at least one short circle of length 8 satisfies $\text{mod}(Hb(a_0, b_0) - Hb(a_0, b_1) + Hb(a_1, b_1) - Hb(a_1, b_2) + Hb(a_2, b_2) - Hb(a_2, b_3) + Hb(a_3, b_3) - Hb(a_3, b_0), Z) = 0$, it is defined that the parity check matrix determined according to the basic matrix and the lifting size Z has a girth of 8. The calculation formula $\text{mod}(x_1, x_2)$ means a modulo operation of x_1 by x_2 , for example, $\text{mod}(10, 3) = 1$. Taking the basic matrix in the above example as an example, it can be seen that the parity matrix determined according to the basic matrix and the lifting size has the girth of 4 because a short circle with the girth of 4 exists for the four elements determined according to $\{a_0, a_1, b_0, b_1\} = \{1, 2, 1, 3\}$ satisfy $\text{mod}(Hb(a_0, b_0) - Hb(a_0, b_1) + Hb(a_1, b_1) - Hb(a_1, b_0), Z) = 0$. For the quasi-cyclic LDPC encoding, if the parity check matrix determined according to the basic matrix and the lifting size has a larger girth, the LDPC codeword has a more random characteristic, better performance, and a larger minimum code distance so that an error floor is more unlikely to occur. Therefore, in the design process of the LDPC code, much importance is attached to the girth characteristic of the basic matrix of the quasi-cyclic LDPC code, and the performance of the quasi-cyclic LDPC encoding can be basically determined according to the girth characteristic of the basic matrix. However, it is more difficult to design the basic matrix of the quasi-cyclic LDPC code with a larger girth.

[0015] From the above-mentioned parity check matrix of the LDPC code, it can be known that indexes of elements equal to 1 in the first row of the parity check matrix are [1 6 9], indicating that in the structured LDPC code, the first bit, the sixth bit, and the ninth bit constitute one parity check code. Similarly, indexes of elements equal to 1 in the second row are [2 7 10], indicating that the second bit, the seventh bit, and the tenth bit constitute one parity check code. In the same manner, it can be known that the LDPC code is practically a codeword with a large number of parity check codes piled up. According to the above-mentioned definition of the quasi-cyclic LDPC encoding, since each row of the parity check matrix corresponds to one parity check code, the number of rows of the parity check matrix must be equal to the number of check columns of the parity check matrix so that effective encoding can be performed. In the parity check matrix described above, the first 8 ($2 \times Z = 2 \times 4$) columns are system columns of the parity check matrix, and the last 8 ($2 \times Z = 2 \times 4$) columns are the check columns of the parity check matrix. Similarly, in the basic matrix, the number of rows of the basic matrix is equal to the number of check columns of the basic matrix. In the basic matrix of the quasi-cyclic LDPC code in the above-mentioned example, the first 2 columns are system columns, and the last two columns are check columns. The number of rows of the basic matrix is exactly equal to the number of check columns of the basic matrix. The quasi-cyclic LDPC encoding can be directly performed according to the parity check matrix determined according to the basic matrix H_b and the lifting size Z . According to the definition of the LDPC code, $H \times C = 0$; the parity check matrix H includes $[H_s \ H_p]$, where H_s is a partial matrix of system columns of the parity check matrix and H_p is a partial matrix of check columns of the parity check matrix; the LDPC codeword C can include $[C_s \ C_p]$, where C_s is a system bit sequence (information bits, known bits) of the LDPC code and C_p is a check bit sequence (unknown bits) of the LDPC code. The LDPC encoding process is a process for calculating the check bit sequence. According to $H \times C = 0$, $H_s \times C_s = H_p \times C_p$ can be obtained and then the check bit sequence can be calculated as $C_p = H_p^{-1} \times H_s \times C_s$. Therefore, the partial matrix of check columns of the parity check matrix must be a square matrix and binary reversible so that a quasi-cyclic LDPC encoded sequence is $[C_s \ C_p]$. Of course, the quasi-cyclic LDPC encoded sequence can also be obtained through the cyclic shift of all the Z -bit blocks.

[0016] Layered decoding, that is, a row parallel decoding method, is generally used for LDPC decoding. The row parallel decoding method can greatly reduce the number of iterations and requires about half the number of iterations of flooding decoding. The parity check matrix described above of the structured LDPC code has 8 rows (the expansion factor $Z=4$ and the basic matrix has 2 rows and 4 columns), indicating the existence of 8 parity check codes or 8 check equations. In the decoding process, each parity check code needs to be decoded, and it requires an iteration to update data for all the 8 parity check codes. When all or part of the rows in the layered decoding process are decoded in parallel, if the parallelism is p in each iteration (p parity check codes are simultaneously updated), the current p parity check codes and the next p parity check codes in the iteration process are updated by the same update module (parity check code update module). In this way, the decoder has much lower complexity, and the currently updated data can be used for updating data at the next layer in the layered decoding process. Therefore, a smaller number of iterations are required and higher decoding throughput is gained. For H shown above, if the parallelism is 2, two parity check codes in every 4 rows of the parity check matrix (corresponding to one row of the basic check matrix) are simultaneously updated.

[0017] Therefore, in the quasi-cyclic LDPC encoding design, the design of the basic matrix is very important. A bad design can result in very poor quasi-cyclic LDPC encoding performance and the error floor. No effective solution has been provided for difficulties in designing the basic matrix such as poor performance of the quasi-cyclic LDPC code.

SUMMARY

[0018] Embodiments of the present disclosure provide a design method and apparatus for quasi-cyclic low-density parity-check encoding, and a computer storage medium to solve at least the problem in the related art of poor performance of a quasi-cyclic LDPC code.

[0018a] In a first aspect, the present disclosure provides a method for wireless communication, comprising: performing low-density parity-check, LDPC, encoding on an input sequence having K bits according to a parity check matrix of an LDPC code; and obtaining an encoded sequence having N bits; wherein the parity check matrix is determined based on at least a basic matrix that is determined according to a lifting size Z and a coefficient matrix in a set of coefficient matrices, wherein K is a positive integer, N is an integer greater than K , and Z is a positive integer, wherein all coefficient matrices in the set of coefficient matrices have $mb = 42$

rows and nb = 52 columns, wherein the lifting size Z is determined to be a minimum lifting size in a lifting size set Zset such that $Z \geq K/kb$, kb being equal to $(nb - mb) = 10$, wherein elements in the lifting size set Zset are grouped in A lifting size subsets, wherein the coefficient matrix corresponds to a 5th subset of the A lifting size subsets that the lifting size Z belongs to, A being an integer greater than 1, and wherein elements of the coefficient matrix comprise:

-1,106,77,-1,-1,43,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
133,-1,-1,-1,25,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-

-1,-1,87,-1,-1,56,-1,104,-1,70,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1;-

-1,80,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,139,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1;-

32,-1,-1,-1,-1,89,-1,-1,-1,-1,-1,-1,71,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1;-

-1,-1,135,-1,-1,-1,-1,6,-1,-1,2,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1;-

37,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,25,114,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1;-

-1,60,-1,-1,-1,137,-1,-1,-1,-1,-1,93,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1;-

121,-1,129,-1,-1,-1,-1,26,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1;-

-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,97,-1,-1,56,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1;-

-1,1,-1,-1,-1,70,-1,-1,-1,-1,-1,1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1;-

119,-1,-1,-1,-1,-1,-1,32,-1,-1,-1,-1,142,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1;-

-1,-1,6,-1,-1,-1,-1,-1,-1,73,-1,-1,102,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1;-

-1,48,-1,-1,-1,47,-1,-1,-1,-1,-1,19,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0;

wherein -1 represents a null value.

[0018b] In a second aspect, the present disclosure provides a method for wireless communication, comprising: performing low-density parity-check, LDPC, encoding on an input

sequence having K bits according to a parity check matrix of an LDPC code; obtaining an encoded sequence having N bits; wherein the parity check matrix is determined based on at least a basic matrix that is determined according to a lifting size Z and a set of coefficient matrices, wherein K is a positive integer, N is an integer greater than K, and Z is a positive integer, wherein all coefficient matrices in the set of coefficient matrices have $m_b = 46$ rows and $n_b = 68$ columns, wherein the lifting size Z is a minimum lifting size in a lifting size set Zset such that $Z \geq K/k_b$, k_b being equal to $(n_b - m_b) = 22$, wherein elements in the lifting size set Zset are grouped in A lifting size subsets, wherein the coefficient matrix corresponds to a 7th subset of the A lifting size subsets that the lifting size Z belongs to, A being an integer greater than 1, and wherein elements of the coefficient matrix comprise:

172,2,-1,131,-1,-1,-1,-1,-1,-1,-1,-1,141,-1,-1,-1,96,-1,-1,99,-1,101,35,-1,116,-1,-1,-1,-1,-
1,0,-1,-
1,-1,-1;

6,10,-1,-1,-1,-1,-1,-1,-1,-1,145,53,-1,201,-1,-1,-1,4,164,-1,173,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,0,-1,-
1,-1;

-1,126,77,-1,156,-1,-1,16,12,-1,-1,-1,-1,-1,70,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,0,-1,-
1;

184,194,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,123,-1,-1,-1,16,-1,-1,-1,-1,104,109,124,-1,-1,-1,-1,-1,-
1,-1,-1,-1,0,-1,-
1,-1,-1;

6,20,-1,-1,-1,-1,-1,-1,-1,-1,203,153,-1,104,-1,-1,-1,-1,207,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,0,-1,-
1,-1;

52,-1,-1,147,-1,-1,-1,1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,16,-1,-1,46,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,0,-1,-
1,-

1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,202,-1,-1,118,130,1,-1,-1,-1,2,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,0,-1,-
1,-

173,6,-1,-1,-1,-1,-1,-1,-1,-1,81,-1,-1,182,-1,-1,-1,53,-1,-1,-1,-1,-1,-1,46,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,0,-1,-
1;

-1,88,-1,198,-1,-1,-1,-1,-1,-1,-1,160,-1,-1,-1,-1,-1,-1,-1,122,-1,182,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,0,-1,-
1,-1;

91,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,184,-1,30,3,-1,-1,-1,155,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,0,-1,-
1;

-1,1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,41,167,-1,-1,-1,-1,68,148,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,0,-1,-
1;

7j

-1,-1,135,-1,-1,-1,-1,6,-1,-1,2,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1;
37,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,25,114,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1;
-1,60,-1,-1,-1,137,-1,-1,-1,-1,-1,93,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1;
121,-1,129,-1,-1,-1,-1,26,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1;
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,97,-1,-1,56,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1;-

-1,1,-1,-1,-1,70,-1,-1,-1,-1,-1,1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1;

119,-1,-1,-1,-1,-1,-1,32,-1,-1,-1,-1,142,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1;

-1,-1,6,-1,-1,-1,-1,-1,-1,73,-1,-1,102,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1;

-1,48,-1,-1,-1,47,-1,-1,-1,-1,-1,19,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0;

wherein -1 represents a null value.

[0018d] In a fourth aspect, the present disclosure provides an apparatus for wireless communications, comprising: a processor; and a memory including processor-executable instructions stored thereon, the processor-executable instructions upon execution by the processor configures the processor to: perform low-density parity-check, LDPC, encoding on an input sequence having K bits according to an LDPC code; obtain an encoded sequence having N bits; wherein the parity check matrix is determined based on at least a basic matrix that is determined according to a lifting size Z and a set of coefficient matrices, wherein K is a positive integer, N is an integer greater than K, and Z is a positive integer, wherein all coefficient matrices in the set of coefficient matrices have mb = 46 rows and nb = 68 columns, wherein the lifting size Z is a minimum lifting size in a lifting size set Zset such that $Z \geq K/kb$, kb being equal to $(nb - mb) = 22$, wherein elements in the lifting size set Zset are grouped in A lifting size subsets, wherein the coefficient matrix corresponds to a 7th subset of the A lifting size subsets that the

lifting size Z belongs to, A being an integer greater than 1, and wherein elements of the coefficient matrix comprise:

[illegible]

[illegible]

wherein -1 represents a null value.

[0019] An embodiment of the present disclosure provides a design method for quasi-cyclic low-density parity-check (LDPC) encoding. The method includes: performing LDPC encoding on a K-bit information sequence to be encoded according to a parity check matrix of a quasi-cyclic LDPC code to obtain an N-bit LDPC encoded sequence, where the parity check matrix is determined according to a basic matrix and a lifting size Z, and the basic matrix is

determined according to the lifting size Z and a coefficient matrix, where K is a positive integer, N is an integer greater than K , and Z is a positive integer.

[0020] Another embodiment of the present disclosure provides a design apparatus for quasi-cyclic LDPC encoding. The apparatus includes a quasi-cyclic LDPC encoding module. The quasi-cyclic LDPC encoding module is configured to perform LDPC encoding on a K -bit information sequence to be encoded according to a parity check matrix of a quasi-cyclic LDPC code to obtain an N -bit LDPC encoded sequence, where the parity check matrix is determined according to a basic matrix and a lifting size Z , and the basic matrix is determined according to the lifting size Z and a coefficient matrix, where K is a positive integer, N is an integer greater than K , and Z is a positive integer.

[0021] Another embodiment of the present disclosure further provides a storage medium which is configured to store program codes for executing a step described below.

[0022] LDPC encoding is performed on a K -bit information sequence to be encoded according to a parity check matrix of a quasi-cyclic LDPC code to obtain an N -bit LDPC encoded sequence.

[0023] The parity check matrix is determined according to a basic matrix and a lifting size Z , and the basic matrix is determined according to the lifting size Z and a coefficient matrix, where K is a positive integer, N is an integer greater than K , and Z is a positive integer.

[0024] In the embodiments of the present disclosure, the LDPC encoding is performed on the K -bit information sequence to be encoded according to the parity check matrix of the quasi-cyclic LDPC code, where the parity check matrix is determined according to the basic matrix and the lifting size Z , and the basic matrix is determined according to the lifting size Z and the coefficient matrix. The present disclosure solves the problem in the related art of poor performance of the quasi-cyclic LDPC code and effectively improves quasi-cyclic LDPC encoding performance.

BRIEF DESCRIPTION OF DRAWINGS

The drawings described herein are used to provide a further understanding of the present invention and form a part of the present application. The exemplary embodiments and

descriptions thereof in the present invention are used to explain the present invention and not to limit the present invention in any improper way. In the drawings:

[0026] FIG. 1 is a flowchart of a design method for quasi-cyclic low-density parity-check encoding according to an embodiment of the present invention;

[0027] FIG. 2 is a block diagram of a design apparatus for quasi-cyclic low-density parity-check encoding according to an embodiment of the present invention;

[0028] FIG. 3 is a flowchart of a design method for quasi-cyclic low-density parity-check encoding according to an example 1 of the present invention; and

[0029] FIG. 4 is a block diagram of a design apparatus for quasi-cyclic low-density parity-check encoding according to an example 2 of the present invention.

DETAILED DESCRIPTION

[0030] The present invention will be described hereinafter in detail with reference to the drawings in conjunction with the embodiments. It is to be noted that if not in collision, the embodiments and features therein in the present application may be combined with each other.

[0031] It is to be noted that the terms “first”, “second” and the like in the description, claims and drawings of the present invention are used to distinguish between similar objects and are not necessarily used to describe a particular order or sequence.

[0032] Embodiment 1

[0033] This embodiment provides a design method for quasi-cyclic low-density parity-check encoding. FIG. 1 is a flowchart of a design method for quasi-cyclic low-density parity-check encoding according to an embodiment of the present invention. As shown in FIG. 1, the process of the method includes steps described below.

[0034] In step S102, a parity check matrix is determined according to a basic matrix coefficient and a lifting size Z .

[0035] In step S104, according to the parity check matrix of a quasi-cyclic low-density parity-check (LDPC) code, LDPC encoding is performed on a K -bit information sequence to be encoded to obtain an N -bit LDPC encoded sequence.

[0036] K is a positive integer, N is an integer greater than K , and Z is a positive integer.

[0037] In the above steps, the LDPC encoding is performed on the K-bit information sequence to be encoded according to the parity check matrix of the quasi-cyclic LDPC code, where the parity check matrix is determined according to a basic matrix and the lifting size Z, and the basic matrix is determined according to the lifting size Z and a coefficient matrix. The method solves the problem in the related art of poor performance of the quasi-cyclic LDPC code and effectively improves quasi-cyclic LDPC encoding performance.

[0038] Optionally, a lifting size set Z_{set} exists in this embodiment. The lifting size Z is an element in the lifting size set Z_{set} , an i-th lifting size subset among A lifting size subsets is denoted as Z_{seti} ($i=1, 2, 3, \dots, A$), all elements in the A lifting size subsets constitute the lifting size set Z_{set} , and the A lifting size subsets have no common elements. A coefficient matrix set exists. The coefficient matrix is a coefficient matrix in the coefficient matrix set, and all lifting sizes supported by an i-th coefficient matrix in the coefficient matrix set constitute the i-th lifting size subset Z_{seti} . The coefficient matrix set includes A coefficient matrices, where A is an integer greater than 1. The A coefficient matrices have a same number of rows, a same number of columns, and same positions at which elements of non“-1” are disposed. All elements in the lifting size set Z_{set} are positive integers.

[0039] Optionally, all lifting sizes corresponding to parity check matrices with a girth greater than or equal to 6 of all basic matrices determined by each lifting size in the lifting size set Z_{set} and the i-th coefficient matrix in the coefficient matrix set constitute a set $Z0_{seti}$. The set $Z0_{seti}$ includes a number $L0i$ of elements which belong to the i-th lifting size subset Z_{seti} and a number $L1i$ of elements which do not belong to the i-th lifting size subset Z_{seti} , where $L0i$ is a positive integer less than or equal to L_i , L_i is a number of elements in the i-th lifting size subset Z_{seti} , and $L1i \geq 2$, $L1i \geq 3$, $L1i \geq 4$, $L1i \geq 5$, $L1i \geq 6$, $L1i \geq 7$, $L1i \geq 8$, $L1i \geq 9$, $L1i \geq 10$, or $L1i \geq 11$.

[0040] Optionally, a first coefficient matrix set (e.g., including the template matrix) exists. All coefficient matrices in the first coefficient matrix set have the same number of rows. All the coefficient matrices have the same number of columns as well. Any coefficient matrix in the first coefficient matrix set has the same number of rows and the same number of columns as the A coefficient matrices in the coefficient matrix set.

[0041] Optionally, the coefficient matrix set at least includes one coefficient matrix in the

first coefficient matrix set. That is, the coefficient matrix set and the first coefficient matrix set share at least one coefficient matrix.

[0042] Optionally, the coefficient matrix set at least includes one adjusted coefficient matrix obtained by adjusting one coefficient matrix in the first coefficient matrix set.

[0043] Optionally, all elements of non-“-1” among a first element to an $(kb+M)$ -th element in an s -th row of the adjusted coefficient matrix are equal to all integers obtained from a modulo calculation of all respective elements of non-“-1” among a first element to an $(kb+M)$ -th element in an s -th row of an original coefficient matrix plus R_s by Z_{max} , where $s=1, 2, 3, \dots, row$, R_s is an integer to be added corresponding to the s -th row, and at least one of $R_0, R_1, \dots, R(row)$ is a non-zero integer. Alternatively, all elements of non-“-1” among a first element to a row -th element in a t -th column of the adjusted coefficient matrix are equal to all integers obtained from a modulo calculation of all respective elements of non-“-1” among a first element to a row -th element in a t -th column of the original coefficient matrix plus C_t by Z_{max} ; where $t=1, 2, 3, \dots, col$, C_t is an integer to be added corresponding to the t -th column, and at least one of $C_0, C_1, \dots, C(col)$ is a non-zero integer. row is a number of rows of the coefficient matrix, col is a number of columns of the coefficient matrix, kb is equal to a difference between col and row , M is a non-negative integer less than 10, Z_{max} a maximum lifting size supported by the adjusted coefficient matrix, and Z_{max} is a positive integer.

[0044] Optionally, an element in an s -th row and a t -th column of the adjusted coefficient matrix is equal to an integer obtained from a modulo calculation result of an element in an s -th row and a t -th column of the original coefficient matrix plus (X_s+Y_t) by Z_{max} . The element in the s -th row and the t -th column of the original coefficient matrix is an element of non-“-1”, $s=1, 2, 3, \dots, row$, $t=1, 2, 3, \dots, kb+M$, kb is the difference between col and row , row is the number of rows of the coefficient matrix, col is the number of columns of the coefficient matrix, M is a non-negative integer less than 10, both X_s and Y_t are integers, and at least one of $X_0, X_1, \dots, X(row)$ and $Y_0, Y_1, \dots, Y(kb+M)$ is a non-zero integer. Adding a same integer to all elements of non-“-1” in a same row and/or a same column of the coefficient matrix is equivalent to interleaving the quasi-cyclic LDPC code, which does not affect a girth characteristic and a code distance characteristic of the quasi-cyclic LDPC code and thus has no influence on performance

of the quasi-cyclic LDPC code. The beneficial effects obtained by adding the same integer to all the elements of non-“-1” in the row or column are described as follows: the basic matrix of the quasi-cyclic LDPC code has certain characteristics, for example, many zero elements are equivalent to performing no cyclic shift on the basic matrix, which may reduce encoding and decoding complexity and has no influence on decoding performance; and in certain cases, the performance of the quasi-cyclic LDPC code may be improved, for example, when an LDPC encoded sequence which is an integer multiple of the lifting size Z is not completely selected in a bit selection process, the same integer with better performance may be selected to adjust the LDPC encoded sequence according to different interleaving cases.

[0045] Optionally, A is equal to 8, and the 8 lifting size subsets are respectively $Z_{set1}=\{2, 4, 8, 16, 32, 64, 128, 256\}$, $Z_{set2}=\{3, 6, 12, 24, 48, 96, 192\}$, $Z_{set3}=\{5, 10, 20, 40, 80, 160\}$, $Z_{set4}=\{7, 14, 28, 56, 112, 224\}$, $Z_{set5}=\{9, 18, 36, 72, 144\}$, $Z_{set6}=\{11, 22, 44, 88, 176\}$, $Z_{set7}=\{13, 26, 52, 104, 208\}$, and $Z_{set8}=\{15, 30, 60, 120, 240\}$.

[0046] Optionally, the set Z_{seti} are respectively $Z_{set1}=\{56, 80, 104, 112, 120, 128, 160, 176, 192, 208, 224, 240, 256\}$, $Z_{set2}=\{44, 48, 72, 88, 96, 144, 160, 176, 192, 208, 224, 240, 256\}$, $Z_{set3}=\{60, 80, 120, 128, 160, 208, 224, 240, 256\}$, $Z_{set4}=\{56, 60, 104, 112, 120, 128, 192, 208, 224, 240, 256\}$, $Z_{set5}=\{72, 128, 144, 160, 176, 192, 224, 240, 256\}$, $Z_{set6}=\{88, 96, 104, 128, 176, 192, 208, 224, 240, 256\}$, $Z_{set7}=\{26, 52, 80, 104, 144, 160, 176, 192, 208, 224, 240\}$, and $Z_{set8}=\{30, 60, 112, 120, 144, 160, 208, 224, 240, 256\}$.

[0047] Optionally, all lifting sizes corresponding to all basic matrices with the girth greater than or equal to 6 and corresponding to a quasi-cyclic LDPC encoding rate of $Rate_0$ determined by the each lifting size in the lifting size set Z_{set} and the i -th coefficient matrix in the coefficient matrix set constitute the set Z_{seti} , and all lifting sizes corresponding to all basic matrices with the girth greater than or equal to 6 and corresponding to a quasi-cyclic LDPC encoding rate of $Rate_1$ determined by the each lifting size in the lifting size set Z_{set} and the i -th coefficient matrix in the coefficient matrix set constitute a set Z_{set1i} , where $Rate_0$ is less than $Rate_1$, a number of elements in the set Z_{set0i} is less than or equal to a number of elements in the set Z_{set1i} , $Rate_0$ and $Rate_1$ are both real numbers greater than 0 and less than 1, and both the set Z_{set0i} and the set Z_{set1i} are non-empty sets.

[illegible]

[illegible]

[0093] 32,166,160,29,-1,-1,73,-1,-1,127,0,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0094] 173,-1,-1,191,49,40,33,19,136,54,-1,0,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0095] 171,183,-1,176,15,-1,-1,-1,65,-1,1,-1,0,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0096] -1,13,73,-1,18,20,125,142,75,141,0,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0097] 74,191,-1,-1,-1,-1,-1,-1,-1,-1,-1,173,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0098] 12,65,-1,-1,-1,21,-1,152,-1,-1,-1,96,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0099] 35,-1,-1,-1,-1,88,-1,28,-1,67,-1,40,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0100] -1,5,-1,-1,-1,86,-1,88,-1,-1,-1,54,-1,95,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0101] 72,43,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,56,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0102] -1,1,-1,-1,-1,-1,-1,-1,124,-1,185,82,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0103] 109,173,-1,-1,-1,-1,168,76,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0104] 34,-1,-1,-1,-1,-1,-1,164,-1,24,-1,-1,-1,106,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0105] -1,60,-1,56,-1,-1,-1,-1,-1,-1,-1,107,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0106] 3,2,-1,-1,-1,-1,-1,-1,28,-1,-1,-1,-1,166,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

$-1,-1;$

[0225] $-1,120,121,-1,22,4,73,49,128,79,0,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1;$

[0226] $42,24,-1,-1,-1,-1,-1,-1,-1,-1,-1,51,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-$
 $-1,-1;$

[0227] $40,140,-1,-1,-1,84,-1,137,-1,-1,-1,71,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1;$

[0228] $109,-1,-1,-1,-1,87,-1,107,-1,133,-1,139,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1;$

[0229] $-1,97,-1,-1,-1,135,-1,35,-1,-1,-1,108,-1,65,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1;$

[0230] $70,69,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,88,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-$
 $-1,-1;$

[0231] $-1,97,-1,-1,-1,-1,-1,-1,40,-1,24,49,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1;$

[0232] $46,41,-1,-1,-1,-1,101,96,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1;$

[0233] $28,-1,-1,-1,-1,-1,-1,30,-1,116,-1,-1,-1,64,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1;$

[0234] $-1,33,-1,122,-1,-1,-1,-1,-1,-1,-1,131,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1;$

[0235] $76,37,-1,-1,-1,-1,-1,-1,62,-1,-1,-1,-1,47,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-$
 $-1,-1;$

[0236] $-1,143,-1,-1,-1,-1,51,-1,-1,-1,-1,130,-1,97,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-$
 $-1,-1;$

[0237] $139,-1,-1,-1,-1,-1,-1,-1,-1,-1,96,128,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,$
 $-1,-1;$

[0238] $-1,48,-1,-1,-1,-1,-1,-1,-1,9,-1,28,8,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,$
 $-1,-1;$

[illegible]

[illegible]

[0256] 37,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,25,114,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1;

[0258] 121,-1,129,-1,-1,-1,-1,26,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1;

[illegible][illegible]

[0264] A coefficient matrix 6 is written as:

[illegible]

[0268] -1,26,18,-1,113,148,41,146,125,21,0,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-

1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0269] 154,40,-1,-1,-1,-1,-1,-1,-1,-1,-1,85,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0270] 93,10,-1,-1,-1,38,-1,88,-1,-1,-1,21,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0271] 56,-1,-1,-1,-1,0,-1,79,-1,26,-1,89,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0272] -1,76,-1,-1,-1,133,-1,173,-1,-1,-1,63,-1,119,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0273] 8,154,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,80,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0274] -1,58,-1,-1,-1,-1,-1,-1,173,-1,64,90,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0275] 158,12,-1,-1,-1,-1,42,62,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0276] 94,-1,-1,-1,-1,-1,-1,57,-1,146,-1,-1,-1,92,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0277] -1,112,-1,26,-1,-1,-1,-1,-1,-1,-1,65,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0278] 15,52,-1,-1,-1,-1,-1,-1,170,-1,-1,-1,-1,63,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0279] -1,6,-1,-1,-1,-1,169,-1,-1,-1,-1,0,-1,151,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0280] 84,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,21,110,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0281] -1,104,-1,-1,-1,-1,-1,-1,-1,108,-1,64,38,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0282] -1,152,-1,-1,-1,128,-1,-1,-1,-1,-1,37,82,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;

[illegible]

[illegible]

-1,-1;
[0313] 142,118,-1,-1,-1,136,-1,126,-1,-1,-1,148,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1;
[0314] 29,-1,-1,-1,-1,188,-1,164,-1,201,-1,128,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1;
[0315] -1,113,-1,-1,-1,110,-1,66,-1,-1,-1,35,-1,10,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
1,-1;
[0316] 135,81,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,65,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1;
[0317] -1,71,-1,-1,-1,-1,-1,-1,32,-1,106,70,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1;
[0318] 152,79,-1,-1,-1,-1,86,3,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
1,-1;
[0319] 63,-1,-1,-1,-1,-1,-1,165,-1,63,-1,-1,-1,139,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1;
[0320] -1,172,-1,50,-1,-1,-1,-1,-1,-1,-1,206,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1;
[0321] 203,198,-1,-1,-1,-1,-1,-1,85,-1,-1,-1,-1,48,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-
1,-1;
[0322] -1,44,-1,-1,-1,-1,124,-1,-1,-1,-1,132,-1,138,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,
-1,-1;
[0323] 186,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,24,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-
1,-1;
[0324] -1,28,-1,-1,-1,-1,-1,-1,6,-1,190,89,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-
1,-1;
[0325] -1,10,-1,-1,-1,94,-1,-1,-1,-1,-1,102,35,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-
-1,-1;
[0326] 15,-1,-1,-1,-1,-1,137,120,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,
-1,-1;

[illegible]

[illegible]

-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0357] 65,-1,-1,-1,-1,16,-1,214,-1,224,-1,116,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1;
[0358] -1,100,-1,-1,-1,148,-1,178,-1,-1,-1,221,-1,217,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1;
[0359] 235,216,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,87,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1;
[0360] -1,214,-1,-1,-1,-1,-1,-1,70,-1,143,205,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1;
[0361] 199,2,-1,-1,-1,-1,184,236,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1;
[0362] 59,-1,-1,-1,-1,-1,-1,35,-1,55,-1,-1,-1,56,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1;
[0363] -1,217,-1,143,-1,-1,-1,-1,-1,-1,-1,123,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1;
[0364] 19,185,-1,-1,-1,-1,-1,-1,123,-1,-1,-1,-1,88,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-
1,-1;
[0365] -1,17,-1,-1,-1,-1,178,-1,-1,-1,-1,46,-1,84,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-
-1,-1;
[0366] 58,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,138,162,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-
-1,-1;
[0367] -1,55,-1,-1,-1,-1,-1,-1,-1,43,-1,22,174,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-
-1,-1;
[0368] -1,63,-1,-1,-1,239,-1,-1,-1,-1,-1,236,187,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-
-1,-1;
[0369] 27,-1,-1,-1,-1,-1,188,53,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-
-1,-1;
[0370] 91,54,-1,-1,-1,-1,-1,-1,-1,-1,82,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-
-1,-1;

[illegible]

[illegible]

$-1,-1;$

[0414] $111,-1,-1,-1,-1,-1,211,225,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-$
 $1,-1;$

[0415] $150,38,-1,-1,-1,-1,-1,-1,-1,-1,66,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,$
 $-1,-1;$

[0416] $-1,191,-1,-1,250,-1,-1,-1,-1,-1,-1,214,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $0,-1;$

[0417] $2,-1,-1,-1,-1,-1,-1,-1,232,-1,-1,-1,-1,76,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $0,-1;$

[0418] $-1,147,134,-1,$
 $1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0419] $25,-1,-1,235,-1,194,-1,$
 $1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0420] $-1,6,11,-1,-1,-1,-1,-1,-1,173,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0421] $11,-1,-1,-1,-1,0,-1,$
 $-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0422] $-1,-1,6,-1,-1,-1,-1,232,-1,-1,-1,-1,226,202,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0423] $232,-1,-1,-1,-1,-1,19,-1,$
 $-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0424] $-1,193,23,-1,-1,142,-1,$
 $1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0425] $0,-1,-1,-1,112,-1,$
 $1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0426] $-1,-1,31,-1,-1,245,-1,12,-1,2,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0427] $-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,152,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[illegible]

[illegible]

1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0458] 97,54,-1,-1,-1,-1,-1,-1,-1,-1,-1,27,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0459] -1,94,-1,-1,28,-1,-1,-1,-1,-1,-1,185,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0460] 83,-1,-1,-1,-1,-1,-1,-1,79,-1,-1,-1,-1,116,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0461] -1,22,67,-1,-
1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0462] 97,-1,-1,43,-1,106,-1,-
1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0463] -1,102,104,-1,-1,-1,-1,-1,-1,-1,169,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0464] 171,-1,-1,-1,-1,121,-
1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0465] -1,-1,183,-1,-1,-1,-1,148,-1,-1,-1,-1,43,160,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0466] 108,-1,-1,-1,-1,-1,176,-1,-
1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0467] -1,118,161,-1,-1,191,-
1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0468] 22,-1,-1,-1,22,-1,-
1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0469] -1,-1,135,-1,-1,74,-1,154,-1,6,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0470] -1,137,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,35,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0471] 105,-1,-1,-1,-1,116,-1,-1,-1,-1,-1,-1,132,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1;

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

-1,-1;
[0546] 19,-1,-1,-1,-1,-1,-1,-1,122,-1,-1,-1,-1,66,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
0,-1;
[0547] -1,0,31,-1,
0,-1;
[0548] 49,-1,-1,134,-1,16,-1,
-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0549] -1,64,72,-1,-1,-1,-1,-1,-1,-1,180,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0550] 221,-1,-1,-1,-1,33,-1,
-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0551] -1,-1,27,-1,-1,-1,-1,87,-1,-1,-1,-1,157,187,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0552] 201,-1,-1,-1,-1,-1,72,-1,
-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0553] -1,219,202,-1,-1,68,-1,
1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0554] 153,-1,-1,-1,196,-1,
1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0555] -1,-1,68,-1,-1,217,-1,131,-1,93,-1,
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0556] -1,193,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,141,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0557] 82,-1,-1,-1,-1,138,-1,-1,-1,-1,-1,-1,204,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0558] -1,-1,142,-1,-1,-1,-1,53,-1,-1,144,-1,
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0559] 79,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,93,54,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;

[0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0590] -1,136,1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0591] 9,-1,-1,142,-1,121,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0592] -1,122,26,-1,-1,-1,-1,-1,-1,125,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0593] 10,-1,-1,-1,-1,128,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0594] -1,-1,111,-1,-1,-1,-1,106,-1,-1,-1,-1,142,98,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0595] 19,-1,-1,-1,-1,-1,7,-1,
-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0596] -1,55,68,-1,-1,120,-1,
-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0597] 40,-1,-1,-1,19,-1,
-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0598] -1,-1,126,-1,-1,113,-1,112,-1,37,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0599] -1,107,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,130,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0600] 2,-1,-1,-1,-1,66,-1,-1,-1,-1,-1,-1,68,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0601] -1,-1,95,-1,-1,-1,-1,115,-1,-1,37,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0602] 56,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,10,16,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1;
[0603] -1,12,-1,-1,-1,43,-1,-1,-1,-1,-1,15,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1;

[illegible]

[illegible]

[illegible]

[illegible]

[0663] -1,49,-1,-1,-1,-1,-1,-1,4,-1,164,128,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[0664] 165,172,-1,-1,-1,-1,11,148,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0665] 85,-1,-1,-1,-1,-1,-1,111,-1,161,-1,-1,-1,113,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0666] -1,84,-1,128,-1,-1,-1,-1,-1,-1,-1,198,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1
,-1;-
[0667] 54,129,-1,-1,-1,-1,-1,-1,10,-1,-1,-1,-1,59,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0668] -1,165,-1,-1,-1,-1,77,-1,-1,-1,-1,125,-1,131,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0669] 74,-1,-1,-1,-1,-1,-1,-1,-1,-1,72,48,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0670] -1,160,-1,-1,-1,-1,-1,-1,-1,134,-1,194,93,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0671] -1,194,-1,-1,-1,78,-1,-1,-1,-1,-1,158,91,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0672] 193,-1,-1,-1,-1,-1,19,14,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0673] 71,134,-1,-1,-1,-1,-1,-1,-1,39,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0674] -1,5,-1,-1,49,-1,-1,-1,-1,-1,-1,159,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0675] 61,-1,-1,-1,-1,-1,-1,-1,48,-1,-1,-1,-1,82,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0676] -1,162,39,-1,
-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[0677] 37,-1,-1,201,-1,174,-1,

[illegible]

[illegible]

[illegible]

$-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0722] $17,-1,-1,-1,-1,115,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0723] $-1,-1,239,-1,-1,-1,-1,173,-1,-1,-1,-1,193,116,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0724] $18,-1,-1,-1,-1,-1,171,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0725] $-1,88,128,-1,-1,66,-1$
 $, -1,-1,-1,-1,-1,-1,0,-1;$

[0726] $181,-1,-1,-1,13,-1,$
 $-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0727] $-1,-1,231,-1,-1,177,-1,12,-1,17,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0728] $-1,6,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,204,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0729] $130,-1,-1,-1,-1,214,-1,-1,-1,-1,-1,-1,67,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0730] $-1,-1,235,-1,-1,-1,-1,192,-1,-1,65,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0731] $123,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,42,2,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0732] $-1,123,-1,-1,-1,135,-1,-1,-1,-1,-1,5,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $, -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0733] $125,-1,190,-1,-1,-1,-1,190,-1,$
 $-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0734] $-1,-1,-1,-1,-1,-1,-1,-1,-1,78,-1,-1,190,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[0735] $-1,90,-1,-1,-1,126,-1,-1,-1,-1,-1,234,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,$
 $-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;$

[illegible]

[illegible]

[0804] 347,-1,-1,88,-1,-1,-1,18,-1,-1,-1,-1,-1,-1,-1,-1,-1,12,-1,-1,261,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,0,-1,-
1,-1;

[0805] 6,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,324,-1,-1,193,23,2,-1,-1,-1,297,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,0,-1,-
1,-1;

[0806] 339,155,-1,-1,-1,-1,-1,-1,-1,-1,109,-1,-1,4,-1,-1,-1,-1,6,-1,-1,-1,-1,-1,-1,178,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,0,-1,-
1,-1;

[0807] -1,163,-1,70,-1,-1,-1,-1,-1,-1,-1,95,-1,-1,-1,-1,-1,-1,-1,27,-1,145,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,0,-1,-
1,-1;

[0808] 4,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,326,-1,57,1,-1,-1,-1,215,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,0,-1,-
1;

[0809] -1,86,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,341,22,-1,-1,-1,-1,349,292,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1;

[0810] 2,269,-1,-1,-1,-1,-1,295,320,-1,46,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1;

[0811] 104,-1,-1,18,-1,-1,-1,-1,-1,136,-1,1,-1,-1,-1,-1,-1,-1,-1,-1,47,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1;

[0812] -1,6,-1,-1,-1,250,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,157,-1,-1,-1,284,2,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-
1,-1;

[0813] 80,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,159,22,-1,-1,-1,45,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,

[illegible]

[illegible]

[illegible]

[illegible]

[0865] -1,8,-1,-1,-1,-1,249,-1,242,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1;

[illegible][illegible][illegible]

73

[0902] 88,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,126,-1,217,54,-1,-1,-1,8,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-
1,-1;

[0903] -1,88,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,142,135,-1,-1,-1,-1,67,189,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-
1,-1,-1;

[0904] 52,166,-1,-1,-1,-1,-1,41,85,-1,4,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-
1;

[0905] 186,-1,-1,43,-1,-1,-1,-1,-1,60,-1,60,-1,-1,-1,-1,-1,-1,-1,-1,-1,40,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-
1,-1;

[0906] -1,53,-1,-1,-1,199,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,168,-1,-1,-1,18,192,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1;

[0907] 125,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,164,156,-1,-1,-1,43,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1;

[0908] -1,18,70,-1,-1,-1,-1,-1,-1,-1,-1,84,-1,-1,-1,-1,-1,-1,-1,49,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1;

[0909] 165,-1,-1,212,16,-1,-1,-1,-1,-1,-1,-1,99,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,118,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-
1,-1,-1;

[0910] -1,179,-1,-1,-1,-1,178,137,-1,-1,-1,-1,-1,-1,83,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1;

[0911] 57,-1,23,-1,16,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,215,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1;-

[illegible]

[0932] 6,120,58,248,-1,240,272,-1,-1,40,9,196,50,254,-1,149,69,-1,177,275,214,150,0,0,-1,-
-1,
-1,-1,-1,-1,-1,-1;

[0933] 247,-1,70,10,99,244,-1,56,208,56,-1,240,251,-1,252,155,97,230,-1,187,-1,181,145,0,0
, -1,
, -1,-1,-1,-1,-1,-1,-1;

[0934] 158,189,10,-1,128,250,3,142,128,42,67,-1,-1,92,201,144,-1,261,54,184,225,-1,-1,-1,0
, 0,-1,
-1,-1,-1,-1,-1,-1,-1;

[0935] 89,21,-1,129,76,-1,0,174,209,-1,138,218,256,26,79,-1,136,200,79,-1,35,255,0,-1,-1,0,
-1,
-1,-1,-1,-1,-1,-1;

[0936] 65,241,-1,0,-1,-1,-1,-1,-
-1,-
1;

[0937] 216,39,-1,226,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,220,-1,-1,-1,-1,241,44,-1,-1,-1,-1,0,-1,-
-1,-
-1,-1,-1;

[0938] 266,-1,-1,-1,-1,-1,45,-1,-1,-1,284,178,-1,265,-1,-1,-1,54,158,-1,11,-1,-1,-1,-1,-1,-1,-1,
0,-1,-
-1,-1,-1,-1;

[0939] 84,110,-1,-1,23,-1,-1,22,250,-1,-1,-1,-1,-1,232,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0
, -1,
, -1,-1;

[0940] 160,270,-1,35,-1,-1,-1,-1,-1,-1,-1,-1,281,-1,-1,-1,225,-1,-1,196,-1,1,63,-1,202,-1,-1,-1,
-1,-1,0,-1,-
-1,-1,-1,-1;

[0941] 252,9,-1,-1,-1,-1,-1,-1,-1,-1,254,223,-1,228,-1,-1,-1,248,142,-1,186,-1,-1,-1,-1,-1,-1,-1

[illegible]

[illegible][illegible][illegible][illegible][illegible]

83

[illegible]

[illegible]

[illegible][illegible][illegible][illegible][illegible][illegible]

[illegible]

[illegible][illegible][illegible][illegible][illegible]

94

[illegible]

[illegible]

[1119] All the lifting sizes supported by the coefficient matrix i correspond to the lifting size subset Z_{seti} , where $i=1, 2, 3, \dots, 8$, and the i -th set $Z0_{seti}$ corresponds to the coefficient matrix i , where $i=1, 2, 3, \dots, 8$.

[1120] In another optional implementation mode, the first coefficient matrix set at least includes one of coefficient matrices of 46 rows and 68 columns described below.

[1121] The coefficient matrix 1 is written as:

[illegible][illegible][illegible][illegible][illegible][illegible][illegible][illegible]

[**I130**] 209,97,-1,55,-1,-1,-1,-1,-1,-1,-1,129,-1,-1,-1,223,-1,-1,182,-1,239,77,-1,225,-1,-1,-1,-1,-1,0,-1;-

[**I131**] 67,40,-1,-1,-1,-1,-1,-1,-1,-1,66,79,-1,156,-1,-1,-1,73,112,-1,136,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1;-

[**I132**] -1,109,3,-1,184,-1,-1,177,254,-1,-1,-1,-1,-1,118,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1;-

[**I133**] 175,92,-1,-1,-1,-1,-1,-1,-1,-1,-1,113,-1,-1,-1,229,-1,-1,-1,-1,217,68,165,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1;-

[**I134**] 36,254,-1,-1,-1,-1,-1,-1,-1,-1,220,150,-1,105,-1,-1,-1,-1,214,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1;-

[**I135**] 108,-1,-1,178,-1,-1,-1,208,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,24,-1,-1,17,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1;-

[**I136**] 69,-1,-1,-1,-1,-1,-1,-1,-1,-1,228,-1,-1,238,39,25,-1,-1,-1,78,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1;-

[**I137**] 229,133,-1,-1,-1,-1,-1,-1,-1,67,-1,-1,176,-1,-1,-1,68,-1,-1,-1,-1,-1,-1,70,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1;-

[**I138**] -1,229,-1,156,-1,-1,-1,-1,-1,-1,83,-1,-1,-1,-1,-1,-1,7,-1,126,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-

[**I139**] 133,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,41,-1,169,29,-1,-1,-1,132,-1,-1,-1,-1,-1,-1,-1,-1,-1,

[illegible]

[illegible]

[illegible]

[**1240**] 104,-1,-1,49,65,-1,-1,-1,-1,-1,-1,74,-1,-1,-1,-1,-1,-1,114,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[1241] -1,39,-1,-1,-1,-1,0,0,-1,-1,-1,-1,-1,-1,107,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
1;
[1242] 10,-1,0,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,36,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;
[1243] -1,31,-1,-1,-1,-1,160,-1,0,-1,-
1,-1;-
1;
[1244] 80,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1;
[1245] -1,113,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1;-
[1246] 0,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1;
[1247] -1,0,-1,-1,-1,-1,-1,97,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,20,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-
-1,-1;-
[1248] 0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,0,-1,-1,-1,-1,-1,-1,-1,15,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1;-
[1249] -1,178,0,-1,-1,-1,-1,-1,-1,-1,-1,160,-1,-1,-1,-1,-1,-1,-1,-1,-1,188,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1;-
1,-1;
[1250] 16,-1,-1,-1,-1,-1,-1,266,-1,-1,-1,-1,-1,-1,-1,240,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1;-
-1;
[1251] -1,0,-1,-1,-1,-1,160,-1,-1,-1,-1,-1,183,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,

[illegible]

[illegible]

[illegible]

[illegible]

[illegible][illegible][illegible][illegible][illegible]

120

[illegible]

[illegible]

[illegible][illegible][illegible]

[1425] -1,120,-1,-1,-1,181,-1,-1,-1,-1,-1,-1,-1,-1,202,-1,-1,-1,143,38,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1;

[illegible]

126

[illegible]

[illegible][illegible][illegible][illegible][illegible]

129

[illegible]

[illegible]

embodiment and preferred implementation modes. What has been described will not be repeated. As used below, a term “module” may be software, hardware or a combination thereof capable of implementing predetermined functions. The apparatus in the embodiment described below is preferably implemented by software, but implementation by hardware or by a combination of software and hardware is also possible and conceived.

[1501] FIG. 2 is a block diagram of a design apparatus for quasi-cyclic low-density parity-check encoding according to an embodiment of the present invention. As shown in FIG. 2, the apparatus includes a quasi-cyclic LDPC encoding module 20.

[1502] The quasi-cyclic LDPC encoding module 20 is configured to perform LDPC encoding on a K-bit information sequence to be encoded according to a parity check matrix of a quasi-cyclic LDPC code to obtain an N-bit LDPC encoded sequence.

[1503] The parity check matrix is determined according to a basic matrix coefficient and a lifting size Z, where K is a positive integer, N is an integer greater than K, and Z is a positive integer.

[1504] Optionally, the apparatus further includes a first storage module and a second storage module. The first storage module is configured to store a lifting size set Zset. The lifting size Z is an element in the lifting size set Zset, A lifting size subsets are respectively Zset_i (i=1, 2, 3, ..., A), all elements in the A lifting size subsets constitute the lifting size set Zset, and the A lifting size subsets have no common elements. The second storage module is configured to store a coefficient matrix set. The coefficient matrix is a coefficient matrix in the coefficient matrix set, and all lifting sizes supported by an i-th coefficient matrix in the coefficient matrix set constitute an i-th lifting size subset Zset_i.

[1505] The coefficient matrix set includes A coefficient matrices, where A is an integer greater than 1. The A coefficient matrices have a same number of rows, a same number of columns, and same positions at which elements of non-“-1” are disposed. All elements in the lifting size set Zset are positive integers.

[1506] Optionally, all lifting sizes corresponding to parity check matrices with a girth greater than or equal to 6 of all basic matrices determined by each lifting size in the lifting size set Zset and the i-th coefficient matrix in the coefficient matrix set constitute a set Z0set_i. The set Z0set_i

includes a number L_{0i} of elements which belong to the i -th lifting size subset Z_{seti} and a number L_{1i} of elements which do not belong to the i -th lifting size subset Z_{seti} , where L_{0i} is a positive integer less than or equal to L_i , L_i is a number of elements in the i -th lifting size subset Z_{seti} , and $L_{1i} \geq 2$, $L_{1i} \geq 3$, $L_{1i} \geq 4$, $L_{1i} \geq 5$, $L_{1i} \geq 6$, $L_{1i} \geq 7$, $L_{1i} \geq 8$, $L_{1i} \geq 9$, $L_{1i} \geq 10$, or $L_{1i} \geq 11$.

[1507] Optionally, the apparatus further includes a third storage module. The third storage module is configured to store a first coefficient matrix set. All coefficient matrices in the first coefficient matrix set have a same number of rows and a same number of columns, and any coefficient matrix in the first coefficient matrix set has a same number of rows and a same number of columns as the A coefficient matrices in the coefficient matrix set stored in the second storage module.

[1508] Optionally, the coefficient matrix set stored in the second storage module includes at least one coefficient matrix in the first coefficient matrix set.

[1509] Optionally, the coefficient matrix set stored in the second storage module includes at least one adjusted coefficient matrix obtained by adjusting one coefficient matrix in the first coefficient matrix set.

[1510] Optionally, all elements of non-“-1” among a first element to an $(kb+M)$ -th element in an s -th row of the adjusted coefficient matrix are equal to all integers obtained from a modulo calculation of all respective elements of non-“-1” among a first element to an $(kb+M)$ -th element in an s -th row of an original coefficient matrix plus R_s by Z_{max} , where $s=1, 2, 3, \dots, row$, R_s is an integer to be added corresponding to the s -th row, at least one of $R_0, R_1, \dots, R(row)$ is a non-zero integer. Alternatively, all elements of non-“-1” among a first element to a row -th element in a t -th column of the adjusted coefficient matrix are equal to all integers obtained from a modulo calculation of all respective elements of non-“-1” among a first element to a row -th element in a t -th column of the original coefficient matrix plus C_t by Z_{max} , where $t=1, 2, 3, \dots, col$, C_t is an integer to be added corresponding to the t -th column, and at least one of $C_0, C_1, \dots, C(col)$ is a non-zero integer. row is a number of rows of the coefficient matrix, col is a number of columns of the coefficient matrix, kb is equal to a difference between col and row , M is a non-negative integer less than 10, Z_{max} a maximum lifting size supported by the adjusted coefficient matrix, and Z_{max} is a positive integer.

[1511] Optionally, an element in an s -th row and a t -th column of the adjusted coefficient matrix is equal to an integer obtained from a modulo calculation result of an element in an s -th row and a t -th column of the original coefficient matrix plus $(X_s + Y_t)$ by $Z_{\max}$. The element in the s -th row and the t -th column of the original coefficient matrix is an element of non-“-1”, $s=1, 2, 3, \dots, row$, $t=1, 2, 3, \dots, kb+M$, kb is the difference between col and row , row is the number of rows of the coefficient matrix, col is the number of columns of the coefficient matrix, M is a non-negative integer less than 10, both X_s and Y_t are integers, and at least one of $X_0, X_1, \dots, X(row)$ and $Y_0, Y_1, \dots, Y(kb+M)$ is a non-zero integer.

[1512] Optionally, A is equal to 8, and the 8 lifting size subsets are respectively as follows:

[1513] $Z_{set1}=\{2, 4, 8, 16, 32, 64, 128, 256\}$, $Z_{set2}=\{3, 6, 12, 24, 48, 96, 192\}$, $Z_{set3}=\{5, 10, 20, 40, 80, 160\}$, $Z_{set4}=\{7, 14, 28, 56, 112, 224\}$, $Z_{set5}=\{9, 18, 36, 72, 144\}$, $Z_{set6}=\{11, 22, 44, 88, 176\}$, $Z_{set7}=\{13, 26, 52, 104, 208\}$, and $Z_{set8}=\{15, 30, 60, 120, 240\}$.

[1514] Optionally, the set Z_{0seti} are respectively $Z_{0set1}=\{56, 80, 104, 112, 120, 128, 160, 176, 192, 208, 224, 240, 256\}$, $Z_{0set2}=\{44, 48, 72, 88, 96, 144, 160, 176, 192, 208, 224, 240, 256\}$, $Z_{0set3}=\{60, 80, 120, 128, 160, 208, 224, 240, 256\}$, $Z_{0set4}=\{56, 60, 104, 112, 120, 128, 192, 208, 224, 240, 256\}$, $Z_{0set5}=\{72, 128, 144, 160, 176, 192, 224, 240, 256\}$, $Z_{0set6}=\{88, 96, 104, 128, 176, 192, 208, 224, 240, 256\}$, $Z_{0set7}=\{26, 52, 80, 104, 144, 160, 176, 192, 208, 224, 240\}$, and $Z_{0set8}=\{30, 60, 112, 120, 144, 160, 208, 224, 240, 256\}$.

[1515] Optionally, A is equal to 8, and the 8 lifting size subsets are respectively as follows:

[1516] $Z_{set1}=\{2, 4, 8, 16, 32, 64, 128, 256\}$, $Z_{set2}=\{3, 6, 12, 24, 48, 96, 192, 384\}$, $Z_{set3}=\{5, 10, 20, 40, 80, 160, 320\}$, $Z_{set4}=\{7, 14, 28, 56, 112, 224\}$, $Z_{set5}=\{9, 18, 36, 72, 144, 288\}$, $Z_{set6}=\{11, 22, 44, 88, 176, 352\}$, $Z_{set7}=\{13, 26, 52, 104, 208\}$, and $Z_{set8}=\{15, 30, 60, 120, 240\}$.

[1517] Optionally, the set Z_{0seti} are respectively $Z_{0set1}=\{16, 32, 48, 64, 80, 96, 112, 128, 144, 160, 176, 192, 208, 224, 240, 256, 288, 320, 352, 384\}$, $Z_{0set2}=\{12, 24, 36, 48, 60, 72, 96, 120, 144, 192, 240, 256, 288, 352, 384\}$, $Z_{0set3}=\{20, 40, 60, 80, 120, 160, 240, 320, 352\}$, $Z_{0set4}=\{56, 112, 224, 288, 352, 384\}$, $Z_{0set5}=\{18, 36, 72, 144, 176, 192, 288, 352, 384\}$, $Z_{0set6}=\{22, 44, 88, 176, 256, 320, 352, 384\}$, $Z_{0set7}=\{13, 26, 52, 104, 176, 208, 224, 240, 256, 288, 320, 352, 384\}$, and $Z_{0set8}=\{30, 60, 120, 240, 320, 352\}$.

[1518] Optionally, all lifting sizes corresponding to all basic matrices with the girth greater than or equal to 6 and corresponding to an encoding rate of Rate0 determined by the each lifting size in the lifting size set Zset and the i-th coefficient matrix in the coefficient matrix set constitute the set Z0seti, and all lifting sizes corresponding to all basic matrices with the girth greater than or equal to 6 and corresponding to an encoding rate of Rate1 determined by the each lifting size in the lifting size set Zset and the i-th coefficient matrix in the coefficient matrix set constitute a set Z0set1i; where Rate0 is less than Rate1, a number of elements in the set Z0set0i is less than or equal to a number of elements in the set Z0set1i, Rate0 and Rate1 are both real numbers greater than 0 and less than 1, and both the set Z0set0i and the set Z0set1i are non-empty sets.

[1519] It is to be noted that the various modules described above may be implemented by software or hardware. Implementation by hardware may, but may not necessarily, be performed in the following manners: the various modules described above are located in a same processor, or the various modules described above are located in their respective processors in any combination form.

[1520] Embodiment 3

[1521] This embodiment is an optional embodiment of the present invention, used for supplementing and describing the present application in detail. This embodiment includes examples described below.

[1522] Example 1

[1523] An embodiment of the present invention provides an example of a design method for quasi-cyclic low-density parity-check encoding. FIG. 3 is a flowchart of a design method for quasi-cyclic low-density parity-check encoding according to an example 1 of the present invention. As shown in FIG. 3, the method includes steps described below.

[1524] In step 302, a lifting size Z for the quasi-cyclic LDPC encoding is determined according to a length K of an information sequence to be encoded.

[1525] In step 304, a coefficient matrix used for the quasi-cyclic LDPC encoding is selected from a coefficient matrix set according to the determined lifting size Z.

[1526] In step 306, a basic matrix corresponding to the lifting size Z is obtained according to

the lifting size Z and the determined coefficient matrix. The basic matrix has mb rows and nb columns, and a maximum number of system columns of the basic matrix is $kb_{max}=nb-mb$.

[1527] In step 308, a parity check matrix for the quasi-cyclic LDPC encoding is acquired according to the basic matrix and the lifting size Z , and the information sequence to be encoded is encoded according to the parity check matrix to obtain an N -bit LDPC encoded sequence, where K is a positive integer, N is an integer greater than K , and Z is a positive integer.

[1528] Specifically, the step 302 in which the lifting size Z for the quasi-cyclic LDPC encoding is determined according to the length K of the information sequence to be encoded includes: selecting a minimum lifting size greater than or equal to K/kb from a lifting size set Z_{set} as the lifting size used for the current quasi-cyclic LDPC encoding, where kb is the determined maximum number of system columns corresponding to the length K of the information sequence to be encoded, kb is less than or equal to the maximum number kb_{max} of system columns of the basic matrix. The beneficial effect of this operation is that a minimum number of padding bits are required for the quasi-cyclic LDPC encoding and performance losses are prevented.

[1529] In step 304, the coefficient matrix set includes $A=8$ coefficient matrices, and lifting sizes supported by the 8 coefficient matrices have no common elements, so an index of a required coefficient matrix in the coefficient matrix set may be determined according to the determined lifting size Z , and the coefficient matrix used for the current LDPC encoding may be determined from the coefficient matrix set according to the index.

[1530] In step 306, the basic matrix corresponding to the lifting size Z may be calculated according to the determined lifting size Z and coefficient matrix. Preferably, a calculation method may be obtained by using the following calculation formula: kb , where $V_{i,j}$ is an element in an i -th row and a j -th column of the coefficient matrix corresponding to the lifting size Z and is an element of non-“-1”, $P_{i,j}$ is an element in an i -th row and a j -th column of the basic matrix corresponding to the lifting size Z , a function $f(a,b)$ is a remainder (modulo) operation, that is, an integer a is divided by an integer b for a remainder. For example, if $a=5$ and $b=10$, $f(a,b) = 5$; if $a=39$ and $b=32$, $f(a,b) = 7$.

[1531] In addition, the calculation formula may also be a floor operation. For example:

$P_{i,j} = f(V_{i,j}, Z, Z_{max}) = \text{floor}(V_{i,j} \times Z/Z_{max})$, where $P_{i,j}$ is related to three variables $V_{i,j}$, Z , and Z_{max} . Therefore, it may be found that the acquired coefficient matrix is practically a basic matrix corresponding to Z_{max} , where Z_{max} is a maximum lifting size supported by the coefficient matrix.

[1532] In step 308, the parity check matrix for the quasi-cyclic LDPC encoding may be obtained according to the acquired basic matrix and the lifting size Z , and the information sequence of length K to be encoded may be encoded according to the parity check matrix to obtain the LDPC encoded sequence. Before the quasi-cyclic LDPC encoding, the information sequence to be encoded needs to be filled with $kb \times Z - K$ padding bits to obtain the information sequence to be encoded of $kb \times Z$ bits. The quasi-cyclic LDPC encoding is performed on the information sequence of $kb \times Z$ bits filled with the dummy bits to obtain an original LDPC encoded sequence of $nb \times Z$ bits. Bit selection is performed on the original LDPC encoded sequence to acquire the N -bit LDPC encoded sequence corresponding to a code rate. A bit selection method is to select from a $(Z \times 2 + 1)$ -th bit and skip the padding bits to cyclically select the N -bit LDPC encoded sequence. The filled $kb \times Z - K$ dummy bits are adopted to assist in encoding and are known for both a transmitting end and a receiving end. Therefore, these bits do not need to be transmitted. The parity check matrix is not necessarily required for the quasi-cyclic LDPC encoding in the encoding process. Since the quasi-cyclic LDPC encoding has very regular construction and strong structurality, the sequence to be encoded may directly be encoded according to the basic matrix and the lifting size.

[1533] In a more specific example, the coefficient matrix set includes 8 coefficient matrices of 46 rows and 68 columns in the embodiment 1, and lifting size subsets supported by the 8 coefficient matrices are respectively $Z_{set1} = \{2, 4, 8, 16, 32, 64, 128, 256\}$, $Z_{set2} = \{3, 6, 12, 24, 48, 96, 192, 384\}$, $Z_{set3} = \{5, 10, 20, 40, 80, 160, 320\}$, $Z_{set4} = \{7, 14, 28, 56, 112, 224\}$, $Z_{set5} = \{9, 18, 36, 72, 144, 288\}$, $Z_{set6} = \{11, 22, 44, 88, 176, 352\}$, $Z_{set7} = \{13, 26, 52, 104, 208\}$, and $Z_{set8} = \{15, 30, 60, 120, 240\}$. All elements in all the lifting size subsets constitute a set of lifting sizes $Z_{set} = \{Z_{set1}, Z_{set2}, Z_{set3}, Z_{set4}, Z_{set5}, Z_{set6}, Z_{set7}, Z_{set8}\}$. The lifting size subset corresponding to a coefficient matrix 1 is Z_{set1} , the lifting size subset corresponding to a coefficient matrix 2 is Z_{set2} , the lifting size subset corresponding to a coefficient matrix 3 is

Zset3, the lifting size subset corresponding to a coefficient matrix 4 is Zset4, the lifting size subset corresponding to a coefficient matrix 5 is Zset5, the lifting size subset corresponding to a coefficient matrix 6 is Zset6, the lifting size subset corresponding to a coefficient matrix 7 is Zset7, and the lifting size subset corresponding to a coefficient matrix 8 is Zset8. According to a number $mb=46$ of rows and a number $nb=68$ of columns of the coefficient matrix, it may be known that $kbmax=68-46=22$, and $kb=kbmax=22$ in this example.

[1534] The LDPC encoding is performed on the information sequence of length $K=6000$ bits to be encoded to output the LDPC encoded sequence of length $N=12000$ bits. The method specifically includes steps described below.

[1535] 1. The minimum lifting size greater than or equal to $K/kb=6000/22$ is selected from the lifting size set Zset. It may be found that $Z=288$ satisfies the preceding condition.

[1536] 2. From the determined lifting size $Z=288$, it may be known that the lifting size $Z=288$ belongs to Zset5 and corresponds to a coefficient matrix index 5. Therefore, a fifth coefficient matrix is selected from the coefficient matrix set as the coefficient matrix corresponding to the lifting size Z .

[1537] 3. The basic matrix corresponding to the lifting size Z is obtained according to the lifting size Z and the determined coefficient matrix. The basic matrix corresponding to the lifting size $Z=288$ is obtained by using the remainder calculation formula described above. Since the lifting size $Z=288$ is the maximum value in a fifth lifting size subset, the basic matrix is equal to the coefficient matrix and has $mb=46$ rows and $nb=68$ columns.

[1538] 4. Since the number of system columns of the basic matrix acquired above is 22, that is, $22 \times 288 - 6000 = 336$ dummy bits need to be filled to perform the quasi-cyclic LDPC encoding, and the length of the information sequence to be encoded becomes $22 \times 288 = 6336$ bits. The parity check matrix for the quasi-cyclic LDPC encoding is acquired according to the basic matrix and the lifting size $Z=288$, the information sequence to be encoded is encoded according to the parity check matrix to obtain the original LDPC encoded sequence of $nb \times Z = 19584$ bits, and the LDPC encoded sequence of $N=12000$ bits is selected from the original LDPC encoded sequence.

[1539] Example 2

[1540] An embodiment of the present invention provides an example of a design apparatus

for quasi-cyclic low-density parity-check encoding. FIG. 4 is a block diagram of a design apparatus for quasi-cyclic low-density parity-check encoding according to an example 2 of the present invention. As shown in FIG. 4, the apparatus includes a determination module 40, a selection module 42, an acquisition module 44, and a processing module 46.

[1541] The determination module 40 is configured to determine a lifting size Z for the quasi-cyclic LDPC encoding according to a length K of an information sequence to be encoded.

[1542] The selection module 42 is configured to select a coefficient matrix used for the quasi-cyclic LDPC encoding from a coefficient matrix set according to the determined lifting size Z .

[1543] The acquisition module 44 is configured to acquire a basic matrix corresponding to the lifting size Z according to the lifting size Z and the determined coefficient matrix. The basic matrix has mb rows and nb columns, and a maximum number of system columns of the basic matrix is $kb_{max}=nb-mb$.

[1544] The processing module 46 is configured to acquire a parity check matrix for the quasi-cyclic LDPC encoding according to the basic matrix and the lifting size Z , and perform encoding on the information sequence to be encoded according to the parity check matrix to obtain an N -bit LDPC encoded sequence, where K is a positive integer, N is an integer greater than K , and Z is a positive integer.

[1545] Specifically, the determination module 40 determines the lifting size Z for the quasi-cyclic LDPC encoding according to the length K of the information sequence to be encoded by selecting a minimum lifting size greater than or equal to K/kb from a lifting size set Z_{set} as the lifting size used for the current quasi-cyclic LDPC encoding, where kb is the determined maximum number of system columns corresponding to the length K of the information sequence to be encoded, kb is less than or equal to the maximum number kb_{max} of system columns of the basic matrix. The beneficial effect of this operation is that a minimum number of padding bits are required for the quasi-cyclic LDPC encoding and performance losses are prevented.

[1546] For the selection module 42, the coefficient matrix set includes $A=8$ coefficient matrices, and lifting sizes supported by the 8 coefficient matrices have no common elements, so

an index of a required coefficient matrix in the coefficient matrix set may be determined according to the determined lifting size Z , and the coefficient matrix used for the current LDPC encoding may be determined from the coefficient matrix set according to the index.

[1547] For the acquisition module 44, the basic matrix corresponding to the lifting size Z may be calculated according to the determined lifting size Z and coefficient matrix. Preferably, a calculation method may be obtained by using the following calculation formula: kb , where $V_{i,j}$ is an element in an i -th row and a j -th column of the coefficient matrix corresponding to the lifting size Z and is an element of non-“1”, $P_{i,j}$ is an element in an i -th row and a j -th column of the basic matrix corresponding to the lifting size Z , a function $f(a, b)$ is a remainder (modulo) operation, that is, an integer a is divided by an integer b for a remainder. For example, if $a=5$ and $b=10$, $f(a, b) = 5$; if $a=39$ and $b=32$, $f(a, b) = 7$.

[1548] In addition, the calculation formula may also be a floor operation. For example: $P_{i,j} = f(V_{i,j}, Z, Z_{max}) = \text{floor}(V_{i,j} \times Z/Z_{max})$, where $P_{i,j}$ is related to three variables $V_{i,j}$, Z , and Z_{max} . Therefore, it may be found that the acquired coefficient matrix is practically a basic matrix corresponding to Z_{max} , where Z_{max} is a maximum lifting size supported by the coefficient matrix.

[1549] For the processing module 46, the parity check matrix for the quasi-cyclic LDPC encoding may be obtained according to the acquired basic matrix and the lifting size Z , and the information sequence of length K to be encoded may be encoded according to the parity check matrix to obtain the LDPC encoded sequence. Before the quasi-cyclic LDPC encoding, the information sequence to be encoded needs to be filled with $kb \times Z - K$ dummy bits to obtain the information sequence to be encoded of $kb \times Z$ bits. The quasi-cyclic LDPC encoding is performed on the information sequence of $kb \times Z$ bits filled with the dummy bits to obtain an original LDPC encoded sequence of $nb \times Z$ bits. Bit selection is performed on the original LDPC encoded sequence to acquire the N -bit LDPC encoded sequence corresponding to a code rate. A bit selection method is to select from a $(Z \times 2 + 1)$ -th bit and skip the padding bits to cyclically select the N -bit LDPC encoded sequence. The filled $kb \times Z - K$ dummy bits are adopted to assist in encoding and are known for both a transmitting end and a receiving end. Therefore, these bits do not need to be transmitted. The parity check matrix is not necessarily required for the

quasi-cyclic LDPC encoding in the encoding process. Since the quasi-cyclic LDPC encoding has very regular construction and strong structurality, the sequence to be encoded may directly be encoded according to the basic matrix and the lifting size.

[1550] In a more specific example, the coefficient matrix set includes 8 coefficient matrices of 46 rows and 68 columns in the embodiment 1, and lifting size subsets supported by the 8 coefficient matrices are respectively $Z_{set1}=\{2, 4, 8, 16, 32, 64, 128, 256\}$, $Z_{set2}=\{3, 6, 12, 24, 48, 96, 192, 384\}$, $Z_{set3}=\{5, 10, 20, 40, 80, 160, 320\}$, $Z_{set4}=\{7, 14, 28, 56, 112, 224\}$, $Z_{set5}=\{9, 18, 36, 72, 144, 288\}$, $Z_{set6}=\{11, 22, 44, 88, 176, 352\}$, $Z_{set7}=\{13, 26, 52, 104, 208\}$, and $Z_{set8}=\{15, 30, 60, 120, 240\}$. All elements in all the lifting size subsets constitute a set of lifting sizes $Z_{set}=\{Z_{set1}, Z_{set2}, Z_{set3}, Z_{set4}, Z_{set5}, Z_{set6}, Z_{set7}, Z_{set8}\}$. The lifting size subset corresponding to a coefficient matrix 1 is Z_{set1} , the lifting size subset corresponding to a coefficient matrix 2 is Z_{set2} , the lifting size subset corresponding to a coefficient matrix 3 is Z_{set3} , ..., and the lifting size subset corresponding to a coefficient matrix 8 is Z_{set8} . According to a number $mb=46$ of rows and a number $nb=68$ of columns of the coefficient matrix, it may be known that $kb_{max}=68-46=22$, and $kb=kb_{max}=22$ in this example.

[1551] The LDPC encoding is performed on the information sequence of length $K=6000$ bits to be encoded to output the LDPC encoded sequence of length $N=12000$ bits. The method specifically includes steps described below.

[1552] 1. The minimum lifting size greater than or equal to $K/kb=6000/22$ is selected from the lifting size set Z_{set} . It may be found that $Z=288$ satisfies the preceding condition.

[1553] 2. From the determined lifting size $Z=288$, it may be known that the lifting size $Z=288$ belongs to Z_{set5} and corresponds to a coefficient matrix index 5. Therefore, a fifth coefficient matrix is selected from the coefficient matrix set as the coefficient matrix corresponding to the lifting size Z .

[1554] 3. The basic matrix corresponding to the lifting size Z is obtained according to the lifting size Z and the determined coefficient matrix. The basic matrix corresponding to the lifting size $Z=288$ is obtained by using the remainder calculation formula described above. Since the lifting size $Z=288$ is the maximum value in a fifth lifting size subset, the basic matrix is equal to the coefficient matrix and has $mb=46$ rows and $nb=68$ columns.

[1555] 4. Since the number of system columns of the basic matrix acquired above is 22, that is, $22 \times 288 - 6000 = 336$ dummy bits need to be filled to perform the quasi-cyclic LDPC encoding, and the length of the information sequence to be encoded becomes $22 \times 288 = 6336$ bits. The parity check matrix for the quasi-cyclic LDPC encoding is acquired according to the basic matrix and the lifting size $Z=288$, the information sequence to be encoded is encoded according to the parity check matrix to obtain the original LDPC encoded sequence of $nb \times Z = 19584$ bits, and the LDPC encoded sequence of $N=12000$ bits is selected from the original LDPC encoded sequence.

[1556] In the technical solutions in this embodiment, the design method for quasi-cyclic LDPC encoding and the corresponding apparatus may effectively solve the problem of poor performance of a quasi-cyclic LDPC code and improve quasi-cyclic LDPC encoding performance.

[1557] Embodiment 4

[1558] An embodiment of the present invention further provides a storage medium. Optionally, in this embodiment, the storage medium may be configured to store program codes for executing a step described below.

[1559] In S1, according to a parity check matrix of a quasi-cyclic low-density parity-check (LDPC) code, LDPC encoding is performed on a K-bit information sequence to be encoded to obtain an N-bit LDPC encoded sequence.

[1560] Optionally, in this embodiment, the storage medium may include, but is not limited to, a USB flash disk, a read-only memory (ROM), a random access memory (RAM), a mobile hard disk, a magnetic disk, an optical disk or another medium capable of storing program codes.

[1561] Optionally, in this embodiment, a processor performs, according to the program codes stored in the storage medium, the following step: according to a parity check matrix of a quasi-cyclic low-density parity-check (LDPC) code, LDPC encoding is performed on a K-bit information sequence to be encoded to obtain an N-bit LDPC encoded sequence.

[1562] Optionally, for specific examples in this embodiment, reference may be made to the examples described in the above-mentioned embodiments and optional implementation modes, and repetition will not be made in this embodiment.

[1563] Apparently, it should be understood by those skilled in the art that each of the

above-mentioned modules or steps of the present invention may be implemented by a general-purpose computing apparatus, the modules or steps may be concentrated on a single computing apparatus or distributed on a network composed of multiple computing apparatuses, and alternatively, the modules or steps may be implemented by program codes executable by the computing apparatus, so that the modules or steps may be stored in a storage apparatus and executed by the computing apparatus. In some circumstances, the illustrated or described steps may be executed in sequences different from those described herein, or the modules or steps may be made into various integrated circuit modules separately, or multiple modules or steps therein may be made into a single integrated circuit module for implementation. In this way, the present invention is not limited to any specific combination of hardware and software.

[1564] The above are only preferred embodiments of the present invention and are not intended to limit the present invention. For those skilled in the art, the present invention may have various modifications and variations. Any modifications, equivalent substitutions, improvements and the like made within the spirit and principle of the present invention should fall within the scope of the present invention.

[1565] It should be understood by those skilled in the art that the embodiments of the present invention may be provided as methods, systems, or computer program products. Therefore, the present invention may adopt a form of a hardware embodiment, a software embodiment, or a combination of hardware and software embodiments. In addition, the present invention may adopt a form of a computer program product implemented on one or more computer-usable storage media (including, but not limited to, a disk memory, an optical memory, and the like) which include computer-usable program codes.

[1566] The present invention is described with reference to flowcharts and/or block diagrams of methods, devices (systems) and computer program products according to the embodiments of the present invention. It should be understood that computer program instructions may implement each flow and/or block in the flowcharts and/or block diagrams and a combination of flows and/or blocks in the flowcharts and/or block diagrams. These computer program instructions may be provided to a general-purpose computer, a special-purpose computer, an embedded processor or a processor of another programmable data processing device to produce a

machine so that instructions executed by a computer or the processor of another programmable data processing device produce a means for implementing functions specified in one or more flows in the flowcharts and/or one or more blocks in the block diagrams.

[1567] These computer program instructions may also be stored in a computer-readable memory which may direct the computer or another programmable data processing device to operate in a particular manner so that the instructions stored in the computer-readable memory produce a manufactured product including an instructing means. The instructing means implements the functions specified in one or more flows in the flowcharts and/or one or more blocks in the block diagrams.

[1568] These computer program instructions may also be loaded onto the computer or another programmable data processing device so that a series of operation steps are performed on the computer or another programmable device to produce processing implemented by the computer. Therefore, instructions executed on the computer or another programmable device provide steps for implementing the functions specified in one or more flows in the flowcharts and/or one or more blocks in the block diagrams.

[1569] The above are only preferred embodiments of the present invention and are not intended to limit the scope of the present invention.

INDUSTRIAL APPLICABILITY

[1570] In the embodiments of the present invention, the LDPC encoding is performed on the K-bit information sequence to be encoded according to the parity check matrix of the quasi-cyclic LDPC code, where the parity check matrix is determined according to the basic matrix and the lifting size Z, and the basic matrix is determined according to the lifting size Z and the coefficient matrix. The present invention solves the problem in the related art of poor performance of the quasi-cyclic LDPC code and effectively improves quasi-cyclic LDPC encoding performance.

1. A method for wireless communication, comprising:
 - performing low-density parity-check, LDPC, encoding on an input sequence having K bits according to a parity check matrix of an LDPC code; and
 - obtaining an encoded sequence having N bits;
 - wherein the parity check matrix is determined based on at least a basic matrix that is determined according to a lifting size Z and a coefficient matrix in a set of coefficient matrices, wherein K is a positive integer, N is an integer greater than K, and Z is a positive integer,
 - wherein all coefficient matrices in the set of coefficient matrices have $m_b = 42$ rows and $n_b = 52$ columns, wherein the lifting size Z is determined to be a minimum lifting size in a lifting size set Zset such that $Z \geq K/k_b$, k_b being equal to $(n_b - m_b) = 10$,
 - wherein elements in the lifting size set Zset are grouped in A lifting size subsets, wherein the coefficient matrix corresponds to a 5th subset of the A lifting size subsets that the lifting size Z belongs to, A being an integer greater than 1, and wherein elements of the coefficient matrix comprise:

[illegible]

4. The method of claim 1 or 2, wherein the parity check matrix is determined further based on a second matrix that has a same number of rows and a same number of columns as the coefficient matrix.

5. The method of claim 1 or 2, wherein the parity check matrix is further determined based on the lifting size Z.

6. A computer-readable storage medium for storing program codes, wherein, when being executed, the program codes implement the method of any one of claims 1 to 5.

7. An apparatus for wireless communications, comprising:

a processor; and

a memory including processor-executable instructions stored thereon, the processor-executable instructions upon execution by the processor configures the processor to:

perform low-density parity check, LDPC, encoding on an input sequence having K bits according to an LDPC code;

obtain an encoded sequence having N bits;

wherein the parity check matrix is determined based on at least a basic matrix that is determined according to a lifting size Z and a coefficient matrix in a set of coefficient matrices, wherein K is a positive integer, N is an integer greater than K , and Z is a positive integer,

wherein all coefficient matrices in the set of coefficient matrices have $mb = 42$ rows and $nb = 52$ columns, wherein the lifting size Z is determined to be a minimum lifting size in a lifting size set Z_{set} such that $Z \geq K/kb$, kb being equal to $(nb - mb) = 10$,

wherein elements in the lifting size set Z_{set} are grouped in A lifting size subsets, wherein the coefficient matrix corresponds to a 5th subset of the A lifting size subsets that the lifting size Z belongs to, A being an integer greater than 1, and wherein elements of the coefficient matrix comprise:

[illegible]

-1,80,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,139,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1;
32,-1,-1,-1,-1,89,-1,-1,-1,-1,-1,-1,71,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1;-

-1,-1,135,-1,-1,-1,-1,6,-1,-1,2,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1;-

37,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,25,114,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1;

-1,60,-1,-1,-1,137,-1,-1,-1,-1,-1,93,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1,-1,-1,-1,-1;

121,-1,129,-1,-1,-1,-1,26,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1,-1;-

-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,97,-1,-1,56,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1,-1;-

-1,1,-1,-1,-1,70,-1,-1,-1,-1,-1,1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1,-1;-

119,-1,-1,-1,-1,-1,-1,32,-1,-1,-1,-1,142,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1,-1;-

-1,-1,6,-1,-1,-1,-1,-1,-1,73,-1,-1,102,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,0,-1;-

-1,48,-1,-1,-1,47,-1,-1,-1,-1,-1,19,-1,-
-1,0;-

wherein -1 represents a null value.

8. An apparatus for wireless communications, comprising:
 - a processor; and
 - a memory including processor-executable instructions stored thereon, the processor-executable instructions upon execution by the processor configures the processor to:
 - perform low-density parity-check, LDPC, encoding on an input sequence having K bits according to an LDPC code;
 - obtain an encoded sequence having N bits;

