the check matrix needs to be performed on the first code word, a code element in the first code word may be updated in the processing process, and if an updated code element exceeds the code element quantization range, this update needs to be abandoned. That is, in this aspect, if an update corresponding to a code element exceeds the code element quantization range, the update corresponding to the code element is abandoned.

[0018] Optionally, in this aspect, the determining whether each second update value of the P second update values exceeds a code element quantization range may be concurrently determining whether each second update value of the P second update values exceeds the code element quantization range, or may be sequentially determining, from left to right according to a sequence of the P first code elements corresponding to the nth row of the check matrix, whether each second update value of the P second update values exceeds the code element quantization range.

[0019] Optionally, in this aspect, the generating a second code word according to the P second code elements corresponding to the nth row of the check matrix includes: generating the second code word according to the P second code elements corresponding to the nth row of the check matrix and another first code element that is in the first code word and unrelated to the nth row of the check matrix.

[0020] Optionally, in this aspect, the generating a second code word according to the P second code elements corresponding to the nth row of the check matrix includes: generating the second code word according to P second code elements corresponding to each row of the check matrix.

[0021] According to another aspect, an embodiment of the present application provides a decoding method, where the decoding method is used to decode an LDPC code and includes: acquiring a first code word, for example, −9, +7, −12, +4, +7, +10, −11, where the first code word is generated by an LDPC code encoding device according to a generator matrix and includes M first code elements, M is a positive integer and M in the previous example is 7; acquiring P first code elements corresponding to the nth row of a check matrix, where for example, when three elements on the first row of the check matrix are 1, P is 3; acquiring P first update values corresponding to the P first code elements; obtaining, according to the P first update values, P second update values corresponding to the P first code elements; determining whether each second update value of the P second update values exceeds a code element quantization range; keeping the P first code elements not updated

DESCRIPTION OF EMBODIMENTS

[0040] To make the objectives, technical solutions, and advantages of the embodiments of the present invention clearer, the following clearly and completely describes the technical solutions in the embodiments of the present invention with reference to the accompanying drawings in the embodiments of the present invention. Apparently, the described embodiments are some but not all of the embodiments of the present invention. All other embodiments obtained by persons of ordinary skill in the art based on the embodiments of the present invention without creative efforts shall fall within the protection scope of the present invention.

[0041] Throughout this specification, the term "low-density parity-check code" (full name in English: Low Density Parity Check Code) refers to a type of linear error-correcting code and is usually used for transmitting a signal through a communications channel or for storing data in a storage medium in the communications field.

[0042] Throughout this specification, the term "code word" (full name in English: Code Word) refers to a signal including a group of code elements (full name in English: Code Element) transmitted through a communications channel or stored in a storage device. For example, in a case in which a code word of an LDPC code has a length of 7, the code word includes 7 code elements.

[0043] Throughout this specification, the term "code element quantization range" may also be referred to as a variable node (Variable Node) quantization range. In a process of decoding an LDPC code, each code element of a to-be-decoded code corresponds to one variable node, and each row of a check matrix corresponds to one check node (Check Node). For details, refer to a Tanner diagram in FIG. 6. A code element quantization range is also a size of address space allocated by an LDPC code decoding device to each code element. For example, if address space allocated to each code element is 6 bits, a code element quantization range is from −31 to +31 (herein after including -31 and +31).

LDPC code encoding and decoding architecture in embodiments of the present invention

[0044] FIG. 1 depicts an LDPC code encoding and decoding architecture

quantization range; if the processing unit 204 determines that the ath second update value exceeds the code element quantization range, the processing unit 204 does not update the ath first code element by using the ath second update value, that is, keeps the ath first code element not updated; and if the processing unit 204 determines that the bth second update value does not exceed the code element quantization range, the processing unit 204 updates the bth first code element by using the bth second update value.

[0054] According to the foregoing example, because P is equal to 3 in this example, a value of a may be 1, 2, or 3 and corresponds to a second update value −5, +3, or −3, respectively. For example, a is set to 1, and the processing unit 204 determines whether the second update value −5 exceeds the code element quantization range; if the second update value −5 exceeds the code element quantization range, −9 is not updated by using −5, that is, the code element −9 keeps unchanged; if the second update value −5 does not exceed the code element quantization range, −9 is updated by using −5, that is, the code element −9 is updated to −5. The processing unit 204 sequentially processes the first code elements −9, +7, and +4 according to the method. For example, the code element quantization range in this embodiment is from −15 to +15 (herein after including -15 and +15). Then, none of the second update values exceeds the code element quantization range. The processing unit 204 generates second code elements after completing the processing corresponding to the first row of the check matrix. The second code elements include −5, +3, and −3. The generation of each code element of the second code elements is to maintain a corresponding first code element (if a corresponding second update value exceeds the code element quantization range) or to update a corresponding first code element by using a corresponding second update value (if the corresponding second update value does not exceed the code element quantization range).

[0055] The processing unit 204 generates the second code word according to the second code elements. Mainly two methods for generating the second code word are shown as follows:

[0056] Method 1: A second code word is generated by updating, to a second code element, a code element (that is, the first code element) that is in the first code word

Device embodiment 3

[0087] Device embodiment 3 provides a decoding device 600. FIG. 5 shows a schematic diagram of a composite structure of the decoding device 600. In this embodiment, an internal composite structure of the processing unit 204 in device embodiment 1 is detailed as a processing unit 604. An overall function of the processing unit 604 in this embodiment is the same as that of the processing unit 204 in device embodiment 1.

[0088] The following describes in detail an operation principle of the decoding device 600 proposed in device embodiment 3 by using an example in which a code word acquired by the decoding device 600 is −9, =±7, −12, =±4, +7, +10, −11, a code element quantization range is from −15 to +15, and the following check matrix is used by the decoding device 600.

$$\left|\begin{array}{ccccccc} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right| \begin{array}{ccccccc} \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{1} \end{array}$$

[0089] After acquiring the code word −9, +7, −12, +4, +7, +10, −11, an acquiring unit 602 first sends the code word to a determining unit 606, and the determining unit 606 maps negative values in the code word to 1 and maps positive values to 0 to obtain a mapped code word 1, 0, 1, 0, 0, 0, 1, and performs an exclusive OR operation, also referred to as a sum modulo 2 or an addition modulo 2 operation, on each code element in the mapped code word according to a check equation. According to an operation on the first row of the check matrix, (1+0+0) mod (2)=1, and the code word fails to be decoded. If the code word acquired by the decoding device 600 passes decision of the determining unit 606 before the code word is processed by the processing unit 604, the code word does not need subsequent processing, and the decoding device 600 directly uses a mapping result of the code word as a decoding result. The code word 1, 0, 1, 0, 0, 0, 1 needs to pass check corresponding to all three rows of the check matrix before the determining unit 606 determines that the code word is decoded successfully.

[0090] A variable node processing unit 6042 receives the code word −9, +7, −12, +4, +7, +10, −11 and stores −9, +7, −12, +4, +7, +10, −11 to a storage module 6043. It can be learned from the first row of the check matrix that the first, the second, and the fourth variable nodes are related to the first row of the check matrix, that is, the

code element quantization range, that is, from −15 to +15, abandons the update of the code element corresponding to the variable node, reads, from the storage module 6043, the code element before the update, and continues to update a code element corresponding to a next variable node; after updates of the first row of the check matrix are complete, continues to update the second row of the check matrix, that is, perform update processing corresponding to a check node 2 in a Tanner diagram shown in FIG. 6. In FIG. 6, a variable node 1 to a variable node 7 correspond to seven code elements received by the variable node processing unit 6042 respectively, and a check node 1 to a check node 3 correspond to the first row to the third row of the check matrix respectively. In FIG. 6, a processing action performed by the variable node 1 to the variable node 7 is also performed by the variable node processing unit 6042 in FIG. 5, and a processing action performed by the check node 1 to the check node 3 is also performed by the check node processing unit 6044 in FIG. 5.

[0095] After the check node 1 returns update values to the variable node 1, the variable node 2, and the variable node 4, if none of updated code elements corresponding to the variable node 1, the variable node 2, and the variable node 4 exceeds the code element quantization range, the variable node 3, the variable node 4, and the variable node 5 continues to send corresponding code elements to the check node 2, and the check node 2 generates corresponding update values and then returns the update values to corresponding variable nodes, and if the code element quantization range is still not exceeded, an update corresponding to the check node 3 continues. In this case, because the code element corresponding to the variable node 4 participates in the processing corresponding to the check node 1 and also participates in the processing corresponding to the check node 2 and the processing corresponding to the check node 3, in FIG. 6, if a sum of the code element +4 corresponding to the variable node 4 and an update value −7 returned by the check node 1 does not exceed the code element quantization range, a sum of a code element corresponding to a variable node and an update value returned by another check node may exceed the code element quantization range. In an example shown in FIG. 6, the updates returned by the check node 1, the check node 2, and the check node 3 to the variable node 4 are −7, −7, and −10 respectively. After the three update values are added to the code ~~word~~ element +4, obtained −20 exceeds the code element quantization range. Because all update values are already stored, the three update values are sorted to obtain an update value that has the greatest impact on the exceeding of the code element quantization range.

Because the code ~~word~~ element in this example exceeds a minimum value of the code element quantization range, the update value with the greatest impact is an update value with a smallest value, that is, −10, the variable node 4 abandons the update value −10 returned by the check node 3, that is, after this round of update, a code ~~word~~ element corresponding to the variable node 4 is +4 + (−7) + (−7), and equals to −10. In solution 2, after a sum of a code element of any variable node and update values returned by multiple check nodes exceeds the code element quantization range, the update values received by the variable node are sorted; when the sum of the code element of the variable node and the update values exceeds a maximum value of the code element quantization range, the variable node first abandons a largest value of the update values; and when the sum of the code element of the variable node and the update values exceeds a minimum value of the code element quantization range, the variable node first abandons a smallest value of the update values, which ensures that the abandoned update value has least impact on a decoding process.

[0096] Each time the variable node processing unit 6042 completes updating code elements corresponding to one row of the check matrix, that is, processing an update value sent by one check node, the variable node processing unit 6042 sends an updated code word to the determining unit 606, and the determining unit 606 maps the updated code word and determines whether this row of code word passes decision. In the previous example, after updates of code elements corresponding to variable nodes corresponding to the first row of the check matrix are complete, an updated code word is −5, +3, −12, −3, +7, +10, −11, and the determining unit 606 performs decision on the updated code word. For details, refer to how the processing unit 204 generates the second code word by using method 1 in device embodiment 1. After updating the code elements corresponding to one row of the check matrix and acquiring the updated code word, the variable node processing unit 6042 may further continue to generate the second code word by using method 2 used by the processing unit 204 in device embodiment 1, and send the second code word to the determining unit 606. It should be noted that, before performing mapping and decision on the code word, the determining unit 606 may further perform minimum offset algorithm processing on the code word, or may further perform normalized min-sum algorithm

processing on the code word.

[0097] This embodiment provides a decoding device. The decoding device performs processing in a case that occurs in a code word update process and in which a value of an updated code element exceeds a code element quantization range, which prevents the decoding device from directly quantizing an updated code word that exceeds the code element quantization range, and improves an error correction capability of the decoding device in a decoding process.

Device embodiment 4

[0098] Device embodiment 4 provides a decoding device 800. FIG. 7 shows a schematic diagram of a composite structure of the decoding device 800. In this embodiment, an internal composite structure of the processing unit 404 in device embodiment 2 is detailed as a processing unit 804. An overall function of the processing unit 804 in this embodiment is the same as that of the processing unit 404 in device embodiment 2.

[0099] The following describes in detail an operation principle of the decoding device 800 proposed in device embodiment 4 by using an example in which a code word acquired by the decoding device 800 is −9, =±7, −12, =±4, +7, +10, −11, a code element quantization range is from −15 to +15, and the following check matrix is used by the decoding device 800.

$$\begin{vmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{vmatrix}$$

[0100] After acquiring the code word −9, +7, −12, +4, +7, +10, −11, an acquiring unit 802 first sends the code word to a determining unit 806, and the determining unit 806 maps negative values in the code word to 1 and maps positive values to 0 to obtain a mapped code word 1, 0, 1, 0, 0, 0, 1, and performs an exclusive OR operation, also referred to as a sum modulo 2 or an addition modulo 2 operation, on each code element in the mapped code word according to a check equation. According to an operation on the first row of the check matrix, (1+0+0) mod (2)=1, and the code word fails to be decoded. If the code word acquired by the decoding device 800 passes decision of the determining unit 806 before the code word is processed by the processing unit 804, the code word does not need subsequent processing, and the decoding device 800 directly uses a mapping result of the code word as a decoding

may be sequentially determining, from left to right according to a sequence of the P code elements corresponding to the nth row of the check matrix, whether each second update value of the P second update values exceeds the code element quantization range.

[0142] Optionally, the acquiring unit 202, the processing unit 204, and the determining unit 206 that are included in the decoding device 200 each are implemented by using a circuit.

[0143] After acquiring the second update values, the processing unit 204 determines whether each second update value exceeds the code element quantization range; if the processing unit 204 determines that the ath second update value exceeds the code element quantization range, the processing unit 204 does not update the ath first code element by using the ath second update value, that is, keeps the ath first code element not updated; if the processing unit 204 determines that the bth second update value does not exceed the code element quantization range, the processing unit 204 updates the bth first code element by using the bth second update value.

[0144] According to the example in device embodiment 1, because P is equal to 3 in this example, a value of a may be 1, 2, or 3 and corresponds to a second update value −5, +3, or −3, respectively. For example, a is set to 1; the processing unit 204 determines whether the second update value −5 exceeds the code element quantization range; if the second update value −5 exceeds the code element quantization range, −9 is not updated by using −5, that is, the code element −9 keeps unchanged; if the second update value −5 does not exceed the code element quantization range, −9 is updated by using −5, that is, the code element −9 is updated to −5. The ~~processor~~ processing unit 204 may concurrently determine whether the second update values −5, +3, and −3 exceed the code element quantization range, or may determine whether the second update values −5, +3, and −3 exceed the code element quantization range according to a sequence of first code elements corresponding to a bit 1 on the first row of the check matrix, that is, first −5, then +3, and then −3.

[0145] In device embodiment 2, the processing unit 404 is configured to acquire P first code elements corresponding to the nth row of a check matrix; acquire P first update values corresponding to the P first code elements; obtain, according to the P

range, the processing unit 404 updates the P first code elements by using the P second update values.

[0149] According to the example in device embodiment 1, because P is equal to 3 in this example, a value of a may be 1, 2, or 3 and corresponds to a second update value −5, +3, or −3, respectively. For example, a is set to 1; the processing unit 404 determines whether the second update value −5 exceeds the code element quantization range; if the second update value −5 exceeds the code element quantization range, −9 is not updated by using −5, −4 is not updated by using +3, and −7 is not updated by using −3; similarly, if another second update value exceeds the code element quantization range, updating the first code elements by using any one of the second update values is abandoned; if none of the second update values −5, +3, and −3 exceeds the code element quantization range, −9 is updated by using −5, −4 is updated by using +3, and −7 is updated by using −3. The ~~processor~~ processing unit 404 may concurrently determine whether the second update values −5, +3, and −3 exceed the code element quantization range, or may determine second update values −5, +3, and −3 according to a sequence of first code elements corresponding to a bit 1 on the first row of the check matrix, that is, first −5, then +3, and then −3, and if any one of the first code elements exceeds the code element quantization range, determining is not required for remaining first code elements.

[0150] In device embodiment 3, the variable node processing unit 6042 is implemented by using a circuit, and the check node processing unit 6044 is implemented by using a circuit.

[0151] In device embodiment 4, the variable node processing unit 8042 is implemented by using a circuit, and the check node processing unit 8044 is implemented by using a circuit.

[0152] In method embodiment 1, step 1004: acquire P first code elements corresponding to the nth row of a check matrix; acquire P first update values corresponding to the P first code elements; obtain, according to the P first update values, P second update values corresponding to the P first code elements; determine whether each second update value of the P second update values exceeds a code