On the Computation of the Minimum Distance of Low-Density Parity-Check Codes

Xiao–Yu Hu† and Marc P.C. Fossorier‡

† IBM Research, Zurich Research Laboratory, 8803 Rüschlikon, Switzerland

Email: xhu@zurich.ibm.com

‡ Dept. Electrical Engineering, Univ. Hawaii at Manoa, Honolulu, HI 96822, USA

Email: marc@spectra.eng.hawaii.edu

***Abstract*— Low-density parity-check (LDPC) codes in their broader-sense definition are linear codes whose parity-check matrices have fewer 1s than 0s. Finding their minimum distance is therefore in general an NP-hard problem; in other words there exists no known polynomial deterministic algorithm to compute the minimum distance of a particular, nontrivial LDPC code. We propose a randomized algorithm called the approximately nearest codewords (ANC) searching approach to attack this hard problem for iteratively decodable LDPC codes. The principle of the ANC searching approach is to search codewords locally around the all-zero codeword perturbed by a minimum level of noise, anticipating that the resultant nearest nonzero codewords will most likely contain the minimum-Hamming-weight codeword whose Hamming weight is equal to the minimum distance of the linear code. The effectiveness of the algorithm is demonstrated by numerous examples.**

Keywords

minimum distance, LDPC codes, algorithm, NP-hardness

I. INTRODUCTION

LOW-DENSITY parity-check (LDPC) codes were originally introduced by Gallager [1]. They have again become interesting because of the success of iterative decoding for Turbo codes [2]. LDPC codes are competitors of these codes, as their performance with iterative decoding closely approaches the Shannon limit [3-5]. Tanner's graphical representation of LDPC codes [6], factor graph [7], and Forney's normal graph [8] are powerful tools for analyzing codes on graphs and their associated iterative decoding algorithms, such as the sum-product algorithm (SPA).

A binary (n, k) linear code $\mathcal{C}$ is a set of binary vectors of length n, called codewords, that satisfy a set of $m \geq (n-k)$ parity check equations. A parity check equation can be conveniently represented as a binary vector of length n in which a 1 in position j, $1 \leq j \leq n$, implies that the jth codeword symbol participates

in the parity check. The set of m parity checks can then be represented by an $m \times n$ binary parity-check matrix H. A 1 in the (i, j)th position in H, $h_{ij} = 1$, indicates that the ith parity check involves the jth symbol of the codeword or, equivalently, that codeword symbol j participates in the ith parity check. This suggests a natural graphical representation of the parity-check matrix H as a bipartite graph with two kinds of nodes: n symbol nodes corresponding to the codeword symbols, and m check nodes corresponding to the parity checks represented by the rows of the matrix H. The connectivity of the bipartite graph, known as the Tanner graph, is such that the parity-check matrix H is its incidence matrix, i.e., for each 1 in the (i, j)th position, the graph has an edge connecting check node i with symbol node j.

A binary (n, k) LDPC code, in its broader definition, is a linear block code described by a sparse $m \times n$ parity-check matrix H, i.e., H has fewer 1s than 0s. Each symbol is checked by a small number of parity checks and each parity check includes a small number of symbols. An LDPC code is called (d_s, d_c)-regular if in the corresponding bipartite graph, every symbol node is connected to d_s check nodes and every check node is connected to d_c symbol nodes; otherwise it is called an irregular LDPC code.

There are several typical classes of methods to construct good Tanner graphs or LDPC codes. One is the class of random constructions, which is used most widely [4, 9]. Another one is the class of geometry/graph theoretic constructions [10-14]. A third class is the progressive-edge-growth (PEG) construction [15-17] aiming at large girth, irregularity and easy encoding. All these codes exhibit excellent empirical performance with the widespread practice of simulating performance only down to a block (bit) error probability of 10^{-5} (10^{-7}). For lower error probability beyond the current computing capability, one has to rely on the theoretically asymptotic performance, namely the asymptote of the union bound, which is exclusively dependent on the minimum distance $d_{\min}$ and the multiplicity $a(d_{\min})$. Unfortunately, the problem of computing $d_{\min}$ and/or $a(d_{\min})$ of a particular LDPC code remains largely open. Our goal of this paper is to provide a solution to deal with this hard problem.

The union bound may not be the end of the story for estimating the performance under iterative decoding at low error rate because of the pseudo-codewords discussed in [18]. Nevertheless it is still a useful tool as it may also suggest error floors as shown by specific examples below.

The remainder of the paper is organized as follows. Section II gives a brief account of the hardness of computing or approximating the minimum distance of linear codes as well as of LDPC codes. We propose the so-called approximately nearest-codewords searching approach to attack the minimum distance problem for iteratively decodable LDPC codes in Section III. It comprises two parts: the perturbing noises that inherit and extend Berrou-Vaton's impulse error pattern [19], and the reliability-based iterative decoder

that performs incomplete information set decoding on the least reliable positions. Section IV presents numerical examples and Section V concludes the paper.

II. Hardness of the Minimum-Distance Problem

The minimum distance (MinDist) problem associated with a linear code is to find the minimum distance $d_{\min}$ of the corresponding code defined by a parity-check matrix H (or generator matrix). The minimum distance of a linear code is obviously related to its error correction capability $\lfloor (d_{\min}-1)/2 \rfloor$ and, therefore, finding $d_{\min}$ is a fundamental computational problem in coding theory. A polynomial time algorithm to compute the distance would be the ideal solution to the problem, as it would be used to construct good error-correcting codes by choosing a parity-check matrix at random and checking whether the associated code has a large minimum distance. Unfortunately, no such algorithm is known. The complexity of this problem (can it be solved in polynomial time?) was first explicitly questioned by Berlekamp, McEliece, and van Tilborg in 1978 [20], who conjectured it to be nondeterministic polynomial time (NP)-hard. This conjecture was finally resolved in the affirmative by Vardy [21] in 1997, who proved that the minimum distance cannot be computed in polynomial time unless P=NP.

In this section we present an alternative, shorter but non-rigorous proof of the hardness of the MinDist problem of linear codes. One outcome of the plausible proof is that it can be easily extended to establish the hardness of the MinDist problem of LDPC codes.

Proposition 1: The MinDist problem of linear codes is NP-hard; so is the MinDist problem of LDPC codes in their broader-sense definition.

Proof: Our approach is to take a known NP-hard problem L and show its equivalence to the MinDist problem in the sense that, if an algorithm A solves the MinDist problem, then L is also solved by the output of A. For this purpose, we introduce the famous **0/1-Linear Programming** (**0/1-LP**) problem which is a known NP-hard optimization problem [22], namely:

> **[Input:]** An $m \times n$ matrix $H = [h_{ij}]_{i=1,\dots,m, j=1,\dots,n}$, and two vectors $b = (b_1, \dots, b_m)'$, $c = (c_1, \dots, c_n)'$ for some integers n, m, where h_{ij}, b_i, c_j are integers for $i = 1, \dots, m$, $j = 1, \dots, n$.
> **[Constraint:]** $\mathcal{M}(H, b, c)$= $\{Z = (z_1, \dots, z_n) \in \{0,1\}^n | HZ = b\}$.
> **[Costs:]** For every $Z = (z_1, \dots, z_n) \in \mathcal{M}(H, b, c)$, $cost(Z, (H, b, c)) = \sum_{i=1}^{n} c_i z_i$.
> **[Goal:]** Minimum.

Now we briefly outline how to relate the **0/1-LP** problem to the MinDist problem. Suppose an algorithm

A solves the MinDist problem of the code $\mathcal{C}$ defined by the parity-check matrix H, i.e. $Z^* = (z_1^*, \ldots, z_n^*)$ is found to be one of the codewords that has the minimum Hamming weight $d_{\min}$. One can choose c^* to be an all 1's vector, i.e. $c^* = (1, \ldots, 1)'$, and concerning the constraint, one just need set $b = b^* = HZ^*$, then $cost(Z^*, (H, b^*, c^*)) = \sum_{i=1}^{n} z_i^* = d_{\min}$. Note that the constraint $\mathcal{M}(H, b^*, c^*)$ forms a hyperplane of the code $\mathcal{C}$ defined by the parity-check matrix H, namely, for $\forall Z \in \mathcal{M}(H, b^*, c^*)$, Z is a codeword of $\mathcal{C}$ satisfying $HZ = b^* = 0$ (mod 2). Therefore, solving the MinDist problem means solving the specific **0/1-LP** problem with $\mathcal{M}(H, b^*, c^*)$. Recall the fact that the **0/1-LP** problem is NP-hard, i.e. there is no known deterministic polynomial-time algorithm for it. Thus the MinDist problem should be NP-hard too, because otherwise this would be a contradiction. As the hardness of the generic **0/1-LP** problem does not rely on how dense the matrix H must be, we argue that the MinDist problem of LDPC codes in their broader-sense definition is NP-hard. ■

Recently the hardness of the MinDist problem has been revealed ever further, which showed that the minimum distance $d_{\min}$ of a linear code is not approximable to within any constant factor in random polynomial time (RP), unless NP=RP [23]. Particularly, it is hard to find approximately nearest codewords even if the number of errors exceeds the unique decoding radius $d_{\min}/2$ by only an arbitrarily small fraction $\epsilon d_{\min}$.

Loosely speaking, all these hopeless statements concerning the MinDist problem apply again to LDPC codes, as long as LDPC codes are viewed as linear codes with the density of 1's in their parity-check matrices less than 1/2. This definition of LDPC codes is quite loose, because almost all practical LDPC codes have a far sparser Tanner graph than general linear codes for the benefit of good iterative decoding performance. For instance, the original Gallager's LDPC codes have only three 1s in each column, as do MacKay's rate-1/2 LDPC codes and Margulis and Ramanujan–Margulis [12] LDPC codes. It is very likely that an LDPC code designated for iterative decoding (i.e. with a degree sequence pair optimized via density evolution) has less than five or six 1s per column on average. To distinguish LDPC codes from linear codes that are not amenable to iterative decoding, we call them *iteratively decodable* LDPC codes. The modifier iteratively decodable, as we shall see later, plays an important role in the approximability of the MinDist problem. The rather sparse characteristic of an iteratively decodable LDPC code has the potential to make it possible for an efficient algorithm to approximate its minimum distance. Indeed, many LDPC codes constructed from Euclidean geometry do not satisfy this definition but the minimum distance of these codes can in general be determined from their structural properties.

The maximum-likelihood (ML) decoding problem of linear codes is known to be NP-hard, however the

iterative decoding of LDPC codes arises as a good approximation approach to the ML decoding problem by exploiting the sparseness of the Tanner graph of iteratively decodable LDPC codes. As the ML decoding problem is closely related to the MinDist problem [21], one would naturally question whether the MinDist problem of iteratively decodable LDPC codes is approximable. Although a theoretical answer is not available yet, this is very likely. Actually we do know an even stronger statement regarding a special subclass of LDPC codes, i.e. cycle codes, whose minimum distance can be exactly computed in polynomial time. By definition, a cycle code is an extremely sparse LDPC code in which each column has only two 1s. Its MinDist problem can be readily solved by computing the girth — the length of the shortest cycle — in a Tanner graph representation.

Proposition 2: Given a cycle Tanner graph with $d_s = 2$ edges incident to each symbol node, assume the girth of the graph is g. Then the minimum distance $d_{\min}$ of the corresponding binary LDPC code satisfies

$$d_{\min} = g/2 \ , \tag{1}$$

which can be computed in polynomial time.

Proof: Referring to [24, Theorem 2.5], we know an upper bound $d_{\min} \leq g/2$. To show that $d_{\min}$ is exactly equal to $g/2$, one can think of an "active" subgraph in the Tanner graph induced by a minimum weight codeword. Adopting the notation of [25], a symbol node whose associated value in the minimum weight codeword is nonzero will be called an *active* symbol node. The edges incident to active symbol nodes will be called *active* edges, and the check nodes with at least one active incident edge will be called *active* check nodes. Note that in a binary LDPC code, any edge incident to an active symbol node must be active, and any active check node must be incident to *even* active edges. Bearing in mind that each active symbol node in a cycle code has only two edges, one can deduce that, starting from any active edge of a symbol node, there must be a closed path (cycle) within the active subgraph coming back from the other edge of the same symbol node. As the shortest cycle in the Tanner graph has a length of g, we have $d_{\min} \geq g/2$, therefore we have proved $d_{\min} = g/2$. The girth of a Tanner graph can be computed in time proportional to n^2. For a detailed algorithm to compute the girth, we refer the reader to [9, 16]. ■

III. Approximately Nearest Codewords Searching Approach

For a particular LDPC code with nontrivial minimum distance, the computation of the minimum distance can essentially be regarded as a combinatorial graph-theoretic question of an even vertex set. Any exhaustive search or branch-and-bound approach requires exponential complexity and is often too complex

to implement. One can remove the requirement that an algorithm find the exact minimum distance, as a fairly good estimate (approximation) of the minimum distance may be satisfactory in most practical cases. One typical approximation approach is local search instead of exhaustive search, in which only a portion of codewords are taken into account.

A concrete local search approach to approximate the minimum distance of an iteratively decodable LDPC code, called the approximately-nearest-codewords (ANC) search, is proposed in this section. The fundamental principle is to perturb the transmit vector corresponding to the all-zero codeword with random and/or deterministic noise, and to use an efficient (iterative) decoding algorithm to obtain approximately nearest codewords. The hope is that, if the perturbing noise is minimized so that the noise-corrupted received vector is decoded just barely away from the all-zero codeword and to the nearest nonzero codeword, then there is a high probability that the Hamming weight of the nearest nonzero codeword will reveal information on the minimum distance. It is evident that the efficiency of the ANC approach depends primarily on the quality of the perturbing noise as well as the ability of the decoder to trap nearest codewords.

A. Perturbing Noise

To obtain a good estimate of the minimum distance of a particular LDPC code, it is critical to apply as little noise as possible so that the noise energy brings the decoder marginally away from the all-zero codeword. Assume $X = (-1, \ldots, -1)'$ is the vector associated with the all-zero codeword by the modulation.

A.1 AWGN

The most straightforward perturbing noise is the additive white Gaussian noise (AWGN), that is to say, the corrupted vector $Y = (y_1, \ldots, y_n)'$, used as the input of the ANC decoder, is given by

$$y_i = -1 + w_i, \tag{2}$$

where w_i is an instance of AWGN with zero-mean and standard derivation σ, which should be appropriately adjusted/chosen. Empirically σ can be chosen to be a very small number, resulting in a relatively high signal-to-noise ratio (SNR). Although with a very low level of disturbance the probability that the output of a decoder deviates from the all-zero codeword is quite small, once the decoder yields a nonzero codeword, its Hamming weight is most likely close to or exactly equal to the minimum distance.

When the perturbing noise is AWGN, a reasonably high SNR should be chosen. The higher the SNR, the more likely the nearby codewords contain the minimum-weight nonzero codeword. However, if the SNR is

set too high, the iterative decoder converges to the all-zero codeword quickly, making it a difficult task for the decoder to find nearby nonzero codewords. Because of the two conflicting factors the computational efficiency turns out to be low, although the use of the AWGN as perturbing noise is intuitive and easily conceivable. Thus we do not recommend the AWGN as perturbing noise in determining the minimum distance of an LDPC code.

A.2 Error Impulse

Another known noise pattern is the error impulse proposed by Berrou *et al.* [19, 26], which was originally proposed for computing the minimum distance of Turbo codes. The corrupted vector, which feeds the decoder, has the form

$$Y = (-1, -1, \cdots, -1, -1 + A_i, -1, \cdots, -1), \tag{3}$$

where A_i is a positive integer called the error impulse, and i is the position of error. It was shown in [26] that

Proposition 3: For any error position i, there exists a positive error impulse A_i^* such that

$$\begin{aligned} A_i^* &= \min\{A_i | Y \text{ decoded to any nonzero codeword}\} \\ &= \max\{A_i | Y \text{ decoded to the all-zero codeword}\} \\ &= \min_{Z \in \mathcal{C}, z_i = 1} w_H(Z), \end{aligned}$$

where Y is decoded according to the ML criterion under the impulse noise model, and $w_H(Z)$ is the weight spectrum of the code. The minimum distance of the code is

$$d_{\min} = \min_i A_i^*. \tag{4}$$

As exact ML decoding of Y is not practically feasible, Berrou, Vaton, Jézéquel and Douillard [26] conjectured that the minimum of individual maximum magnitudes of the error impulse corrected by a Soft-In decoder based on the AWGN assumption can directly be used as a good estimate of the minimum distance.

It is noteworthy that this pattern of noise, featuring no noise at all except in position i where A_i is large, is artificial and very improbable on the AWGN channel. In addition, the iterative decoding performance may be far away from that of the ML decoding performance under the impulse noise model. Therefore it

is quite arguable that (4) holds for computing the minimum distance of any linear code if the threshold A_i^* is computed by a Soft-In iterative decoder.

Nevertheless, the error impulse pattern can be used as a class of perturbing noise in the ANC approach, where A_i values are set to be the minimum so that the iterative decoder deviates from the all-zero codeword. It should be noted that we do not use A_i^* directly as the estimate of the minimum distance, as does in [26]. Instead, our approach depends primarily on the ability of an ANC decoder to find nearby nonzero codewords around the impulse-error corrupted vector Y.

A.3 Bit Reversing

For medium and large block length LDPC codes, say more than 2,000 bits, the threshold A_i^* under iterative decoding can be found to be extremely large or tending to infinity, as the nature of iterative decoding is quite different from the ML decoder under single impulse noise model. In this case, there might be no valid error-impulse excitation for the ANC decoder. To deal with this issue, we propose a new concept called "bit reversing".

Denote $\tilde{y}_i$ as the hard decision of ith component y_i of Y, and define the cost metric (to be minimized) as the discrepancy value:

$$L(Z, Y) = \sum_{z_i \neq \tilde{y}_i} |y_i|. \tag{5}$$

Considering the error impulse Y with ϵ to break ties, $\epsilon \ll 1$,

$$Y = (-1, -1, \cdots, -1, -1 + A_i - \epsilon, -1, \cdots, -1), \tag{6}$$

we obtain

$$L(0, Y) = A_i - 1 - \epsilon,$$

and for a codeword $Z(: z_i = 1)$ of Hamming weight w with a 1 at the impulse position, we have

$$L(Z(: z_i = 1), Y) = w - 1.$$

With a suboptimum decoder it is therefore possible that for a given value A of A_i, a metric $L(Z, Y) < A - 1$ is found, whereas nothing is detected with Berrou's method for an impulse of the value of $A_i = w$. This is particularly the case when the block length is large.

The minimum distance is related to bit-reversing metric by

$$d_{\min} = \min_i L(Z(: z_i = 1), Y) + 1. \tag{7}$$

However we can not use (7) directly. As the A_i does not always reflect the information of minimum distance, we simply reverse that bit, i.e. we set $A_i - \epsilon = +2$,

$$Y = (-1, -1, \cdots, -1, +1, -1, \cdots, -1). \tag{8}$$

Besides this reversing, we also need a decoder able to find nearby codewords under the constraint of $z_i = 1$.

A.4 Generalization

The single-position impulse error and bit reversing can be generalized to multiple positions in a straightforward way. For simplicity, the case of two impulses is discussed in the following.

For one impulse, we have

$$L(0, Y) = A_i - \epsilon - 1 \tag{9}$$

$$L(Z(: z_i = 1), Y) = w - 1 \ . \tag{10}$$

Similarly, for two impulses at positions i and j with magnitude $A_{ij} - \epsilon - 1$ each, we obtain

$$L(0, Y) = 2(A_{ij} - \epsilon - 1) \tag{11}$$

$$L(Z(: z_i = 1, z_j = 1), Y) = w - 2 \ . \tag{12}$$

The other cases are not introducing as they reduce the problem to that of (9), (10). Hence we have

$$d_{\min} - 2 \leq w - 2 \leq 2(A_{ij} - \epsilon - 1), \tag{13}$$

which leads to a direct estimate of minimum distance

$$d_{\min} \leq 2(A_{ij} - \epsilon) \ \text{ or} \tag{14}$$

$$d_{\min} \leq L(Z(: z_i = 1, z_j = 1), Y) + 2. \tag{15}$$

Again we do not use (14) and (14) directly, but use an ANC decoder to find the nearby codewords around the corrupted vector. In principle, multiple bit-reversing noise patterns can also be used as perturbing noise for the ANC decoder.

B. Approximately Nearest Codewords (ANC) Decoder

The ANC approach for computing the minimum distance of a particular LDPC code relies on the heuristic that the minimum perturbing noise for the decoder to marginally deviate from the all-zero codeword

probably leads to the minimum-weight codeword. An exact nearest-codewords decoder under certain criteria, e.g. ML, is computationally infeasible. We define an approximate nearest-codewords decoder, which is a variant of Fossorier's iterative reliability-based (IRB) decoding algorithm [27]. The IRB algorithm is a near-ML decoding method combining the SPA and the order-statistic-decoding (OSD) algorithm. At each iteration, the reliability information, i.e. the *a posteriori probabilities*(APPs) for each bit delivered by the SPA decoder, is sorted to find the most reliable basis (MRB). Then a systematic reprocessing of candidate codewords expressed in the MRB is performed. The difference between the ANC and the IRB decoder is that the former has only to select the least weight nonzero codeword from the candidate list as its output, whereas the latter has to select the most likely codeword. Also, owing to the different nature of the noise patterns, a different reprocessing strategy is used in the ANC decoder. The ANC decoder per each iteration is summarized as follows.

- *Step 0*: Perform the SPA, delivering reliability informations for each bit.

- *Step 1*: Determine and permute the positions of N^e least reliable bits (marked as erasures) based on the APPs delivered by the SPA decoder to the left of the parity-check matrix. The remaining positions (non-erasures) are preloaded with hard decisions 0 or 1 based on the SPA decoder outputs.

- *Step 2*: Perform semi-Gaussian elimination on the permuted parity-check matrix from left to right, yielding an approximate upper diagonal matrix (see Figure 1). The dependent columns encountered during the Gaussian elimination are permuted next to the last independent column. Denote the number of dependent columns by N^d.

- *Step 3*: Try all the combinations on the N^d erasures corresponding to the dependent columns. For each combination, check the set of parity-check equations $N^e - N^d + 1, \ldots, m$. If any equation is violated, abort this combination. If all equations are satisfied, then the remaining $N^e - N^d$ erasures can be determined uniquely by a simple recursive procedure, first the bit position $N^e - N^d$ and last the left-most bit position, by satisfying the check equations $N^e - N^d, N^e - N^d - 1, \ldots, 1$ in sequence. In this way a valid codeword is obtained. At most 2^{N^d} codewords can be found at each iteration.

If the input to the ANC decoder is the bit-reversing noise pattern specified by (8), the SPA decoder does not update the symbol node at the reversed bit position so that the hard decision of that bit is always 1.

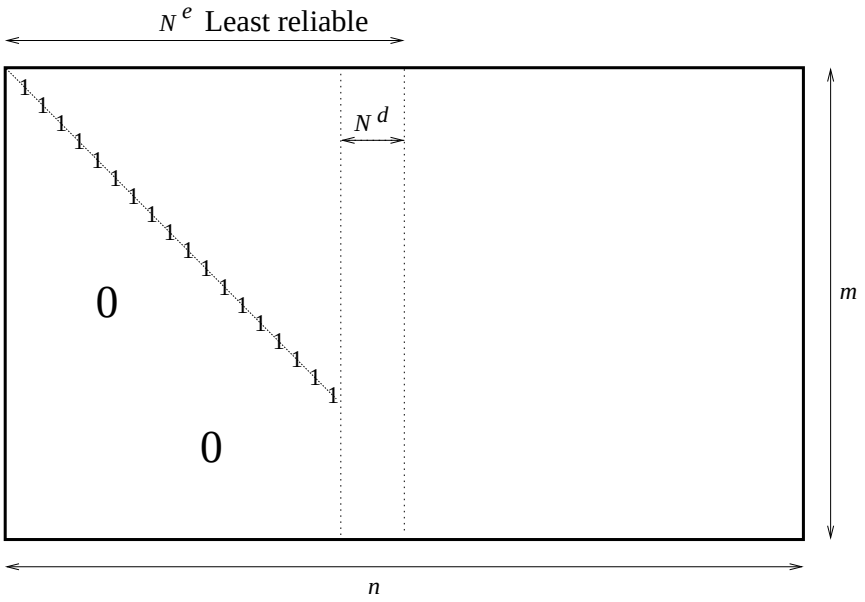

Fig. 1. A graphical view of an approximate upper triangle parity–check matrix after the permutations and semi-Gaussian elimination in *Step 2*.

The above procedure halts until a valid codeword has been reached by the SPA decoder, or a maximum number of iterations has been fulfilled. The complexity of the ANC decoder per iteration depends on the complexity of SPA decoder, the values of N^e and N^d. In practice, the value of N^e can be adaptively adjusted such that N^d is kept to a small number, say $N^d = 6 \sim 10$.

Note that step 3 is the dual formulation of OSD [28]. With respect to [27], step 3 covers only a small part of the MRB (namely the N^d least reliable positions of the MRB). However, this is perfectly matched with error patterns (an impulse or a bit-reverse) and the goal of the procedure i.e. finding a small weight codeword, which implies that many most reliable positions ($k - N^d$) and their dependences can be assumed to be error-free.

The conceptual difference between the ANC approach and the impulse method of Berrou and Vaton is that the ANC does not rely on the correctness of (4), but on the codewords found around minimally perturbed corrupted vectors, i.e. the minimum distance in the ANC approach is estimated on the basis of all codewords found (thus an upper bound), rather than the integer values of A_i's. As a by-product, the ANC approach also yields a lower bound on the multiplicity based on the collected distinct codewords.

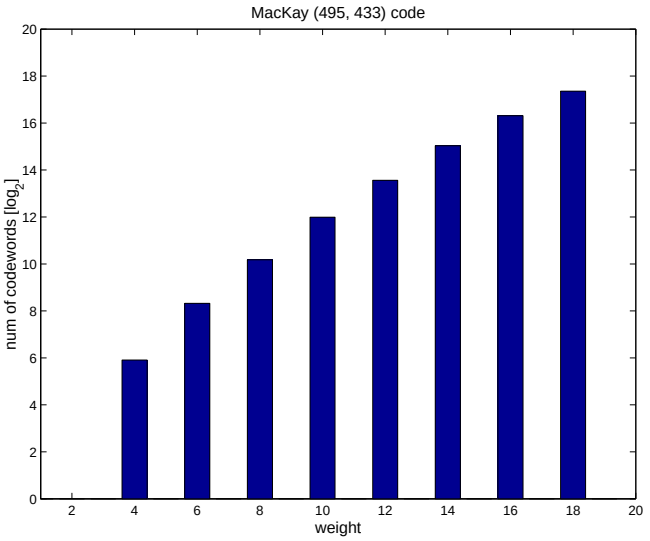

Fig. 2. Approximately low-part Hamming weight spectrum of MacKay's (495, 433) code.

IV. Numerical Results

In this section we report numerical results obtained by applying the ANC approach to different classes of LDPC codes. In all examples, the SPA used by the ANC decoder is the min-sum approximation version which is independent of the noise variance initialization, and the perturbing noise is the error impulse and/or the bit-reversing operating on individual information bits. We first consider a MacKay's regular, rate-433/495 LDPC code (495, 433) [29]. We record all the distinct codewords with Hamming weight less than 20 of this code encountered by the ANC approach, as shown in Figure 2. It turns out that this code has a minimum distance of 4 and a multiplicity of at least 60. The minimum distance can be verified exhaustively by taking the generator matrix of this (495, 433) code in the systematic form and computing the input redundancy weight distribution for inputs of weight 1, 2, and 3 with the complexity $\sum_{i=1}^{3} \binom{433}{i} \approx 2^{23.7}$. Figure 3 shows the simulated block error rate at a low SNR region and the approximate union bound based on the Hamming weights of found codewords. The convergence of the simulated block error rate to the approximate union bound at high SNR confirms that the ANC approach is a powerful tool to compute the minimum distance and approximate the corresponding multiplicity for a particular LDPC code.

We next investigate the MinDist problem of an irregular rate-1/2 LDPC code constructed by the progressive-edge-growth (PEG) construction. The block length is 200, the symbol-node edge distribution is selected as $0.31570x + 0.26758x^2 + 0.41672x^6$ and the check-node edge distribution is concentrated as $0.4381x^5 +$

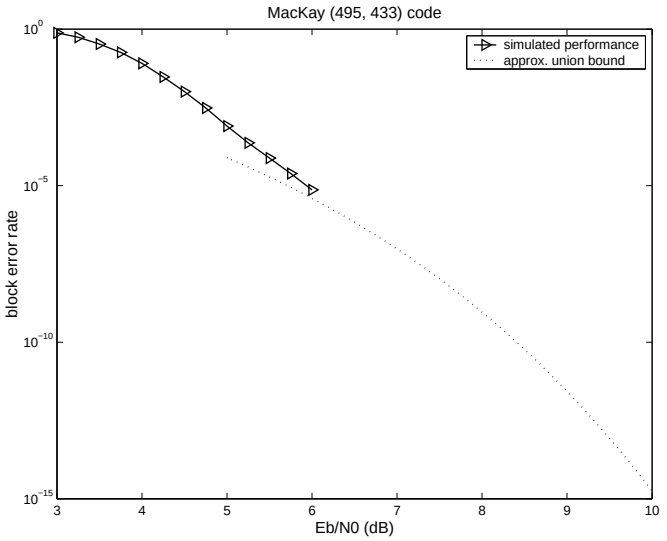

Fig. 3. Block error rate performance and the approximate union bound of MacKay's (495, 433) code.

$0.5619x^6$. Such a distribution pair is selected from Table I in [30] and has an iterative decoding threshold of 0.5153 dB. It is likely that this distribution pair does not provide the best solution for a length 200 code but this issue is not essential for our purpose. Figures 4 and 5 show the approximately low-part Hamming weight spectrum and block error rate performance, respectively. The minimum distance is found to be at most 8, and the corresponding multiplicity is at least 2. Specifically, the obtained weight distribution with one position of bit-reversing is $A(z) = 1 + 2z^8 + z^9 + 5z^{10} + 12z^{11} + 32z^{12} + 54z^{13} + 97z^{14} + 224z^{15} + 579z^{16} + 1265z^{17} + 2678z^{18} + 5586z^{19} + \cdots$. This result can be further refined by also considering the results of two positions for bit-reversing as $A(z) = 1 + 2z^8 + z^9 + 5z^{10} + 12z^{11} + 32z^{12} + 54z^{13} + 97z^{14} + 224z^{15} + 592z^{16} + 1339z^{17} + 2964z^{18} + 6515z^{19} + \cdots$. It can be seen in Figure 5 that the approximate union bound predicts well the block error rate performance at the high SNR region.

Using the ANC approach together with one-position bit reversing, we have investigated the minimum distance and multiplicity issues of the following LDPC codes: Mackay's (3, 6)-regular (504, 252) and (1008, 504) codes, Margulis code $p = 11$, Ramanujan–Margulis (13, 5) and (17, 5) codes. The results are reported in Table I. We have also tried to apply Berrou *et al.*'s impulse method, which was originally applied to determine the minimum distance of Turbo codes, to these codes, but the results are not encouraging: For Mackay's (504, 252) and (1008, 504) codes the minimum distance is estimated to be 15 and 38, respectively; for the Ramanujan–Margulis (13, 5) code it is 74; and for the Margulis code $p = 11$ and the Ramanujan–Margulis (17, 5) code, the impulse method simply failed and cannot yield a reasonable

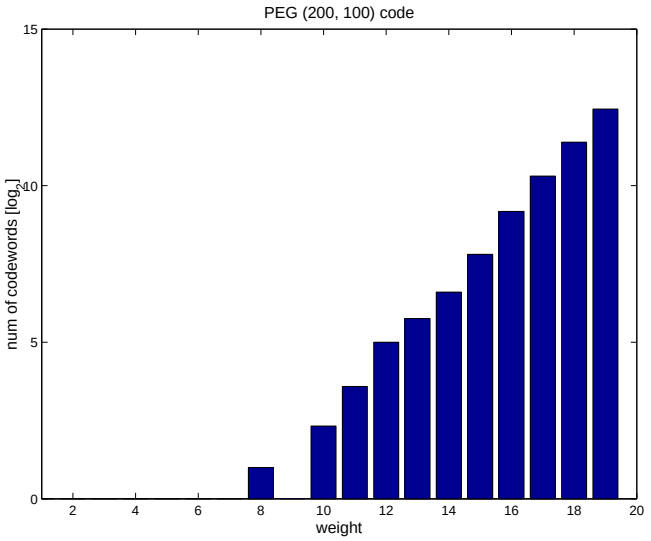

Fig. 4. Approximately low-part Hamming weight spectrum of PEG (200, 100) code.

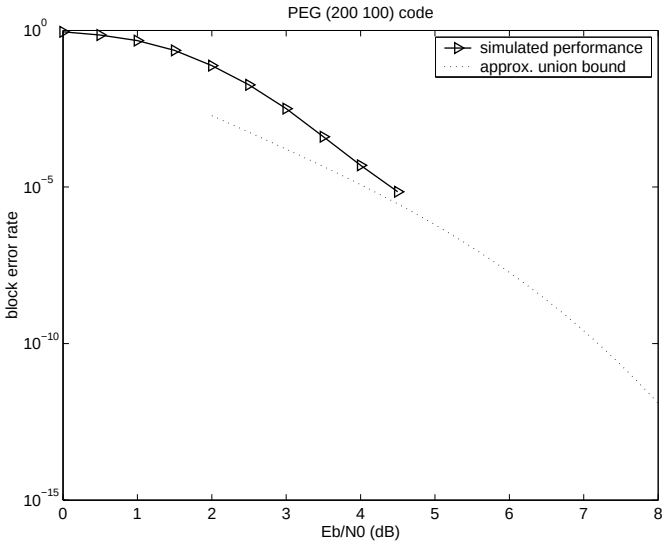

Fig. 5. Block error rate performance and the approximate union bound of PEG (200, 100) code.

number. It should be emphasized that the values of the estimated minimum distance given in Table I are based on recorded distinct codewords and thus upper bounds, although we trust that the estimated minimum distance by the ANC approach is most likely the true minimum distance of these codes. It is possible in principle that some codewords with smaller Hamming weight are missed by the ANC approach. It is also likely that some codewords with the same Hamming weight as the minimum distance are not captured, and thus the multiplicity might be improved provided more patience or more computing power is available.

TABLE I

ESTIMATED MINIMUM DISTANCE AND MULTIPLICITY OF SOME WELL-KNOWN LDPC CODES USING THE ANC APPROACH.

code name	MacKay	MacKay	Margulis ($p = 11$)	(13, 5) Ramanujan –Margulis	(17, 5) Ramanujan –Margulis
code length	504	1008	2640	4368	4896
code dimension	252	504	1320	2184	2474
minimum distance	**20**	**34**	**40**	**14**	**24**
multiplicity	**2**	**1**	**66**	**2184**	**204**

To cross-check our results, we point out that these minimum distance results are in good agreement with the analysis in [18], demonstrating the effectiveness of the ANC approach for computing the minimum distance of iteratively decodable LDPC codes. Referring to [18], it can also be pointed out that the union bound based on the code distance structure may not always provide an accurate representation of iterative decoding at high SNR, but nevertheless remains valuable.

V. CONCLUSIONS

Starting with the hardness analysis of the problem of computing the minimum distance of LDPC codes, we have proposed the approximately nearest codewords (ANC) approach to attack this MinDist problem for iteratively decodable LDPC codes. The principle of the ANC approach is to search codewords locally around the all-zero codeword perturbed by a minimum level of noise. The resultant nearest codewords most likely contain the minimum Hamming weight codeword whose Hamming weight is equal to the minimum distance of the linear code.

We have considered three classes of perturbing noise: one is the conventional AWGN, [1] one is the error impulse noise from Berrou *et al.* and the third is the bit-reversing noise. The ANC decoder can be viewed as a modified version of Fossorier's IRB near-ML decoder in which the reprocessing phase has been modified based on the noise patterns used. The effectiveness of this approach has been verified by numerical examples. It is worth mentioning that the proposed modifications of Berrou *et al.*'s impulse method can be applied to the MinDist problem of turbo codes.

The ANC approach has its limitations. For linear block codes, which are not amenable to iterative

[1] Note that for AWGN, the reprocessing of [27] is better than step 3 proposed here. But it was not used here anyway.

decoding, the iterative ANC decoder generally fails to find the nearest nonzero codewords because the SPA does not work well unless a sparse graph representation with a good degree distribution is used. Nevertheless, it seems that in this area not much can be done from an algorithmic point of view, since the approximation of the MinDist problem for generic linear codes, as shown in [23], is unfortunately NP-hard.

ACKNOWLEDGMENT

The authors are grateful to Dr. P. O. Vontobel, who provided the parity-check matrices of Margulis ($p = 11$), Ramanujan–Margulis (13, 5) and (17, 5) codes for testing the effectiveness of the iterative ANC approach. They thank Prof. MacKay for providing his (504, 252) and (1008, 504) codes and for the interesting discussion at IBM Zurich Res. Lab. during his visit. They also thank Prof. C. Berrou for providing preprints of [19, 26].

REFERENCES

[1] R. G. Gallager, "Low-density parity-check code," *IRE Trans. Inform. Theory*, vol. 8, pp. 21–28, 1962.

[2] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," in *Proc. IEEE Intl. Conf. Commun. (ICC), Geneva, Switzerland*, pp. 1064–1070, 1993.

[3] D. J. C. MacKay and R. M. Neal, "Good codes based on very sparse matrices," in *Cryptography and Coding, 5th IMA Conference (Lecture Notes in Computer Science)*, pp. 100–111, 1995. C. Boyd, Ed. Berlin, Germany: Springer vol. 1025.

[4] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 45, pp. 399–431, Mar. 1999.

[5] T. Richardson and R. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, vol. 47, pp. 599–618, Feb. 2001.

[6] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inform. Theory*, vol. 27, pp. 533–547, Sept. 1981.

[7] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum–product algorithm," *IEEE Trans. Inform. Theory*, vol. 47, pp. 498–519, Feb. 2001.

[8] G. D. J. Forney, "Codes on graphs: Normal realizations," *IEEE Trans. Inform. Theory*, vol. 47, pp. 520–548, Feb. 2001.

[9] Y. Mao and A. Banihashemi, "A heuristic search for good low-density parity-check codes at short block lengths," in *Proc. IEEE Intl. Conf. Commun. (ICC), Helsinki, Finland*, pp. 41–44, 2001.

[10] Y. Kou, S. Lin, and M. P. C. Fossorier, "Low-density parity-check codes based on finite geometrices: a rediscovery and new results," *IEEE Trans. Inform. Theory*, vol. 47, pp. 2711–2736, Nov. 2001.

[11] J. Lafferty and D. Rockmore, "Codes and iterative decoding on algebraic expander graphs," in *Proc. IEEE Intl. Symp. Inform. Theory and Applications (ISITA), Honolulu, Hawaii, USA*, Nov. 2000.

[12] J. Rosenthal and P. O. Vontobel, "Construction of LDPC codes based on Ramanujan graphs and ideas from Margulis," in *Proc. 38th Annual Allerton Conf. on Communication, Computing and Control, Monticello, IL*, Oct. 2000.

[13] P. O. Vontobel and R. M. Tanner, "Construction of codes based on finite generalized quadrangles for iterative decoding," in *Proc. IEEE Intl. Symp. Inform. Theory, Washington, DC*, June 2001.

[14] P. O. Vontobel, *Algebraic Coding for Iterative Decoding.* PhD thesis, Swiss Federal Institute of Technology Zurich (ETHZ), Switzerland, Jan. 2003.

[15] X.-Y. Hu, E. Eleftheriou, and D.-M. Arnold, "Progressive edge-growth Tanner graphs," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM), San Antonio, Texas, USA*, Nov. 2001.

[16] X.-Y. Hu, E. Eleftheriou, and D.-M. Arnold, "Irregular progressive edge-growth Tanner graphs," in *Proc. IEEE Intl. Symp. Inform. Theory, Lausanne, Switzerland*, July 2002. Submitted to IEEE trans. on Information Theory (revised).

[17] X.-Y. Hu, *Low-delay low-complexity error-correcting codes on sparse graphs*. PhD thesis, Swiss Federal Institute of Technology Lausanne (EPFL), Switzerland, Oct. 2002. *Available on request.*

[18] D. J. C. MacKay and M. S. Postol, "Weaknesses of Margulis and Ramanujan–Margulis low-density parity-check codes," *Electronic Notes in Theoretical Computer Science*, vol. 74, 2002.

[19] C. Berrou and S. Vaton, "Computing the minimum distance of linear codes by the error impulse method," in *Proc. IEEE Intl. Symp. Inform. Theory, Lausanne, Switzerland*, July 2002.

[20] E. R. Berlekamp, R. J. McEliece, and H. C. A. van Tilborg, "On the inherent intractability of certain coding problems," *IEEE Trans. Inform. Theory*, vol. 24, pp. 384–386, Mar. 1978.

[21] A. Vardy, "The intractability of computing the minimum distance of a code," *IEEE Trans. Inform. Theory*, vol. 43, pp. 1757–1766, Nov. 1997.

[22] J. Hromkovič, *Algorithmics for Hard Problems: Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics*. Springer-Verlag Berlin Heidelberg, 2001.

[23] I. Dumer, D. Micciancio, and M. Sudan, "Hardness of approximating the minimum distance of a linear code," *IEEE Trans. Inform. Theory*, vol. 49, pp. 22–37, Jan. 2003.

[24] R. G. Gallager, *Low Density Parity Check Codes*. MIT Press, Cambridge, MA, 1963.

[25] R. M. Tanner, "Minimum distance bounds by graph analysis," *IEEE Trans. Inform. Theory*, vol. 47, pp. 808–821, Feb. 2001.

[26] C. Berrou, S. Vaton, M. Jézéquel, and C. Douillard, "Computing the minimum distance of linear codes by the error impulse method," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM), Taiwan*, Nov. 2002.

[27] M. P. C. Fossorier, "Iterative reliability-based decoding of low-density parity-check codes," *IEEE J. Select. Areas Commun.*, vol. 19, pp. 908–917, May 2001.

[28] M. P. C. Fossorier, S. Lin, and J. Snyders, "Reliability-based syndrome decoding of linear block codes," *IEEE Trans. Inform. Theory*, vol. 44, pp. 388–398, Jan. 1998.

[29] D. J. C. MacKay, *Online database of low-density parity-check codes*. `http://wol.ra.phy.cam.uk/mackay/codes/data.html`.

[30] T. Richardson, A. Shokrollahi, and R. Urbanke, "Design of provably good low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 619–637, Feb. 2001.