Geometrical and Performance Analysis of GMD and Chase Decoding Algorithms

Eran Fishler, *Student Member, IEEE*, Ofer Amrani, and Yair Be'ery, *Senior Member, IEEE*

***Abstract*—The overall number of nearest neighbors in bounded distance decoding (BDD) algorithms is given by $N_{o,\text{eff}} = N_o + N_{\text{BDD}}$, where N_{BDD} denotes the number of additional, noncodeword, neighbors that are generated during the (suboptimal) decoding process. We identify and enumerate the nearest neighbors associated with the original Generalized Minimum Distance (GMD) and Chase decoding algorithms. After careful examination of the decision regions of these algorithms, we derive an approximated probability ratio between the error contribution of a noncodeword neighbor (one of N_{BDD} points) and a codeword nearest neighbor. For Chase Algorithm 1 it is shown that the contribution to error probability of a noncodeword nearest neighbor is a factor of 2^{d-1} less than the contribution of a codeword, while for Chase Algorithm 2 the factor is $2^{\lceil d/2 \rceil - 1}$, d being the minimum Hamming distance of the code. For Chase Algorithm 3 and GMD, a recursive procedure for calculating this ratio, which turns out to be nonexponential in d, is presented. This procedure can also be used for specifically identifying the error patterns associated with Chase Algorithm 3 and GMD. Utilizing the probability ratio, we propose an improved approximated upper bound on the probability of error based on the union bound approach. Simulation results are given to demonstrate and support the analytical derivations.**

***Index Terms*— Bounded-distance decoding, decision region, nearest neighbors, pseudo neighbors, volume ratio.**

I. Introduction

THE computational burden involved in optimal soft-decision decoding of good linear block codes is often prohibitive. As an alternative, various suboptimal soft-decision algorithms, which trade performance for computational complexity, have been devised along the years. While measuring the decoding complexity is an interesting problem in its own right, finding a unified yet feasible method with which to evaluate performance of decoding algorithms is a much more complicated task (be it a computer simulation or an analytic approach). For the additive white Gaussian noise (AWGN) channel with variance σ^2, the performance of a specific decoding algorithm is determined by the decision regions of the codewords. When all these regions are congruent, one must first determine the shape of a region (see, e.g., [1]), and then integrate the noise density function within this region to obtain the probability for correct decoding. Except for some trivial cases, this problem is analytically intractable.

Alternatively, a computationally simple, yet often loose, upper bound is commonly employed—the *union bound*. For optimal decoding, the decision region of a specific codeword is the *Voronoi region*, which is determined by certain "face defining" neighboring codewords also known as the *Voronoi neighbors*. In general, the union bound requires precisely that knowledge about the (linear) code, namely, its spectrum of distances:

$$P_e \leq \sum_{d_i \in \mathcal{F}} A(d_i) Q(d_i/2\sigma)$$

where $\mathcal{F}$ is the set of Euclidean distances between some codeword and its Voronoi neighbors and $A(d_i)$ denotes the number of codewords at Euclidean distance d_i. Let $d_{\min}$ denote the minimum Euclidean distance of a code. For high signal-to-noise ratios (SNR) the number of the nearest neighbors $N_o = A(d_{\min})$ usually suffices, as the contribution of the other neighbors (located further away) to performance degradation is relatively negligible.

A suboptimal decoding algorithm which decodes correctly at least within the spheres of radius $d_{\min}/2$ centered on the codewords is known as a *bounded distance* decoding (BDD) algorithm. The shape of the decision regions for bounded distance algorithms is much more complex than that of optimal algorithms as recently shown in [2]. More specifically, the number of nearest neighbors is increased due to the suboptimal decoding process, and hence the overall number of nearest neighbors, $N_{o,\text{eff}} = N_o + N_{\text{BDD}}$, is usually plugged into the union bound in order to evaluate the performance of the algorithm. This, however, yields a loose bound because the error contribution of a nearest neighbor that is generated during the suboptimal decoding process is smaller than that of nearest neighbor which is a codeword [2]. Note that bounded distance decoding algorithms do not necessarily increase the number of nearest neighbors [7], [12]. In this case, performance degradation is mainly due to additional shells of neighboring points, located close to the first shell (the shell of nearest neighbors) [2], [9].

Suppose that we can somehow obtain the ratio between the error contributions of the two types of nearest neighbors, i.e., a nearest neighbor that is generated during the bounded distance decoding process (noncodeword nearest neighbor), and a codeword nearest neighbor. Denote this ratio by η. Then, we may write a modified approximated upper bound in the

Manuscript received September 11, 1997; revised February 22, 1999. The material in this paper was presented in part at the Israeli-French Workshop on Coding and Information Integrity, Dead Sea, Israel, October 1997.

The authors are with the Department of Electrical Engineering–Systems, Tel-Aviv University, Ramat-Aviv 69978, Tel-Aviv, Israel.

Communicated by F. Kschischang, Associate Editor for Coding Theory.

Publisher Item Identifier S 0018-9448(99)04168-1.

form

$$P_e \approx (N_0 + \eta \cdot N_{\mathrm{BDD}})Q\left(\frac{d_{\min}}{2\sigma}\right). \tag{1}$$

In the sequel, such bounds are derived for the prominent Generalized Minimum Distance (GMD) and Chase decoding algorithms [6], [3]. The bounds are obtained after a careful examination of the decision regions of these algorithms. Surprisingly, the dominant parameter influencing the decision region of the algorithms, and hence the error ratio η, is the minimum Hamming distance of the code, d.

While this paper focuses on the nearest neighbors, we note that the error ratio η can be defined more generally as a function of the Euclidean distance corresponding to the error regions caused by neighbors at various distances. In this case, the union bound will have the form

$$P_e \le \sum_{d_i \in \mathcal{F}} A(d_i)Q(d_i/2\sigma) + \sum_{\rho \in \tilde{\mathcal{F}}} \eta(\rho)A(\rho)Q(\rho/2\sigma)$$

where $\tilde{\mathcal{F}}$ is the set of Euclidean distances of the neighbors that are generated in the decoding process. Obviously, in (1), $\eta = \eta(d_{\min})$. For algorithms with no additional nearest neighbors, i.e., $N_{\mathrm{BDD}} = 0$, it would be of practical importance to evaluate $\eta(\rho)$ for the second shell of neighbors (e.g., $d + \frac{2}{3}$ for the modified GMD [9]) as their contribution to performance loss may be significant. This, however, is not discussed in the paper.

In the next section we establish notations and briefly review the Chase and GMD decoding algorithms. For clarity of exposition we start our treatment with the Chase rather than the GMD algorithm. Also in Section II, the *union bound* and *nearest neighbors* are revisited. In particular, a refined definition is given to the term *nearest neighbor*. The Chase and GMD decoding algorithms are analyzed in detail in Sections III and IV, respectively. Conclusions and simulation results are presented in Section V. Finally, the Appendix contains some proofs and a supplementary example.

II. Preliminaries

Let $\mathcal{C}$ denote an $[n, k, d]$ binary linear block code, with $M = 2^k$ codewords. Each codeword $C_i \in \mathcal{C}$ is a vector in $\mathrm{GF}(2)^n$. A one-to-one mapping between each codeword C_i and $X_i \in R^n$, R^n being the Euclidean n-space, is obtained using the transformation

$$\begin{aligned} X_i &= (-1)^{C_i} \\ &= \{(x_{i1}, x_{i2}, \cdots, x_{in}): \\ &\qquad x_{ij} = (-1)^{c_{ij}}, (c_{i1}, c_{i2}, \cdots, c_{in}) \in \mathcal{C}\}. \end{aligned} \tag{2}$$

The resultant set $\mathcal{X} = \{X_1, X_2, \cdots, X_M\}$ contains M distinct points, one for each binary codeword. Henceforth, we also refer to these points as codewords. The minimum squared Euclidean distance of the code $\mathcal{C}$ is given by

$$d^2_{\min}(\mathcal{C}) \triangleq \min_{i \ne j} ||X_i - X_j||^2 = 4d$$

where $||\cdot||$ denotes Euclidean distance. The codewords X_i are assumed to be transmitted over an AWGN channel, wherethe received vector $\boldsymbol{r} \in R^n$, is a transmitted codeword $X \in \mathcal{C}$ perturbed by a noise vector $\boldsymbol{n} \sim N(\mathbf{0}, \sigma^2 I)$, and $\boldsymbol{r}$ is given by $\boldsymbol{r} = X + \boldsymbol{n}$.

A complete decoding scheme is a mapping rule $f: R^n \longmapsto \mathcal{C}$, such that for every vector $\boldsymbol{r} \in R^n$ it assigns a codeword, $C_i \in \mathcal{C}$. Denote by $\hat{X}$ the estimated codeword given the received vector is $\boldsymbol{r}$, $\hat{X} = f(r)$ via (2). A decoding scheme that achieves minimum mean probability of error is called optimal. For an AWGN channel with equal probability for transmitting a particular codeword, the task of optimal decoding is equivalent to finding the closest codeword to the received vector $\boldsymbol{r}$, namely, $\hat{X} = \arg\min_{X_i \in \mathcal{X}} ||X_i - \boldsymbol{r}||$.

Several methods have been proposed for performing this task without explicitly computing the Euclidean distances between $\boldsymbol{r}$ and all possible codewords. One such method employs channel measurement and error vectors. We briefly describe this method below as it will become handy in following sections. Denote by Y the bitwise hard-decision vector whose elements are given by $y_i = 0$ for $r_i > 0$, and $y_i = 1$ otherwise. Also, let Z_m denote the error vector satisfying $Z_m \triangleq Y \oplus C_m$, where C_m is a codeword and $\oplus$ denotes modulo-2 addition. Then, the estimated transmitted codeword $\hat{X}$ is the one with minimum "analog weight" W

$$\hat{X} = \arg \min_{X_m \in \mathcal{X}} W(Z_m)$$

where

$$W(Z_m) = \sum_{i=1}^{n} |r_i| Z_{m_i}.$$

We shell henceforth refer to $|r_i|$ as the *confidence level* of the ith symbol, and to r_i as the *soft value* of that symbol. Note that r_i is actually a signed likelihood measure of the ith symbol.

A. Description of Chase and GMD Decoding Algorithms

Chase algorithms [3] are suboptimal decoding algorithms, that is, the algorithms do not achieve minimum mean probability of error. In general, each of these algorithms generates a set of error vectors, where each vector has 1's in the coordinates suspected as errors and 0 elsewhere. By summing each error vector with the original hard-decision vector, a new set of vectors is generated with the symbols inverted wherever an error was suspected to have occurred. All the newly generated vectors are then decoded with a binary decoder. This binary decoder guarantees finding a codeword *iff* the Hamming distance between the candidate vector and some codeword is less than or equal to $\lfloor \frac{d-1}{2} \rfloor$; otherwise, the decoder will declare a decoding failure. Following the decoding stage, a set of codewords is obtained and the "analog weight" of all these codewords is computed. The estimated transmitted codeword is the one with the minimum "analog weight." If no codeword was generated in the decoding stage, the hard-decision vector is assumed to have been transmitted. Chase algorithms differ in the manner in which they generate error patterns and in the number of the error patterns they produce.

- *Chase Algorithm 1* (CA1): CA1 generates a large set of error patterns. The error patterns generated are all the

vectors which have $\lfloor d/2 \rfloor$ 1's in them. There are $\binom{n}{\lfloor \frac{d}{2} \rfloor}$ such error patterns to be considered. A sufficient condition for an error in the decoding process is that $d(C_i, Y) \geq d$, where C_i is the transmitted codeword.

- *Chase Algorithm 2* (CA2): CA2 generates a smaller set of error patterns. The set of error patterns consists of all vectors having any combination of 1's in the $\lfloor \frac{d}{2} \rfloor$ coordinates with the lowest confidence levels. There are $2^{\lfloor \frac{d}{2} \rfloor}$ such error patterns to be considered.
- *Chase Algorithm 3* (CA3): CA3 generates the smallest set of error patterns. Each error pattern is a vector containing 1's in the j symbols with the lowest confidence levels. For a code with d even, j takes the values $j = 0, 1, 3, \cdots, d-1$. When d is odd, j takes the values, $j = 0, 2, 4, \cdots, d-1$. The number of patterns is only $\lfloor \frac{d}{2} + 1 \rfloor$.

An upper bound on the probability of error for CA1–CA3 was derived in [3]. The error exponent of that bound is the same for all three algorithms, as well as for optimal decoding.

An even more prominent suboptimal algorithm is the GMD algorithm proposed in [6]. A brief description of the algorithm follows. Initially, the symbols of the hard-decision vector are ordered according to their confidence levels. Assume that there exists a binary *errors and erasures* (EE) decoder. Such a decoder is capable of decoding correctly *iff* the number of erasures and twice the number of errors, in the decoded vector, is less than d. Then, a sequence of EE decoding trials is performed, where in each trial a different number of least reliable symbols are erased: $d-1, d-3, \cdots$ least reliable symbols. Finally, the generated codeword, with the minimum distance from the received vector $\boldsymbol{r}$ is the estimated transmitted codeword. If no codeword was generated in the decoding trails, decoding failure is declared and is treated as a decoding error.

B. The Union Bound and Nearest Neighbors

The Voronoi region of X_i, $V(X_i)$, is the portion of R^n such that for the points belonging to this portion, X_i is the closest codeword. When the received vector belongs to the Voronoi region, $\boldsymbol{r} \in V(X_i)$, optimal decoding will decode $\boldsymbol{r}$ to X_i. The Voronoi region $V(X_i)$ is a convex polytope whose faces lie in the hyperplane, midway between X_i and as many as $M-1$ other codewords.

Probably the best known method for estimating the probability of error for optimal decoding is the union bound. The union bound is an upper bound on the probability of error given that X_i was transmitted

$$P_e(X_i) \leq \sum_{j=1, j \neq i}^{M} \int_{d_{ij}/2}^{\infty} \frac{1}{\sqrt{2\pi\sigma^2}} \exp^{-(y^2/2\sigma^2)} \, dy \qquad (3)$$

where d_{ij} is the Euclidean distance between X_i and X_j.

For moderate to high SNR's, the union bound can be approximated using only the terms for which $d_{ij} = d_{\min}$. Denoting by N_0 the number of codewords at distance $d_{\min}$ from X_i, (3) may be approximated by

$$N_0 \int_{d_{\min}/2}^{\infty} \frac{1}{\sqrt{2\pi\sigma^2}} \exp^{-(y^2/2\sigma^2)} \, dy. \qquad (4)$$

When suboptimal decoding is employed, the decision region is no more $V(X_i)$. Let $D(X_i)$ denote the decision region of an X_i under suboptimal decoding. That is, $D(X_i)$ is the set of all the points in R^n, where for each $\boldsymbol{r} \in D(X_i)$, $\boldsymbol{r}$ will be decoded to X_i. Clearly, $D(X_i)$ determines the exact probability of error given that X_i was transmitted

$$Pe(X_i) = 1 - \int_{D(X_i)} \frac{1}{(2\pi\sigma^2)^{n/2}} e^{-\|\boldsymbol{\alpha} - X_i\|^2/2\sigma^2)} \, d\boldsymbol{\alpha}.$$

When optimal decoding is employed, there exists a hyper-sphere of radius $d_{\min}/2$ centered on X_i, denoted by $B(X_i, (d_{\min}/2))$, such that every received vector satisfying $\boldsymbol{r} \in B(X_i, (d_{\min}/2))$ is decoded to X_i. A decoding algorithm that guarantees correct decoding whenever the received vector $\boldsymbol{r}$ is within a hyper-sphere of radius $d_{\min}/2$ centered on some codeword, is a bounded distance algorithm.

The approximation of the union bound (4) takes into account only the terms corresponding to the codewords at distance $d_{\min}$ from X_i. All other terms are exponentially smaller and thus can be neglected for moderate to high SNR's. Each of the terms considered corresponds to half-space error region whose closest point is at distance $d_{\min}/2$ from X_i in the direction of a neighboring point. In other words, the hyper-planes that serve as the border of the half-space error regions are tangent to the hyper-sphere $B(X_i, (d_{\min}/2))$ at one point. Consequently, the approximation for the union bound can be described as follows: the number of tangent points between $B(X_i, (d_{\min}/2))$, and $V(X_i)$, multiplied by the probability that the noise in the direction of the tangent point is greater than $d_{\min}/2$.

This approach can be adapted for BDD algorithms. The probability of error in this case is approximately upper-bounded by the number of tangent points between $D(X_i)$ and $B(X_i, (d_{\min}/2))$, multiplied by the probability that the noise in the direction of the tangent point is greater than $d_{\min}/2$. The number of tangent points is traditionally denoted by $N_{0,\text{eff}}$. $N_{0,\text{eff}} = N_0 + N_{\text{BDD}}$, where N_0 is the number of codewords at distance $d_{\min}$ from X_i, and N_{BDD} is the number of points at distance $d_{\min}$ from X_i generated by the suboptimal decoding process. Thus an approximated bound on the probability of error when using a BDD algorithm is

$$\begin{aligned} Pe(X_i) &\approx N_{0,\text{eff}} \int_{d_{\min}/2}^{\infty} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(y^2/2\sigma^2)} \, dy \\ &= N_{0,\text{eff}} Q\left(\frac{d_{\min}}{2\sigma}\right). \end{aligned} \qquad (5)$$

The above approximation has two main deficiencies. The first arises when $N_{0,\text{eff}}$ is very big. Note that in [2] it is shown that some algorithms may have an infinite number of tangent points between $D(X_i)$ and $B(X_i, (d_{\min}/2))$. In those cases, and also when the SNR is not sufficiently high, this bound is useless as it may produce values greater than one. It is, however, the following deficiency that we address in the sequel. The aforementioned bounds assume that every tangent point between $D(X_i)$ and $B(X_i, (d_{\min}/2))$ causes a decoding failure whenever the magnitude of the noise (along the axis between X_i and the tangent point) is greater than

$d_{\min}/2$. Additionally, the bound assumes that the border of the decision region is, at least locally, a hyper-plane that is tangent to $B(X_i,(d_{\min}/2))$. It was recently discovered [2] that in several cases, locally, the decision region has the shape of hyper-polygon rather than a hyper-plane. The aforementioned bounds do not take this into consideration.

We now focus on the conventional nearest neighbors of a BDD algorithm. The "common definition" states that a nearest neighbor in a BDD algorithm is a point $l \in R^n$ at distance $d_{\min}$ from X_i such that there exists a corresponding tangent point $\frac{l+X_i}{2}$ between $D(X_i)$ and $B(X_i,(d_{\min}/2))$. This definition of nearest neighbors may in certain instances lead to incorrect performance estimation, as will be shown for CA3. Hence, we propose an alternative definition for nearest neighbors.

Definition 1: (Conventional nearest neighbor of a codeword X_i.) Every point $X \in R^n$ at distance $d_{\min}$ from X_i, such that X can be obtained by the transformation (2) of a GF$(2)^n$ vector, shall be called a conventional nearest neighbor if for every ϵ greater than zero, there is an error region (for X_i) with nonzero volume within the hyper-sphere $B(\frac{X_i+X}{2},\epsilon)$.

As expected, according to this definition, any codeword X_j at distance $d_{\min}$ from X_i is a conventional nearest neighbor of X_i, since one hemisphere of $B(\frac{X_i+X_j}{2},\epsilon)$ will not be decoded to X_i. The motivation for Definition 1 is the following. Let X and X_i be defined as in Definition 1, except that X is not a codeword. For $\boldsymbol{r}$ on the axis connecting X and the transmitted codeword X_i, CA3 can decode correctly even when $\|X_i-\boldsymbol{r}\| > (d_{\min}/2)+\epsilon$, where ϵ is small. We prove, however, that there is a region within $B(\frac{X_i+X}{2},\epsilon)$ where CA3 fails to decode, even for infinitely small ϵ. Thus the contribution of any such point X should be taken into consideration when estimating the error probability. Indeed, Definition 1 also accounts for this type of points.

Finally, following the definition of *pseudo neighbors* given in [2], *pseudo nearest neighbors* are defined as follows. Let X_i be the transmitted codeword. Denote by L the set of all the points $l \in R^n$ satisfying $\|l-X_i\| = d_{\min}$, where l is not a conventional nearest neighbor. Then, a pseudo nearest neighbor is a point $l \in L$ such that, for every $\epsilon > 0$, there is a nonzero volume within $B(\frac{l+X_i}{2},\epsilon)$ in which decoding error occurs.

III. Performance Analysis of Chase Algorithms

To employ the union bound (5) for estimating the performance of Chase algorithms, one must first prove that they are indeed BDD algorithms. Our proof employs a result derived in [3, Appendix I] and is stated as follows. Assume without loss of generality (w.l.o.g.) that the $\mathbf{1} \in R^n$ (all-zero binary) codeword was transmitted and let $\boldsymbol{r}$ denote the received vector. A necessary condition for error in decoding $\boldsymbol{r}$ is the existence of a set of indices $S(d)$, where $|S(d)| = d$, such that

$$\sum_{i\in S(d)} r_i \le 0. \tag{6}$$

Theorem 1: Chase algorithms are BDD algorithms.

Proof: Assume that the codeword $\mathbf{1}$ was transmitted and that decoding error has occurred. Recall that $d_{\min} = 2\sqrt{d}$. It will be proved that the distance between the received vector $\boldsymbol{r}$ and $\mathbf{1}$ is no smaller than $(d_{\min}/2) = \sqrt{d}$.

First, assume that there are more than $d-1$ errors in the hard-decision vector. This means that at least d symbols satisfy $r_i \le 0$ and thus $\|\boldsymbol{r}-\mathbf{1}\|^2 \ge d$.

The second possible event is when there are $d-j$ errors in the hard-decision vector, $1 \le j \le d-1$. Let e denote the set of indices, such that $r_i \le 0$. Also, let $\{\delta_i\}$ and $\{\epsilon_i\}$ be the confidence levels of the symbols with indices belonging to e and $\overline{e}$, respectively. The received vector $\boldsymbol{r}$ will be inside the hyper-sphere $B(\mathbf{1},(d_{\min}/2))$ if

$$\sum_{i\in e}(1+\delta_i)^2 + \sum_{i\in\overline{e}}(1-\epsilon_i)^2 = \|\boldsymbol{r}-\mathbf{1}\|^2 < \left(\frac{d_{\min}}{2}\right)^2 = d. \tag{7}$$

Let $S(\overline{e})$ be the set of all indices belonging to both $S(d)$ and $\overline{e}$, $S(\overline{e}) = \{S(d)\cap\overline{e}\}$. Assume that $|S(\overline{e})| = m$, where $j \le m \le d$. Let $S(e)$ be the set of all symbols belonging to $S(d)$ and e, $S(e) = \{S(d)\cap e\}$, $|S(e)| = d-m$. According to (7)

$$\begin{aligned}
&\sum_{i\in S(e)}(1+\delta_i)^2 + \sum_{i\in S(\overline{e})}(1-\epsilon_i)^2 \\
&\quad\le \sum_{i\in e}(1+\delta_i)^2 + \sum_{i\in\overline{e}}(1-\epsilon_i)^2 < d \\
&d-m+2\sum_{i\in S(e)}\delta_i + \sum_{i\in S(e)}\delta_i^2 + m \\
&\quad -2\sum_{i\in S(\overline{e})}\epsilon_i + \sum_{i\in S(\overline{e})}\epsilon_i^2 < d \\
&2\left(\sum_{i\in S(e)}\delta_i - \sum_{i\in S(\overline{e})}\epsilon_i\right) + \sum_{i\in S(e)}\delta_i^2 + \sum_{i\in S(\overline{e})}\epsilon_i^2 < 0.
\end{aligned} \tag{8}$$

From (6)

$$\sum_{i\in S(e)}\delta_i - \sum_{i\in S(\overline{e})}\epsilon_i \ge 0 \tag{9}$$

thus the left-hand side of (8) is a sum of nonnegative components and hence greater or equal to 0, thereby encountering a contradiction. Concluding, if (6), which is an error condition, holds, then $\boldsymbol{r} \notin B(\mathbf{1},(d_{\min}/2))$. □

It can be seen from the proof of Theorem 1, that the received vector $\boldsymbol{r}$, at distance $d_{\min}/2$ from the transmitted word, might cause a decoding error *iff* there are d symbols with confidence level 0, and $n-d$ symbols with confidence level 1. Otherwise, the left-hand side of (8) will be greater than zero and a contradiction will be encountered. Every vector $\boldsymbol{r}$ that satisfies the above condition corresponds to one of the $\binom{n}{d}$ conventional nearest neighbors. It also follows from the proof of Theorem 1, that the received vector $\boldsymbol{r}$ at distance $(d_{\min}/2)+\epsilon$, where ϵ is small, that will cause a decoding error, must be in the "area" of the midpoint between a conventional nearest neighbor and the transmitted codeword; if not, we will get contradiction in (8). This leads us to the corollary

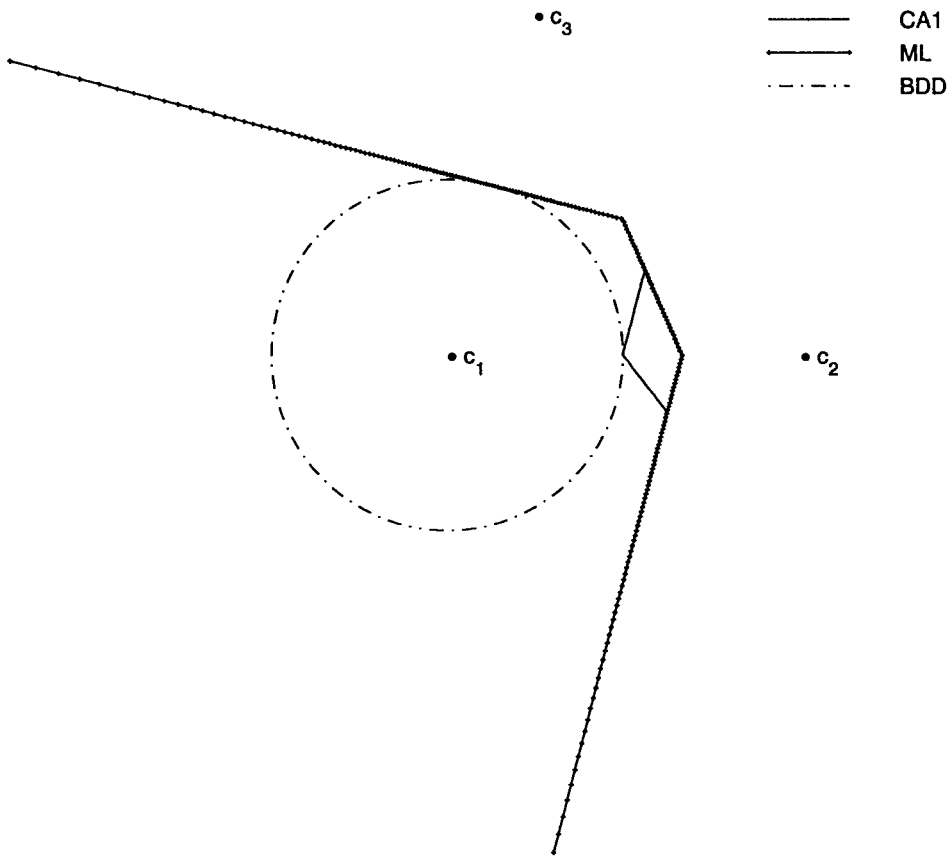

Fig. 1. Chase Algorithm 1, borders of decision regions for codeword c_1: c_2 noncodeword; c_3 codeword.

Corollary 1: Chase algorithms do not have pseudo nearest neighbors.

Additionally, as proved in the Appendix for the new definition of a conventional neighbor, we have

Proposition 1: Chase algorithms have $\binom{n}{d}$ conventional nearest neighbors.

It follows that, for Chase algorithms, the approximation of the union bound as described by (5), is

$$Pe(X_i) \approx \binom{n}{d} Q\left(\frac{d_{\min}}{2\sigma}\right). \tag{10}$$

From here and throughout this section, the term nearest neighbor refers to a conventional nearest neighbor. Later, an improvement to this bound will be presented. This improvement will be achieved by virtue of the fact that a noncodeword nearest neighbor induces, locally, a different decision region than a codeword nearest neighbor.

A. Chase Algorithm 1

CA1 is a rather inefficient decoding process requiring $\binom{n}{\lfloor d/2 \rfloor}$ binary algebraic decoding trials and producing at most that number of candidate codewords. Note that this number can be even greater than the overall number of codewords 2^k. According to (5), a large increase in the number of nearest neighbors should cause a large degradation in the performance; surprisingly, CA1 performs practically optimal [3] (see also Section V) in spite of its large number of nearest neighbors. This contradiction can be easily resolved by recalling that there are various types of nearest neighbors differently affecting the decision region [2]. The bound (5) does not take this fact into consideration.

Fig. 1 demonstrates this phenomenon for the $[8,4,4]$ extended Hamming code. A two-dimensional cross section of R^8 is presented. The plane shown is defined by the three points c_1, c_2, c_3, where $c_1 = \mathbf{1}$, c_2 is a noncodeword nearest neighbor to c_1, and c_3 is a nearest neighbor which is a codeword. Fig. 1 shows fragments from the boundaries of the decision regions $D(\mathbf{1})$, and $V(\mathbf{1})$, as well as the hyper-sphere $B(\mathbf{1}, (d_{\min}/2))$ They are denoted by CA1, ML, and BDD, respectively. The codeword c_3 induces, locally, a straight boundary line that is tangent to $B(\mathbf{1}, (d_{\min}/2))$ midway between c_3 and c_1. The noncodeword nearest neighbor induces a tipped shape boundary line that touches the hyper-sphere. This figure suggests that a noncodeword nearest neighbor has lesser probability of causing decoding error than a codeword nearest neighbor.

In order to tighten the approximation for the union bound (5), we will quantify the difference in the probabilities for causing a decoding error, between a codeword and a noncodeword nearest neighbors. Assume w.l.o.g. that the $\mathbf{1}$ codeword was transmitted. Denote by X_0 a codeword nearest neighbor and by X_{BDD} a noncodeword nearest neighbor. Denote by $\boldsymbol{y}_0 = \frac{\mathbf{1}+X_0}{2}$ the midpoint between X_0 and $\mathbf{1}$, and by $\boldsymbol{y}_{\mathrm{BDD}} = \frac{\mathbf{1}+X_{\mathrm{BDD}}}{2}$ the midpoint between X_{BDD} and $\mathbf{1}$. For clarity of exposition, let T_0 and T_{BDD} denote the hyper-spheres with

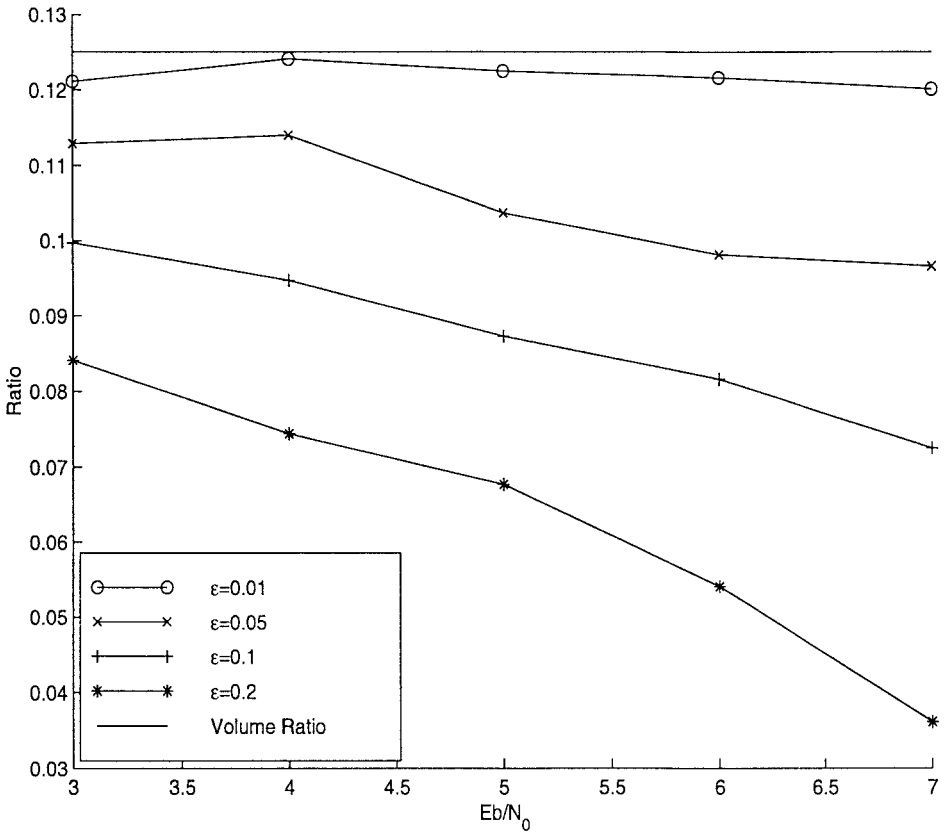

Fig. 2. Chase Algorithm 1, $[8, 4, 4]$ code: volume ratio versus empirical probability ratio.

a small radius ϵ, centered on $\boldsymbol{y}_0$ and $\boldsymbol{y}_{\mathrm{BDD}}$, respectively. Denote by e_0 and by e_{BDD} the regions in T_0 and T_{BDD}, where decoding error occurs: $e_0 = \{p\colon\ p \in T_0 \setminus D(\mathbf{1})\}$ $e_{\mathrm{BDD}} = \{p\colon\ p \in T_{\mathrm{BDD}} \setminus D(\mathbf{1})\}$. Note that e_0 and e_{BDD} are the same for all X_0, X_{BDD}, and for any "transmitted" codeword. This is evident for e_0, while for e_{BDD} it follows from the fact that for a binary linear code with the mapping (2), the analyzed algorithms [3], [6], do not make any distinction between codewords or symbol values $\{0, 1\}$. Since e_0 and e_{BDD} are close to the transmitted codeword (relative to the complete error region $\{p\colon\ p \in R^n \setminus D(\mathbf{1})\}$), the probability ratio

$$\eta = \frac{P(e_{\mathrm{BDD}})}{P(e_0)} \triangleq \frac{P(\boldsymbol{r} \in e_{\mathrm{BDD}})}{P(\boldsymbol{r} \in e_0)}$$

provides significant information on the relation between a codeword and a noncodeword nearest neighbor in terms of their contribution to the error probability. Using η as a multiplying factor of N_{BDD} yields a better approximation for the upper bound (5). In general, $\eta(\rho)$ is defined as the ratio between $P(e_{\mathrm{BDD}})$ corresponding to the error region whose closest point is at distance $\frac{1}{2}\rho$ from $\mathbf{1}$ and $P(e_0)$ corresponding to the hemisphere-shaped error region at distance $\frac{1}{2}\rho$ from $\mathbf{1}$. In this work, our attention is restricted to the particular case $\eta = \eta(d_{\min})$ since the first term of the union bound, corresponding to the nearest neighbors, usually (and the analyzed algorithms [3], [6], are no exception) dominates the error performance.

Computing η involves integrating the term e^{x^2} over a volume whose boundaries are rather complex. This is a difficult problem, for which no explicit expression is known. One way of approximating the probability ratio η is by using the ratio, $V(e_{\mathrm{BDD}})/V(e_0)$, between the volumes of e_{BDD} and e_0. For ϵ small, alternatively for high enough SNR, all the points contained in the hyper-spheres T_0 and T_{BDD} have approximately the same probability. In light of the above, we can write

$$(P(e_{\mathrm{BDD}})/P(e_0)) \approx (V(e_{\mathrm{BDD}})/V(e_0)).$$

The region e_0 is simply a hemisphere, its volume is thus given by [4] $V(e_0) = \frac{1}{2}V_n\epsilon^n$, where $V_n = (\pi^{n/2}/\Gamma((n/2)+1))$. The volume of the region e_{BDD} is given by the next theorem.

Theorem 2: For Chase Algorithm 1

$$V(e_{\mathrm{BDD}}) = (V_n\epsilon^n/2^d).$$

Proof: Let $g(\boldsymbol{y}_{\mathrm{BDD}})$ be the set of indices such that $y_{\mathrm{BDD}_i} = 0$. Note that $g(\boldsymbol{y}_{\mathrm{BDD}})$ is one of the $\binom{n}{d}$ sets $S(d)$ defined from (6). T_{BDD}, whose volume is $V(T_{\mathrm{BDD}}) = V_n\epsilon^n$, can be partitioned into 2^d equal and disjoint portions as follows. Each portion is composed of all the vectors $\boldsymbol{p} \in T_{\mathrm{BDD}}$, such that for every index $i \in g(\boldsymbol{y}_{\mathrm{BDD}})$, p_i has a constant sign. For instance, one such portion is obtained by letting all the symbols with indices belonging to $g(\boldsymbol{y}_{\mathrm{BDD}})$ to have a positive soft value. It is clear that all portions have the same volume, and the union of all the portions is T_{BDD}. When ϵ is small enough, the transmitted codeword is the closest codeword to

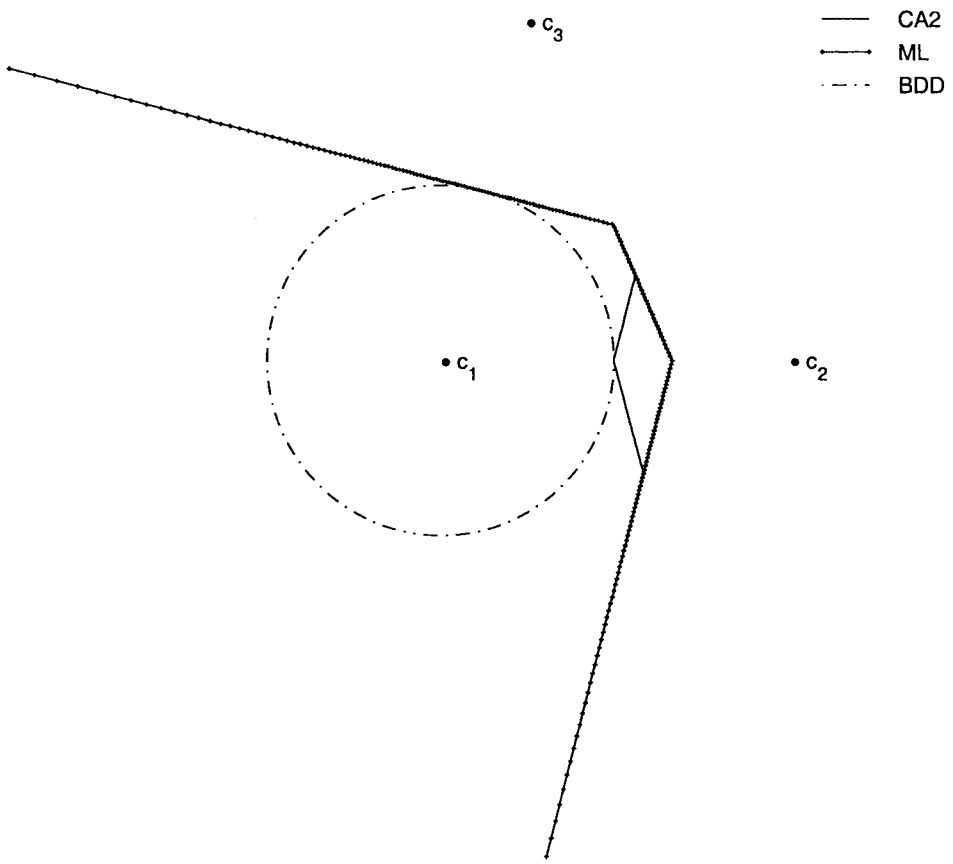

Fig. 3. Chase Algorithm 2, borders of decision regions for codeword c_1: c_2 noncodeword; c_3 codeword.

every point in $T_{\rm BDD}$. Hence, decoding error will occur (within $T_{\rm BDD}$) only if the transmitted codeword is not one of the generated candidates. It follows from [3, proof of Theorem 1] that this occurs only in the portions where all symbols with indices belonging to $g(\boldsymbol{y}_{\rm BDD})$ have negative soft value. In all other portions, no decoding error will occur as there will be fewer than d hard-decision errors and the transmitted codeword will be one of the candidates. This leads us to the conclusion that decoding error occurs in only $1/2^d$ of the volume $V(T_{\rm BDD})$. Clearly now, $V(e_{\rm BDD})$ is the same for all $X_{\rm BDD}$. □

Consequently, for Chase Algorithm 1, we obtain

$$\eta = \frac{P(e_{\rm BDD})}{P(e_0)} \approx \frac{V(e_{\rm BDD})}{V(e_0)} = \frac{1}{2^{(d-1)}} \tag{11}$$

and, therefore,

$$Pe(X_i) \approx \left(N_0 + \frac{N_{\rm BDD}}{2^{(d-1)}}\right) Q\left(\frac{d_{\min}}{2\sigma}\right). \tag{12}$$

In fact, the volume ratio is an upper bound on the probability ratio [5]. The proof is rather lengthy and therefore only sketched in the following. Partition T_0 as $T_{\rm BDD}$ into 2^d disjoint portions. Clearly, there exists one portion $e_0^1 \in T_0$, obtained via isometric mapping of $e_{\rm BDD}$, such that $P(e_0^1) = P(e_{\rm BDD})$. All the portions of T_0 are isometrically equivalent (to each other) by construction; however, it can be shown that e_0^1 is the portion furthest away from $\mathbf{1}$ and hence satisfying $P(e_0^1) \le P(e_0^j)$ for $j = 1, \cdots, 2^d$. Then clearly

$$\frac{P(e_0)}{P(e_{\rm BDD})} = \frac{P(e_0)}{P(e_0^1)} = \frac{1}{P(e_0^1)} \sum_{j=1}^{2^{d-1}} P(e_0^j) > 2^{d-1} = \frac{V(e_0)}{V(e_{\rm BDD})}.$$

It should be emphasized that the proof is independent of the SNR, and holds for a wide range of ϵ. This is demonstrated in Fig. 2 for the $[8,4,4]$ code by means of computer simulation The trace representing the volume ratio $1/2^{4-1}$ is uniformly higher than the probability ratio traces corresponding to several values of ϵ, ranging from 0.01 to 0.2, and for the entire SNR range. Similar behavior has also been observed for the rest of the algorithms discussed in this work.

B. Chase Algorithm 2

CA2 trades performance for computational complexity. It is more efficient than CA1, as it considers just a subset of the error patterns used by the latter. As in Fig. 1, Figs. 3–5 present two-dimensional cross sections of R^8 for the $[8,4,4]$ extended Hamming code. In Fig. 3, $c_1 = \mathbf{1}$, c_2 is a noncodeword nearest neighbor to c_1, and c_3 is a nearest neighbor which is a codeword. Clearly, the noncodeword nearest neighbor c_2 affects the probability of decoding error less than a nearest neighbor which is a codeword. In Fig. 4, $c_1 = \mathbf{1}$ and c_2, c_3 are noncodeword nearest neighbors to c_1. Here, a nearest neighbor

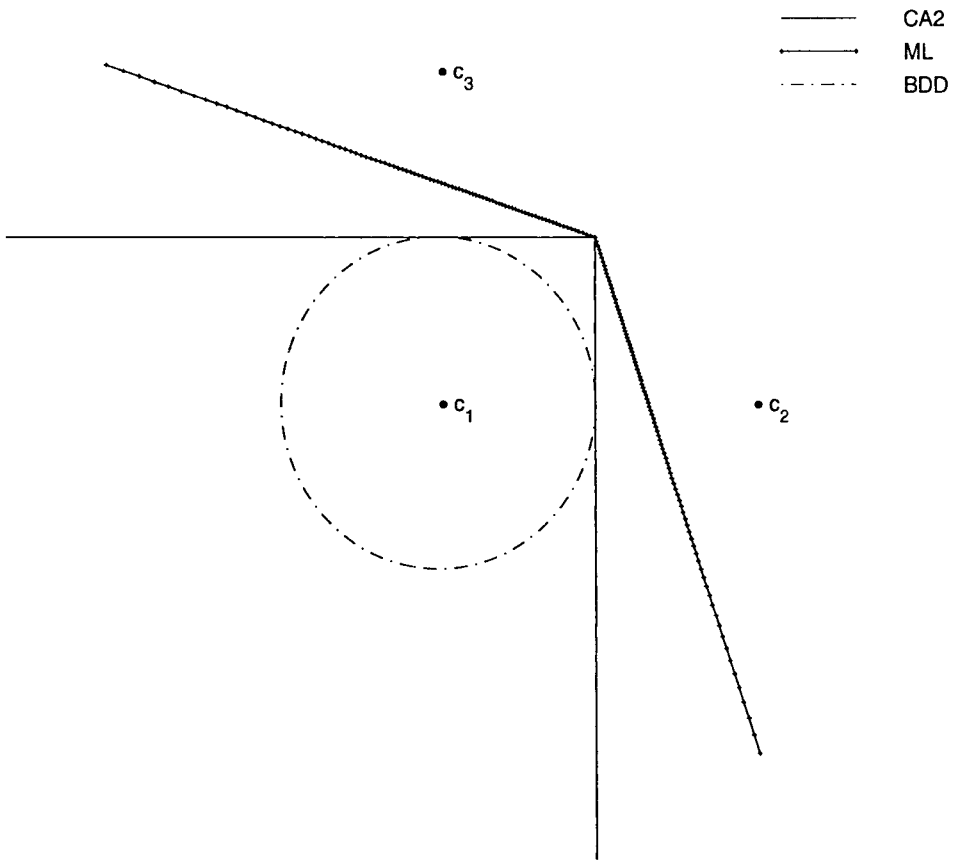

Fig. 4. Chase Algorithm 2, c_2 and c_3 are noncodeword neighbors.

which is not a codeword seems to behave as if it were a codeword. Note that in Figs. 3 and 4 the same nearest neighbor, c_2, behaves differently since the figures present different cross sections. The points c_1, c_2, and c_3 in Fig. 5 are the same as in Fig. 3. The figure presents a fragment from the border of the decision regions of Chase Algorithms 1 and 2, on the same plot. Note that the same noncodeword neighbor, c_2, contributes to the decision error of CA2 more than to CA1. This difference in the decision regions of c_1 demonstrates where CA1 gains (in performance) over CA2. We now turn to compute $V(e_{\mathrm{BDD}})$ for CA2. But first, the following conclusion is drawn from [3, proof of Theorem 1].

Conclusion 1: Let X, $\boldsymbol{r}$, and Y be the transmitted, received, and hard-decision vectors, respectively. If there are at least $\lceil d/2 \rceil$ errors in Y outside the set of $\lfloor d/2 \rfloor$ symbols with the smallest level of confidence, the transmitted codeword will not be a candidate, and an error will occur while decoding $\boldsymbol{r}$.

Without loss of generality, assume that the codeword $\mathbf{1}$ was transmitted. X_0, X_{BDD}, $\boldsymbol{y}_0$, $\boldsymbol{y}_{\mathrm{BDD}}$, e_0, e_{BDD}, and T_{BDD} are defined as before. The next lemma gives the the volume of the region e_{BDD}.

Theorem 3: for Chase Algorithm 2

$$V(e_{\mathrm{BDD}}) = V_n \epsilon^n / 2^{\lceil d/2 \rceil}.$$

Proof: Assume w.l.o.g. that the set of d indices, satisfying $y_{BDD_i} = 0$, is $\{1, 2, \cdots, d\}$. T_{BDD} can be partitioned into $d!$ equal and disjoint portions. Each portion comprises all the points (vectors) in T_{BDD} for which the confidence levels of the first d symbols are ordered similarly. Denote by T^1_{BDD} the portion satisfying

$$T^1_{\mathrm{BDD}} = \{\boldsymbol{t} | \boldsymbol{t} \in T_{\mathrm{BDD}} \,\&\, |t_1| \leq |t_2| \leq \cdots \leq |t_d|\}.$$

Clearly, $V(T^1_{\mathrm{BDD}}) = (V_n \epsilon^n / d!)$. The portion T^1_{BDD} can be further partitioned into 2^d subportions, where each subportion is comprised of all the vectors, such that the soft values of the first d symbols have constant signs. Note that since ϵ is small, hard-decision errors can occur only within the first d symbols. From Conclusion 1, decoding error will occur only in those subportions of T^1_{BDD}, for which the $\lceil d/2 \rceil$ symbols with indices $\{\lfloor d/2 \rfloor + 1, \cdots, d\}$ have negative soft values. There are altogether $2^{\lfloor d/2 \rfloor}$ such subportions in T^1_{BDD}, corresponding to all combinations of signs in the first $\lfloor d/2 \rfloor$ symbols. (In all other subportions the transmitted codeword $\mathbf{1}$ will be one of the generated candidates, and hence selected as the decoder output.) We thus have

$$V(e^1_{\mathrm{BDD}}) = V(T^1_{\mathrm{BDD}})\,(2^{\lfloor d/2 \rfloor}/2^d) = V(T^1_{\mathrm{BDD}})\,(1/2^{\lceil d/2 \rceil}).$$

Finally, since the above derivation is exactly the same for any of the $d!$ portions of T_{BDD}, we get

$$V(e_{\mathrm{BDD}}) = (V_n \epsilon^n / 2^{\lceil d/2 \rceil}). \qquad \square$$

Consequently, for Chase Algorithm 2

$$\eta \approx \frac{1}{2^{\lceil d/2 \rceil - 1}} \tag{13}$$

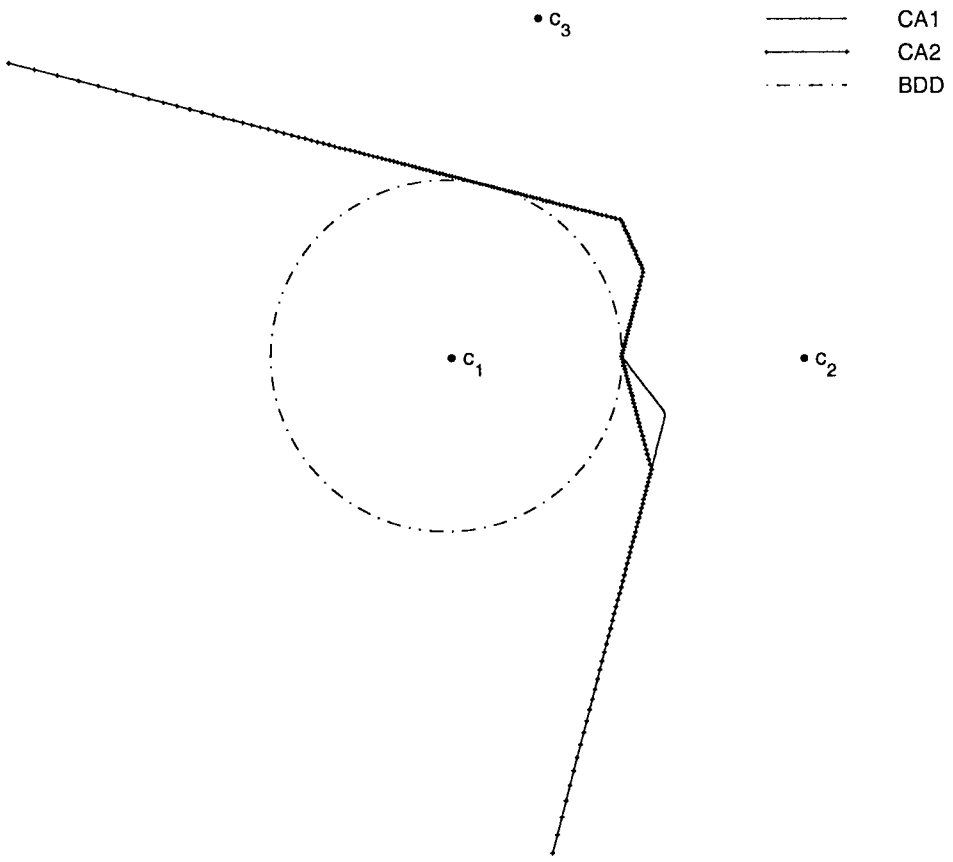

Fig. 5. Chase Algorithm 1 and 2, different decision borders for the same point c_2.

and, therefore,

$$Pe(X_i) \approx \left(N_0 + \frac{N_{\mathrm{BDD}}}{2^{\lceil d/2 \rceil - 1}}\right) Q\left(\frac{d_{\min}}{2\sigma}\right). \tag{14}$$

Equations (11) and (13), reveal an interesting property of Chase Algorithms 1 and 2, respectively. The contribution to error probability of a noncodeword nearest neighbor drops exponentially with d and $d/2$, for CA1 and CA2, respectively As in the case of CA1, the volume ratio is an upper bound on the probability ratio. The proof is not much different than for CA1 and is supported by simulation results (not presented in this paper). Recently, an upper bound has been derived on the bit-error rate (BER) performance of CA2 [8]. This bound is based on a probabilistic rather than a geometrical method and is more complex to evaluate than the proposed bound.

C. Chase Algorithm 3

CA3 is the most efficient of the Chase algorithms, the price is further degradation in performance. The decision region of CA3 has quite an interesting shape. In Fig. 6 , $c_1 = \mathbf{1}$, c_2, and c_3 are noncodeword nearest neighbors to c_1. In the depicted cross section, the decision region is not at all tangent to the bounded distance hyper-sphere at the midpoints $\frac{c_1+c_2}{2}$ and $\frac{c_1+c_3}{2}$ as is usually expected. Referring to c_1 and c_2, the magnitude of the noise can be greater than $d_{\min}/2$, on the imaginary line connecting c_1 and c_2, so that there are d symbol errors in the hard-decision vector, yet CA3 decodes correctly. We refer to this phenomenon as a "*tunnel effect*." This figure, however, is misleading. It seems to contradict Proposition 1, for it appears that there exists a hyper-sphere centered on $\frac{c_1+c_2}{2}$, in which no decoding error occurs. Fig. 7 presents a shifted and rotated cross section about the midpoint $\frac{c_1+c_2}{2}$ (p_1, p_2, and p_3 were chosen such that $\frac{p_1+p_2}{2} = \frac{c_1+c_2}{2}$), there is no tunnel here. In fact, the decision region in this cross section appears just as if c_2 were a codeword.

Figs. 6 and 7 also indicate that computing the volume ratio for CA3 is not an easy task. While for the previous algorithms a necessary and sufficient condition for decoding error was derived, for CA3 we develop recursive formulas for computing the volume ratio and thus eliminating the need for such a condition.

Assume w.l.o.g. that the codeword $\mathbf{1}$ was transmitted. Let X_{BDD}, $\boldsymbol{y}_{\mathrm{BDD}}$, T_{BDD}, and T^1_{BDD} be defined as before. Let e^1_{BDD} be the portion of T^1_{BDD} where decoding error occurs. Recall that $V(T^1_{\mathrm{BDD}}) = (V_n \epsilon^n / d!)$, and that $\boldsymbol{y}_{\mathrm{BDD}}$ is 0 in the first d symbols and 1 in the next $n-d$ symbols. T^1_{BDD} can be further partitioned into 2^d equal and disjoint subportions, each of volume $V(T^1_{\mathrm{BDD}})/2^d$. Clearly, within $V(\mathbf{1})$, CA3 will fail to decode correctly *iff* the transmitted codeword is not one of the generated candidates. Henceforth, we shall also assume that the symbols are arranged according to their confidence values in a nondecreasing order.

Let us define the *L-order Chase Algorithm 3* as the original CA3, only considering a smaller set of error patterns. This set

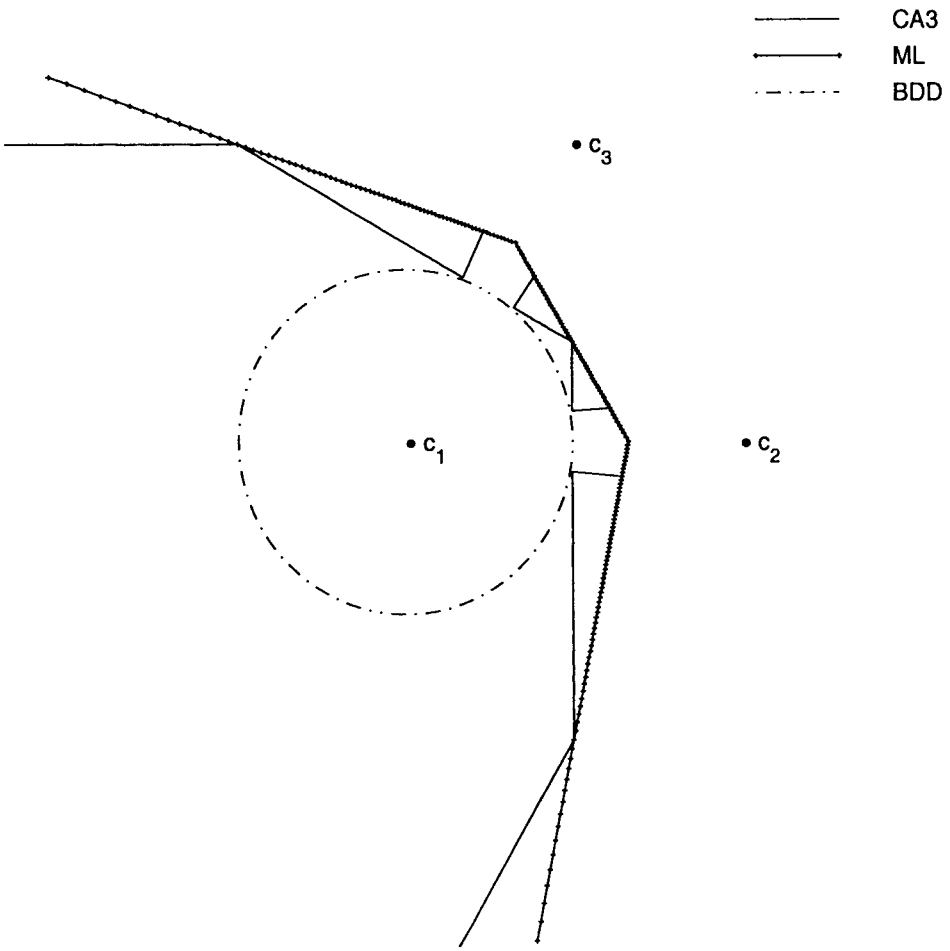

Fig. 6. Chase Algorithm 3, *tunnel effect*: c_2 and c_3 are noncodeword neighbors.

is comprised of the all-zero error pattern, and the error patterns with 1's in the $d-1, d-3, \cdots, d-L$ symbols with the lowest confidence levels. Finally, we denote by $a_{i,j}^{L}$ the number of combinations (error patterns) of exactly i errors among the symbols with indices $d-L+1, \cdots, d$—assuming j errors among the first $d-L$ symbols and no errors among the last $n-d$ symbols—such that the *L-order* CA3 fails to generate the transmitted codeword. Note that L is an odd integer, where for d odd, L can range between 1 and d, and for d even, L can range between 1 and $d-1$.

Example 1: For $d=6$, $a_{2,1}^{3}=2$. If and only if there are errors in the symbols with indices $4,6$ or $5,6$, and assuming that there is only one error in the symbols with indices $1,2,3$, than the transmitted codeword will not be generated by the *3-order CA3*.

$a_{i,j}^{L}$ will be employed in a recursive manner to compute $V(c_{\mathrm{BDD}})$. The general notion is demonstrated by the following. For, say, d odd, $a_{i,0}^{d}$ represents the number of (equal) subportions of T_{BDD}^{1}, each subportion corresponding to i errors, such that CA3 fails to generate the transmitted codeword. Thus compute

$$\alpha(d)=\sum_{i=1}^{d-1} a_{i,0}^{d},$$

and then

$$V(c_{\mathrm{BDD}}^{1})=(\alpha(d)/2^{d})V(T_{\mathrm{BDD}}^{1}).$$

Not all values of $a_{i,j}^{L}$ need be computed (clearly, $a_{0,0}^{d}=a_{d,0}^{d}=0$). The following proposition, proved in the Appendix, summarizes all the cases for which $a_{i,j}^{L}=0$.

Proposition 2: $a_{i,j}^{L}=0$ if any of the following conditions holds:

i) $i<0 \quad \text{or} \quad i>L$;
ii) $j<0 \quad \text{or} \quad j>d-L$;
iii) $i+j \le \frac{d-1}{2}$;
iv) $i+(d-L)-j \le \left\lfloor \frac{d-1}{2} \right\rfloor$.

The next lemma establishes the recursive relation between $\{a_{i,j}^{L}\}_{i,j\in Z}$ and $\{a_{i,j}^{L-2}\}_{i,j\in Z}$.

Lemma 1: If i,j,d, and L do not satisfy any of the conditions of Proposition 2, then

$$a_{i,j}^{L}=a_{i,j}^{L-2}+2a_{i-1,j+1}^{L-2}+a_{i-2,j+2}^{L-2}. \tag{15}$$

Proof: Let us denote the set of $a_{i,j}^{L}$ error patterns by $p_{i,j}^{L}$. The proof consists of four parts as follows. I) It is first proved that for every error pattern $p\in p_{i,j}^{L-2}$, there exists at least one pattern $p'\in p_{i,j}^{L}$. Let p' represent the same i errors as does p

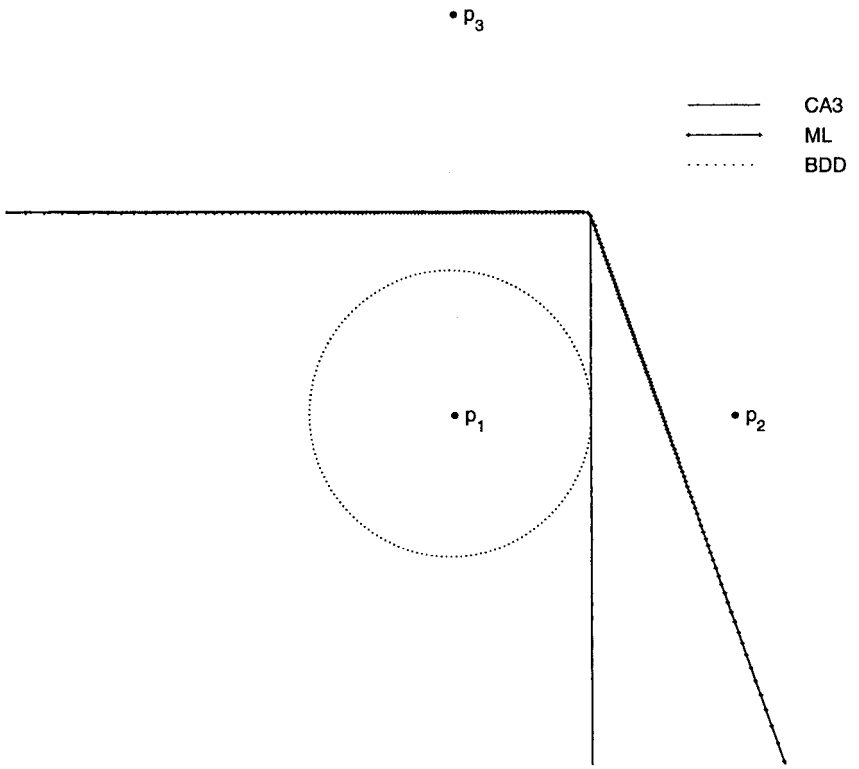

Fig. 7. Chase Algorithm 3, a shifted and rotated version of Fig. 6, no tunnel here.

among the symbols with indices $d-L+3,\cdots,d$, and no errors in the indices $d-L+1$, $d-L+2$. For the error pattern p', the *L-order* CA3 will fail to generate the transmitted codeword in any of the decoding trials: $i+(d-L)-j>\lfloor d-1/2\rfloor$ guarantees failure in the trial where the $d-L$ lowest confidence symbols are complemented; $p\in p_{i,j}^{L-2}$ guarantees failure in all the remaining trials. Thus evidently, $p'\in p_{i,j}^{L}$. II) It is shown that for every error pattern $p\in p_{i-1,j+1}^{L-2}$, there exist at least two patterns $\{p',p''\}$ that belong to $p_{i,j}^{L}$. Simply, let p', respectively, p'', represent the same $i-1$ errors as does p, among the symbols with indices $d-L+3,\cdots,d$, and one error in the symbol with index $d-L+1$, respectively, $d-L+2$. Using the above arguments, it is straightforward to verify that $\{p',p''\}\in p_{i,j}^{L}$. III) Finally, for every pattern $p\in p_{i-2,j+2}^{L-2}$, there exists at least one pattern $p'\in p_{i,j}^{L}$. The pattern p' represents the same $i-2$ errors in the symbols with indices $d-L+3,\cdots,d$, and two errors in the symbols with indices $d-L+1$, $d-L+2$. Thus far, we have shown that

$$a_{i,j}^{L}\geq a_{i,j}^{L-2}+2a_{i-1,j+1}^{L-2}+a_{i-2,j+2}^{L-2}.$$

IV) The opposite inequality

$$a_{i,j}^{L}\leq a_{i,j}^{L-2}+2a_{i-1,j+1}^{L-2}+a_{i-2,j+2}^{L-2}$$

is easily derived using similar arguments. For every pattern $p'\in p_{i,j}^{L}$ as described in Part I above, there exists at least one pattern $p\in p_{i,j}^{L-2}$, for instance the pattern p described in Part I. For every pair of error patterns $\{p',p''\}\in p_{i,j}^{L}$ as described in Part II above, there exists at least one error pattern $p\in p_{i-1,j+1}^{L-2}$, for instance the pattern p described in part II For every pattern $p'\in p_{i,j}^{L}$ as described in Part III above, there exists at least one pattern, $p\in p_{i-2,j+2}^{L-2}$, for instance the pattern p described in Part III. This concludes the proof of the opposite inequality and the Lemma. □

Computing the number of subportions of T_{BDD}^{1} in which decoding error occurs is rather simple using Proposition 2 and Lemma 1. For d odd, respectively, d even, compute $a_{i,j}^{d}$, respectively, $a_{i,j}^{d-1}$, for $i,j\in Z$. Let $\alpha(d)$ denote the result of the summation of all the relevant terms, i.e., for d odd

$$\alpha(d)=\sum_{i=1}^{d-1}a_{i,0}^{d}$$

and for d even

$$\alpha(d)=\sum_{i=1}^{d-1}(a_{i,0}^{d-1}+a_{i-1,1}^{d-1}).$$

Then $V(e_{\mathrm{BDD}}^{1})=(\alpha(d)/2^{d})V(T_{\mathrm{BDD}}^{1})$. It is now left only to establish the initial conditions for the recursive relations. When d is even (using Proposition 2), the nonzero initial terms are $a_{1,(d/2)}^{1}=a_{1,(d/2)-1}^{1}=1$. When d is odd, the only nonzero initial term is $a_{1,(d-1/2)}^{1}=1$.

In the table at the bottom of the following page, $\alpha(d)$ is listed for some values of d. It is noteworthy that $\alpha(d)$ has the same order of magnitude as $2^{(d-1)}$, for the presented values. Thus $\alpha(d)/2^{(d-1)}$ is close to one, as can be seen in Table II of Section V. In the Appendix we give the complete derivation of $\alpha(d)$ for $d=5$, as an illustrating example. Also for CA3, simulation results show that the volume ratio is an upper bound on the probability ratio.

In conclusion, since the derivation of $V(e_{\mathrm{BDD}}^{1})$ is the same for any other T_{BDD}^{i}, then clearly $V(e_{\mathrm{BDD}})=(\alpha(d)/2^{d})V_{n}\epsilon^{n}$.

Consequently, for CA3, $\eta = (\alpha(d)/2^{d-1})$, and, therefore,

$$Pe(X_i) \approx \left(N_0 + \frac{\alpha(d)}{2^{d-1}} N_{\mathrm{BDD}}\right) Q\left(\frac{d_{\min}}{2\sigma}\right). \qquad (16)$$

IV. Performance Analysis for the GMD Algorithm

The GMD algorithm [6] is one of the first suboptimal decoding algorithms. It is noteworthy that the results recently presented in [12] and [7] suggest a modified GMD algorithm which has the same number of nearest neighbors as maximum-likelihood decoding, i.e., $N_{0,\mathrm{eff}} = N_0$. In [12] it is proved that the GMD decoding algorithm is BDD. Moreover, from [12, proof of Theorem 1] it follows that the GMD algorithm does not have pseudo nearest neighbors. Thus by a nearest neighbor we shall henceforth be referring to a conventional nearest neighbor. The above properties allow us to evaluate the performance of the GMD algorithm by using (5). This bound, however, can be tightened by taking into consideration the different effect of the nearest neighbors on the decision region.

Assuming the same geometrical scenario (for X_{BDD}) and notations as with CA3, we next describe a method for calculating the number of subportions of T^1_{BDD} in which decoding error occurs. Let us define the *L-order GMD algorithm* as the original GMD algorithm, only considering a smaller set of *erasure patterns*. This subset is comprised of the erasure patterns of the $d-L, d-L+2, \cdots, d-1$, symbols with the lowest confidence levels. Denote by a^j_{d-L} the number of combinations (error patterns) of exactly j hard-decision errors, among the symbols with indices $d-L+1, \cdots, d$, such that the *L-order* GMD algorithm fails to generate the transmitted codeword $\mathbf{1}$. The following examples are given to clarify this definition.

Example 2: a^j_{d-d} represents the number of combinations of j errors among the d symbols with the lowest confidence levels, such that the GMD algorithm fails to generate the transmitted codeword. Note that each combination corresponds to one of the 2^d portions of T^1_{BDD}.

Example 3: For $d = 8, a^0_{8-1} = 0$. Since the received vector will undergo $d-1$ erasers but it contains no hard-decision errors—the EE decoder will generate the transmitted codeword.

Example 4: For $d = 8, a^2_{8-3} = 2$. The received vector undergoes two erasure trials: the first five, and then first seven symbols are erased. When the symbols with indices $7, 8$ are in error, the EE decoder will fail to generate the transmitted codeword in both erasure trials. The same holds for the symbols with indices $6, 8$.

a^j_{d-L} will be employed for calculating the number of subportions of T_{BDD} in which decoding error occurs. The terms a^j_{d-L} needed for this task can be computed recursively based on the following two lemmas. Lemma 2 is readily applicable for odd values of d.

Lemma 2: For L odd, and $1 \le L \le d$

$$a^j_{d-L} = \begin{cases} a^j_{d-L+2} + 2a^{j-1}_{d-L+2} + a^{j-2}_{d-L+2}, & \text{for } \lceil \frac{L}{2} \rceil \le j \le L \\ 0, & \text{otherwise.} \end{cases}$$

Proof: Clearly, $j > L$ is irrelevant because the number of errors cannot be greater than the number of symbols. Also, if $j < \lceil L/2 \rceil$, when erasing the $d-L$ symbols with the lowest confidence levels the transmitted codeword will always be generated by the EE decoder. The remainder of the proof closely follows the proof of Lemma 1, with the exception that GMD decoding uses erasures rather then bit completion. □

For even values of d the next lemma is employed along with Lemma 2 in a complimentary fashion, as will be described below.

Lemma 3: For d even, and $(d/2) \le j \le d$

$$a^j_{d-d} = a^j_{d-(d-1)} + a^{j-1}_{d-(d-1)}.$$

Proof: Let us denote the set of a^j_{d-L} error patterns by p^j_{d-L}. First, it is shown that for every pattern $p \in p^j_{d-(d-1)}$ there exists at least one pattern $p' \in p^j_{d-d}$. p' is the pattern representing j errors in the same positions as in p. For p', the order-d GMD algorithm, which is simply the original GMD algorithm, will fail to generate the transmitted codeword in any of the decoding trials: the decoding trial with no erasures (conventional algebraic decoder) will fail since $j \ge (d/2)$; failure is guaranteed in all remaining decoding trials due to the fact that $p \in p^j_{d-(d-1)}$. Next, it is shown that for every pattern $p \in p^{j-1}_{d-(d-1)}$ there exists at least one pattern $p' \in p^j_{d-d}$. Let p' represent the same $j-1$ errors as does p, and one additional error in the symbol with the lowest confidence level. Using the above arguments, it is straightforward to verify that $p' \in p^j_{d-d}$. In conclusion, we have shown that

$$a^j_{d-d} \ge a^j_{d-(d-1)} + a^{j-1}_{d-(d-1)}.$$

The opposite inequality, i.e.,

$$a^j_{d-d} \le a^j_{d-(d-1)} + a^{j-1}_{d-(d-1)}$$

is derived using similar arguments. □

The initial conditions, given below, are easy to derive

$$a^j_{d-1} = \begin{cases} 1, & \text{for } j = 1 \\ 0, & \text{otherwise.} \end{cases}$$

The number of subportions of T^1_{BDD}, in which decoding error occurs, is obtained in the following manner.

- For d *odd*: use Lemma 2 to determine a^j_{d-d} for all j satisfying $\lfloor \frac{d-1}{2} \rfloor < j \le d$. Calculate the sum

$$\beta(d) = \sum_j a^j_{d-d}.$$

d	3	4	5	6	7	8	9	10	11	12	30	32
$\alpha(d)$	2	5	5	14	14	42	42	132	132	429	$\approx 35 \cdot 10^6$	$\approx 130 \cdot 10^6$

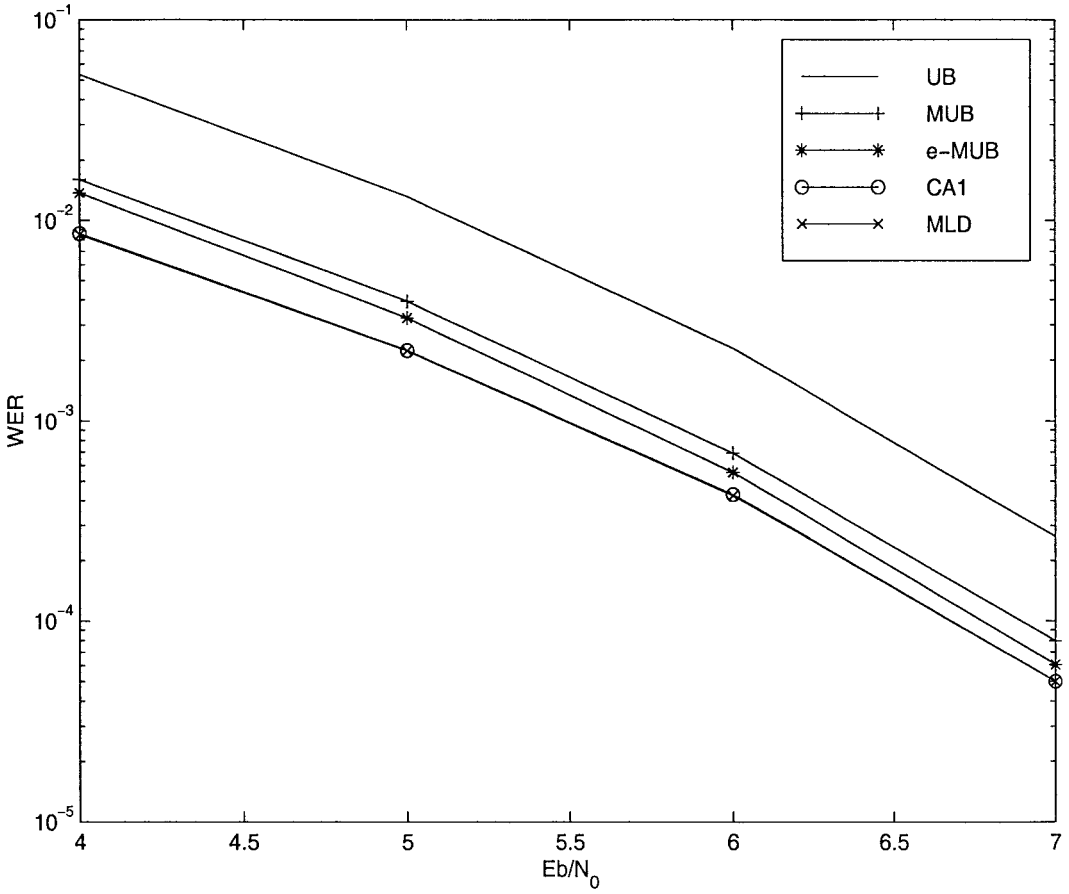

Fig. 8. Chase Algorithm 1, simulation results and bounds.

Out of the 2^d possible subportions of $T^1_{\rm BDD}$, $\beta(d)$ represents the number of subportions in which decoding error occurs.

- For d *even*: use Lemma 2 to determine $a^j_{d-(d-1)}$ for all j satisfying $\lfloor d-1/2 \rfloor < j \le d$. Calculate the sum

$$\gamma = \sum_j a^j_{d-(d-1)}.$$

From Lemma 3 it follows that

$$\sum_j a^j_{d-d} = 2 \sum_j a^j_{d-(d-1)}$$

hence let $\beta(d) = 2\gamma$. As in the previous case, $\beta(d)$ represents the number of subportions in which decoding error occurs.

In the table at the bottom of this page $\beta(d)$ is listed for some values of d. Similar to $\alpha(d)$, $\beta(d)$ has the same order of magnitude as $2^{(d-1)}$, thus $\beta(d)/2^{(d-1)}$ is close to one, see Table II of Section V.

Given $\beta(d)$, we may write $V(e^1_{\rm BDD}) = (\beta(d)/2^d)V(T^1_{\rm BDD})$. Since the derivation of $V(e^1_{\rm BDD})$ is exactly the same for any other portion $T^i_{\rm BDD}$ of $T_{\rm BDD}$, then clearly $V(e_{\rm BDD}) = (\beta(d)/2^d)V_n\epsilon^n$. Simulation results, not presented in this paper, show that the volume ratio is an upper bound on the probability ratio also for the GMD algorithm. Finally, for the GMD decoding algorithm, the approximated bound given by (5) can be tightened as follows:

$$Pe(X_i) \approx \left(N_0 + \frac{\beta(d)}{2^{d-1}} N_{\rm BDD}\right) Q\left(\frac{d_{\min}}{2\sigma}\right). \qquad (17)$$

V. Conclusions and Computer Simulations

A refined definition of the term *nearest neighbors* is suggested. Using this definition we identify and enumerate the nearest neighbors associated with the Chase decoding algorithms, not before proving that these algorithms are indeed bounded distance. Then, an improved approximated upper bound on the probability of codeword error for the Chase and GMD decoding algorithms is presented. This bound is based on the union bound approach while taking into consideration the different influence of the different types of nearest neighbors on the decision region.

In order to quantify this difference, we introduce the *probability ratio*, η. η represents the ratio between the error

d	3	4	5	6	7	8	9	10	11	12	30	32
$\beta(d)$	3	6	10	20	35	70	126	252	462	924	$\approx 155 \cdot 10^6$	$\approx 601 \cdot 10^6$

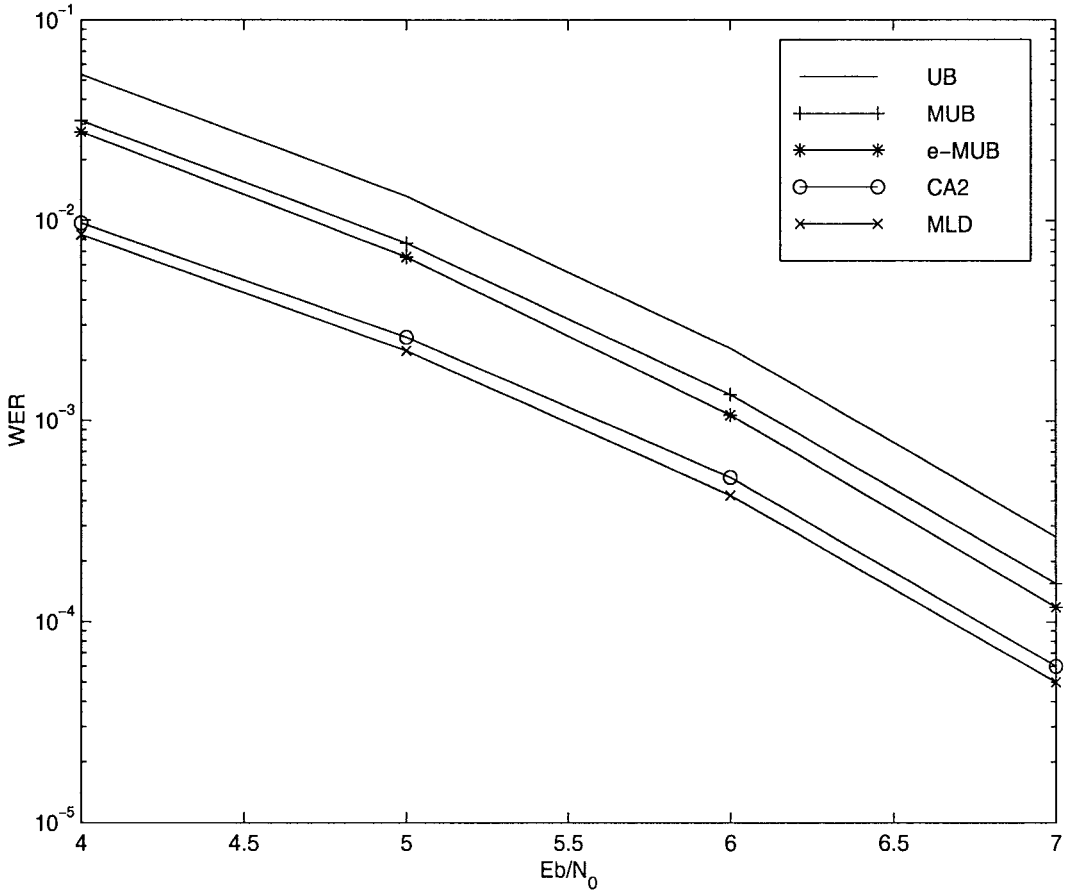

Fig. 9. Chase Algorithm 2, simulation results and bounds.

TABLE I
THE VOLUME RATIO

ALGORITHM	CA1	CA2	CA3	GMD
$\eta \approx$	$\frac{1}{2^{(d-1)}}$	$\frac{1}{2^{\lceil \frac{d}{2} \rceil - 1}}$	$\frac{\alpha(d)}{2^{(d-1)}}$	$\frac{\beta(d)}{2^{(d-1)}}$

contribution of a noncodeword nearest neighbor and a nearest neighbor which is a codeword. Since the probability ratio is too difficult to calculate directly, we approximate η by using the ratio between the corresponding error volumes. The improved bound is of the form $(N_0 + \eta N_{\mathrm{BDD}})Q(d_{\min}/2\sigma)$.

Table I summarizes the obtained volume ratios for the aforementioned decoding algorithms. Specific values of $\alpha(d)$ and $\beta(d)$ should be calculated according to the procedure in the corresponding sections. In Table II we list the approximated η associated with the decoding algorithm for several values of the minimum Hamming distance d. Note that the volume ratio is exponential in d for CA1 and CA2, but not for CA3 or GMD. This accounts for the different performance of the algorithms as obtained from simulations.

Finally, we present some simulation results. A simple example, the $[8, 4, 4]$ extended Hamming code, has been chosen to demonstrate the gain of the improved bound over the union bound. For each of the bounded distance decoding algorithms treated in this work, we plot the simulation results of: maximum-likelihood decoding (MLD); the bounded distance algorithm; and a modified union bound (e-MUB) based on the empirical computation of the ratio $P(e_{\mathrm{BDD}})/P(e_0)$. Additionally, we plot the error probability as given by the union bound (UB); and the modified union bound based on the volume ratio (MUB).

The results for CA1 are depicted in Fig. 8. As can be seen in this figure, CA1 is practically optimal. This may be explained as follows. Although the number of noncodeword nearest neighbors N_{BDD} is much higher than N_0, the contribution (to error probability) of a noncodeword neighbor is considerably smaller than that of a codeword neighbor. Note that at word-error rate (WER) of $\approx 4 \cdot 10^{-4}$ the modified bound, MUB, is only 0.25 dB from the actual simulation results, while the union bound is 0.8 dB away. The modified bound is much tighter than the union bound also for low SNR. The empirical bound, e-MUB, is even tighter, only 0.15 dB from the actual results. The 0.1-dB difference between MUB and e-MUB suggests that, in the case of CA1, the volume ratio is a very close approximation for the probability ratio.

CA2 is much more efficient than CA1. It involves only four algebraic decoding trials as compared to 28 trials required by CA1. Nevertheless, its performance, presented in Fig. 9, are only 0.125 dB worse than the optimal. For this algorithm, at WER of $\approx 2 \cdot 10^{-4}$, the empirical modified bound, the modified bound, and the original union bound, respectively, are 0.35, 0.45, and 0.7 dB away from the simulation results. Again, the 0.1-dB difference between

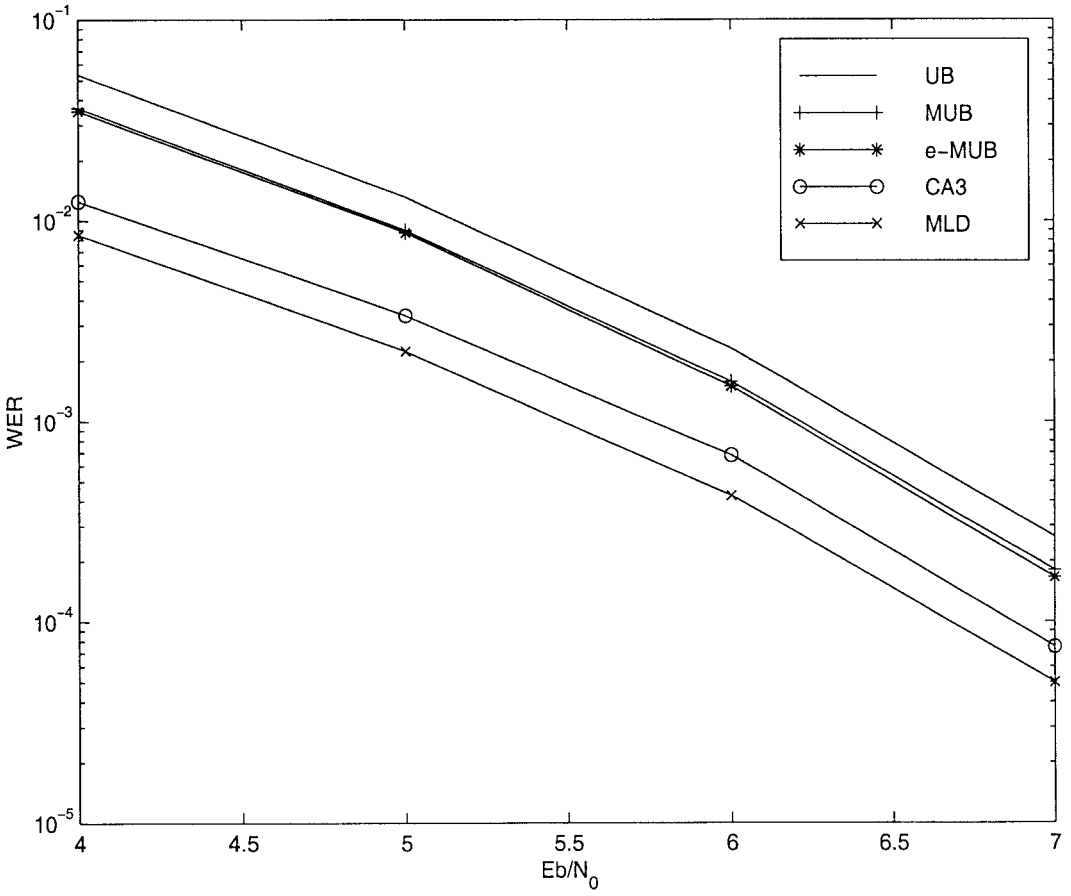

Fig. 10. Chase Algorithm 3, simulation results and bounds.

TABLE II
SOME SPECIFIC VALUES OF THE VOLUME RATIO

d	3	4	5	6	7	8	9	10
$CA1$	0.2500	0.1250	0.0625	0.0312	$1.562\cdot10^{-2}$	$7.812\cdot10^{-3}$	$3.9\cdot10^{-3}$	$1.95\cdot10^{-3}$
$CA2$	0.5000	0.5000	0.2500	0.2500	0.1250	0.1250	0.0625	0.0625
$CA3$	0.5000	0.6250	0.3125	0.4375	0.2187	0.3281	0.1640	0.2570
GMD	0.7500	0.7500	0.6250	0.6250	0.5496	0.5496	0.4922	0.4922

d	11	12	28	29	30	31
$CA1$	$9.765\cdot10^{-4}$	$4.88\cdot10^{-4}$	$3.72\cdot10^{-9}$	$1.862\cdot10^{-9}$	$9.31\cdot10^{-10}$	$4.65\cdot10^{-10}$
$CA2$	0.0312	0.0312	$6.1\cdot10^{-5}$	$6.1\cdot10^{-5}$	$3.05\cdot10^{-5}$	$3.05\cdot10^{-5}$
$CA3$	0.1280	0.2094	0.0361	0.0659	0.0329	0.0603
GMD	0.4512	0.4512	0.2989	0.2889	0.2889	0.2779

MUB and e-MUB suggests that the volume ratio is a very close approximation for the probability ratio.

CA3 is the most efficient among the Chase Algorithms, indeed, for the price of performance. At WER of $\approx 2\cdot10^{-4}$, it is 0.2 dB worse than the optimal, as can be seen in Fig. 10. The empirical modified bound and the modified bound almost coincide, 0.35 dB from the ML simulation results. The original union bound is 0.6 dB away from the ML simulation results. For this algorithm, the volume ratio is evidently closer to the probability ratio, even more than in the previous algorithms. Unfortunately, this may imply, certainly for the presented SNR range, that the probability ratio is not a tight enough approximation. Note that since all three algorithms have the same nearest neighbors $N_{0,\text{eff}}$, the original union bound cannot distinguish between them, and is thus the same for all. As noted in [2], however, in different decoding algorithms, the same neighbors may differently influence the decision region. This is evident from the presented simulation results, and indeed taken into consideration by the modified union bound.

The results for the GMD decoding algorithm are depicted in Fig. 11. The simulation range has been extended here up to SNR $= 9$ dB. At WER of $\approx 10^{-4}$, the GMD algorithm loses 0.25 dB as compared to the ML algorithm. That is 0.05 dB

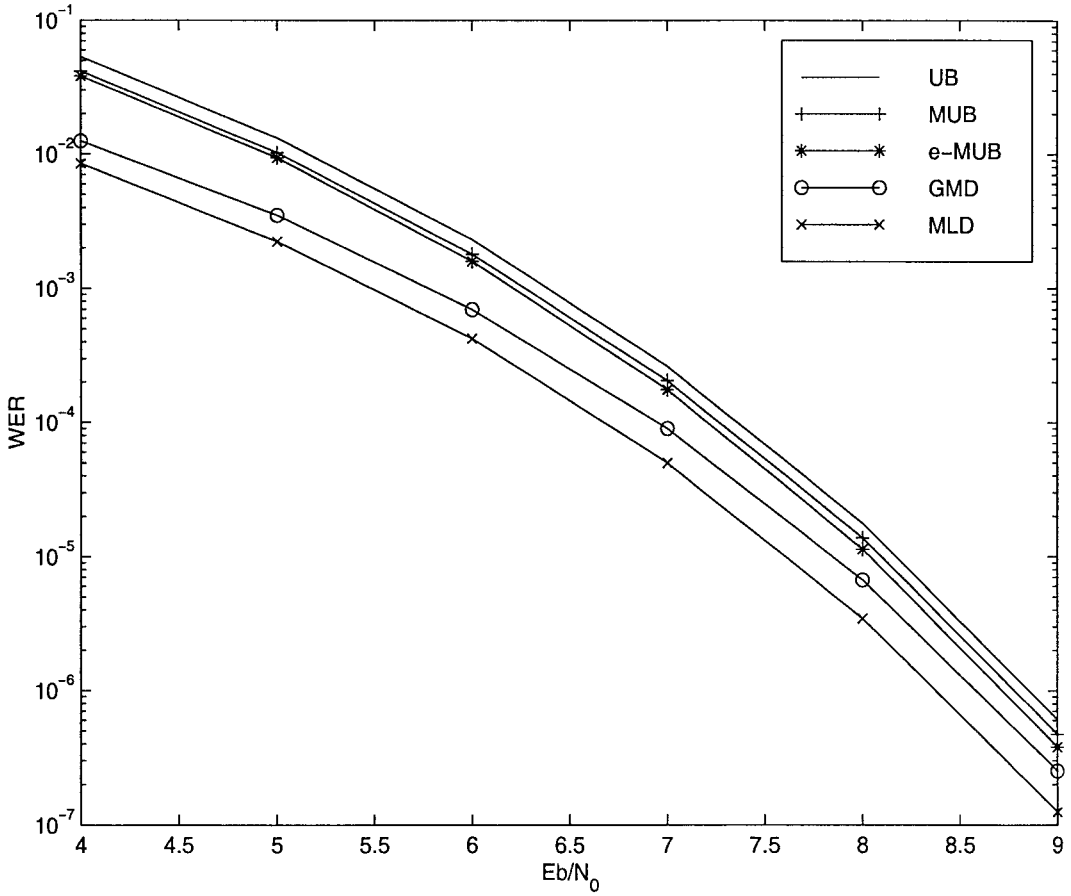

Fig. 11. GMD, simulation results and bounds.

more than CA3, while their decoding complexity and nearest neighbors are identical. The empirical modified bound, the modified bound, and the original union bound, respectively, are 0.3, 0.4, and 0.55 dB, away from the ML simulation results.

APPENDIX

Proof of Proposition 1

Assume (w.l.o.g.) that the $\mathbf{1} \in R^n$ codeword was transmitted. Let $\mathcal{V} \in \text{GF}(2)^n$ be a vector at Hamming distance d from the all-zero binary codeword $d(\mathcal{V}, \mathbf{0}) = d$.There are exactly $\binom{n}{d}$ such vectors. Denote by X the Euclidean version of V via the transformation (2). X has d symbols equal to -1, and $n-d$ symbols equal to 1. Let $\boldsymbol{y} \in R^n$ be the midpoint between X and $\mathbf{1}$, that is, $\boldsymbol{y} = \frac{X+\mathbf{1}}{2}$. $\boldsymbol{y}$ has d symbols equal to 0, and $n-d$ symbols equal to 1. Let $T_{\boldsymbol{y}}$ be the hyper-sphere with a small radius ϵ, centered on $\boldsymbol{y}$.

Chase Algorithm 1: Let $T_{\boldsymbol{y}}^1$ be the portion of $T_{\boldsymbol{y}}$, such that $T_{\boldsymbol{y}}^1 = \{\boldsymbol{t} | \boldsymbol{t} \in T_{\boldsymbol{y}}, t_i \leq 0, \forall y_i = 0\}$. According to [3], decoding failure will occur for every vector $\boldsymbol{t} \in T_{\boldsymbol{y}}^1$.This proves that X is a conventional nearest neighbor, as there is a region with nonzero volume, centered on the midpoint $\boldsymbol{y}$, where decoding error occurs.

Chase Algorithm 2: The error patterns considered by this algorithm are a subset of the patterns considered by CA1. This fact, along with the arguments used for CA1, concludes the proof.

Chase Algorithm 3: Although it has the same nearest neighbors as do CA1 and CA2, the proof is different in this case. Assume w.l.o.g. that $y_i = 0$ for $i \in \{1, \cdots, d\}$. For ϵ sufficiently small, such that $\forall\, \boldsymbol{t} \in T_{\boldsymbol{y}}$

$$\max_{i \in \{1, \cdots, d\}} |t_i| < \min_{j \in \{d+1, \cdots, n\}} |t_j|$$

$T_{\boldsymbol{y}}$ can be partitioned into $d!(n-d)!$ equal and disjoint portions. Each portion corresponds to one combination of ordering of the symbols according to their confidence levels. Let

$$T_{\boldsymbol{y}}^1 = \{\boldsymbol{t} | \boldsymbol{t} \in T_{\boldsymbol{y}} \,\&\, |t_1| \leq, \cdots, \leq |t_n|\}.$$

$T_{\boldsymbol{y}}^1$ can be further partitioned into 2^d equal and disjoint subportions, where each subportion is comprised of all the vectors $\boldsymbol{t} \in T_{\boldsymbol{y}}^1$, such that the soft values of the first d symbols have constant sign. Now, consider the subportion of $T_{\boldsymbol{y}}^1$ where the first $\lfloor d/2 \rfloor$ symbols have positive soft values (correct symbols) and the following $\lceil d/2 \rceil$ symbols have negative soft values (errors). Evidently, the volume of this subportion is nonzero, and CA3 will fail here. In conclusion, we have shown that every vector $\mathcal{V}$ as defined above, satisfies Definition 1, and is thus a conventional nearest neighbor for CA3. □

Proof of Proposition 2

Proposition 2 has several conditions, we separately consider each one. Condition i): clearly, there cannot be a negative number of hard-decision errors, or more than L errors, among

L symbols. Condition ii): similarly, there cannot be a negative number of hard-decision errors, or more than $d-L$ error, among $d-L$ symbols. Condition iii): since the overall number of errors satisfies $i+j \le \lfloor \frac{d-1}{2} \rfloor$, the algebraic decoder will certainly generate the transmitted codeword in the decoding trial corresponding to the all-zero error pattern. Condition iv): $(d-L)-j$ represents the number of errors in the first $d-L$ symbols when those are complimented, i.e., when the pattern $\boldsymbol{p}=(1^{d-L},0^{L})$ is added to the received hard-decision vector. Since the overall number of errors in this case satisfies $i+(d-L)-j \le \lfloor \frac{d-1}{2} \rfloor$, the algebraic decoder will generate the transmitted codeword in the decoding trial corresponding to the pattern $\boldsymbol{p}$. It is one of the decoding trials of the L-*order* CA3. □

Illustrating Example: Derivation of $\alpha(5)$ for Chase Algorithm 3

Clearly, $\alpha(5)=\Sigma_{i=1}^{4}\ a_{i,0}^{5}$. Hence, we compute the five terms $a_{i,0}^{5}$, where the initial condition in this case is $a_{1,2}^{1}=1$.

First, each term is tested for satisfying the conditions of Proposition 2. If $a_{i,j}^{L}$ does not satisfy any of these conditions (or the initial condition), it is computed recursively using Lemma 1. Thus from Proposition 2 iii) it follows that the first two terms of $\alpha(5)$ satisfy $a_{1,0}^{5}=a_{2,0}^{5}=0$. For each of the other two terms Lemma 1 is used recursively as follows:

$$\begin{array}{ll}
\boldsymbol{a}_{3,0}^{5}=\boldsymbol{a}_{3,0}^{3}+2\boldsymbol{a}_{2,1}^{3}+\boldsymbol{a}_{1,2}^{3} & \boldsymbol{a}_{4,0}^{5}=\boldsymbol{a}_{4,0}^{3}+2\boldsymbol{a}_{3,1}^{3}+\boldsymbol{a}_{2,2}^{3} \\
\boldsymbol{a}_{3,0}^{3}=\boldsymbol{a}_{3,0}^{1}+2\boldsymbol{a}_{2,1}^{1}+\boldsymbol{a}_{1,2}^{1}=1 & \boldsymbol{a}_{4,0}^{3}=0 \\
\boldsymbol{a}_{2,1}^{3}=\boldsymbol{a}_{2,1}^{1}+2\boldsymbol{a}_{1,2}^{1}+\boldsymbol{a}_{0,3}^{1}=2 & \boldsymbol{a}_{3,1}^{3}=\boldsymbol{a}_{3,1}^{1}+2\boldsymbol{a}_{2,2}^{1}+\boldsymbol{a}_{1,3}^{1}=0 \\
\boldsymbol{a}_{1,2}^{3}=0 & \boldsymbol{a}_{2,2}^{3}=0
\end{array}$$

where

$$\boldsymbol{a}_{4,0}^{3}=\boldsymbol{a}_{3,0}^{1}=\boldsymbol{a}_{2,1}^{1}=\boldsymbol{a}_{3,1}^{1}=\boldsymbol{a}_{2,2}^{1}=0$$

due to Proposition 2 i), and

$$\boldsymbol{a}_{1,2}^{3}=\boldsymbol{a}_{2,2}^{3}=\boldsymbol{a}_{1,3}^{1}=\boldsymbol{a}_{0,3}^{1}=0$$

due to Proposition 2 iv). Therefore,

$$\alpha(5)=\boldsymbol{a}_{3,0}^{5}=\boldsymbol{a}_{3,0}^{3}+2\boldsymbol{a}_{2,1}^{3}=5.$$

ACKNOWLEDGMENT

The authors thank M. P. C. Fossorier and S. Lin for preprint of their paper [9]. The authors also thank the referees for their constructive comments.

REFERENCES

[1] E. Agrell, "Voronoi regions for binary linear block codes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 310–316, 1996.
[2] O. Amrani, and Y. Be'ery, "Bounded-distance decoding algorithms: decision regions, and pseudo nearest neighbors," *IEEE Trans. Inform. Theory*, vol. 44, pp. 3072–3082, 1998.
[3] D. Chase, "A class of algorithms for decoding block codes with channel measurement information," *IEEE Trans. Inform. Theory*, vol. IT-18, pp. 170–182, 1972.
[4] J. H. Conway, and N. J. A. Sloane, *Sphere Packings, Lattices and Groups*. New York: Springer-Verlag, 1993.
[5] E. Fishler, "Nearest neighbors, decision regions and performance analysis of Chase and GMD decoding algorithms," M.Sc. thesis, Tel-Aviv Univ., Tel-Aviv, Israel, Sept. 1997.
[6] G. D. Forney, Jr. "Generalized minimum distance decoding," *IEEE Trans. Inform. Theory*, vol. IT-12, pp. 125–131, 1966.
[7] G. D. Forney, Jr. and A. Vardy, "Generalized minimum distance decoding of Euclidean-space codes and lattices," *IEEE Trans. Inform. Theory*, vol. 42, pp. 1992–2026, 1996.
[8] M. P. C. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," *IEEE Trans. Inform. Theory*, vol. 41, pp. 1379–1396, 1995.
[9] ———, "A unified method for evaluating the error-correcting radius of reliability-based soft decision algorithms for linear block codes," *IEEE Trans. Inform. Theory*, vol 44, pp. 691–700, 1998.
[10] T. Kaneko, T. Nishijima, H. Inazumi, and S. Hirasawa, "An efficient maximum-likelihood-decoding algorithm for linear block codes with algebraic decoder," *IEEE Trans. Inform. Theory*, vol. 40, pp. 320–327, 1994.
[11] H. Tanaka and K. Kakigahara, "Simplified correlation decoding by selecting possible codewords using erasure information," *IEEE Trans. Inform. Theory*, vol. IT-29, pp. 743–748, 1983.
[12] B.-Z. Shen, K. K. Tzeng, and C. Wang, "A bounded-distance decoding algorithm for binary linear block codes achieving the minimum effective error coefficient," *IEEE Trans. Inform. Theory*, vol. 42, pp. 1987–1991, 1996.