The Complexity of Information Set Decoding

JOHN T. COFFEY, MEMBER, IEEE AND RODNEY M. GOODMAN, MEMBER, IEEE

Abstract—**Information set decoding is an algorithm for decoding *any* linear code. Expressions for the complexity of the procedure that are logarithmically exact for virtually all codes are presented. The expressions cover the cases of complete minimum distance decoding, bounded hard-decision decoding, and the important case of bounded soft-decision decoding. It is demonstrated that these results are vastly better than for the trivial algorithms of searching through all codewords or through all syndromes and are significantly better than those for any other general algorithm currently known. For codes over large symbol fields the procedure tends towards a complexity that is subexponential in the symbol size.**

I. Introduction

THE COMPLETE DECODING PROBLEM for a linear code is the problem of determining, for any received vector, the nearest codeword of the code. It has been shown by Berlekamp *et al.* [1] that the complete decoding problem—or *minimum distance decoding* problem (MDD), as it is commonly known—is NP-complete. This strongly suggests, but does not rigorously imply, that no general polynomial time algorithm for the problem exists. Many researchers have interpreted this result as suggesting that no general algorithm that is significantly less complex than the exhaustive searches through all codewords or all syndromes can exist. Indeed, Berlekamp *et al.* state that "the discovery of an algorithm which runs significantly faster than this would be an important achievement." Bassalygo *et al.* [2] have suggested much the same idea. We show that a significant improvement is possible for virtually all linear codes over any symbol field. We give an expression for the *complexity coefficient* $F(q,R)$ obtained by using generalized information set decoding for complete decoding of a linear code of rate R over GF(q) that is *exact* for virtually all such codes. The complexity coefficient is defined by

$$F(q,R) = \lim_{n\to\infty} \frac{1}{n} \log_q E(q,R)$$

where $E(q,R)$ is the number of computations necessary (i.e., we have $E(q,R) = q^{nF(q,R)+o(n)}$). Similar exact solutions are given for information set decoding used to achieve bounded hard-decision decoding and for the important case of bounded soft-decision decoding. Furthermore, we show that the gain over exhaustive search procedures becomes more significant as the symbol size q rises; in fact, the complexity coefficient tends towards zero with increasing q. We demonstrate that these results are significantly better than those obtained for the recently proposed zero neighbors algorithm [3], [4].

Manuscript received August 8, 1988. Revised February 7, 1990.

J. T. Coffey was with the California Institute of Technology, Pasadena, CA. He is now with the Department of Electrical Engineering and Computer Science, University of Michigan, 1301 N. Beal Ave., Ann Arbor, MI 48109-2122.

R. M. Goodman is with the Department of Electrical Engineering, California Institute of Technology, Pasadena, CA 91125.

IEEE Log Number 9036390.

II. Information Set Decoding

Information set decoding was first suggested by Prange [5] for decoding cyclic codes and has been extensively examined and modified by many other researchers [6]–[22]. Information set decoding in all its forms works by exploiting the redundancy of the code. In an (n,k) code, if we have a set of k linearly independent bits from a received vector, we can construct the unique codeword of the code that agrees with the received vector in these bits. Thus these k bits determine the codeword uniquely, and hence the set is called an *information set*. If the received vector contains no errors in an information set, then we can recover the error pattern by the following procedure: reencode from the information set to find the unique codeword that agrees with the received vector in the information set bits, then subtract the codeword from the received vector. Because the information set now contains only zeros, the modified received vector now contains the error pattern added to the all-zero codeword, and so the errors have been *trapped*. Of course, *any* correctable error pattern lying outside the information set is corrected by this method, and this is where the procedure derives its efficiency.

Many embellishments of this basic idea exist. In permutation decoding [6] sets of k positions are obtained by using the automorphism group of the code. If the code is in systematic form, the first k bits form an information set, and so do all valid permutations thereof. This deals with the problem of how the information sets are to be generated, but an exact analysis of complexity, even for correction of a very small number of errors, is very cumbersome [7], [8]. Nevertheless, this method has been the focus of much attention [6]–[8], [16].

A modification of the basic idea is to drop the condition that the information set is error-free. We then search

0018-9448/90/0900-1031$01.00

systematically through precomputed patterns of information set errors. These patterns are called *covering polynomials* by Kasami [9], who applies the method to cyclic codes, using n information sets and concentrating the computational effort in the use of covering polynomials. The obvious idea of using general information sets with covering polynomials has been suggested many times, for example by Dmitriev [10] and Evseev [11].

Information sets have also been applied, in a different context, to the soft-decision decoding problem [20]–[22].

The approach we refer to as generalized information set decoding uses k-tuples that are not necessarily information sets. If the k-tuple has fewer than k independent bits, we augment the set by adding more bits till there are k independent bits in the set. All possible binary patterns in the augmented bits are then searched. This approach is equivalent to both *decoding with multipliers* [12] and combined information set and covering polynomial methods.

Despite the great amount of interest in algorithms based on the information set idea, no precise estimates of the decoding complexity have been produced. Clark and Cain [13] discuss some reasons why the problem is difficult. First, it is related to a long-standing unsolved problem in combinatorics, the (n,l,t) covering problem [14]. Given a set of n objects, we seek the minimum number of subsets of cardinality l, such that any subset of cardinality t is contained in at least one of the subsets of cardinality l. We refer to the minimum number of subsets required as the (n,l,t) covering coefficient, denoted by $b(n,l,t)$. In our problem, the subsets of cardinality t are the error patterns, and the subsets of cardinality l are the sets of parity positions, so that $l=n-k$. A t-tuple that is covered by an $(n-k)$-tuple is said to be *trapped* by the corresponding k-tuple. Our problem is thus to find an approximation for the $(n,n-k,t)$ covering coefficient. However, the problem is more difficult for two reasons: the k-tuples selected must represent an information set, and (for complete decoding) we must decode all patterns that are coset leaders, not just all patterns of a fixed weight or less. Despite these difficulties, we present a solution which is logarithmically accurate for virtually all linear codes.

First we derive a logarithmically accurate expression for $b(n,l,t)$, the (n,l,t) covering coefficient.

Theorem 1: Let R and ρ be constants such that $0<\rho<1-R<1$. Then

$$\lim_{n\to\infty}\frac{1}{n}\log b(n,\lfloor n(1-R)\rfloor,\lfloor n\rho\rfloor) = H_2(\rho)-(1-R)H_2(\rho/(1-R))$$

where

$$H_2(x) = -x\log_2 x-(1-x)\log_2(1-x)$$

is the binary entropy function.

Proof: A lower bound for $b(n,l,t)$ is easy to obtain. We must trap all t-tuples. Each selected k-tuple can trap $\binom{n-k}{t}$ t-tuples. Even in the most optimistic scenario where each t-tuple is trapped by exactly one k-tuple, we still need

$$\binom{n}{t}\Big/\binom{n-k}{t}$$

k-tuples. Using the relation [19]

$$2^{nH_2(\lambda)-o(n)}\le\binom{n}{\lambda n}\le 2^{nH_2(\lambda)}$$

for $0<\lambda<1$, where $o(n)$ denotes a function satisfying $\lim_{n\to\infty} o(n)/n=0$, we have

$$\binom{n}{\rho n}\ge 2^{nH_2(\rho)-o(n)} \quad\text{and}$$

$$\binom{n(1-R)}{n\rho}\le 2^{n(1-R)H_2(\rho/(1-R))}$$

and thus

$$b(n,\lfloor n(1-R)\rfloor,\lfloor n\rho\rfloor)\ge\binom{n}{n\rho}\Big/\binom{n(1-R)}{n\rho} \ge 2^{n[H_2(1-R)-(1-R)H_2(\rho/(1-R))]+o(n)}.$$

For the upper bound, we adopt the following argument. We select a large number $f(n,k,t)$ of k-tuples independently and at random. The probability that a given t-tuple is *not* trapped is

$$\left[1-\binom{n-t}{k}\Big/\binom{n}{k}\right]^{f(n,k,t)}$$

because for each choice of t-tuple, there are $\binom{n-t}{k}$ "good" k-tuples, out of a total of $\binom{n}{k}$ k-tuples. The expected number of t-tuples not trapped is

$$\binom{n}{t}\left[1-\binom{n-t}{k}\Big/\binom{n}{k}\right]^{f(n,k,t)}.$$

Now let

$$f(n,k,t)=\frac{\binom{n}{k}}{\binom{n-t}{k}}g(n,k,t)$$

for some function $g(n,k,t)$. Then the expected number of t-tuples not trapped is

$$\binom{n}{t}\left[1-\binom{n-t}{k}\Big/\binom{n}{k}\right]^{\binom{n}{k}/\binom{n-t}{k}g(n,k,t)}.$$

Using the relation

$$\lim_{x\to\infty}(1-1/x)^x=e^{-1}$$

we see that this expression tends towards

$$\binom{n}{t}e^{-g(n,k,t)}=2^{nH_2(t/n)+o(n)-g(n,k,t)\log e}.$$

Setting

$$g(n,k,t)>\frac{1}{\log e}[nH_2(t/n)+o(n)]$$

now gives an expected number of t-tuples not trapped less than one. This is possible only if there is at least one set of $f(n,k,t)$ k-tuples which traps all t-tuples. Thus we need

$$f(n,k,t) = \frac{\binom{n}{k}}{\binom{n-t}{k}} \cdot cn$$

for some constant c. Using the identity

$$\binom{n}{t}\binom{n-t}{k} = \binom{n}{k}\binom{n-k}{t},$$

we have

$$\binom{n}{k}\Big/\binom{n-t}{k} = \cdot\binom{n}{t}\Big/\binom{n-k}{t}.$$

Thus the upper bound is such that

$$b(n,n-k,t) \le f(n,k,t) = cn\binom{n}{t}\Big/\binom{n-k}{t}$$
$$= 2^{n[H_2(\rho)-(1-R)H_2(\rho/(1-R))]+o(n)}$$

which has the same form as the lower bound. □

To analyze the complete decoding problem for linear codes, it is necessary to have some knowledge of the covering radius of these codes. The covering radius r is the weight of the highest weight coset leader of the code. The Goblick bound [24] states that

$$r \ge nH_q^{-1}(1-R) + o(n)$$

for all codes, but no general upper bound has been known until recently, when Blinovskii [25] and Levitin [26] showed independently that the Goblick bound is tight for virtually all linear codes, i.e., that

$$r = nH_q^{-1}(1-R) + o(n)$$

for all but a fraction of codes that tends to zero as $n \to \infty$. Thus the coset leaders form an almost perfect sphere around the zero codeword with very sharp *spikes* extending inwards to half the Goblick bound—an "inverted hedgehog," to use Levitin's term.

We seek the number of k-tuples to be selected such that any *coset leader* of the code is disjoint from some k-tuple. This number is given by the following theorem.

Theorem 2: For virtually all linear (n,k) codes over GF(q), the minimum number $M(\mathscr{C})$ of k-tuples required to ensure that each coset leader is disjoint from at least one k-tuple satisfies

$$\frac{1}{n}\log_2 M(\mathscr{C}) = H_2\left(H_q^{-1}(1-R)\right) - (1-R)H_2\left(\frac{H_q^{-1}(1-R)}{1-R}\right) + o(1).$$

Proof: An upper bound is obtained by considering the number of k-tuples necessary to trap *all* patterns of up to r errors, whether the patterns are coset leaders or not. By the definition of covering radius, this set includes all coset leaders. From Theorem 1, we have the upper bound

$$2^{n[H_2(r/n)-(1-R)H_2(r/n(1-R))]+o(n)}$$
$$= 2^{n[H_2(H_q^{-1}(1-R))-(1-R)H_2(H_q^{-1}(1-R)/(1-R))]+o(n)}.$$

For the lower bound, we note that we must trap q^{n-k} coset leaders, and that virtually all coset leaders have weight greater than $nH_q^{-1}(1-R) - o(n)$. Each k-tuple can trap no more than

$$\sum_{n\rho - o(n) \le i \le n\rho + o(n)} \binom{n-k}{i}(q-1)^i$$

such coset leaders. Now

$$q^{n-k} = \binom{n}{n\rho}(q-1)^{n\rho}\cdot q^{o(n)}$$

where

$$\rho = H_q^{-1}(1-R)$$

so the lower bound has the form

$$\binom{n}{n\rho}\Big/\binom{n(1-R)}{n\rho}$$
$$= 2^{n[H_2(H_q^{-1}(1-R))-(1-R)H_2(H_q^{-1}(1-R)/(1-R))]+o(n)}$$

and the theorem follows. □

The only remaining problem is how we deal with the case when a selected k-tuple does not contain k linearly independent bits. If a k-tuple has only $k-l$ independent bits, we say that the k-tuple is *l-defective*. We can remedy this condition by finding l bits from the remaining $n-k$ in such a way that the $(k+l)$-tuple has k linearly independent bits, and then exhaustively searching through all possible error patterns in that l-tuple—these are just the covering polynomials mentioned earlier. This will cause an increase in complexity of 2^l for that k-tuple. We need to show that this increase in complexity is subexponential. Given fixed R and α, with $0 < R$, $\alpha < 1$, we say that an $\lfloor nR \rfloor$-tuple is *seriously α-defective* if the $\lfloor nR \rfloor$-tuple contains less than $\lfloor nR(1-\alpha) \rfloor$ independent bits. We will show that for any fixed $\alpha > 0$ and sufficiently large n, there are virtually no linear (n,k) codes that contain *any* k-tuple that is seriously α-defective. To do this, we employ the useful analytical tool of Kolmogorov complexity as applied to codes; a detailed discussion of this approach can be found in [27]. For our present purposes, a brief synopsis of the main points will suffice. We use a universal model of computation, the universal Turing machine (UTM) [28]. Informally, this consists of a finite state machine, a read-write head, and a two-way infinite tape ruled into cells. At each time instant, the UTM can change state and/or simultaneously perform one of the following actions: overprint the current symbol on the tape with a new one or a blank, move one cell right, or move one cell left. Church's Thesis [28] states that any function which is computable by a mechanical procedure can be computed by a UTM. The Kolmogorov complexity of a string s is the length of the shortest string ("program") which, if placed on the tape to the immedi-

ate right of the read-write head, would cause the UTM to run and halt eventually leaving s to the right of the read-write head. A string is said to be *random* if its Kolmogorov complexity is at least equal to its length. We need the following key result [28].

- The fraction of all q-ary strings of length n with Kolmogorov complexity less than $n-c$ is less than q^{-c} for any c.

This follows because each such "low complexity" string requires a unique program of length less than $n-c$ to represent it. The number of such q-ary strings is $\sum_{i=0}^{n-c-1} q^i = (q^{n-c}-1)/(q-1)$, and this is a fraction of less than q^{-c} of all strings of length n. Thus almost all strings of length n are almost totally random.

We are now ready for the following result.

Theorem 3: For any fixed R and α satisfying $0 < \alpha$, $R < 1$, and for all sufficiently large values of n, virtually all linear $(n, \lfloor nR \rfloor)$ codes over any symbol field contain no $\lfloor nR \rfloor$-tuple with fewer than $\lfloor nR(1-\alpha) \rfloor$ independent bits.

Proof: Let G be the $k \times n$ generator matrix of a linear code $\mathscr{C}$. With this generator matrix we associate the string $s(G)$ of length nk obtained by writing G out row by row. Each generator matrix corresponds to exactly one such string and vice versa (note that we do not insist that each code be represented by exactly one string; nor do we insist that $\dim \mathscr{C} = k$). From the key lemma on Kolmogorov complexity, the fraction of these strings with Kolmogorov complexity less than $nk - c$ is less than q^{-c}. Suppose a code contains a k-tuple that is seriously α-deficient. We can specify the generator matrix (and hence the full code string) as follows:

- specify the deficient k-tuple (taking $\log_q \binom{n}{k}$ symbols);
- write out the other $n-k$ columns in full (taking $k(n-k)$ symbols);
- write out the $k(1-\alpha)$ independent columns in the defective k-tuple (taking $k^2(1-\alpha)$ symbols);
- specify each of the remaining columns in the defective k-tuple by specifying the linear combination of the independent columns that yields it (taking $k\alpha k(1-\alpha)$ symbols).

The total length of this program is

$$C + \log_q \binom{n}{k} + k(n-k) + k^2(1-\alpha) + k\alpha k(1-\alpha)$$
$$= n^2R - n^2R^2\alpha^2 + o(n^2).$$

The fraction of such strings is thus less than $q^{-n^2R^2\alpha^2 + o(n^2)}$, as required. □

Putting together the results of Theorems 1, 2, and 3, we have the following result.

Theorem 4: For virtually all linear (n,k) codes $\mathscr{C}$ over GF(q), the complexity $M(\mathscr{C})$ of complete minimum distance decoding using the generalized information set decoding algorithm satisfies

$$\frac{1}{n}\log_2 M(\mathscr{C}) = H_2\left(H_q^{-1}(1-R)\right) - (1-R)H_2\left(\frac{H_q^{-1}(1-R)}{1-R}\right) + o(1).$$

By the convexity of the entropy function, we have $H_2(xy) > xH_2(y)$ for $0 < x, y < 1$, and so the function is always greater than zero, as we would expect.

The behavior of this function versus R for the case $q = 2$ can be seen in Fig. 1. Clearly, it represents a huge improvement over exhaustive search procedures for any fixed rate. For $R = 1/2$, generalized information set decoding requires less than the fourth root of the number of computations required by a search through all codewords.

For bounded distance hard-decision decoding, we need to decode all error patterns of weight up to t, where $t = \lfloor (d-1)/2 \rfloor$. It is well known that virtually all linear codes satisfy the Gilbert–Varshamov bound, $d \geq nH_q^{-1}(1-R) + o(n)$. In the binary case ($q = 2$), it is also known [29] that virtually no codes exceed the bound by any significant amount. We assume that for any q, the bounded distance decoding problem involves only decoding all error patterns up to half the Gilbert–Varshamov bound, i.e., up to $nH_q^{-1}(1-R)/2 + o(n)$ errors.

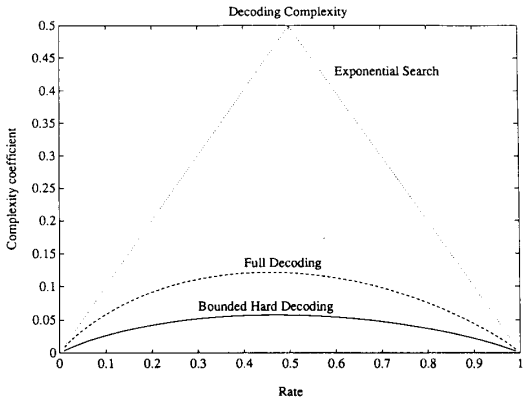

Fig. 1. Complexity of various decoding rules for binary codes.

Theorem 5: Bounded distance decoding using generalized information set decoding has, for virtually all linear codes, a complexity $M(\mathscr{C})$ satisfying

$$\frac{1}{n}\log_2 M(\mathscr{C}) = H_2\left(H_q^{-1}(1-R)/2\right) - (1-R)H_2\left(\frac{H_q^{-1}(1-R)}{2(1-R)}\right) + o(1).$$

Proof: The proof follows from Theorems 1–3 with $\rho = H_q^{-1}(1-R)/2 + o(1)$. □

This function is plotted versus rate for the binary case ($q = 2$) in Fig. 1. The number of computations is far less than for exhaustive search, and is also much less than for complete minimum distance decoding, requiring slightly less than the square root of the number of computations

required for complete decoding at rate one-half. This represents a complexity of slightly more than the ninth root of the number of codewords.

The problem of soft decision decoding for binary linear codes has been extensively examined. It is well known [30] that an error pattern can have at most $d-1$ hard errors if it is to be within the guaranteed soft error correcting power of the code. Conversely, given any set of $d-1$ locations, we can construct an error pattern within the guaranteed soft-error-correcting power of the code with hard errors in all those $d-1$ positions. Thus to derive *a priori* an algorithm that achieves soft-decision decoding up to the guaranteed soft-error-correcting power of the code, it is necessary and sufficient that we should be able to correct all patterns of up to $d-1$ hard errors. For most codes, we have

$$d-1 = nH_q^{-1}(1-R) + o(n) \approx \rho$$

so the computational requirement is as given in Theorem 4.

Theorem 6: Bounded soft-decision decoding using generalized information set decoding has, for virtually all binary linear codes, a complexity $M(\mathscr{C})$ satisfying

$$\frac{1}{n}\log_2 M(\mathscr{C}) = H_2\left(H_2^{-1}(1-R)\right) - (1-R)H_2\left(\frac{H_2^{-1}(1-R)}{1-R}\right) + o(1)$$

$$= (1-R)\left[1 - H_2\left(\frac{H_2^{-1}(1-R)}{1-R}\right)\right] + o(1).$$

Again, the complexity is plotted as a function of R in Fig. 1, where the exhaustive search procedures involve searching through all codewords (for $R \le 1/2$) or decoding with a trellis [31] (for $R > 1/2$). In practical applications, bounded soft-decision decoding asymptotically (in SNR) doubles the error-correcting power. Fig. 1 shows that for virtually all codes, it also doubles the exponent in the number of computations, assuming generalized information set decoding is used. In some applications, we may not wish to decode out to double the guaranteed hard distance, but rather to three halves the hard distance, or some other multiple η. In general, this requires a complexity coefficient of

$$H_2\left(\frac{\eta}{2}H_2^{-1}(1-R)\right) - (1-R)H_2\left(\frac{\eta H_2^{-1}(1-R)}{2(1-R)}\right) + o(1).$$

Another parameter of interest is the behavior of the algorithm when q becomes very large. We have the following result.

Theorem 7: For large q, we have $F(q,R) \to H_2(1-R)/\log_2 q$. Thus $\lim_{q\to\infty} F(q,R) = 0$.

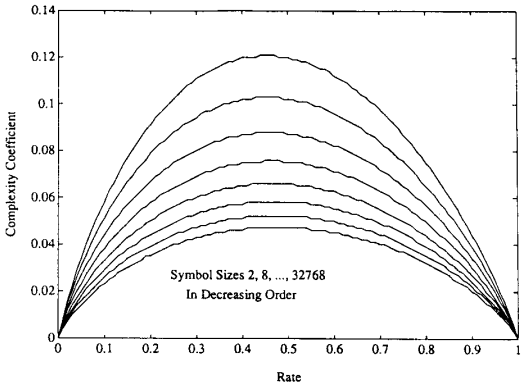

Fig. 2. Complexity for various symbol fields.

Proof: This follows from the behavior of the function $H_q^{-1}(x)$ as q becomes large. We have

$$H_q(x) = x\log_q(q-1) + x\log_q x + (1-x)\log_q(1-x)$$
$$= x\left(1 + \log_q\left(1-\frac{1}{q}\right)\right) + H_2(x)/\log_2 q$$
$$= x\left(1 - \left(\frac{1}{q} - \frac{1}{2q^2} + \cdots\right)\Big/\ln q\right) + (H_2(x))/\log_2 q$$
$$\to x \quad \text{as} \quad q \to \infty.$$

Then by continuity of $H_q(x)$, we must have

$$\lim_{q\to\infty} H_q^{-1}(x) = x.$$

From Theorem 4, we have

$$\lim_{n\to\infty}\frac{1}{n}\log_2 M(\mathscr{C}) = H_2\left(H_q^{-1}(1-R)\right) - (1-R)H_2\left(\frac{H_q^{-1}(1-R)}{1-R}\right)$$
$$= H_2(1-R-\epsilon) - (1-R)H_2\left(\frac{1-R-\epsilon}{1-R}\right)$$
$$= H_2(1-R) + \epsilon' - (1-R)\epsilon''$$
$$\to H_2(1-R).$$

Thus the computational effort $M(\mathscr{C})$ has the form $2^{nH_2(1-R)+o(n)}$ for large n, independent of q. This is equivalent to $q^{(nH_2(1-R)+o(n))/\log_2 q}$, so the complexity coefficient is $H_2(1-R)/\log_2 q$ as claimed. □

Fig. 2 shows the complexity coefficient for complete minimum distance decoding for many values of q. The fact that the complexity coefficient tends to zero with increasing q is intuitively quite surprising and, we feel, quite significant.

III. Comparisons

The question of the complexity of general decoding algorithms has attracted a great deal of attention, and many algorithms have been proposed. Recently, Levitin

and Hartmann have proposed new algorithms based on the novel and intuitively appealing concept of "zero neighbors" [3] and the related concept of "zero guards" [4] and have given upper bounds on the complexity of these algorithms that are valid for virtually all linear codes, and which are much lower than the $2^{n[\min(R,(1-R))]}$ obtained from the exhaustive search algorithms. Because of this, the algorithms have attracted much attention [15], [33]. However, Fig. 3 shows that the complexity of the zero neighbors algorithm (ZNA),

$$\left(F_{\mathrm{ZNA}}(2,R) \approx H_2\left(2H_2^{-1}(1-R)\right)-(1-R)\right.$$
$$\left.\text{for } R > 0.1887 \text{ and for } q=2\right),$$

is much higher than that for generalized information set decoding—for example, in the case $R = 1/2$, generalized information set decoding requires less than the *square root* of the complexity required by the zero neighbors algorithm. This is even greater than the gain made by the ZNA over exhaustive search (the ZNA requires marginally more than the square root of the number of codewords at $R = 1/2$). In addition to this favorable comparison, generalized information set decoding has two further major advantages. First, as we have seen, it can be modified easily to perform bounded soft-decision decoding, without any significant rise in complexity. For the ZNA, on the other hand, soft-decision decoding requires a prohibitive rise in complexity. Second, the complexity characteristic for large q is much less favorable for the ZNA, approaching the complexity required by exhaustive search, rather than zero. We have $F_{\mathrm{ZNA}}(q,R) \approx H_q(2H_q^{-1}(1-R))-(1-R)$ for R such that $H_q^{-1}(1-R) \le 1/2$, and $F_{\mathrm{ZNA}}(q,R) = R$ otherwise. It can easily be demonstrated that this yields $F_{\mathrm{ZNA}}(q,R) \to \min(R,(1-R))$ as $q \to \infty$.

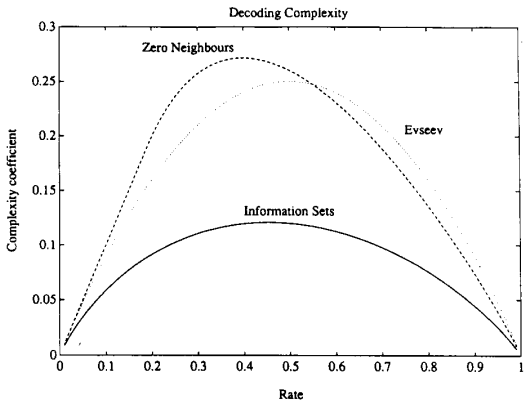

Fig. 3. Comparison of complexity for complete minimum distance decoding—various algorithms.

Previous analysis of algorithms based on the error trapping idea have usually given the lower bound for bounded distance hard-decision decoding [9], [13]. Evseev [11] discusses an algorithm—Q-decoding—which is basically the same as information set decoding. He shows that with soft-decision decoding, the probability of error for the algorithm is no more than double that for maximum likelihood decoding, with complexity coefficient $F(R) \le R(1-R)$. We plot this bound in Fig. 3. Clearly, it lies far above our (exact) solution.

IV. Conclusion

Exact solutions for the complexity coefficient for generalized information set decoding for the cases of complete minimum distance decoding, bounded hard-decision decoding, and bounded soft-decision decoding have been obtained. These are significantly less than the best available bounds from other algorithms, and vastly less than the requirements from the trivial exhaustive search algorithms. Indeed, for large symbol fields, the gain over the full search algorithms is essentially unlimited.

Comparison of the complexity requirements for the various decoding strategies yields an insight into the tradeoffs of performance versus complexity that are available. In particular, bounded soft-decision decoding gives a performance asymptotically twice as good as that for bounded hard-decision decoding for the AWGN channel. Using generalized information set decoding, it requires a complexity coefficient twice as high.

Although we do not claim that the generalized information set algorithm is the best general decoding scheme for linear codes, it seems clear that an important principle is being exploited in the algorithm, and that any optimal decoding scheme should possess the same advantages. Elsewhere, we have proposed a unified structure [34] for decoding algorithms that incorporates both information set decoding and the zero neighbors algorithm, among others, which we believe may yield general algorithms of even lower complexity. We intend to report on this work at a later date.

References

[1] 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. IT-24, pp. 384–386, 1978.

[2] L. A. Bassalygo, V. V. Zyablov, and M. S. Pinsker, "Problems of complexity in the theory of correcting codes," *Probl. Peredach. Inform.*, vol. 13, pp. 5–17, 1977.

[3] L. B. Levitin and C. R. P. Hartmann, "A new approach to the general minimum distance decoding problem—The zero neighbors algorithm," *IEEE Trans. Inform. Theory*, vol. IT-31, pp. 378–384, 1985.

[4] C. R. P. Hartmann and L. B. Levitin, "An improvement of the zero-neighbors minimum distance decoding algorithm: The zero guards algorithm," *IEEE Int. Symp. Inform. Theory*, Kobe, Japan, 1988.

[5] E. Prange, "The use of information sets in decoding cyclic codes," *IRE Trans.*, vol. IT-8, pp. S5–S9, 1962.

[6] F. J. MacWilliams, "Permutation decoding of systematic codes," *Bell Syst. Tech. J.*, vol. 43, pp. 485–505, 1964.

[7] S. G. S. Shiva and K. C. Fung, "Permutation decoding of certain triple-error-correcting binary codes," *IEEE Trans. Inform. Theory*, vol. IT-18, pp. 444–446, 1972.

[8] A. Benyamin-Seeyar, S. G. S. Shiva, and V. K. Bhargava, "Capability of error-trapping technique in decoding cyclic codes," *IEEE Trans. Inform. Theory*, vol. IT-32, pp. 166–180, 1986.

[9] T. Kasami, "A decoding procedure for multiple-error-correcting cyclic codes," *IEEE Trans. Inform. Theory*, vol. IT-10, pp. 134–138, 1964.

[10] O. F. Dmitriev, "An algorithm for the correction of independent errors by cyclic codes," *Probl. Peredach. Inform.*, vol. 3, pp. 102–104, 1967.
[11] G. S. Evseev, "Complexity of decoding for linear codes," *Probl. Peredach. Inform.*, vol. 19, pp. 3–8, 1983.
[12] L. D. Baumert, R. J. McEliece, and G. Solomon, "Decoding with multipliers," Jet Propulsion Laboratory Deep Space Network Progress Report, 42-34, pp. 42–46.
[13] G. C. Clark and J. B. Cain, *Error-Correcting Coding for Digital Communications*. New York: Plenum Press, 1981.
[14] A. H. Chan and R. A. Games, "(n,k,t)-covering systems and error-trapping decoding," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 643–646, 1981.
[15] P. G. Farrell, M. Rice, and F. Taleb, "Minimum weight decoding for cyclic codes," in *Proc. IMA Conf. on Cryptography and Coding*, Cirencester, 1986.
[16] R. M. F. Goodman and A. D. Green, "Microprocessor-controlled permutation decoding of error-correcting codes," in *Proc. IERE Conf. on Microprocessors in Automation and Commun.*, Kent, no. 41, 1978, pp. 365–376.
[17] D. M. Mandelbaum, "On vote-taking and complete decoding of certain error-correcting codes," *Inform. Contr.*, vol. 43, pp. 195–197, 1979.
[18] ____, "On complete decoding of linear error-correcting codes," *Inform. Contr.*, vol. 47, pp. 195–200, 1980.
[19] V. K. Wei, "An error-trapping decoder for nonbinary cyclic codes," *IEEE Trans. Inform. Theory*, vol. IT-30, pp. 538–541, 1984.
[20] J. P. Gao, L. D. Rudolph, and C. R. P. Hartmann, "Soft-decision decoding using a modified gradient method and multiple information sets," *IEEE Int. Symp. Inform. Theory*, St. Jovite, Canada, 1983.
[21] K. H. Farrell, L. D. Rudolph, C. R. P. Hartmann, and L. D. Nielsen, "Decoding by local optimization," *IEEE Trans. Inform. Theory*, vol. IT-29, pp. 740–743, 1983.
[22] L. D. Rudolph, C. R. P. Hartmann, K. H. Farrell, J. P. Gao, and N. Tendolkar, "Further results on decoding by local optimization," *IEEE Int. Symp. Inform. Theory*, Les Arcs, France, 1982.
[23] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. Amsterdam: North-Holland, 1977.
[24] T. J. Goblick, Jr., "Coding for a discrete information source with a distortion measure," Ph.D. dissertation, Dept. of Elect. Eng., M.I.T., Cambridge, MA, 1962.
[25] V. M. Blinovskii, "Lower asymptotic bound on the number of linear code words in a sphere of given radius in F_q^n," *Probl. Peredach. Inform.*, vol. 23, pp. 50–53, 1987.
[26] L. B. Levitin, "Covering radius of almost all linear codes satisfies the Goblick bound," recent results session, *IEEE Int. Symp. Inform. Theory*, Kobe, Japan, 1988.
[27] J. T. Coffey and R. M. F. Goodman, "Any code of which we cannot think is good," submitted to *IEEE Trans. Inform. Theory*.
[28] G. J. Chaitin, "Information-theoretic computational complexity," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 10–15, 1974.
[29] J. N. Pierce, "Limit distribution of the minimum distance of random linear codes," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 595–599, 1967.
[30] 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.
[31] J. K. Wolf, "Efficient maximum likelihood decoding of linear block codes using a trellis," *IEEE Trans. Inform. Theory*, vol. IT-24, pp. 76–80, 1978.
[32] P. G. Farrell, M. Rice, and F. Taleb, "Division algorithms for hard- and soft-decision decoders," in *Proc. Int. Conf. Digital Signal Processing*, Florence, Italy, 1987.
[33] W. Godoy, Jr. and D. S. Arantes, "Suboptimum soft-decision decoding of block codes using the zero-neighbors algorithm," *IEEE Int. Symp. Inform. Theory*, Kobe, Japan, 1988.
[34] J. T. Coffey, R. M. F. Goodman, and P. G. Farrell, "Mapping vector decoding," in *Proc. Second Conf. Error-Correcting Codes*, IBM Research Center, Almaden, CA, 1987.