Note: It is easy to extend Proposition 4 to the repetition and Golay codes and thereby to all binary perfect codes.

Proposition 5: Shortened Hamming codes do not necessarily obey the 2^{-p} bound.

Proof: Consider a Hamming (1023,1013) code which has its parity-check bits at positions 2^i, $i = 0, 1, \cdots, 9$ [4]. This implementation is useful for error correction because the syndrome gives the position of the error. If we shorten this code to a (12,2) code by eliminating the leading 1011 information bits, it is easily seen that the codewords for this shortened code are

$$\begin{array}{cccccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0. \end{array}$$

So in this case $P(e) = \varepsilon^3(1-\varepsilon)^9 + \varepsilon^{10}(1-\varepsilon)^2 + \varepsilon^{11}(1-\varepsilon)$. For example, for $\varepsilon = 0.25$, $P(e) = 1.174 \times 10^{-3} > 2^{-10} = 9.766 \times 10^{-4}$.

We might also consider shortening a Hamming code in cyclic form. Recall that an (n, k) Hamming code is a cyclic code whose generator polynomial $g(x)$ is a primitive polynomial of degree m, where $n = 2^m - 1$ and $k = 2^m - m - 1$. From [3] we find that $x^{10} + x^3 + 1$ is a primitive polynomial so that the generator matrix for a cyclic Hamming (1023,1013) code might be

$$\begin{array}{c} \\ 1013 \end{array} \overset{\longleftarrow 1012 \longrightarrow}{\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & \cdots & 0 & 0 \\ & \cdot & & & & \cdot & & & & & & & & & & \\ & & \cdot & & & & \cdot & & & & & & & & & \\ & & & \cdot & & & & \cdot & & & & & & & & \end{bmatrix}}.$$

If we shorten the code [3, p. 241] to an (11,1) code, the generator matrix will be just [10010000001]. This gives $P(e) = \varepsilon^3(1-\varepsilon)^8$ which has its maximum at $\varepsilon = 3/11$. At $\varepsilon = 3/11$, $P(e) = 1.59 \times 10^{-3} > 2^{-10} = 9.77 \times 10^{-4}$.

It should be noted that we chose a primitive polynomial which had the minimum number of nonzero coefficients. This reduces the cost for hardware implementation.

III. Conclusion

It has been demonstrated that Hamming codes always obey the 2^{-p} bound on $P(e)$ for $\varepsilon \leq \frac{1}{2}$. However, this is not necessarily true for more general cyclic codes. Therefore, caution is needed when using the bound, especially in the design of systems in which the probability of undetected error is critical. Fortunately, the bound is not badly violated by the cyclic codes we have used as counterexamples, providing hope that a slight overdesign should suffice. More careful analysis or simulation should be used to check out such designs.

Even for Hamming codes, $P(e)$ can be larger than 2^{-p} for $\varepsilon > \frac{1}{2}$. This leads one to wonder if there are any codes which are uniformly good for all $0 \leq \varepsilon \leq 1$. We have been able to show that an (n,k) Hamming code altered to an $(n,k-1) \equiv (n',k')$ code by deleting all codewords of weight greater than $n/2$ has $P(e) < 2^{-(n-k)} = 2^{-(n'-k')+1}$, for all $0 \leq \varepsilon \leq 1$. The proof is straightforward and relies on evaluation of $P(e)$ through use of the weight enumerator polynomial. We have further found, through numerical evaluation, that, when the (7,4) code is thus shortened to a (7,3) code, $P(e) < 2^{-4}$ as opposed to the 2^{-3} value predicted by the above argument, for all $0 \leq \varepsilon \leq 1$. And it is easily shown that the code obtained by affixing an overall parity-check bit to a maximal length shift register code has its maximum undetected error rate at $\varepsilon = \frac{1}{2}$, and thus has $P(e) \leq (2^k - 1)/2^n$, for all $0 \leq \varepsilon \leq 1$. Thus, for these codes, $\varepsilon = \frac{1}{2}$ is the "worst of all possible channels" for error detection. However, we doubt that this is true for codes with more reasonable rates.

Acknowledgment

The authors would like to thank Prof. John T. Gill for his help in computing the weight distribution of some BCH codes.

References

[1] M. E. Hellman, "Error detection made simple," in *Conference Record, Int. Conf. on Communications*, Minneapolis, Minn., June 17–19, 1974, pp. 9A-4.
[2] S. Lin, *An Introduction to Error-Correcting Codes*. Englewood Cliffs, N.J.: Prentice-Hall, 1970.
[3] W. W. Peterson and E. J. Weldon, *Error Correcting Codes*, 2nd ed. Cambridge, Mass.: M.I.T. Press, 1972.
[4] W. Bennett and J. Davey, *Data Transmission*. New York: McGraw-Hill, 1965.

On the Complexity of Decoding Reed–Solomon Codes

JØRN JUSTESEN

***Abstract*—Certain q-ary Reed–Solomon codes can be decoded by an algorithm requiring only $0(q \log^2 q)$ additions and multiplications in $GF(q)$.**

The fast Fourier transform has been used to obtain algorithms of low complexity for multiplying, dividing, and evaluating polynomials and for computing greatest common divisors [1]. When $q - 1$ has many small factors, a fast transform in $GF(q)$ has been suggested for evaluating syndromes of Reed–Solomon codes [2]. The computations are particularly convenient when q is a Fermat prime [3], and we shall simplify the presentation by referring to this particular case.

The fast Fourier transform is based on a factorization of binomials (or other polynomials of low weight) of the type

$$\begin{aligned} x^n - 1 &= (x^{n/2} - 1)(x^{n/2} + 1) \\ &= (x^{n/4} - 1)(x^{n/4} + 1)(x^{n/4} - \alpha^{n/4})(x^{n/4} + \alpha^{n/4}) \\ &= \cdots = \prod_{j=0}^{n-1} (x - \alpha^j) \end{aligned}$$

where α is a primitive nth root of unity. When $q = 2^{2^r} + 1$ is a prime, such a factorization is possible in $GF(q)$ for all n that are powers of 2. A polynomial $a(x)$ is transformed into the values $a(\alpha^j)$. Thus a polynomial can be evaluated for all nonzero values of its argument in $0(n \log n)$ operations. The inverse transform of the product $a(\alpha^j)b(\alpha^j)$ yields the product $a(x)b(x)$ modulo $(x^n - 1)$ in $0(n \log n)$ operations. When this algorithm is applied to Reed–Solomon codes of length n, it is an efficient method for encoding, calculating the syndromes, and computing the error locations and error values. When the error locator $\sigma(x)$ and error evaluator polynomial $\omega(x)$ [4] are known, the values of $\sigma(x)$ and $\omega(x)/\sigma'(x)$ are obtained through the transform.

An algorithm for solving the key equation based on Euclid's algorithm was suggested by Sugiyama *et al.* [5]. Starting from the syndrome polynomial $S(x)$ and x^{2t}, the remainder sequence r_j is computed until $\deg r_{k-1} \geq t$ and $\deg r_k \leq t - 1$. We suggest a modification of this approach based on an algorithm that computes the greatest common divisor in $0(n \log^2 n)$

Manuscript received May 30, 1975.
The author is with the Aalborg Universitets Center, Aalborg, Denmark.

operations. Procedure 8.7 in [1] recursively computes the last remainder whose degree exceeds $\frac{1}{2}$ deg r_{-1}. In the notation of [5], the output is r_{i-1}, r_i, V_{i-1}, U_{i-1}, and U_i, where r_{i-1} is the last remainder of degree greater than t. Thus deg $r_i \leq t$, and an additional iteration of Euclid's algorithm may be needed to make deg $r_k \leq t - 1$. If $2t$ is a power of 4, then the algorithm reduces the calculation of $\gcd(p_1,p_2)$ to two new greatest common divisor calculations with polynomials of degree at most $\frac{1}{2}$ deg p_1. The steps of this algorithm, as well as the additional iteration, require that a q_i and an r_i satisfying

$$r_{i-2} = q_i r_{i-1} + r_i$$

can be computed efficiently. This calculation has complexity $0(\deg r_{i-2})$ if deg q_i is upperbounded by a constant. However, if we need to divide polynomials of very different degrees, we first compute $r_{i-1}^{-1} = \lfloor x^{\deg r_{i-2}}/r_{i-1}(x) \rfloor$ by procedure 8.2 of [1], and then determine q_i as the leading coefficient of $r_{i-2}r_{i-1}$. The procedure for computing the approximate reciprocal of a polynomial of degree $k - 1$ involves the computation of a polynomial of degree $(k/2) - 1$ and multiplication of polynomials of degree k. Thus, when fast multiplications of the appropriate lengths are used, the procedure will recursively determine the reciprocal in $0(k \log k)$ operations. We conclude that the complexity of the decoding algorithm is $0(q \log^2 q)$ in terms of $GF(q)$ multiplications and additions.

In this correspondence we have indicated only the order of magnitude of the complexities, but inspection of the algorithms in [1] shows that the constant factors are small. Thus the use of the fast algorithm for evaluating polynomials is advantageous for codes of medium rate and length $2^8 + 1 = 257$. The modified algorithm for solving the key equation should be used when t^2 is large compared to $n \log^2 n$. For medium rate codes, this is the case for $n = 2^{16} + 1 = 65537$. The algorithm actually applies also to Goppa (and thus BCH) codes. However, the asymptotic results are most interesting for codes that remain good for large n.

References

[1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*. Reading, Mass.: Addison-Wesley, 1974.

[2] V. B. Afanasyev, "Time saving Reed–Solomon coding and error detection," in *Third Int. Symp. on Information Theory*, Tallin, 1973.

[3] R. C. Agarwal and C. S. Burrus, "Fast convolution using Fermat number transforms with applications to digital filtering," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-22, pp. 87–97, Apr. 1974.

[4] E. R. Berlekamp, *Algebraic Coding Theory*. New York: McGraw-Hill, 1968.

[5] Y. Sugiyama, M. Kasahara, S. Hirasawa, and T. Namekawa, "A method for solving key equations for decoding Goppa codes," *Inform. Contr.*, vol. 27, pp. 87–99, Jan. 1975.

An Erasures-and-Errors Decoding Algorithm for Goppa Codes

YASUO SUGIYAMA, MASAO KASAHARA, MEMBER, IEEE, SHIGEICHI HIRASAWA, AND TOSHIHIKO NAMEKAWA, MEMBER, IEEE

***Abstract*—An erasures-and-errors decoding algorithm for Goppa codes is presented. Given the Goppa polynomial and the modified syndrome polynomial, a modified key equation is solved using Euclid's algorithm to determine the error locator polynomial and the errata evaluator polynomial.**

Manuscript received August 20, 1974; revised July 12, 1975.

Y. Sugiyama and S. Hirasawa are with the Communication Equipment Works, Mitsubishi Electric Corporation, Amagasaki, Japan.

M. Kasahara and T. Namekawa are with the Faculty of Engineering, Osaka University, Suita, Japan.

I. Introduction

It is well known that the receiver with an erasure option can improve the probability of decoding error. It is desirable to find as simple a decoding algorithm as possible for correcting erasures as well as errors. Such an algorithm enables one to carry out generalized minimum distance decoding [1] for further improving the probability of decoding error. Forney first formulated an erasures-and-errors decoding algorithm for Bose–Chaudhuri–Hocquenghem (BCH) codes [2]. Berlekamp formulated an elegant erasures-and-errors decoding algorithm for BCH codes based on his previous errors-only decoding algorithm [3].

Goppa has discovered a wide class of linear codes which includes BCH codes and Srivastava codes as subclasses [4]–[6]. Goppa has described an errors-only decoding algorithm for Goppa codes similar to Peterson's algorithm for BCH codes. Retter [7] has shown that Goppa codes can be decoded with a BCH decoder as given by Berlekamp. Patterson has formulated an errors-only decoding algorithm for Goppa codes in a manner similar to Berlekamp's algorithm for BCH codes [8]. We have formulated an errors-only decoding algorithm for Goppa codes using Euclid's algorithm [9]. In this correspondence, we present an erasures-and-errors decoding algorithm that includes our errors-only algorithm as a special case.

In Section II, Goppa codes and Euclid's algorithm are briefly described. In Section III, the erasures-and-errors decoding algorithm for Goppa codes is described together with its correction capability. In Section IV, a method for solving the key equation for the erasures-and-errors decoding is described.

II. Preliminaries

The q-ary Goppa code of length n with Goppa polynomial $g(z)$ is defined as the set of vectors $(a_1,a_2,a_3,\cdots,a_n)$ that satisfy

$$\sum_{i=1}^{n} \frac{a_i}{z - \alpha_i} \equiv 0 \bmod g(z),$$

where q is a prime power, $g(z)$ is a polynomial over $GF(q^m)$ of degree $2t$, α_i is an element of $GF(q^m)$ excluding roots of $g(z)$, and m and t are positive integers. All α_i are distinct. The minimum distance of the code is at least $(2t + 1)$.

In the following we summarize the properties of Euclid's algorithm as described in [9]. Euclid's algorithm computes the greatest common divisor of two polynomials $r_{-1}(z)$ and $r_0(z)$ where we assume deg r_{-1} > deg r_0. Given polynomials $r_{i-2}(z)$ and $r_{i-1}(z)$ such that deg r_{i-2} > deg r_{i-1}, it determines quotient polynomials $q_i(z)$ and remainder polynomials $r_i(z)$ which satisfy the following relations

$$r_{i-2}(z) = q_i(z)r_{i-1}(z) + r_i(z)$$

and

$$\deg r_{i-1} > \deg r_i \tag{1}$$

iteratively until it finds $r_i(z) = 0$. We define the following polynomials

$$U_i(z) = q_i(z)U_{i-1}(z) + U_{i-2}(z)$$

and

$$V_i(z) = q_i(z)V_{i-1}(z) + V_{i-2}(z)$$

where $U_0(z) = 1$, $U_{-1}(z) = 0$, $V_0(z) = 0$, and $V_{-1}(z) = 1$. Then we have the following relations

$$r_i(z) = (-1)^i\{-V_i(z)r_{-1}(z) + U_i(z)r_0(z)\}, \tag{2}$$