COMPUTATIONAL & APPLIED MATHEMATICS

Volume 24, N. 3, pp. 393–398, 2005

ISSN 0101-8205
www.scielo.br/cam

Computing the minimal distance of cyclic codes

JOSÉ FELIPE VOLOCH

Department of Mathematics, University of Texas, Austin, TX 78712, USA

E-mail: voloch@math.utexas.edu

Abstract. We describe an algorithm that improves on the standard algorithm for computing the minimal distance of cyclic codes.

Mathematical subject classification: 94B15.

Key words: Cyclic codes, minimal distance, algorithm.

1 Introduction

The standard algorithm for computing the minimal distance of cyclic codes, due to Chen [2], proceeds by searching for low weight codewords among the codewords with few nonzero information symbols. We will give a more detailed description below. We present a new algorithm that follows the same strategy, adding a tradeoff of time for space. This a standard computational principle, but the actual use of it requires some extra work which we will describe below. Chen produced a table of the minimal distances of binary cyclic codes of length at most 65 which was extended to length at most 99 by Promhouse and Tavares [6].

Some improvements on Chen's algorithm were made by Coppersmith and Seroussi [3] who used their algorithm to compute the minimal distance of some binary and ternary quadratic residue codes. The binary quadratic residue codes are defined as follows. Given a prime $p \equiv \pm 1 \bmod 8$, let ζ be a primitive p-th root of unity in the algebraic closure of $\mathbf{F}_2$, the field of two elements. The hypothesis on p entails that the monic polynomial $a(x)$, say, whose roots are ζ^r, with r running over the non-zero quadratic residues modulo p, is defined

#618/04. Received: 13/X/04. Accepted: 22/XI/04.

over $\mathbf{F}_2$ and the cyclic code of length p whose generator polynomial is $a(x)$ is, by definition, the binary quadratic residue code of length p. Different choices of ζ lead to different choices of $a(x)$ that give different but equivalent codes. Their minimal distance d is, in general, not known although the lower bound $d \geq \sqrt{p}$ and minor improvements are known, see [5]. Prior to Coppersmith and Seroussi, the known values for the minimal distance of the binary quadratic residue codes were tabulated in [5]. It appears that the above mentioned lower bound can be often improved and perhaps, at least when $p-1$ has few factors, is far from the truth. The largest value dealt with by Coppersmith and Seroussi was $p = 113$ which has $d = 15$. In 1998, Boston [1], showed that the software package Magma could, in about ten days, compute the next value corresponding to $p = 137$, namely $d = 21$. Some of these calculations were extended by Grassl [4] who showed that for $p = 167$, the minimal distance is $d = 23$. Magma's code is proprietary but, from the vague description of the algorithm in the user's guide, it appears that it uses an algorithm similar to Chen's algorithm, which is attributed to A. Brouwer.

2 The algorithm

In this session we consider cyclic codes over an arbitrary finite field $\mathbf{F}_q$ with q elements. The basic fact underlying Chen's algorithm is the following lemma.

Lemma 1. *Let c be a codeword of weight w in an $[n, k]$ cyclic code and $r = \lfloor kw/n \rfloor$. Then there exists a cyclic shift of c with exactly r nonzero coordinates among its first k coordinates.*

See [3], Lemma 1, for a proof.

Starting with a known lower bound w_0 for the minimal weight ($w_0 = 1$ if no better bound is available), for $w = w_0, w_0+1, \ldots$, the algorithm finds a codeword of weight w or shows that none exists. The first value of w having a codeword of that weight is the minimal distance. Lemma 1 ensures that an exhaustive search through all possibilities for codewords with exactly $r = \lfloor kw/n \rfloor$ nonzero coordinates among its first k coordinates suffices to decide whether or not there exists a codeword of weight w.

Our improved algorithm is the following. We proceed as above computing only with codewords with exactly $r = \lfloor kw/n \rfloor$ nonzero coordinates among its first k coordinates but we do not do so exhaustively. Instead, we choose an integer $r', 0 \leq r' \leq r$ and compute a table of all codewords with exactly r' nonzero coordinates among its first k coordinates. We also take t as large as possible with $\lfloor (n-k)/t \rfloor > w - r$. For $m = 0, \ldots, w - r$ we sort the table first according to the value of the t consecutive coordinates beginning at $k + 1 + mt$ and, within this sorting we further sort according to the value of the largest index of a nonzero coordinate. We now loop over all codewords with exactly $r'' = r - r'$ nonzero coordinates among its first k coordinates. For each such codeword c'' we compute the value of its t consecutive coordinates beginning at $k + 1 + mt$ and select from the table those c' which have the **same** value as c'' in those coordinates and for which the smallest index with a nonzero coordinate in c'' is larger than the largest index with a nonzero coordinate in c'. We then compute the weight of $c'' + c'$ for the selected c'.

We claim that, if a codeword of weight w exists on the code, a codeword of weight w will be found by this procedure. This will show that our algorithm computes the minimal distance of the code. To verify this claim, it is enough to show, by Lemma 1, that if there exists a codeword c of weight w with exactly r nonzero coordinates among its first k coordinates, it will be found among the codewords whose weight are being computed. Indeed, let $c = c' + c''$ where c' (resp. c'') has exactly r' (resp. r'') nonzero coordinates among its first k coordinates and the indices with nonzero coordinates of c' are all smaller than the ones of c''. As $c = c' + c''$ has weight w and exactly r nonzero coordinates among its first k coordinates, we have that c' and c'' disagree in exactly $w - r$ of the their $n - k$ last coordinates. Therefore they must agree in some block of t consecutive coordinates beginning at $k + 1 + mt$ for some $m = 0, \ldots, w - r$, since $\lfloor (n-k)/t \rfloor > w - r$. Thus, for this value of m, when c'' comes up in the loop, the codeword c' will be selected from the table and the weight of $c'' + c'$ computed, as we wished to show.

We now compare the running time of our algorithm to that of Chen's. At the r-th step, the Chen algorithm computes the weight of $\binom{k}{r}(q-1)^r$ codewords. Our algorithm requires the precomputation, storage and sorting of $\binom{k}{r'}(q-1)^{r'}$

codewords. The sorting is negligible in relation to the other computations. The next step is a loop over the $\binom{k}{r''}(q-1)^{r''}$ codewords c'' with r'' nonzero coordinates among their first k coordinates. We will make the following assumption and will discuss its validity later. Namely we assume that the number of codewords $c \in C$ with with r nonzero coordinates among their first k coordinates and with its t consecutive coordinates beginning at $k+1+mt$ for some fixed $m \in \{0, \ldots, w-r\}$ being all zero is approximately $\binom{k}{r}(q-1)^r/q^t$. These codewords c will be exactly the ones for which we will compute the weight. So the total number of steps is approximately

$$(w-r+1)\left(\binom{k}{r'}(q-1)^{r'}+\binom{k}{r''}(q-1)^{r''}+\binom{k}{r}(q-1)^r/q^t\right).$$

This is often a substantial improvement. We still can choose r' so as to minimize the above expression for the number of steps, and the best values are r' near $r/2$. However, in practice, we cannot take r' too large, as its size determines the amount of storage required, which unlike time, is a limited resource.

It remains to discuss the assumption that the number of codewords $c \in C$ with r nonzero coordinates among their first k coordinates and with its t consecutive coordinates beginning at $k+1+mt$ for some fixed $m \in \{0, \ldots, w-r\}$ being all zero is approximately $\binom{k}{r}(q-1)^r/q^t$. Fix $m \in \{0, \ldots, w-r\}$ and consider the code $C_0 \subset C$ consisting of codewords of C having their t consecutive coordinates beginning at $k+1+mt$ being all zero. Consider the code C_1 of length k consisting of the projection of C_0 to its first k coordinates. We wish to estimate therefore the number of codewords of weight r in C_1. From the results of [5], Chapter 9, section 10, it follows that this number is approximately $\binom{k}{r}(q-1)^r/q^t$ if the minimal weight of the dual of C_1 is large.

Lemma 2. *The minimal distance of the dual of C_1 is at least $d^\perp - t$, where $d^\perp$ is the minimal distance of the dual of C.*

Proof. Let $a(x)$ be the generator polynomial of C of degree $n-k$. Let

$$f(x) = \ell_0 + \cdots + \ell_{k-1}x^{k-1} + x^k(h_0 + \cdots + h_{n-k-1}x^{n-k-1}) = \ell(x) + x^k h(x)$$

be a polynomial of degree at most $n-1$. In order that $f(x)$ belongs to C we must have $\ell(x) + x^k h(x) \equiv 0 \bmod a(x)$ which is equivalent to $h(x) \equiv$

$-x^{n-k}\ell(x) \bmod a(x)$. If we write $x^{n-k+i} \equiv \sum_{j=0}^{n-k-1} b_{ij}x^j \bmod a(x)$, then $f(x)$ belongs to C if and only if $h_j = -\sum_{j=0}^{k-1} b_{ij}\ell_i$. It follows that the code C_1 is given by the equations $\sum_{j=0}^{k-1} b_{ij}\ell_i = 0$ for $j = k+1+mt, \ldots, k+(m+1)t$. The dual of C_1 is thus spanned by the vectors $(b_{0,j}, \ldots, b_{k-1,j})$, $j = k+1+mt, \ldots, k+(m+1)t$. Now consider the code D spanned by the vectors of length n of the form $(b_{0,j}, \ldots, b_{k-1,j}, 0, \ldots, -1, \ldots, 0)$, where the -1 occurs at position j for $j = k+1+mt, \ldots, k+(m+1)t$. On one hand, D is a subcode of the dual of C and so its minimal distance is at least $d^\perp$. On the other hand, the projection of D onto its first k coordinates is the dual of C_1 and in this projection any codeword loses at most t nonzero coordinates, so the minimal distance of the dual of C_1 is at least $d^\perp - t$, as was to be shown. □

Since t is small, we conclude that our algorithm will have the running time discussed above if $d^\perp$ is large. This is the case for quadratic residue codes, since a quadratic residue code is essentially self-dual, and is also the case for most BCH codes since the minimal distance of their dual can be estimated via the Carlitz-Uchiyama bound [5] and other techniques such as [7].

3 Computations

We used (an earlier version of) our algorithm to check that the minimal distance of the binary quadratic residue code of length 167 is 23 before we learned of Grassl's calculation. Our calculation took longer but was done with different software and hardware, so the comparison doesn't mean much.

For comparison we checked that the quadratic residue code of length 79 has no codeword of weight 11 both using Chen and our algorithm in similar implementations using Pari/GP and on the same computer. Chen took 22.5 seconds, while ours with $r' = 4$ took 14.7 seconds. Apparently, for $r' = 5$, Pari/GP needs more RAM to store the table than what the computer had available.

Acknowledgments. The calculation was programmed using Pari/GP. I would also like to thank N. Boston, M. Grassl and M. Zou.

REFERENCES

[1] N. Boston, "The minimum distance of the [137,69] binary quadratic residue code" *IEEE Trans. Info. Theory*, **45** (1999), 282.

[2] C. L. Chen, "Some results on algebraically structured error-correcting codes" *Ph.D. dissertation*, Univ. of Hawaii, (1969).

[3] D. Coppersmith and G. Seroussi, "On the minimum distance of some quadratic residue codes" *IEEE Trans. Info. Theory*, **30** (1984), 407–411.

[4] M. Grassl, "On the minimum distance of some Quadratic-Residue Codes" *IEEE International Symposium on Information Theory, Proceedings*, p. 253, (2000).

[5] J. MacWilliams and N. Sloane, "The theory of error-correcting codes", North-Holland, (1977).

[6] G. Promhouse and S. Tavares, "The minimum distance of all binary cyclic codes of odd lengths from 69 to 99" *IEEE Trans. Info. Theory*, **24** (1978), 438–442.

[7] Voloch, J.F., "On the duals of binary BCH codes" *IEEE Trans. Info. Theory*, **47** (2001), 2050–2051.