Minimum Distance of Binary Nonlinear Codes*

Jaume Pujol Mercè Villanueva Fanxuan Zeng

May 4, 2012

Abstract

A binary nonlinear code can be represented as the union of cosets of a binary linear subcode. Using this representation, new algorithms methods to compute the minimum Hamming weight and distance are presented. The performance of these algorithms is also studied.

1 Introduction

Let $\mathbb{Z}_2$ be the ring of integers modulo 2 and let $\mathbb{Z}_2^n$ be the set of all binary vectors of length n. The *Hamming distance* $d(u, v)$ between two vectors $u, v \in \mathbb{Z}_2^n$ is the number of coordinates in which u and v differ. The *Hamming weight* $w(u)$ of $u \in \mathbb{Z}_2^n$ is $w(u) = d(u, \mathbf{0})$, where $\mathbf{0}$ is the all-zero vector of length n. A (n, M, d) *binary code* C is a subset of $\mathbb{Z}_2^n$ with M codewords and minimum Hamming distance d. The vectors of a code are called *codewords* and the *minimum Hamming distance* is the minimum value of $d(u, v)$ for all $u, v \in C$ and $u \neq v$.

Two binary codes C_1 and C_2 of length n are said to be *permutation equivalent* if there exists a coordinate permutation π such that $C_2 = \{\pi(c) \mid c \in C_1\}$. They are said to be *equivalent* if there exists a vector $a \in \mathbb{Z}_2^n$ and a coordinate permutation π such that $C_2 = \{a + \pi(c) \mid c \in C_1\}$. Note that two equivalent codes have the same minimum distance.

Given a binary code C, the problem of storing C in memory is a well-known problem. If C is linear, that is, it is a subgroup of $\mathbb{Z}_2^n$, then it can be compactly represented using a binary generator matrix. On the other hand, if C is nonlinear, then a solution would be to know whether it has another structure or not. For example, there are binary codes which have a $\mathbb{Z}_4$-linear or $\mathbb{Z}_2\mathbb{Z}_4$-linear structure and, therefore, they can also be compactly represented using a quaternary generator matrix [2], [5]. In general, binary codes without any of these structures can be represented as the union of cosets of a binary linear subcode of C [1]. This will allow us to represent a binary code as a set of representative codewords instead of as a set with all codewords. Moreover, this representative codewords can be organized as a matrix, called *parity-check system* [6], which is a generalization of the binary parity-check matrix for binary linear codes [7].

The problems of computing the minimum weight of a binary code C, denoted by $w(C)$, and minimum distance, denoted by $d(C)$, are important, and also necessary in order to establish its error-correcting capability. If C is linear, the minimum weight coincides with the minimum distance, and the Brouwer-Zimmerman minimum weight algorithm for linear codes over finite fields [10], [11] can be used. This algorithm has been implemented in the computational algebra system Magma [3], [4], [9]. On the other hand, if C is nonlinear, the minimum weight and minimum distance does not always coincide, and as far as we know there is not any efficient algorithm to compute them.

In this paper, we will propose a new algorithm to compute the minimum weight and minimum distance of a binary nonlinear code C, based on the coset structure and the known algorithms for linear codes. From now on, we will assume that $\mathbf{0} \in C$. Note that if C is linear, then $\mathbf{0} \in C$; but if C is nonlinear, then $\mathbf{0}$ does not need to belong to C. In this case, we can always consider a new binary code $C' = C + c$ for any $c \in C$, which is equivalent to C, such that $\mathbf{0} \in C'$.

2 Binary nonlinear codes

Two structural properties of binary codes are the rank and dimension of the kernel. The *rank* of a binary code C, r, is simply the dimension of the linear span, $\langle C \rangle$, of C. The *kernel* of a binary code

*This work has been partially supported by the Spanish MICINN under Grant TIN2010-17358, and by the Catalan AGAUR under Grant 2009SGR1224.

The authors are with the Department of Information and Communications Engineering, Universitat Autònoma de Barcelona, 08193-Bellaterra, Spain.

C is defined as $K(C) = \{x \in \mathbb{Z}_2^n \mid x + C = C\}$. Since $\mathbf{0} \in C$, $K(C)$ is a binary linear subcode of C. We will denote by k the dimension of the kernel of C. In general, C can be written as the union of cosets of $K(C)$, and $K(C)$ is the largest such linear code for which this is true [1]. Therefore, $C = \cup_{i=0}^{t}(K(C) + c_i)$, where $c_0 = \mathbf{0}$, $t + 1 = M/2^k$ and $M = |C|$. The parameters r and k can be used to distinguish between nonequivalent binary codes, since equivalent ones have the same r and k.

Let C be a binary code of length n with kernel $K(C)$ of dimension k and t coset leaders given by the set $S = \{c_1, \ldots, c_t\}$. Note that we can represent C as the kernel $K(C)$ plus the coset leaders S. Since $K(C)$ is linear, it can be compactly represented by its binary generator matrix G of size $k \times n$. Therefore, considering S as the matrix where in the t rows there are the coset leaders, the binary code C can be also represented by the matrix $\binom{G}{S}$. Since the kernel takes up a memory space of order $O(nk)$, the kernel plus the t coset leaders take up a memory space of order $O(n(k + t))$.

For the case $t = 0$, that is, when the binary code C is linear, we have that $C = K(C)$ and the code can be represented by its binary generator matrix, so the memory space is of order $O(nk)$. On the other hand, for the case $t + 1 = M$, this solution is as bad as representing the code as a set of all its codewords, so it takes up a memory space of order $O(nM)$, where $M \gg k$.

3 Minimum weight and minimum distance

Using the representation given in the previous section, we can define new algorithms to compute the minimum weight and minimum distance of a binary nonlinear code.

Given a binary code C and a vector $v \in \mathbb{Z}_2^n$, let $K_v = K(C) \cup (K(C) + v)$. Since $K(C)$ is linear, then K_v is also linear. Let w_v be the minimum weight of the binary linear code K_v.

Proposition 1 *Let $C = \bigcup_{i=0}^{t} \big(K(C) + c_i\big)$ with $t \geq 2$. The minimum weight of C can be computed as $\min(\{w_{c_i} \mid i = 1, \ldots, t\})$.*

Proposition 2 *Let $C = \bigcup_{i=0}^{t} \big(K(C) + c_i\big)$ with $t \geq 2$. The minimum distance of C can be computed as $\min(\{w_{c_i} \mid i = 1, \ldots, t\} \cup \{w_{c_i + c_j} \mid i, j = 1, \ldots, t \text{ and } i < j\})$.*

Using Propositions 1, 2 and applying the known algorithms to compute the minimum weight of a linear code, we can define Algorithms 1 and 2 to compute the minimum weight and minimum distance of a binary nonlinear code, respectively. Note that the complexity of these two algorithms depends strongly on the number of coset leaders t. For the minimum weight, we compute t times the minimum weight of a linear code K_v, and for the minimum distance, $\binom{t+1}{2}$ times.

Algorithm 1: MinimumWeight

```
Data: A binary code C.
Result: The minimum weight w(C).
begin
    w(C) ← Length(C)
    for i ∈ [1, ..., t] do
        K_{c_i} ← K(C) ∪ (K(C) + c_i)
        w(C) ← min(w(K_{c_i}), w(C))
    return w(C)
```

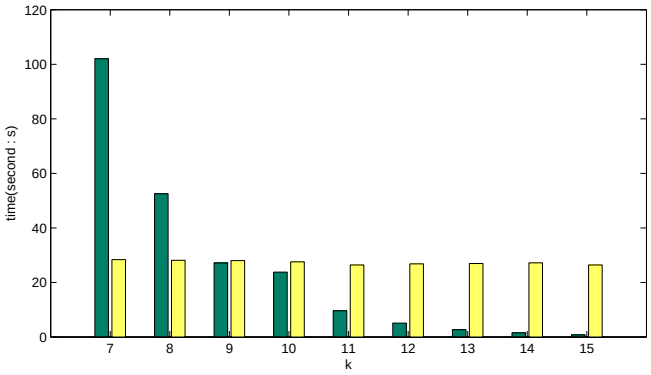
In order to know the efficiency of these two algorithms, we compare them with brute force method. Note that a nonlinear binary code with a kernel of dimension k and t coset leaders has $M = 2^k(t + 1)$ codewords. Computing minimum weight: By using brute force, we need to exam the weight of every codeword to decide the minimum weight, so it will take $M = 2^k(t + 1)$ times the computation of a codeword weight. By using Algorithm 1, we need to compute t times the minimum weight of a linear code of dimension $k + 1$. Computing minimum distance: By using brute force, we need to exam the distance between every pair of codewords, so it will take $\binom{M}{2} = \binom{2^k(t+1)}{2}$ times the computation of a vector distance to get the minimum distance. On contrary, by using Algorithm 2, we need to compute $\binom{t+1}{2}$ times the minimum weight of a linear code of dimension $k + 1$. Note that when k is large, Algorithms 1 and 2 will save a lot of time.

Example 1 *Let K be the binary linear code of length $n = 31$, dimension 5, and minimum distance 16, constructed as the dual of the binary Hamming code of length $n = 31$. Let $C_{31} = \bigcup_{i=0}^{3} \big(K(C_{31}) + c_i\big)$,*

Algorithm 2: MinimumDistance

Data: A binary code C.
Result: The minimum distance $d(C)$.
begin
  $d(C) \leftarrow \text{Length}(C)$
  for $i \in [0, \ldots, t-1]$ **do**
    for $j \in [i+1, \ldots, t]$ **do**
      $K_{c_i+c_j} \leftarrow K(C) \cup (K(C) + c_i + c_j)$
      $d(C) \leftarrow \min(w(K_{c_i}), d(C))$
  return $d(C)$

Figure 1: Time of computing $w(C)$ using Algorithm 1 compared with brute force, for binary codes of length $n = 100$, size $M = 2^{19} \cdot 31$, and kernel of dimension $k \in \{7, \ldots, 15\}$.



where the kernel $K(C_{31}) = K$, $c_0 = \mathbf{0}$, *and the coset leaders are:*

$$c_1 = (00100011100110100111100010111 10)$$
$$c_2 = (01011010101111001011101001111 01)$$
$$c_3 = (00000111000111011010001111010 11)$$

It is easy to check that the minimum weight of C_{31} *is* $w(C_{31}) = 10$ *and its minimum distance* $d(C_{31}) = 8$. *The time of computing* $w(C_{31})$ *using brute force and Algorithm 1 are 0.00018 and 0.00060 seconds, respectively. Note that sometimes a brute force calculation can be a faster way to get the minimum weight. However, to compute* $d(C_{31})$ *it is much faster to use Algorithm 2, since the time of computing it using brute force and Algorithm 2 are 0.00840 and 0.00126 seconds, respectively.*

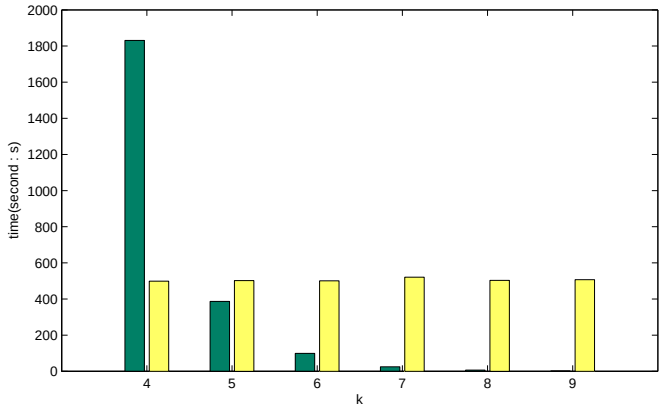
In order to show the difference of time between computing the minimum weight with Algorithms 1, 2 and brute force, we have chosen random binary codes of length $n = 100$. In Figures 1 and 2, the dark beam represents the time of using Algorithm 1 and 2, respectively. The light beam represents the time of using brute force. All the tests have been done in MAGMA version V2.18-3, running on a server with Intel Xeon (clock speed 2.40GHz) and 32GB memory. As we can see, keeping the same length and number of codewords, the time of Algorithms 1 and 2 decreases sharply while the kernel dimension k increases (or equivalently, while the number of cosets t decreases).

4 Conclusions

In this paper, we have presented algorithms to compute the minimum weight and minimum distance of binary nonlinear codes. These algorithms are especially suitable for codes with a big kernel while brute force works better for codes with a small kernel.

In a near future study, we will establish the relationship between the performance of these algorithms and the parameters of the code: length n, dimension of the kernel k, and number of coset leaders t. In this way, we will be able to decide which algorithm to use for given parameters. In order to get this relationship, we will include both work factor and practical experiments. We will also focus on improving these algorithms, and generalizing them to q-ary nonlinear codes.

Figure 2: Time of computing $d(C)$ using Algorithm 2 compared with brute force, for binary codes of length $n = 100$, size $M = 2^{10} \cdot 31$, and kernel of dimension $k \in \{4, \ldots, 9\}$.



References

[1] H. Bauer, B. Ganter and F. Hergert, "Algebraic techniques for nonlinear codes," *Combinatorica*, vol. 3, pp. 21-33, 1983.

[2] J. Borges, C. Fernández, J. Pujol, J. Rifà and M. Villanueva, "$\mathbb{Z}_2\mathbb{Z}_4$-linear codes: generator matrices and duality," *Designs, Codes and Cryptography*, vol. 54, pp. 167-179, 2010.

[3] J. J. Cannon and W. Bosma (Eds.), *Handbook of* Magma *Functions*, Edition 2.13, 4350 pages, 2006. (`http://magma.maths.usyd.edu.au/magma/`)

[4] M. Grassl, *Searching for linear codes with large minimum distance*, in: W. Bosma and J. Cannon (Eds.) *Discovering Mathematics with Magma*, Springer, 2006.

[5] A. R. Hammons, P. V. Kumar, A. R. Calderbank, N. J. A. Sloane and P. Solé, "The $\mathbb{Z}_4$-linearity of kerdock, preparata, goethals and related codes", *IEEE Trans. on Information Theory*, vol. 40, pp. 301-319, 1994.

[6] O. Heden, "On perfect p-ary codes of length $p + 1$," *Designs, Codes and Cryptography*, vol. 46, no. 1, pp. 45-56, 2008.

[7] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, Amsterdam: North-Holland Publishing Company, 1977.

[8] A. Vardy, "The intractability of computing the minimum distance of a code," *IEEE Trans. Inform. Theory*, vol. 43, no. 6, pp. 17571766, 1997.

[9] G. White, "Enumeration-based Algorithms in Coding Theory," PhD Thesis, U. of Sydney, 2006.

[10] K.-H. Zimmerman, "Integral Hecke Modules, Integral Generalized Reed-Muller Codes, and Linear Codes," Tech. Rep. 3-96, Technische Universität Hamburg-Harburg, 1996.

[11] A. Betten, H. Fripertinger, A. Kerber, A. Wassermann, and K.-H. Zimmerman, *Codierungstheorie: Konstruktionen und Auwendungen linearer Codes*, Berlin: Springer, 1998.