Representation, constructions and minimum distance computation of binary nonlinear codes

Jaume Pujol, Mercè Villanueva, and Fanxuan Zeng
Universitat Autònoma de Barcelona
fanxuan@deic.uab.cat

Abstract

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* $wt(u)$ of $u \in \mathbb{Z}_2^n$ is $wt(u) = d(u, \mathbf{0})$, 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 *minimum Hamming distance*, denoted by $d(C)$, 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 *equivalent* if there exists a vector $a \in \mathbb{Z}_2^n$ and a coordinate permutation π such that $C_2 = \{a + \pi(c) : c \in C_1\}$. Note that two equivalent codes have the same minimum distance. 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 $\mathbf{0} \in C'$. Therefore, from now on, we assume that $\mathbf{0} \in C$.

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. In general, binary codes without any of these structures can be represented as the union of cosets of a binary linear subcode of C. This allows us to represent them as a set of representative codewords instead of as a set with all codewords.

The *kernel* of a binary code C is defined as $K(C) = \{x \in \mathbb{Z}_2^n \ : \ x + C = C\}$. Since $\mathbf{0} \in C$, $K(C)$ is a binary linear subcode of C. We denote by k the dimension of $K(C)$. 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,

$$C = \bigcup_{i=0}^{t} \Big(K(C) + c_i \Big), \tag{1}$$

where $c_0 = \mathbf{0}$, $t + 1 = M/2^k$ and $M = |C|$. Note that we can represent C as the kernel $K(C)$ plus the coset leaders $L = \{c_1, \ldots, c_t\}$. It is important to emphasize that the codewords in L are not necessarily the ones having minimum weight in the coset. Since $K(C)$ is linear, it can be compactly represented by its binary generator matrix G of size $k \times n$. Therefore, considering L as the matrix where in the t rows there are the coset leaders, the binary code C can be also represented by the matrix $\binom{G}{L}$. 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 example, applying this representation to the set of all completely classified binary perfect codes of length 15 and extended perfect codes of length 16, we obtain very significant compression rates. It is known that there are exactly 5983 binary perfect codes of length 15 and 2165 binary extended perfect codes of length 16, each one having 2048 codewords [2]. In the first case, instead of taking up $5983 \cdot 2048 \cdot 4 = 49012736$ hexadecimal numbers by encoding each codeword in hexadecimal notation, it only takes 3677928 hexadecimal numbers by storing the codewords of a generator matrix of the kernel and the set of coset leaders for each binary code. This gives a compression rate of 92.5%. Similarly, in the second case, the extended perfect codes of length 16 can be compressed from $2165 \cdot 2048 \cdot 4 = 17735680$ hexadecimal numbers to 1439336, which gives a compression rate of 91.9%.

In order to compute the kernel and coset leaders of a binary code C of length n, according to the definition of $K(C)$, it is necessary to classify the M codewords of C. Since $M = 2^k(t+1)$, the algorithm must be at least exponential on k, the dimension of $K(C)$. A straightforward algorithm to compute the kernel from the definition of $K(C)$ requires $M^2 \log M$ operations,

if C is sorted. However, this algorithm can be improved using the following two properties: (1) if $K' \subseteq K(C)$, then $v \in K(C)$ if and only if $K' + v \subseteq K(C)$; (2) if $K' \subseteq K(C)$, $v \in C$ and $(C \backslash K') + v \subseteq C$, then $v \in K(C)$. Therefore, depending on k, the complexity can be reduced. If $k = 0$ we still need $M^2 \log M$ operations, but if $k > 0$ we obtain a complexity of order $O(kM \log M)$. Note that, for large M, $kM \ll M^2$.

Although the exponential behaviour of the kernel computation, using the representation given above, we can manipulate and construct new binary nonlinear codes from old ones in a more efficient way. Specifically, we show how to establish the equality and inclusion of two given nonlinear codes from their kernels and coset leaders, and how to compute the kernel and coset leaders of related new codes (union, intersection, extended, punctured, shorten, direct sum, Plotkin sum) from given ones, which are represented in this structure. All these results will be written to be implemented easily as algorithms.

Given a binary code C, the problem of computing its minimum distance is also important, and necessary in order to establish its error-correcting capability. This problem is computationally difficult, and has been proven to be NP-hard. If C is linear, the minimum distance coincides with the minimum weight, denoted by $wt(C)$, and the Brouwer-Zimmerman minimum weight algorithm for linear codes over finite fields [3] can be used. We propose new algorithms 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. 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.

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

Using Proposition 1 and applying the known Brouwer-Zimmermann algorithms, we can compute the minimum weight and distance of a binary nonlinear code. 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. An estimate of the total work an algorithm performs is referred to as *work factor* [4]. We study the work factors for these algorithms to compare them with brute force. An improvement is given to the proposition by avoiding repeated computations in each coset.

Finally, the previous algorithm can also be used to decode a binary linear code C. For a received vector $u \in \mathbb{Z}_2^n$, in order to decode it as a codeword from C, we look for a vector e of minimum weight such that $u - e \in C$. This is equivalent to find a vector e of minimum weight in the coset containing u, which is $C + u$.

Proposition 2 *Let C be a binary linear code with minimum distance d. For a received vector $u = c + e \notin C$, where $c \in C$, let $C_u = C \cup (C + u)$. If $wt(e) < d$, then the received vector u can be decoded as $c' = u - e' \in C$, where e' is a vector of minimum weight in C_u, so $wt(e) = wt(e')$. Note that if $wt(e) \leq \lfloor \frac{d-1}{2} \rfloor$, then $e' = e$ and $c' = c$.*

In this way, we can decode a received vector as long as less than d errors have been added to the transmitted codeword. When d or more than d errors occurs during the transmission, the minimum vector of C_u could come from C, and an error vector e can not be found. Therefore, the method provides a complete decoding but only up to $d-1$ errors. Note that if the covering radius of C, denoted by ρ, satisfies $\rho \leq d - 1$, that is when C is a maximal code, we actually obtain a complete decoding.

References

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

[2] P. R. J. Östergård and O. Pottonen, "The perfect binary one-error-correcting codes of length 15: part I-classification." *IEEE Trans. Inform. Theory*, vol. 55, no. 10, pp. 4657-4660, 2009.

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

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

Keywords

binary nonlinear codes, kernel, minimum distance, decoding