On the Complexity of the Greedy Construction of Linear Error-Correcting Codes

Dejan Spasov[1], Marjan Gusev[1]

[1] Institute of Informatics, Faculty of Natural Science, Ss. Cyril and Methodius University, Skopje, Macedonia
{dejan, marjan}@ii.edu.mk

Abstract. Greedy algorithms are one of the oldest known methods for code construction. They are simple to define and easy to implement, but require exponential running time. Codes obtained with greedy construction have very good encoding parameters; hence, the idea of finding faster algorithms for code generation seems natural. We start with an overview of the greedy algorithms and propose some improvements. Then, we study the code parameters of long greedy codes in attempt to produce stronger estimates. It is well known that greedy-code parameters give raise to the Gilbert-Varshamov bound; improving this bound is fundamental problem in coding theory.

Keywords: Linear codes, Greedy Codes, Lexicodes, Gilbert-Varshamov Bound, Greedy Algorithms.

1 Introduction

Given an n-dimensional vector space F_q^n over some finite field F_q, a *code* C is any subset of M elements. Let $d(x,y)$ denotes the *Hamming distance* between two vectors x and y, i.e. the number of coordinates in which they differ; then we can define *minimum distance* d of a code as $d=\min(d(x,y))$, $\forall x,y\in C$. We write (n,M,d) to denote a code of M elements and minimum distance d over F_q^m.

The main focus in this paper is on the *linear codes*. The $M=q^k$ codewords of a linear code form a k-dimensional subspace in F_q^n. We write $[n,k,d]$ to denote a linear code of dimension k and minimum distance d. A linear code C has a $k\times n$ generator matrix G and $(n-k)\times n$ parity check matrix H, such that $HG^T=0$. Throughout the paper, we will assume that the generator and parity check matrices are in standard form $G=[I \quad A]$, and $H=[-A^T \quad I]$.

We use $wt(x)\in \mathrm{N}$ to denote the Hamming weight of the vector x, i.e. the number of nonzero positions of x, $wt(x)=dist(x,0)$. In addition, δ will denote the relative distance of the code $\delta=d/n$, and R will be the code rate $R=\log_q(M)/n$. A string of n ones, $11\ldots1$, will be written as 1^n, and the concatenation of two strings a and b will be represented with $(a\,|\,b)$.

In estimating the complexity of an algorithm, we adopt Random Access Machine (RAM) as a computational model. The time complexity is measured as the number of basic (sequential) steps needed for instance of the algorithm to end. It is considered that RAM has unlimited memory with instant access. Thus the space complexity is simply the number of registers used by an instance of the algorithm.

In Section 2, we study the complexity of the greedy algorithms for code generation. In 2.3, we will introduce an algorithm that we call the Jenkins' construction. We will show that this algorithm has better time complexity than the similar constructions (2.1 and 2.2). In 2.4, we will explain the Lexicographic construction- a greedy approach that originally was introduced in [1]. In this paper we will present faster and less memory demanding version of the algorithm, generalized over arbitrary alphabet $GF(q)$. Our contribution to the algorithm is underlined with lemma 1 and theorem 1. Using them we avoid using long coset leaders and we work with their coset weights instead.

It is obvious that due to the exponential nature of the greedy algorithm it is impossible to generate long codes in an acceptable time. Thus, in section 3, we want to estimate the code parameters of the long greedy codes. In 3.1, we develop counting mechanisms (theorems 2, 3, and 4) that give better estimate on code parameters than the well-known Gilbert-Varshamov bound. In 3.2, we use theorem 3 to improve some published results (see (12) and (13)). Some parts in section 3 have already been published in [14]; we present them again for the reason of completeness.

2 Greedy Algorithms for Code Construction

Fundamental problem in coding theory is how to find *optimal codes*. The code (n,M,d) is optimal if it has maximal number of codewords M for a given n and d. In general, finding an optimal code is considered to be a difficult problem. Trivial way to do this is by super-exponential search over all possible orderings of the field F_q^n. For small fields ($q \le 9$, $n \le 256$), there exist tables of best known (some of them optimal) codes [7], but for larger spaces optimal-code parameters can be estimated with the Gilbert-Varshamov bound and its asymptotical variant.

It is well-known fact that a simple greedy search produces a code with parameters δ and R that follow the Gilbert-Varshamov bound. In binary case, unproven conjecture is that this is the best known method for code construction. A particularly mysterious truth is that almost all random linear codes meet the asymptotic Gilbert-Varshamov bound.

2.1 The Gilbert's Construction

In general, Gilbert's Construction produces a nonlinear (n, M, d) code. Given the length n and the minimum distance d, the algorithm searches over the entire space F_q^n and greedily adds to C the first vector x, such that $d(x,c) \geq d, \forall c \in C$. The time complexity of the algorithm is $O\left(nq^{(1+R)n}\right)$, because there are q^n vectors to be checked against a set of at most M codewords, $M = q^{Rn}$.

Important to notice is that the space needed to store all M codewords is nq^{Rn}. This is one of the reasons why nonlinear codes are not very popular for practical purposes.

2.2 The Varshamov's Construction

Given the codimension $m = n - k$ and the distance d, the Varshamov's algorithm searches over F_q^m and produces the parity check matrix H of a linear code. The algorithm greedily adds to H the first vector x that is not linear combination of $d-2$ or less columns of H. It can be verified that with this construction every combination of $d-1$ columns of H is linearly independent; hence the resulting code is linear with minimum distance d.

The time complexity of the algorithm is $O\left(n^2 q^{n(1-R+H(\delta))}\right)$, because each of the q^{n-k} vectors of F_q^m must be compared with at most $q^{nH(\delta)}$ linear combinations, where $H(\delta)$ is the entropy function, and $q^{nH(\delta)}$ represents all $d-2$ or less combinations from an n-element set, when $n \to \infty$. The space complexity of the Varshamov's construction is proportional with the dimensions of the parity check matrix H.

2.3 The Jenkins' Construction

The idea in this construction is to build the generator matrix of a systematic code $G = [I \quad A]$. For each $x \in F_q^m$ we form the vector $c_{k+1} = (10\ldots0 \mid x)$, $c \in F_q^n$. Then we check if all linear combinations of rows of the matrix $G_{k+1} = \begin{bmatrix} G \\ c_{k+1} \end{bmatrix}$ have Hamming weight at least d. There are two important facts about the systematic codes that help to reduce the number and size of the row checks: every linear combination of d or more rows of G_{k+1} has Hamming weight $\geq d$, and

$$wt\left(c_{i_1} + c_{i_2} + \cdots + c_{i_N}\right) = N + wt\left(a_{i_1} + a_{i_2} + \cdots + a_{i_N}\right)$$

where each c_{i_j} is a row of G_{k+1}, and a_{i_j} are the parity bits of c_{i_j} that belong to the A-matrix.

The worst-case running time of the algorithm is $O\left(n^2 q^{n(1-R+RH(\delta/R))}\right)$, because q^{n-k} vectors must be checked in $q^{nRH(\delta/R)}$ linear combinations, while the space complexity is proportional with the size of the A-matrix. We named this algorithm Jenkins' construction because, to our knowledge, it was first implemented by B. Jenkins in [2], by using divide-and-conquer strategy in such a manner that first it checks all linear combinations of one vector, then all linear combinations of two vectors, and so on.

2.4 The Lexicographic Construction

Any linear code C, with parity check matrix H, partitions the entire space F_q^n into q^{n-k} disjoint sets of q^k elements called *cosets*. Two vectors $x, y \in F_q^n$ belong to the same coset if and only if $x - y \in C$. Each coset has two special vectors: a unique *syndrome* $s \in F_q^{n-k}$ and a *coset leader* $e(s) \in F_q^n$. The coset leader $e(s)$ is the minimum weight vector in the coset. The syndrome s is obtained with $s = H \cdot x^T$, where x is any vector from the coset. Usually, all q^{n-k} pairs $(e(s), s)$ are stored in a table of size nq^{n-k} called *standard array*. The standard array is used to describe minimum distance decoding principle called *syndrome decoding*. In syndrome decoding, the error pattern $e \in F_q^n$ that scrambled the transmitted message $y \in F_q^n$ is considered to be the coset leader $e(s)$ associated with the syndrome $s = H \cdot y^T$. The weight of the heaviest coset leader in the array is called *covering radius* ρ of the code.

In [1], A. Trachtenberg published an algorithm that uses the standard array of a binary $[n, k, d]$ code C to build a new $[n_{k+1}, k+1, d]$ code in a greedy fashion. The algorithm, named Lexicographic Construction, runs in time $\mathrm{O}\left(n^2 2^{n-k}\right)$ and space $\mathrm{O}\left(n 2^{n-k}\right)$. In [4], D. Spasov improved the space complexity of the Lexicographic Construction to $\Theta\left(2^{n-k}\right)$, by demonstrating that the algorithm can work with the weights of the coset leaders $w(s) \in \mathrm{N}$ instead of the coset leaders $e(s)$. In addition, in [4], the time complexity was reduced to $\mathrm{O}\left(n 2^{n-k}\right)$ by making the greedy choice less time consuming. In this section, we will describe the Lexicographic Construction, as implemented in [4], but generalized over $GF(q)$.

Given the code $[n, k, d]$, let assume that the pairs syndrome-coset weight $(s, w(s))$ are stored in a table. The Lexicographic Construction is iterative algorithm that can be described in three steps. In the first step the covering radius of the code is found, $\rho = \max(w(s)), \forall s \in F_q^m$. In the second step, a vector from the maximum weight cosets is chosen, and a new basis codeword is formed. In the third step the table $(s_{k+1}, w(s_{k+1}))$ for the new $[n_{k+1}, k+1, d]$ code is computed from $(s, w(s))$.

Even though, for the second step, it seems that we can not obtain a coset member from $(s, w(s))$, the following lemma shows how to get one for systematic codes:

Lemma 1. Vectors l and $(0^k \mid s)$ belong to the same coset if and only if $Hl^T = s$.

Proof. $H \cdot (0^k \mid s)^T = s$ ■

Hence, in the second step, the algorithm chooses the syndrome s of a maximum weight coset, $w(s) = \rho$, and attaches to it k zeroes, $(0^k \mid s)$. Then the new basis codeword c_{k+1} is constructed by adding $d - \rho$ ones:

$$c_{k+1} = (1^{d-\rho} \mid 0^k \mid s) \tag{1}$$

A proof that in binary case the code $[n_{k+1}, k+1, d]$ will have minimum distance d can be found in [1]. This proof can be generalized for any $GF(q)$.

The third step is an effective mechanism to build the table $(s_{k+1}, w(s_{k+1}))$ from the existing $(s, w(s))$. First, note that any vector s_{k+1} that belongs to $\mathbf{F}_q^{n_{k+1}-k-1}$ is a syndrome of a coset. This syndrome s_{k+1} is considered to be a concatenation of two vectors $s_{k+1} = (v \mid s)$, such that v and $s, s \in \mathbf{F}_q^{n-k}$. Second, we need the following definition:

Definition 1. Given two syndromes s_k and s, the *syndrome companion set* of s_k with respect to s is the set:

$$\{y_i \mid y_i = \mathrm{s_k} + i \cdot s,\ \ i \in \mathrm{F_q}\} \tag{2}$$

There are q^{n-k-1} disjoint syndrome companion sets and each syndrome belongs to only one companion set. Next, we can proceed to the main result:

Theorem 1. Given ρ, c_{k+1}, and $(s_k, w(s_k))$. The table $(s_{k+1}, w(s_{k+1}))$ associated with $[n_{k+1}, k+1, d]$ can be efficiently constructed by carrying out the following minimization for each syndrome $s_{k+1} = (v \mid s_k)$

$$w(s_{k+1}) = \min_{\substack{i=0..q-1 \\ y_i = s_k + i \cdot s}} \{wt(v + i^{d-\rho}) + w(y_i)\} \quad . \tag{3}$$

Proof Outline. Any coset from $\mathbf{F}_q^{n_{k+1}-k-1}$ can be seen as concatenation of q companion cosets from $\mathbf{F}_q^{n-k}$. In a simple case, the new coset leader is chosen to be the minimum weight coset leader among the companion cosets. Companion cosets are found by finding the syndrome companion set (2) for each syndrome ■

The Lexicographic Construction starts with the repetition code and iterates as long as there are available memory resources. The space complexity of the algorithm is

$O\left(\log(n)2^{\lceil \lg(q)\cdot\rceil(1-R)n}\right)$. In practice, we can speak about implementation only for the case $d = const$, thus the size of the registers needed for storing $w(s_k)$ can be considered constant and the space complexity becomes $\Theta\left(2^{\lceil \lg(q)\cdot\rceil(1-R)n}\right)$.

For reference purposes, theorem 2.2 published in [5] is a generalization of theorem 3 from [1] for q-ary alphabet. Our Theorem 1, not only shows that the same conclusion can be derived for coset weights $w(s)$, but also shows how to find the coset companions. In contrast, the method used for finding coset companions in [1] is binary search.

3 Estimate of the Parameters of the Greedy Codes. The Gilbert-Varshamov bound.

Let H is the parity check matrix of some binary code $[n-1, k-1, d]$. Let $H(n-k, d-2)$ denotes the set of all unique $(n-k)$-tuples that are linear combination of $(d-2)$ columns of H. Then a code with parameters $[n,k,d]$ does exist provided

$$|H(n-k, d-2)| \leq 2^{n-k} - 2 \tag{4}$$

The existence of $[n,k,d]$ lower-bounds the existence of the optimal code for given n and d. Let $V(n,d)$ denotes the number of all possible d or less combinations from an n-element set

$$V(n,d) = \sum_{i=0}^{d} \binom{n}{d}. \tag{5}$$

Then, since $|H(n-k, d-2)|$ cannot be larger than $V(n-1, d-2)$ we obtain a simple combinatorial estimate of (4)

$$V(n-1, d-2) \leq 2^{n-k} - 2 \tag{6}$$

known as the Gilbert-Varshamov (GV) bound. One of the most challenging problems in coding theory is how to improve the GV bound, especially for infinite code length n. In binary case, so far, only one asymptotic improvement is known [8], and a few non-asymptotic ones (see [10], [12], and [13]).

3.1 Some results on the Gilbert-Varshamov bound

We start with a simple improvement of the GV bound for the case when d is even number:

Theorem 2. Let the minimum distance d is even number. Then the code $[n,k,d]$ does exist provided

$$V(n-2,d-3) \le 2^{n-k-1} - 2 . \tag{7}$$

Proof. Construct the code $[n,k,2t]$ from the code $[n-1,k,2t-1]$ by adding overall parity check. Use (6) to find $[n-1,k,2t-1]$. ■

It can be shown that the right-hand of (7) is always smaller than the right-hand of (6) by a factor n/d. Despite the simplicity of theorem 2, we have not found any publication that mentions it. Theorem 3 was presented in [14]; however later we discovered that it was already published in similar fashion in [13]:

Theorem 3. [13] The code $[n,k,d]$ can be extended to a code with parameters $[n+l+1,k+l,d]$ provided that

$$\sum_{\substack{i=1 \\ i=i+2}}^{\min(l,d-2)} \binom{l}{i} V(n,d-2-i) \le 2^{n-k} . \tag{8}$$

Interesting to note is that the existence of $[n,k,d]$ can be supported with the Varshamov bound (6), or by using (8) recursively. The recursion ends with the repetition code.

Proof Outline. We build the parity check matrix H_{m+1} recursively from H_m, $H_{m+1} = [H_m \quad L_m]$. In order to estimate $|H(n-k,d-2)|$, we count only those linear combinations that include odd number of vectors from L_m. The parameter l is the number of columns of the matrix L_m. ■

In absence of stronger evidence, simulations suggest that theorem 3 non-asymptotically improves the GV bound when $\delta = const$.

If we compare code parameters that are solution of (8) with the parameters obtained from running the greedy algorithm [4], we will notice a considerable gap. This implies that many linear combinations from $H(n-k,d-2)$ are counted multiple times. Theorem 4 attempts to improve this over-counting:

Theorem 4. The code $[n,k,d]$ can be extended to a code with parameters $[n+l+1,k+l,d]$ provided that

$$\begin{aligned} &\sum_{\substack{i=1 \\ i=i+2}}^{\min(l,d')} \binom{l}{i} V(n,d'-i) - \\ &-\frac{1}{2} \sum_{\substack{t=2 \\ t=t+2}}^{\min(l,d')} \binom{l}{t} \sum_{\substack{i=1 \\ i=i+2}}^{t} \binom{t}{i} \sum_{j=t-i}^{d'-i} \binom{d'}{j} V(n-d',d'-\max(i+j,t-i+d'-j)) \le 2^{n-k} \end{aligned} . \tag{9}$$

where $d' = d-2$.

Proof Outline. We will show an example of the case when a vector is counted twice with (8). Let $H_{m+1} = [H_m \quad L_m]$. Then for any two columns $l_1, l_2 \in L_m$, there exist $d-2$ columns in H_m such that $l_1 + l_2 + h_1 + h_2 + \cdots + h_{d-2} = 0$. If we transfer some vectors on the right-hand side we obtain

$$l_1 + h_1 + h_2 + \cdots + h_i = l_2 + h_{i+1} + h_{i+2} + \cdots + h_{d-2} .$$

Hence, we observe that the vector $l_1 + h_1 + h_2 + \cdots + h_i$ is counted twice. Theorem 4 is simply a generalization of this observation that includes all linear combinations of even number of vectors from L_m. ■

For Hamming codes $(d = 3)$, there is no difference between theorem 3 and 4; however, for $d = 5$ inequality (9) becomes slightly better, namely

$$l\left(1 + n + \binom{n}{2}\right) - 3\binom{l}{2} + \binom{l}{3} \le 2^{n-k-1} . \tag{10}$$

3.2 Comparison with prior work

To the best of our knowledge, theorem 3 was first published in [13] in order to estimate the parameters of the greedy lexicodes. However, in [13] it is not mentioned that $[n,k,d]$ needs not to be a greedy code, and that (8) can be used recursively, first to guarantee existence of $[n,k,d]$, then the existence of $[n+l+1, k+l, d]$.

Elia [10] reported the following result: Let the code $[n-2, k-1, d]$ does exist. Then the code $[n,k,d]$ does exist too, provided

$$V(n-2, d-3) \le 2^{n-k-1} . \tag{11}$$

If we restrict l to be at most 1 then (8) is precisely Elia's result. Moreover, letting $l \le 2$ we obtain improvement of (11); namely, assuming prior existence of the code $[n-3, k-2, d]$, the code $[n,k,d]$ does exist if the following holds true

$$V(n-3, d-3) \le 2^{n-k-2} . \tag{12}$$

Even though (7) and (11) are the same inequality, they are used in a different context. In [10], inequality (11) is used only after the existence of $[n,k,d]$ is secured. In theorem 2, prior existence of $[n,k,d]$ is not required, but (7) is restricted only for codes with even minimum distance.

Barg, Guritman, and Simonis [12] reported the following remark: The code $[n,k,d]$ with covering radius $\rho \le d-2$ can be extended to $[n+d-\rho-1, k+1, d]$. In this context, if the covering radius of $[n,k,d]$ is strictly less than $d-2$, then (8) guarantees existence of the trivial lengthening $[n+1, k, d]$. However if we have prior knowledge of the covering radius, we can modify (8) so that we obtain at least the same result as in [12]. For example, similar to (11), we can extend remark 13 from [12], i.e. if

$$V(n-1,\alpha) \le 2^{n-k-1}. \tag{13}$$

for some $\alpha \le d-1$, then any $[n,k,d]$ code can be extended to an $[n+d-\rho, k+2, d]$ code. For $\alpha = d-3$ this reduces to (12).

Jiang and Vardy have developed a graph-theoretic approach to asymptotically improve the GV bound for nonlinear codes [8], [9]. They were able to show that the code (n,M,d) does exist provided

$$c\frac{V(n,d-1)}{n} \le 2^{n-\lceil \log_2 M \rceil}. \tag{14}$$

where the constant c is at least $1/2 + o(1)$, as reported in [9]. How does (8) compares with (14)? So far, we were unable to prove that the left-hand of (8) can be smaller by a factor n. Hence, one may assume that (14) guarantees existence of a code with better parameters than (8). However, in general inequality (14) guarantees existence of a non-linear code, while (8) pertains to the linear codes. Gaborit and Zemor [11] proved that some linear double circulant codes follow (14), but only for code rates of $1/2$. If a linear code is proved to comply with (14), then (8) and (14) will complement each other. Namely, Jiang and Vardy reported that (14) improves the GV bound when the relative distance δ is constant. On the other hand, (8) improves the GV bound even when δ approaches to zero.

Conclusion

In section II, we have introduced four exponential-time greedy algorithms. In binary case these algorithms remain asymptotically the best known method for code construction. An open problem is to find polynomial-time construction that meets the greedy-code parameters, or to prove non-existence of such an algorithm.

The exponent in the growth rate of an exponential algorithm is the key factor that determines the running time. Our goal was to find algorithms with smaller exponents in the worst-case running time; though improving the worst-case running time not necessarily guarantees faster algorithm. More important is the average-case running time. In the case of the Lexicographic Construction the worst-case equals the average-case. However, in the case of the Jenkins' construction, we leave the average-case complexity as open problem. On the other hand, the best-case complexity of the Gilbert's construction is $O(nq^k)$. Comparing this best-case with the Lexicographic construction's worst-case, we concluded that not only the Lexicographic Construction has better space complexity than the Gilbert's construction, but also it is faster for, at least, code rates $R \ge 1/2$.

In general, finding a faster algorithm is a difficult task, since a faster algorithm will have to check only a fraction of the q^{n-k} codeword candidates or only a fraction of the $\begin{pmatrix} k \\ d \end{pmatrix}$ row checks. The solution that we propose is combination of the Lexicographic construction and the Jenkins' construction. First, as long as there are available memory resources, run the Lexicographic construction. Then, after the entire memory is used, continue with the Jenkins' construction, while the table $(s, w(s))$ is still kept in memory for reducing the number of row checks.

In section III, we tried to count only once as many linear combinations as possible from of the parity check matrix H . The complex theorem 4 has the best possible estimate on $|H(n-k, d-2)|$. However, even for $d = 5$, there is a big difference between the estimated results (11) and simulated results [4]. The obvious conclusion is that there are still many combinations that are counted multiple times, but we believe that asymptotical improvements similar to [8] may exist for linear codes.

References

1. Trachtenberg, A.: Designing Lexicographic Codes with a Given Trellis Complexity. In: IEEE Trans. Information Theory, vol. 48, No. 1, January 2001, pp. 89-100
2. Jenkins, B.: Tables of Binary Lexicodes. In: http://www.burtleburtle.net/bob/math/lexicode.html
3. Barg, A.: Complexity Issues in Coding Theory. In: Handbook of Coding Theory. Elsevier Science, (1998)
4. Spasov, D.: Implementing the Lexicographic Construction. Available at: http://nislab.bu.edu/nislab/projects/lexicode/index.html
5. O'Brien, K., Fitzpatrick, P.: Covering radius construction codes with minimum distance at most 8 are normal. Available at: http://www.bcri.ucc.ie/BCRI_01.pdf
6. Vardy, A.: Algorithmic Complexity in Coding Theory and the Minimum Distance Problem. In: *STOC* (1997)
7. Grassl, M.: Bounds on the minimum distance of linear codes and quantum codes. Available at: http://www.codetables.de. Accessed on 2009-09-02
8. Jiang, T., Vardy, A.: Asymptotic improvement of the Gilbert-Varshamov bound on the size of binary codes. In: IEEE Trans. Inform. Theory vol. 50, pp. 1655-1664, 2004.
9. Vu, V., Wu, L.: Improving the Gilbert-Varshamov bound for q-ary codes. In: IEEE Trans. Inform. Theory 51, 2005, pp. 3200-3208.
10. Elia, M.: Some results on the existence of binary linear codes. IEEE Trans. Inform. Theory 29, 1983, pp. 933-934.
11. Gaborit, P., Zemor, G.: Asymptotic improvement of the Gilbert-Varshamov bound for binary linear codes. IEEE Trans. Inform. Theory 54, 2008, pp. 3865-3872.
12. Barg, A., Guritman, S., Simonis, J.: Strengthening the Gilbert-Varshamov Bound. Lin. Alg. Appl. 307, 2000, 119-129.
13. O'Brien, K., Fitzpatrick, P.: Improving the Varshamov bound by counting components in the Varshamov graph. In: Designs, Codes, and Cryptography, Vol. 39, No. 3, (2006)
14. Spasov, D., Gusev, M.: Some notes on the binary Gilbert-Varshamov bound. In: Sixth International Workshop on Optimal Codes and Related Topics, Varna, Bulgaria, (2009)