Lexicographic Codes: Constructions Bounds, and Trellis Complexity

Ari Trachtenberg
Digital Computer Laboratory
University of Illinois at Urbana-Champaign
1304 W. Springfield Avenue, Urbana, IL 61801

Alexander Vardy
Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
1308 W. Main Street, Urbana, IL 61801

Abstract. *We study lexicographic codes, which are generated by an iterative greedy construction. We analyze the relationship between successive iterations in this construction, and derive bounds on the parameters of the resulting codes that are tighter than the presently known bounds. Furthermore, we generalize the lexicographic construction to produce families of codes that combine the excellent rate and distance of the lexicodes with other desired properties. Among the various families of generalized lexicodes that we investigate, of particular interest are trellis-oriented lexicodes whose trellis decoding complexity is locally optimized. We also show how to construct generalized lexicographic codes whose trellis state-complexity satisfies a prescribed bound.*

1. Introduction

Lexicographic codes, or *lexicodes* for short, were introduced by Conway and Sloane in [3, 4] as algebraic codes with surprisingly good parameters. Binary lexicodes include, among other famous optimal codes, the Hamming codes, the Golay code, and certain quadratic residue codes [4, 8]. Several authors [2, 4] have proved that lexicodes are always linear. Comparison with optimal linear codes of the same length and dimension [4] shows that lexicodes are usually within one of the optimal minimum distance. Hence, lexicodes may be regarded as a reasonably good "approximation" to the optimal codes. Brualdi and Pless [2] have examined a generalization of lexicodes, known as greedy codes, and presented certain bounds on their parameters. The constructions studied herein are somewhat similar to those of [2].

In this work, we show that lexicodes of increasing length, with a given minimum distance d, can be produced by an iterative procedure we call the *lexicographic construction.* The lexicographic construction is, in turn, a special case of a more general iterative procedure to which we will refer as the *generalized lexicographic construction*, or $\mathfrak{G}$-construction for short. The lexicographic construction and its generalized counterpart establish a relationship between the coset leaders at one iteration and the codes obtained in the next iteration. Using this relationship we derive bounds on the parameters of lexicodes, which are often tighter that the bounds of Brualdi and Pless [2].

Furthermore, the $\mathfrak{G}$-construction makes it possible to improve upon the trellis complexity of lexicodes, in a greedy fashion at each iteration. Thus, we study "trellis-oriented" lexicodes, and show that their trellis complexity is often much lower than that of the original lexicodes. In some cases, it is the lowest trellis complexity known for a given length, rate, and minimum distance. A different application of the $\mathfrak{G}$-construction enables us to design lexicodes that satisfy a predetermined bound on trellis state-complexity. Thus, for any given constants d and s, we can produce reasonably good high-rate binary codes whose minimum distance is at least d and whose trellis state-complexity is at most s.

2. The lexicographic construction

A length n, dimension k, minimum distance d lexicode is traditionally defined constructively based on a lexicographic (i.e. dictionary) ordering of vectors. For example, 01111 comes before 10000 in the lexicographic order. The construction starts with the set $\mathcal{S} = \{\mathbf{0}\}$ and iteratively adds, until exhaustion, the lexicographically earliest vector in $\mathbb{F}_2^n$ whose Hamming distance from $\mathcal{S}$ is at least d.

For example, the codewords of the binary lexicode of length $n = 3$ and minimum distance $d = 2$ are marked by a $\bullet$ in the following lexicographically ordered table:

000	001	010	011	100	101	110	111
$\bullet$			$\bullet$		$\bullet$	$\bullet$	

Figure 1. *A simple* $(3, 4, 2)$ *binary lexicode.*

On the other hand, the *lexicographic construction* starts with a code $\mathbb{C}$ and iteratively adds to the generator matrix of $\mathbb{C}$ the lexicographically earliest generator. If ρ is

*A version of this paper appeared in the Proceedings of the 31st Annual Conference on Information Science s and Systems (1997). This work was supported by the Packard Foundation, the National Science Foundation, and by the Computational Science and Engineering Program at the University of Illinois.

the covering radius of $\mathbb{C}$, then this generator is a binary vector at distance ρ from the code, padded with $d-\rho$ extra ones on the left. This iterative process obviously preserves the minimum distance d of the code and is usually bootstrapped by taking $\mathbb{C}_0 = \{\mathbf{0}\}$. The resulting dimension k code is denoted $\mathcal{L}_k^d$, with d being sometimes omitted in context. For example, Table 1 depicts the generator matrix for a lexicographically constructed code $\mathcal{L}_4^3$; the padding bits on the left are set in boldface.

0000**111**
00**11**001
0**1**01010
1001011

Table 1. *Generator matrix for the* $(6,4,3)$ *lexicode*

As it is known that the lexicodes are linear [2, 4], it is not surprising that the linear codes generated by the lexicographic construction described above are precisely the lexicodes.

Theorem 1. An (n,k,d) code $\mathbb{C}$ is a lexicode if and only if it can be produced by the lexicographic construction, namely $\mathbb{C} = \mathcal{L}_k^d$.

Proof. We first prove that $\mathcal{L}_k^d$ is always a lexicode, by induction on k for an arbitrary, fixed d. For $k=1$ it is clear that $\mathcal{L}_k = \{0^d, 1^d\}$ where 1^d has the usual meaning of d successive 1's. Thus $\mathcal{L}_1^1$ is clearly a $(d,1,d)$ lexicode.

Now assume that $\mathcal{L}_k$ is equal to the (n,k,d) lexicode $\mathbb{C}_k$. ¿From the definition of the lexicographic construction, $\mathcal{L}_{k+1}$ has parameters $(n+\rho, k+1, d)$, where ρ is the covering radius of $\mathcal{L}_k$. Consider the $(n+\rho, k', d)$ lexicode $\mathbb{C}_{k'}$ constructed by repeatedly choosing the appropriate lexicographically-earliest vectors in $\mathbb{F}_2^{n+\rho}$. Clearly, such a choice necessitates that $\mathbb{C}_k \subset \mathbb{C}_{k'}$ which implies, by the induction hypothesis, that $\mathcal{L}_k \subset \mathbb{C}_{k'}$. We may apply our lexicode definition to see that the vector $v = \left(1^{d-\rho}|w\right)$ is in $\mathbb{C}_{k'}$, where w is the lexicographically earliest vector of distance d from $\mathbb{C}_k$ and $(\cdot|\cdot)$ denotes concatenation. Our inductive hypothesis applies to show that $\mathcal{L}_{k+1} \subseteq \mathbb{C}_{k'}$.

In fact, any vector $v \in \mathbb{F}_2^{n+\rho}$ that is in $\mathbb{C}_{k'}$ but not in $\mathcal{L}_{k+1}$ would necessarily have its n right-most bits at distance $\leq \rho$ from $\mathcal{L}_{k+1}$, and its other bits at distance $\leq \lceil (d-\rho)/2 \rceil$ from $\mathcal{L}_{k+1}$. The triangle inequality now shows that the distance from v to $\mathcal{L}_{k+1}$ is at most

$$\rho + \lceil (d-\rho)/2 \rceil \;=\; \lceil (d+\rho)/2 \rceil \;<\; d$$

This contradicts our constructive definition of $\mathbb{C}_{k'}$, Thus, it must be that $\mathcal{L}_{k+1} = \mathbb{C}_{k'}$.

For the second part of the proof, we need to show that an (n,k,d) lexicode $\mathbb{C}(n,k,d)$ can be lexicographically constructed. Suppose that $\mathcal{L}_k^d$, the code produced by k iterations of the minimum-distance d lexicographic construction, has length n'. ¿From the first part of the proof, we know that $\mathcal{L}_k^d$ is a minimum distance d lexicode. Thus, either $\mathbb{C}(n,k,d) \subset \mathcal{L}_k^d$ or else $\mathcal{L}_k^d \subset \mathbb{C}(n,k,d)$. Either way, since $\mathcal{L}_k^d$ and $\mathbb{C}$ both have dimension k, it follows that they must be equal codes (trivial length-altering modifications non-withstanding). □

Theorem 1 makes it possible to restrict our attention to the lexicographic construction, which is easier to handle, and then apply the results to lexicodes.

3. Generalizations

The lexicographic construction may be extended to produce a wide variety of codes through a *generalized lexicographic construction.* The generalized lexicographic construction replaces the "lexicographically earliest" criterion used in building lexicodes with an arbitrary function w, so as to combine the good code parameters of the lexicodes with other optimization criteria. More formally, we have

Definition 2. The generalized lexicographic construction bootstrapped with a linear (n,k,d) seed code $\mathbb{C}$ iteratively constructs the family of codes

$$\mathfrak{G}(w,\mathbb{C}) \;=\; \{\mathfrak{G}_i(w,\mathbb{C}) \;:\; i \in \mathbb{Z}\}$$

using a mapping

$$w(\cdot) : \mathbb{K} \subset \mathbb{F}_2^n \longmapsto v \in \mathbb{F}_2^n \tag{1}$$

where n is arbitrary, according to the following scheme:

- $\mathfrak{G}_0(w,\mathbb{C})$ is trivially the code $\mathbb{C}$
- $\mathfrak{G}_i(w,\mathbb{C})$ is computed by adding to $\mathfrak{G}_{i-1}(w,\mathbb{C})$ the generator

$$\left(1^{n-\delta(v,\mathbb{C})} \;\middle|\; v\right)$$

  where $v = w(\mathfrak{G}_{i-1}(w,\mathbb{C}))$ and $\delta(v,\mathbb{C})$ is the minimum Hamming distance from v to the code $\mathbb{C}$.

The familiar lexicode family is the simple special case $\mathfrak{G}(w_l,\mathbb{S})$, where $w_l(\mathbb{C})$ returns the lexicographically earliest vector with maximum distance from $\mathbb{C}$, and the seed code is $\mathbb{S} = \{0^d, 1^d\}$.

Method 1 greedily computes w_l in linear time and space for appropriate set $\mathcal{V}$. It assumes that the generator matrix of the code has been transformed into a minimal span generator matrix (MSGM) form, in which each generator has minimum span. Adapting the notation in [9], the span of a binary n-vector $x = (x_n, x_{n-1}, x_{n-2}, \ldots, x_1)$ is $\mathrm{L}(x) - \mathrm{R}(x)$, where $\mathrm{L}(\cdot)$ and $\mathrm{R}(\cdot)$ are the smallest (leftmost) and the largest (rightmost) index i, respectively, such that $x_i \neq 0$.

Method 1. Consider a set of vectors $\mathcal{V}$ representing some cosets of the length n code $\mathbb{C}$. Let G be a MSGM for $\mathbb{C}$. The following method computes the lexicographically earliest vector among the represented cosets.

for $v \in \mathcal{V}$ **do**

while $v_{\mathrm{L}(G_i)} \neq 0$

do $v \leftarrow v + G_i$

store the modified v;

among all stored v, return the lexicographically earliest

This method is similar to the greedy algorithm for finding a minimal span generator matrix presented in [9, p.32]. The proof of correctness and time-bound for this method is detailed in [11, Chapter 4].

4. Trellis-oriented lexicodes

The trellis originated [5] in the study of finite-state automata, and was conceived as a means to better explains the inner workings of the Viterbi algorithm for decoding convolutional codes. In 1974, Bahl, Cocke, Jelinek, and Raviv [1] observed that, just like convolutional codes, block codes can be also represented by a trellis. This observation created a potential for applying the well-developed machinery of algebraic and combinatorial theory of block codes to the study of their trellis representations. In this work, we will attempt doing so for the class of lexicodes and their generalizations.

It is shown in [1, 9, 7] how to construct, for an arbitrary linear block code, the BCJR trellis, which has no more edges or vertices at each time index than any other trellis for the code. Hence the BCJR trellis is the unique minimal trellis, as defined by Muder [10]. It is also shown in [9, 7] that the minimal span generator matrix for a linear code, in which each binary generator has minimum span, reflects the properties of the corresponding BCJR trellis for the code. Namely, if G is an MSGM of a code V, comprised of the generators g_j for $j = 1, 2, \ldots, k$, the number of vertices V_i and edges E_i in the corresponding BCJR trellis is given by:

$$\begin{aligned} |V_i| &= 2^{k-p_i-f_i} \\ |E_i| &= 2^{k-p_i-f_{i+1}} \end{aligned}$$

where p_i and f_i are the dimensions of the past and future subcodes of $\mathbb{C}$. These dimensions may be computed for $i = 0, 1, \ldots, n$ as follows:

$$\begin{aligned} p_i &= |\{j : \mathrm{R}(g_j) \geq n + 1 - i\}| \\ f_i &= |\{j : \mathrm{L}(g_j) \leq n - i\}| \end{aligned}$$

We consider a minor modification of the lexicographic construction that exploits the above relations to minimize trellis complexity. It turns out that the family of codes $\mathfrak{G}(w_t, \mathbb{S})$, where $w_t(\mathbb{C})$ is the vector whose bit-reversal is lexicographically earliest among vectors of maximum distance from $\mathbb{C}$, locally minimizes trellis complexity. Hence, these codes may be justly called "trellis-oriented" lexicodes.

Lemma 2. Let $\mathbb{C}$ be a linear code. Over all functions w that result in $\mathfrak{G}_1(w, \mathbb{C})$ having optimal code parameters (i.e. the shortest possible code length), the minimum trellis complexity is attained when $w(\cdot) = w_t(\cdot)$.

Proof. Suppose that G is a MSGM for $\mathbb{C}$, and the corresponding trellis for $\mathbb{C}$ has $V_i = 2^{k-p_i-f_i}$ vertices and $E_i = 2^{k-p_i-f_i}$ edges at time i. Appending to G one more generator $\left(1^{n-\delta(v,\mathbb{C})}|v\right)$, where $v \neq \mathbf{0}$, that does not violate the MSGM condition will produce the following past and future subcode dimensions for the new (n'=n+d-wt(v),k+1,d) code, respectively:

$$p'_i = \begin{cases} 0 & \text{if } i < n' - n \\ p_{i-n'+n} & \text{if } n' - n \leq i \leq n' - R(v) \\ p_{i-n'+n} + 1 & \text{if } n' - R(v) < i \leq n' \end{cases}$$

$$f'_i = \begin{cases} k+1 & \text{if } i = 0 \\ k & \text{if } 0 < i \leq n' - n \\ f_{i-n'+n} & \text{if } n' - n < i \end{cases}$$

Clearly, if $\mathrm{wt}(v) < \rho$, where ρ is the covering radius of $\mathbb{C}$, then the resulting $\mathfrak{G}_1(w, \mathbb{C})$ has longer length than $\mathfrak{G}_1(w_t, \mathbb{C})$. Thus, it must be that $\mathrm{wt}(v) = \rho$. However, if $\mathrm{wt}(v)$ is fixed, then all the future dimensions f'_i are also fixed. Hence higher $\mathrm{R}(v)$ decreases the Viterbi decoding complexity of $2|E| - |V| + 1$. In this a case, picking v such that $R(v)$ is maximum, as is done by $w_t(\cdot)$, optimizes trellis complexity. □

Method 1 can be easily modified to compute w_t by replacing $\mathrm{L}(\cdot)$ with $\mathrm{R}(\cdot)$; the resulting running time is unaffected.

5. Trellis-bounded lexicodes

It is useful to bound various parameters of a trellis so that decoding can be efficiently handled by a system with complexity constraints, such a VLSI chip with certain layering limits. We consider bounding the maximum number of states, as well as the decoding complexity of a trellis for generalized lexicodes.

For any given minimum distance d and dimension k, we can usually generate a reasonably good code whose trellis state complexity is bounded by a predetermined constant s, using the $\mathfrak{G}$-construction. This family of codes is $\mathfrak{G}(w_b, \mathbb{S})$, where $w_b(\mathbb{C})$ is the vector v whose bit-reversal is lexicographically earliest among vectors of maximum distance from $\mathbb{C}$, with the property that the linear code determined by $\mathbb{C} \cup \{(1^{n-\delta(v,\mathbb{C})} \mid v)\}$ has state complexity at most s. This family can be computed by a straightforward application of Theorem 2, in which state complexity is read off directly from the MSGM matrix as the coset leaders are being computed. Nevertheless, this method is not quite as fast as Method 1.

A similar construction (and implementation) yields codes whose Viterbi decoding complexity is bounded either by an arbitrary function. Interesting examples include various polynomial bounds as function of the code length.

6. A relation between cosets

The $\mathfrak{G}$-construction establishes an interesting relationship between the coset leaders of the codes produced at successive iterations. This relationship is best described by associating a *companion* with each coset leader.

Definition 1. The companion of a coset leader u, with respect to $v \in \mathbb{F}_2^n$ is the leader of the coset containing $u+v$. It is denoted $\kappa_v(u)$, or $\kappa(u)$ when v is clear from context.

We show that one iteration of the lexicographic construction produces a code whose coset leaders are the lower-weight vectors between $(a|u)$ and $(\bar{a}|\kappa_v(u))$, for each vector a and coset leader u in the original code $\mathbb{C}$. Here, $\bar{a}$ stands for the complement of a.

Theorem 2. Consider an (n,k,d) code $\mathbb{C}$ with a fixed set $\mathcal{S}$ of coset leaders, and the code $\mathbb{C}' = \mathrm{Span}\left\{(1^\Delta|v), \mathbb{C}\right\}$ for some $v \in \mathbb{F}_2^n$ and $\Delta \in \mathbb{Z}$. Then the set $\mathcal{S}'$ of coset leaders of $\mathbb{C}'$ can be obtained from $\mathcal{S}$ according to the following bijective mapping:

$$\phi : a,\, u \in \mathcal{S} \longmapsto u' \in \mathcal{S}'$$

where u' is defined by the property

$$u' = \begin{cases} (a|u) & \text{if } \mathrm{wt}(a|u) \leq \mathrm{wt}\left(\bar{a}|\kappa_v(u)\right), \\ (\bar{a}|\kappa_v(u)) & \text{if } \mathrm{wt}(a|u) > \mathrm{wt}\left(\bar{a}|\kappa_v(u)\right). \end{cases}$$

and $a \in \{1|0^{\Delta-1}\} \cup \mathbb{F}_2{}^\Delta - 1$.

Proof. Let us introduce the notation

$$\mathrm{f}(a,l) \mathrel{\hat{=}} (a \mid l) \text{ and } \mathrm{h}(a,l) \mathrel{\hat{=}} (\bar{a} \mid \kappa(l))$$

This proof rests upon two observations. The first observation is that $\mathrm{f}(a,l)$ and $\mathrm{h}(a,l)$ are always in the same coset in the new code $\mathbb{C}'$. This is clear because:

$$\begin{aligned} \mathrm{f}(a,l) \;+\; \mathrm{h}(a,l) &= a+\bar{a} \;\mid\; [l+\kappa(l)] \\ &= 1^\Delta \;\mid\; [l+(l+v+c)]\,, \text{ for } c \in \mathbb{C} \\ &= \left(1^\Delta \mid v\right) + \left(0^\Delta \mid c\right) \in \mathbb{C}' \end{aligned}$$

The second observation of this proof is that, $\mathrm{f}(a,l)$ and $\mathrm{f}(a',l')$ are generally not in the same coset. This is specifically true for any two, different coset leaders l and l' of $\mathbb{C}$ and for all $a, a' \in \mathbb{F}_2^\Delta$ such that $\bar{a} \neq a'$. To see this we note two possibilities for $\mathrm{f}(a,l)+\mathrm{f}(a',l')$. In the first case, its leading bits are neither 0^Δ nor 1^Δ. Alternatively, its leading bits are 0^Δ but its trailing bit are $l+l' \notin \mathbb{C}$. Both possibilities preclude $\mathrm{f}(a,l)$ and $\mathrm{f}(a',l')$ from being in the same coset.

Based on the above two observations, f and h represent the same two-to-one correspondence between (vector, coset-leader) pairs (a,l) and the cosets of $\mathbb{C}'$. In fact, for each $a \in \mathbb{F}_2^n$ and coset leader l of $\mathbb{C}$, the coset containing $\mathrm{f}(a,l)$ contains only the vectors:

$$\begin{cases} (a \mid c) & \text{for } c \text{ in the same coset as } l, \\ (\bar{a} \mid (v+c')) & \text{for } v+c' \text{ in the same coset as } \kappa(l). \end{cases}$$

In addition, $\mathrm{f}(a,l)$ cannot have weight greater than the weight of $(a|c)$ because l is a coset leader of the $\kappa(c)$. Similarly, $\mathrm{h}(a,l)$ cannot have weight greater than $(\bar{a}|(v+c'))$. Thus, all the vectors in the coset containing $\mathrm{f}(a,l)$ will have weight $\leq \min\{\mathrm{wt}(\mathrm{f}(a,l)), \mathrm{wt}(\mathrm{h}(a,l))\}$. Since $\mathrm{f}(a,l)$ and $\mathrm{h}(a,l)$ are correspondences, the set $\mathcal{S}'$ as stated is the complete set of coset leaders $\mathcal{C}'$. □

7. Bounds on code parameters

Theorem 2 yields a description of the coset leaders of the code $\mathfrak{G}_k$, generated by the k-th iteration of the $\mathfrak{G}$-construction, in terms of the coset leaders of its predecessor $\mathfrak{G}_{k-1}$. The maximum weight coset leader of $\mathfrak{G}_k$, in turn, determines the covering radius of this lexicode and, hence, the code parameters of the subsequent lexicode in the lexicographic chain. This intuition allows us to improve upon the known bounds on the parameters of lexicodes.

Our first bound is a recursive *upper* bound on the covering radius ρ_m of any code generated using Theorem 2.

Lemma 1.

$$\rho_m \;\leq\; \left\lfloor \frac{d+\rho_{m-1}}{2} \right\rfloor$$

Proof. Consider constructing $\mathfrak{G}_m$ from $\mathfrak{G}_{m-1}$ using Theorem 2. Any two coset leaders l and $\kappa(l)$ of $\mathfrak{G}_{m-1}$ must have weight at most ρ_{m-1}. Therefore we have

$$\begin{aligned} &\min\left\{\mathrm{wt}(a \mid l), \mathrm{wt}\left(\bar{a} \mid \kappa_v(l)\right)\right\} \\ &\qquad \leq \min\left\{\mathrm{wt}(a)+\rho_{m-1}, \mathrm{wt}(\bar{a})+\rho_{m-1}\right\} \\ &\qquad\qquad \leq \lfloor \Delta/2 \rfloor + \rho_{m-1} \end{aligned}$$

that for all $a \in \mathbb{F}_2^\Delta$. In the specific case of $\Delta = d-\rho_{m-1}$, Theorem 2 applies to show that each coset leader l' of $\mathfrak{G}_m$ has weight no greater than

$$\left\lfloor \frac{d-\rho_{m-1}}{2} \right\rfloor + \rho_{m-1} \;=\; \left\lfloor \frac{d+\rho_{m-1}}{2} \right\rfloor$$

This bounds the covering radius of $\mathfrak{G}_m$ and the lemma is proved. □

With some combinatorial manipulation, we can also establish a recursive *lower* bound on this covering radius.

Theorem 3. For $m \geq 3$,