Linear codes over the ring $F_2 + uF_2$

S. Azab [*] M. Zayed [**] M.Atrash [***] M. Ashker [****]

Abstract

In this paper we prove that the code and self-orthogonal codes that we get using an algorithm that is "almost" greedy over the ring F_2+uF_2 with Lee distance are linear when we use B-ordering of vectors for a basis B of $(F_2+uF_2)^n$.

AMS Subject Classification 2000:11T71

Key words:Greedy codes, F_2+uF_2 -linear codes

1 Introduction

Greedy codes over Z_2 are studied by Levenstein [1]. He proved that binary greedy codes are linear when using lexicographic ordering. In [2] Conway and Sloane proved that the greedy codes are linear when using lexicographic ordering over a field of order 2^{2^a}, where a is a positive integer. Pless and Brualdi [3] , Laura Monroe [4], [5] and El-Atrash [6] showed that the binary greedy codes are linear, following different approaches, when they are generated by using a B-ordering and Hamming distance. In [7] El-Atrash showed that greedy and self-orthogonal greedy codes over a field Z_p where p is prime integer using B* -ordering (a modification of B -ordering) and Hamming distance are linear. In [11] El-Atrash and Al-Ashker showed that the codes over Z_4 when they are generated by B-ordering of any ordered basis using an algorithm that is "almost" greedy and Lee distance are linear codes.

In this paper we will give an inductive proof that we get linear codes and linear self orthogonal greedy codes with Lee distance when using a B -ordering of any ordered basis over the ring $F_2 + uF_2$ when using the "almost" greedy algorithm. This paper is the first to study greedy codes over the ring $F_2 + uF_2$ using Lee distance.

2 Definitions and preliminaries

The ring $F_2 + uF_2$ is introduced in [9]. $F_2 + uF_2$ is a commutative ring $\{0, 1, u, 1 + u\}$ with $u^2 = 0$, where F_2 is the binary field with two elements $\{0, 1\}$. Addition and multiplication operations for $F_2 + uF_2$ are given in the following tables:

+	0	1	u	1+u
0	0	1	u	1+u
1	1	0	1+u	u
u	u	1+u	0	1
1+u	1+u	u	1	0

·	0	1	u	1+u
0	0	0	0	0
1	0	1	u	1+u
u	0	u	0	u
1+u	0	1+u	u	1

For convenience we set $v = 1 + u$ and $R = F_2 + uF_2$. A set C of n - tuples over R is called a code over R or R- code. An element of C is called a codeword. If C is an R- submodule of $(R)^n$ we say that C is a linear code over R. As in [9], [10] we use the following terminology. The Hamming weight of a codeword is the number of non-zero components.

The Lee weight for a codeword $x = (x_1, x_2, ...x_n)$ is defined by

$$wt_L(x) = \sum_{i=1}^{n} wt_L(x_i) ,$$

where

$$wt_L(x_i) = \begin{cases} 0 & \text{if } x_i = 0 \\ 1 & \text{if } x_i = 1, \text{ or } 1+u \\ 2 & \text{if } x_i = u \end{cases}$$

The Lee distance between vectors x and $y \in (R)^n$ is defined as

$$d_L(x, y) = \sum_{i=1}^{n} wt_L(x_i - y_i).$$

It follows that

$$d_L(x, y) = wt_L(x - y)$$

Let $n_0(x)$ be the number of components i for which $x_i = 0$, $n_2(x)$ be the number of components i for which $x_i = u$ and $n_1(x) = n - n_0(x) - n_2(x)$ i.e.,

a_1 be the 1's and $1+u$'s in x. Then the Lee weight $wt_L(x)$ of $x = (x_1, x_2, ...x_n)$ can also be defined as :

$$wt_L(x) = n_1(x) + 2n_2(x)$$

If $d = \min\{d_L(x,y) : x, y \in C, x \neq y\}$ then d is called the minimum distance of C.

The inner product of $u = (x_1, x_2, ..., x_n)$ and $y = (y_1, y_2, ..., y_n)$ on $(R)^n$ is defined by

$$x.y = x_1y_1 + x_2y_2 + ... + x_ny_n.$$

The dual code $C^\perp$ of C is defined as

$$C^\perp = \{x \in (R)^n : x.y = 0 \text{ for all } y \in C\}.$$

C is called self-orthogonal if $C \subseteq C^\perp$ and C is called self-dual if $C = C^\perp$. Two codes are equivalent if one can be obtained from the other by permuting the coordinates and exchanging 1 and $1+u$ in certain coordinates.

Following [8] A non-zero linear code C over $R = F_2 + uF_2$ has a generator matrix which after a suitable permutation of coordinates can be written in the form

$$G = \begin{pmatrix} I_{k_1} & A & B \\ 0 & uI_{k_2} & uD \end{pmatrix},$$

where A and B are matrices over R and D is an F_2 matrix. The code C then contains all code words v_0, v_1 G. Where v_0 is a vector of length k_1 over R and v_1 is a vector of length k_2 over F_2. Thus C contains a total of $4^{k_1}2^{k_2}$ codewords. The parameters of C are given by $n, 4^{k_1}2^{k_2}, d_L$ where d_L represents the minimum Lee distance of C.

3 Greedy Codes

Greedy codes are defined in [1], [2], [3], [4], [5], [6], [7], [11]. For the purpose of this paper we define the ordering and the greedy codes in the following general way:

3.1 Definition

In the binary ordering case there is a " natural " way of ordering the vectors called the lexicographic ordering [1] in which $0 \leq 1$ and a vector $(a_1, a_2, ..., a_n) \leq (b_1, b_2, ..., b_n)$ if $i = k$ such that $a_k = b_k$ for all $1 \leq k \leq i$ and $a_i = 0$, $b_i = 1$.

Pless and Brualdi [3] have generalized this and defined what they called B-ordering of vectors over Z_2 as in the following definition .

3.2 Definition [4]

A B-ordering is an ordering of vectors of length n over the binary field obtained recursively from an ordered basis $B = \{b_1, b_2, ..., b_n\}$, which can be any ordered basis of the binary vectors of length n . The first vector in the B-ordering is the zero vector and the next is b_1. The B-ordering is then generated recursively , where if the first 2^{i-1} vectors of the ordering have been generated using the basis elements $\{b_1, b_2, ..., b_{i-1}\}$, then the next 2^{i-1} vectors are generated by adding b_i to those vectors already produced in order.

3.3 Definition

A greedy code of vectors of length n listed in some ordering over a ring R and designed distance d is generated as follows: The first vector in the ordering is selected for the code. Then proceeding once through the ordering , a vector is selected if its distance from all previously chosen vectors is at least d. If we use Lee distance, we will call it Lee greedy code over R. Clearly, greedy codes depends on the ordering of the vectors and the type of the distance .

4 Main Results

Here we define the B-ordering over $(R)^n$ in the following way:

4.1 Definition

Let $B = \{b_1, b_2, ..., b_n\}$ be a basis for the module $(R)^n$ over R . We define the B-ordering as follows: The first 4 vectors are $0, b_1, ub_1, vb_1$.

The B-ordering is then generated recursively , where if 4^k vectors of the ordering have been generated using basis elements , $b_1, b_2, ..., b_k$, then the

next $3(4^k)$ vectors are generated by adding vb_{k+1} to those vectors already produced , in order $k = 1, u, v$.

Note: Let M be an abelian group and R be a ring , iff M is a unitary R-module and $B \subseteq M$ then we say that B is a basis for M iff B is a linearly independent subset of M that spans it , in this case M is said to be a free R-module. A ring theorist would point out that codes over $R = F_2 + uF_2$ are not in general a free module , and so need not have a basis. Although this true: in coding theory every linear code over the ring $R = F_2 + uF_2$ has a generator matrix whose rows form a basis for the code. i.e., a basis for a code over R is a minimal subset of vectors that spans the code.

4.2 Example

Let $B = \{b_1, b_2\}$ be a basis over R .

Then the B-ordering is

$0, b_1, ub_1, vb_1,$

$b_2, b_2 + b_1, b_2 + ub_1, b_2 + vb_1,$

$ub_2, ub_2 + b_1, ub_2 + ub_1, ub_2 + vb_1,$

$vb_2, vb_2 + b_1, vb_2 + ub_1, vb_2 + vb_1.$

If we consider the standard basis

$b_1 = 000..1, b_2 = 000..10, ...b_n = 10...00$

In this case we call the B -ordering of R the lexicographic ordering where x comes before y in the ordering when $x_i = \alpha$ and $y_i = \gamma$, where $\alpha \leq \gamma$, α and $\gamma \in R$ and where i is the last position in which x and y differ. where $0 \leq i \leq u \leq v$.

Notation: We write vectors without parenthesis and comma's i.e., we write $u1u$ for $(u, 1, u)$

4.3 Example.

Let $B = \{u01 \ , 1u1 \ , u1u\}$ be a basis over R. Then the Lee greedy codes generated by the B-ordering is as follows:

B-ordering			Greedy codes		
000	vvl	v0v	$d_1 = 2$	$d_1 = 4$	$d_1 = 6$
n0l	ivO	i0u	000	000	000
00u	0lO	v0l	n0l	iul	uuu
n0v	nul	i00	00u	n0u	
iul	0lu	uvu	n0v	vuv	
vu0	nlv	0vv	iul		
iuv	ivl	uv0	vu0		
vuu	vv0	0vl	iuv		
n0u	ivv	vlv	vuu		
00v	vvu	ilu	nlu		
n00	0u0	vll	0lv		
00l	uul	ill	nl0		
vuv	0uu	0v0	0ll		
iuu	uuv	uvl	vvv		
vul	i0l	0vu	ivO		
iu0	v00	uvv	0u0		
nlu	i0v	ill	nul		
0lv	v0u	vl0	uuv		
nl0	uuu	ilv	uvu		
0ll	0uv	vlu	uv0		
vvv	uu0		0vl		
ivu	0ul				

This example shows that greedy codes for $d_1 = 2$ in column 4 is not linear but for $d_1 = 4$ and for $d_1 = 6$ it is linear. It follows that the greedy code when generated by B-ordering over R is not always linear.

As in paper [11], we can modify the greedy algorithm to another one called the (almost-greedy algorithm) in order to produce linear codes over R as follows:

4.4 Almost-Greedy Algorithm

1. Use B-ordering to order all vectors in $(R)^n$.

2. Divide the whole list into $n + 1$ bins in the following way: the 0-bin contains only the zero vector and the i-th bin contains all vectors as

ordered in the B-ordering of the form $\alpha b_j + w$ where w ranges over all vectors in the previous bins , $\alpha \in F_2 + F_2$.

3. Set $C_0 = \{0\}$

4. Assume we have defined C_{i-1}. To define C_i we do the following.

   We look for the first vector x in the ith bin such that $d_L(x, c) \geq d$, $d_L(ux, c) \geq d$, for all $u \in C_{i-1}$. If such an x is found in the ith bin then we set

   $$C_i = C_{i-1} \cup \{\alpha x + c \mid \alpha \in C_{i-1}, \alpha \in R\}$$

   and we jump to the next bin, if no such x is found then we set $C_i = C_{i-1}$.

5. Process continues until the end of $n+1$ st bin . Set $C = C_{n+1}$.

In the greedy algorithm, we test all vectors in the ordering for suitability. However in this algorithm we only pick up first vector we find that suits and then jump to the next bin. This is why it is called almost greedy. The reason for using such an algorithm and not using greedy algorithm is the fact that in the case of greedy algorithm we may encounter a vector x that suits but it would make the code non-linear. This does not happen in case of vector space over fields.

4.5 Example

Consider the basis as in example (4.3) . Let $d = 2$ then the greedy code using the almost-greedy algorithm is given in the following table.

B-ordering			Greedy code using almost greedy algorithm
000	vv1	v0v	$C_0 = \{000\}$
$b_1 = u01$	1v0	10u	$C_1 = \{000, u01, 00u, u0v\}$
00u	010	v01	
u0v	u11	100	
1u1	01u	uvu	
vu0	u1v	0vv	
1uv	1v1	uv0	
vuu	vv0	0v1	
u0u	1vv	v1v	
00v	vvu	1u1	
u00	0u0	v11	
001	uu1	110	
vuv	0uu	0v0	
1uu	uuv	uv1	
vu1	101	0vu	
1u0	v00	uvv	
$b_2 = u1u$	10v	1u1	C_2
01v	v0u	v10	
u10	uuu	1uv	
011	0uv	v1u	
vvv	uu0		
1vu	0u1		

Where $C_2 = \{000, u01, 00u, u0v, u1u, 0u0, uvu, 01v, u10, 011, uu1, 0uu, uuv, 0vv, uv0, 0v1\}$
This example shows that the greedy code with $d = 2$ when generated by B-ordering and using almost greedy algorithm over R is linear.

Here, we state and prove the first of two main theorems of this paper.

4.6 Theorem

Let C be the code we get using the B-ordering and the almost-greedy algorithm with Lee distance $d \geq 1$. Then C is a linear code over the ring R.

Proof: We will use induction on i to show that C_i is a linear code (C_i as in the algorithm). This is obviously true for $i = 0$. Assume that it is true for $i = k$.

To prove linearity of C_{k+1}, let x be the first vector chosen in the $(k+1)$ bin, then x satisfies:

$d_L(x, c) \geq d$ and $d_L(ux, c) \geq d$ for all $c \in C_k$.

We will show that

$$d_L(\alpha x + c, \beta x + t) \geq d$$

for all $c, t \in C_k$ and $\alpha, \beta \in R$.

$$d_L(\alpha x + c, \beta x + t) = wt_L(\alpha x + c - \beta x - t) = wt_L((\alpha - \beta)x + (c - t))$$

$$= wt_L(\gamma x + w),$$

where $\gamma = \alpha - \beta \in R$, $w = c - t$.

$w \in C_k$ by linearity of C_k, and by induction $wt_L(w) \geq d$.

For $\gamma = 0$, $wt_L(w) \geq d$ by induction on C_k.

For $\gamma = 1$, $wt_L(x + w) = wt_L(x - w) \geq d$ by the choice of x and $w \in C_k$.

Since

$$wt_L(x) = n_1(x) + 2n_2(x) = n_1(vx) + 2n_2(vx) = wt_L(vx).$$

Then for $\gamma = u$ this implies that

$$wt_L(vx + w) = wt_L(v^2x + vw) = wt_L(x + vw) = wt_L(x - vw) \geq d$$

by the choice of x, $v^2 = 1$ and since $vw \in C_k$.

Finally for $\gamma = u$ we have $wt_L(ux + w) = wt_L(ux - w) \geq d$, since $w \in C_k$ and by the choice of x.

It follows that $d_L(\alpha x + c, \beta x + t) \geq d$ for all $c, t \in C_k$.

It follows that $C_{k+1} = \langle x, C_k \rangle$ is a linear code. This completes the proof.

4.7 Corollary.

The linear code C over R of Lee distance $d \geq 1$ when is generated by the B-ordering and using the almost greedy algorithm contains only codewords that satisfy the following:

(1) If $x \in C$ and x contains only 0 and u components then $n_2(x) \geq \frac{d}{2}$.

(2) If $x \in C$, x contains $0, 1, u$ and v components then $n_1(x) \geq \frac{d}{2}$.

Proof:

(1) If $x \in C$ and u contains only 0 and w components then

$$wt_L(x) - 2n_2(x) \geq w.$$

This implies that $n_2(x) \geq \frac{w}{2}$.

(2) If $x \in C$and u contains $0, 1, w$ and v components then

$$wt_L(x) - wt_L(vx) - 2n_2(x) + n_1(x) \geq w.$$

Since the code C is linear then $ux \in C$ and

$$wt_L(ux) - 2n_2(ux) - 2n_1(x) \geq w$$

Then $n_2(ux) \geq \frac{w}{2}$ and hence $n_1(x) \geq \frac{w}{2}$.

4.8 Definition.

A self orthogonal code oven R of length n and designed Lee distance w when it is generated by the B-ordering and using the almost greedy algorithm is a code with additional restraint that the vectors must be orthogonal to themselves and each other.

The following is the second main theorem in this paper.

4.9 Theorem.

Let C be the self-orthogonal code we get using the B-ordering and the almost-greedy algorithm with Lee distance $d \geq 1$. Then C is a linear code over R.

Proof: The proof is almost same as the proof at Theorem (4.6), however, We have to make sure that our codewords are self orthogonal.

We prove by induction that the self- orthogonal code C is linear.

We will use induction on k to show that C_i is linear self-orthogonal code (C_i as in the algorithm). This is clearly true for $k = 0$. Assume it is true for $k = k$.

To prove linearity of C_{k+1}, let u be the first vector chosen in the $k+1$ bin. then u satisfies.

(1) $d_L(x, c) \geq d$ and $d_L(ux, c) \geq w$ for all $c \in C_k$

(2) $xu - 0$, $xc - 0$ for all codewords $c \in C_k$.

We will show that

(1) $d_L(\alpha x + c, \beta x + t) \geq d$, for all $c, t \in C_k$ and $\alpha, \beta \in R$.

(2) $(\alpha x + c).(\beta x + t) - 0$

Since

$$d_L(\alpha x + c, \beta x + t) - wt_L(\alpha x + c - \beta x - t) - wt_L((\alpha - \beta)x + (c - t))$$

$$- wt_L(\gamma x + w),$$

where $\gamma - \alpha - \beta \in R$, $w - c - t$

$w \in C_k$ by linearity of C_k, and by induction $wt_L(w) \geq d$.

For $\gamma - 0$, $wt_L(w) \geq d$ by induction on C_k.

For $\gamma - 1$, $wt_L(x + w) - wt_L(x - w) \geq d$ by the choice of x since $w \in C_k$.

Since

$$wt_L(x) - n_1(x) + 2n_2(x) - n_1(vx) + 2n_2(vx) - wt_L(vx).$$

Then for $\gamma - u$ this implies that

$$wt_L(vx + w) - wt_L(v^2x + vw) - wt_L(x + vw) - wt_L(x - vw) \geq d$$

by the choice of x, $v^2 - 1$ and since $uw \in C_k$.

Finally for $\gamma - u$ we have $wt_L(ux + w) - wt_L(ux - w) \geq d$, since $w \in C_k$ and by the choice of x.

It follows that $d_L(\alpha x + c, \beta x + t) \geq d$ for all $c, t \in C_k$.

Also

$$(\alpha x + c).(\beta x + t) - \alpha\beta x.x + \alpha x.t + \beta c.x + c.t - 0,$$

since $xu - 0$, $xt - 0$, $xc - 0$ and $ct - 0$.

It follows that $C_{k+1} - \langle x, C_k \rangle$ is a linear self-orthogonal code.

This completes the proof.

4.10 Example.

Let $n = 4$, B = {(1000), (0100), (0010), (0001)}be a basis over R.

The following table shows the B-ordering in columns 1 to 8 and the codes over R with minimum Lee distances $d_1 = 4$, $d_1 = 6$, $d_1 = 8$ in columns 9 to 11.

This example shows that the self orthogonal codes generated using the R-ordering over R and the almost-greedy algorithm are linear.

B-ordering						
0000	0001	0002	0003	0010	0011	0012
1000	1001	1002	1003	1010	1011	1012
2000	2001	2002	2003	2010	2011	2012
3000	3001	3002	3003	3010	3011	3012
0100	0101	0102	0103	0110	0111	0112
1100	1101	1102	1103	1110	1111	1112
2100	2101	2102	2103	2110	2111	2112
3100	3101	3102	3103	3110	3111	3112
0200	0201	0202	0203	0210	0211	0212
1200	1201	1202	1203	1210	1211	1212
2200	2201	2202	2203	2210	2211	2212
3200	3201	3202	3203	3210	3211	3212
0300	0301	0302	0303	0310	0311	0312
1300	1301	1302	1303	1310	1311	1312
2300	2301	2302	2303	2310	2311	2312
3300	3301	3302	3303	3310	3311	3312
0010	0011	0012	0013	0020	0021	0022
1010	1011	1012	1013	1020	1021	1022
2010	2011	2012	2013	2020	2021	2022
3010	3011	3012	3013	3020	3021	3022
0110	0111	0112	0113	0120	0121	0122
1110	1111	1112	1113	1120	1121	1122
2110	2111	2112	2113	2120	2121	2122
3110	3111	3112	3113	3120	3121	3122
0210	0211	0212	0213	0220	0221	0222
1210	1211	1212	1213	1220	1221	1222
2210	2211	2212	2213	2220	2221	2222
3210	3211	3212	3213	3220	3221	3222
0310	0311	0312	0313	0320	0321	0322
1310	1311	1312	1313	1320	1321	1322
2310	2311	2312	2313	2320	2321	2322
3310	3311	3312	3313	3320	3321	3322

4.11 Corollary.

The self-orthogonal code for odd designed Lee distance d is the same as the self-orthogonal code for $d+1$ when they are generated by the B-ordering using the almost -greedy algorithm.

Proof: The code is linear, so all sums of vectors must be in the code. The code is self-orthogonal, so the weights of these sums must be even. This is the same as saying that all vectors in the code differ from each other in at least $d+1$ Lee distance. Henceforth, when we refer to the chosen minimum distance d, it will be understood that d is even.

4.12 Example

Consider the self-orthogonal codes in example (4.10) for $d = 5$ and for $d = 6$. They are the same code $\{0000, uuu0\}$.

4.13 Theorem

Let F be a field. Let V be a vector space over F. Let $B = \{b_1, b_2, ...b_n\}$ be a basis for V over F. Use the B-ordering to order the vectors in V. If the greedy code using the greedy algorithm is linear code then we get the same code using the almost-greedy algorithm.

Proof: We will use induction on i to show that the code C_i we get using the greedy algorithm is the same code C_i' we get using the almost-greedy algorithm. Where C_i is the code generated by $\langle b_1, b_2, ...b_i \rangle$ using the greedy algorithm, and C_i' is the code generated by $\langle b_1, b_2, ...b_i \rangle$ using the almost greedy algorithm.

If $0 \leq k \leq i$ and both are the linear. Then we will show that $C_{k+1} = C'_{k+1}$. It is obviously true for the case $i = 0$. Assume that $C_k = C'_k$ for

Now if there exist a vector x in the $k+1$ bin that satisfies the greedy algorithm, choose x to be the first.

It follows that $d(x, c) \geq d$ for all $c \in C_k \cup C'_k$.

And

$$C'_{k+1} = \{\alpha x + c \mid \alpha \in F, c \in C'_k\}.$$

But $C_k = C'_k$ then

$$C'_{k+1} = \{\alpha x + c \mid \alpha \in F, c \in C_k\}.$$

Now we would like to show that $u \in C_{k+1}$ if and only if u has the form

$$\alpha x + c, \text{ where } \alpha \in F \text{ and } c \in C_k.$$

Suppose that $u \in C_{k+1}$, then $d(v,u) \geq d$ for every vector u chosen before v.

u has the form

$$\beta b_{k+1} + w \text{ where } w \in \langle b_1, b_2, \ldots b_k \rangle.$$

Since u is in the $k+1$ st bin, it follows that u has the form

$$u = \gamma b_{k+1} + w_0, \text{ where } \gamma \neq 0, \ w_0 \in \langle b_1, b_2, \ldots b_k \rangle.$$

It follows that

$$b_{k+1} = \gamma^{-1} x - \gamma^{-1} w_0,$$

then

$$u = \beta(\gamma^{-1} x - \gamma^{-1} w_0) + w = \beta\gamma^{-1} x + w',$$

where $w' = w - \beta\gamma^{-1} w_0$.

All we need to show that $w' \in C_k$.

For simplicity we let $\gamma = 1$ since the vector $b_{k+1} + \gamma^{-1} w_0$ comes before u in the ordering and it satisfies the greedy algorithm. Thus $u = b_{k+1} + w_0$ and $u = \beta x + w'$.

We first need to show that $\beta x + c \in C_{k+1}$.

since for $c' \in C_k$,

$$d(\beta x + c, c') = wt(\beta x + c - c') = wt(\beta x + c'')$$

$$= wt(\beta x - (-c'')) = wt(x - (-\beta^{-1} c'')),$$

where $c'' = c - c'$ and $(-\beta^{-1} c'') \in C_k$, so $\beta x + c \in C_{k+1}$ it follows that

$$d(w', c) = wt(w' - c) = wt(\beta x + w' - \beta x - c) \geq d,$$

for all previously chosen vectors $c \in C_k$. Thus $w' \in C_k$. i.e., $u = \beta x + w'$, $w' \in C_k$.

Conversely if $u = \beta x + c$, $c \in C_k$, then for $c' \in C_k$,

$$d(v, c') = wt(v - c') = wt(\beta x + c - c') \geq d ,$$

by the linearity of C_k and the choice of x. Thus $u \in C_{k+1}$.

This proves that $C_{k+1} = C'_{k+1}$.

4.14 Theorem

Let R be the ring $F_2 + uF_2$. Let V be an R-module over R. Let B be a basis of V over R. Let W_k be the submodule generated by $\{b_1, b_2, ...b_k\}$. Then no matter how vectors are ordered within W_k, the almost greedy algorithm produces equivalent codes.

Proof: Suppose that C and C' be two codes we get using almost-greedy algorithm and two different rearrangements of ordered vectors within W_k. We will use induction to show that C and C' are two equivalent codes. Let C_i and C'_i be the two codes we get using almost-greedy algorithm and two different rearrangements of ordered vectors in W_i. This is true for $i = 0$, since both consists of the zero vector. Assume it is true for $i = j$ and assume that there is a one to one correspondence between C_i and C'_i. To prove it is true for $i = j + 1$. Let x_i be the first vector chosen in the $j + 1$th bin for C_{j+1} such that

$$d_L(x_i, c) \geq d \quad \text{and} \quad d_L(ux_i, c) \geq d \text{ for all } c \in C_j$$

Let y_i be the first corresponding vector chosen in the $j+1$th bin for C'_{j+1} such that

$$d_L(y_i, c') \geq d \quad \text{and} \quad d_L(uy_i, c') \geq d \text{ for all } c' \in C'_j$$

Define $\Psi(x_i) = y_i$, and extend Ψ linearly.

This shows that

$$\Psi : C_{j+1} \Longrightarrow C'_{j+1}$$

is a one to one correspondence and C_{j+1} is equivalent to C'_{j+1}. This completes the proof.

5 References

1 | V.Levenstine, "A class of systematic codes" Dokl.Akad ii (1960),368-371.

2 | J.Conway and N.J.A.Sloane, "Lexicographic codes;Error correcting codes from game theory ",IEEE Trans.Inform.Theory IT-32(1986),337-348.

3 | R.Brualdi and V.Pless,"Greedy codes" JCT(A)64(1993),10-30.

[4] Laura,Monroe, "Binary greedy codes",to appear in congressus numeratium , vol. 100-104

[5] Laura,Monroe, "Self orthogonal greedy codes", Designs,codes and cryptography ,9(1):79-83,August 1996

[6] M.EL-Atrash," Linearity of binary greedy codes",The Islamic University Journal,Volume-8-NO.2 Part 2-June 2000

[7] M.EL-atrash, "Greedy codes over Z_p", conference on " Information Technology and its future role", Musil, Iraq, Mar., 11-13, 2000

[8] A. Bonnecaze and P. Udaya, "Cyclic codes and self-dual codes over $F_2 + uF_2$ ", IEEE IT, Vol 45, no. 4, pp. 1250-1254, May 1999

[9] Bachoc,C.: "Application of coding theory to the construction of modular lattices", JI Combin. Theory Ser. A 78, pp. 92-119, (1997).

[10] Dougherty, S. T., Gaborit, P., Harada, M., Sole', P.: "Type II codes over $F_2 + uF_2$", IEEE Trans. Inform. Theory 45, pp. 32-45, (1999).

[11] M. EL-Atrash and M. AL-Ashker "Linear codes over Z_4 using almost-greedy algorithm ", The Islamic University Journal, Vol. 11, No. 1, Natural Sciences Series, pp. 20-34, January, 2003